Tripoli University

Faculty of Engineering

Computer Engineering Department

A graduation project is submitted in partial fulfilment of requirements for the degree of Bachelor in Computer Engineering

DESIGN AND IMPLEMENTATION OF A SMART AUTOMATED ROTARY CAR PARKING SYSTEM ON “FPGA”

By:

Adnan Abdurazaq Alhengari

Supervised By:

Dr. Mohamed Muftah Eljhani.

Fall 2022

Specially dedicated

My dear Father and beloved Mother and my Brothers

Acknowledgment

A lot of praise and 'syukur' to ALLAH.

First and foremost, I would like to thank my supervisor Dr. Mohamed Eljhani, whose its guidance has been invaluable in my time as a undergraduate student and its patience and encouragement throughout this project; I would never finish this project as Is now. Thank you very much!

I would like to thank the faculty members of the Department of Computer Engineering, whose their lectures driven me to here I had the privilege of attending It while I studying there and making my study at University of Tripoli a pleasant experience.

Last but not least, I would like to use this opportunity to say thank you to my family for all the love and persistent incorporeal and materialistic support. Finally, I would like to express my appreciation to all my friends, thanks for all support and help.

Thank you to all of you.

Abstract

This project presents the design and implementation of a rotary car parking system to improve and develop the mechanical model that was designed and implemented in the Mechanical Engineering Department, University of Tripoli [1], and to make this system more useful and reliable, and we used field programmable gate array (FPGA) to control all mechanical parts and make them fully automatic, so that FPGA controls a movement of the system when cars enter and exit and it also controls whether or not to allow entry if the car violates the conditions in terms of weight, and for protection the FPGA controls whether or not the car is requested to exit asked the customer is to enter the password of the room. If it is correct the car will be delivered to him, if it is not correct the exit request is rejected.

The system consists of a rotary platform that can rotate 360 degrees, allowing cars to be parked and retrieved from any direction. also, this system has sixteen rooms, in each room several sensors are installed to control the direction of rotation and determine its location within the system.

And for protection from fires a fire extinguishing system that is controlled by the FPGA has been included, by measuring the room temperature so that there is a temperature sensor inside each room.

The FPGA-based control system provides a flexible and scalable solution that can be easily adapted to different parking requirements. Experimental results demonstrate that the proposed system can effectively park and retrieve cars in a timely and efficient manner, making It an ideal solution for urban areas with limited parking space.

Table of contents

List of Figures

Chapter 1

Introduction

An overview of the problem statement, purpose, and methodology of the project will be provided in this chapter.

1.1. **Statement of problem**

Congestion results from the increase of cars being out of proportion to the parking spaces that are available. A prototype rotational parking system is then developed from the issue, as shown in Fig. (1.1). The idea of a rotational car park is one of the mechanical car park systems where cars are placed on vertical rack chains arranged on both sides and its controls operate automatically with drives. The main idea is to piled up sixteen cars in the space usually occupied by two cars.

Figure 1.1:Modern day parking system.

Rotary parking systems stack automobiles both vertically to make the most of the available space. There are many tiers of parking places built around a circular platform that revolves around a central axis. Vehicles are parked on these levels and are raised or lowered to their predetermined locations by mechanical lifts.

Rotary car parking systems provide several advantages, including better efficiency, reduced land utilization, and improved safety. They take up a lesser amount of area than standard parking lots since they can fit more cars per square foot. They also reduce the need for ramps and driving lanes, which can consume precious real estate.

Rotary car parking systems have become increasingly popular in densely populated cities around the world. They are also used in airports and other public facilities where space is limited.

Overall, rotating car parking systems present a novel approach to the issue of limited parking space in urban areas. They will undoubtedly be crucial in the planning and growth of cities in the future because of their effective design and capacity to make the most of available space.

1.2. **Mechanical design**

The mechanical part of this project has been designed and built in the Department of Mechanical, University of Tripoli [1], Fig. (1.2.b) shows this design and we will review below some of its main parts and how they work.

- **Frame**

  System it connects the most of pieces with together. Every component like the assembly of pallet, motor drive chain, sprocket, is installed over it Fig. (1.2.a).

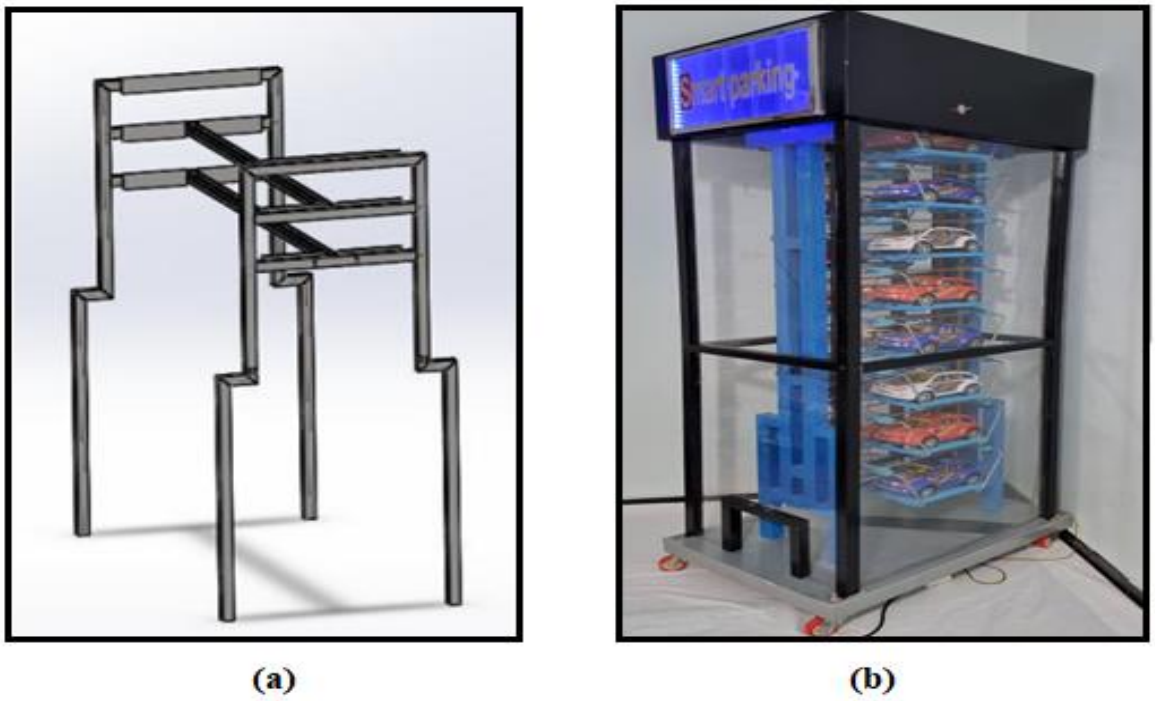

Figure1.2 : a. Frame structure. b. Mechanical design. .

- **Pallet**

  The main function of the pallet in mechanical system to hold the car in its place; so, it has to be with stand the load of the car and to be stable while the system is moving Fig. (1.3.a).

- **Roller chain**

  The goal of this element is to connects between gearbox and sprocket Roller chain or bush roller chain is the type of chain drive most commonly used for transmission of mechanical power on many kinds of domestic, industrial and agricultural machinery, including conveyors, wire- and tube-drawing machines, printing presses, cars, motorcycles, and bicycles. It consists of a series of short cylindrical rollers held together by side links. It is driven by a toothed wheel called a sprocket. It is a simple, reliable, and efficient means of power transmission Fig. (1.3.b).

- **Sprocket**

  A sprocket or sprocket-wheel is a profiled wheel with teeth, cogs, or even sprockets that mesh with a chain, track or other perforated or indented material. The name 'sprocket' applies generally to any wheel upon which radial projections engage a chain passing over it. It is distinguished from a gear in that sprockets are never meshed together directly and differs from a pulley in that sprockets have teeth and pulleys are smooth Fig. (1.3.c).

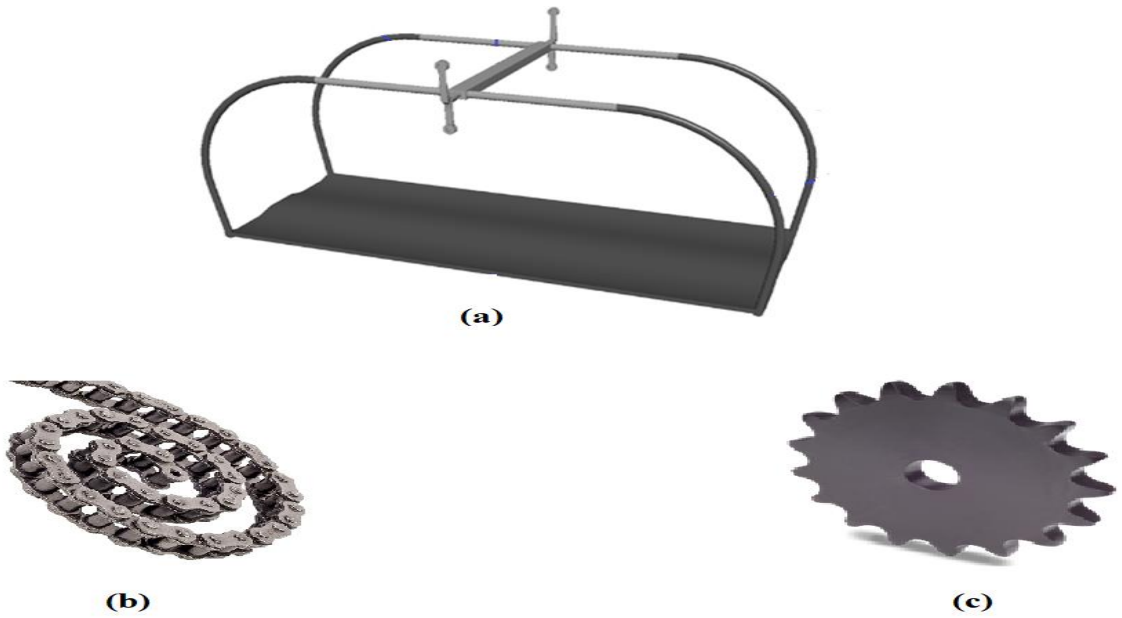

Figure 1.3: a. Pallet b. Roller chain c. Sprocket

- **Beam Rod**

  Used for connecting between the sprockets and allow the system to rotate also it is connected to the gearbox Fig. (1.4.a).

- **Triangle**

  Used to Connect between Pallet and frame and make rotating pallets much easier Fig. (1.4.b).

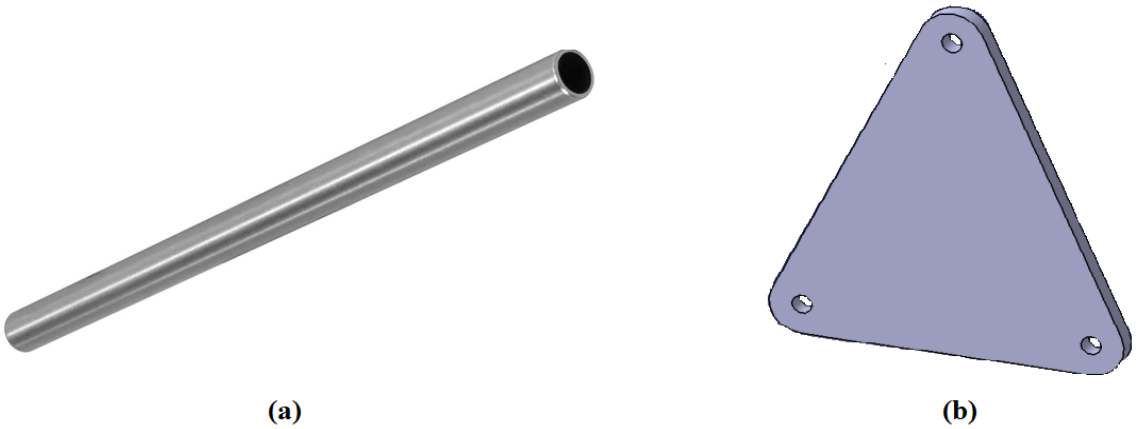

Figure 1.4: a.Beam Rod. b.Triangle.

1.3. **Purpose**

The main purpose of this project is to construct a prototype of a rotary car parking system. This project contains the three fundamental components of mechatronics, which are mechanics, electronics and programming. The following research inquiries will then be investigated:

1. **Are there any vacant places?**

   One of the first issues that drivers may encounter is the inability to always tell if there are empty baskets in the system.

2. **Calculating the parking time within system**.

When the digital system is not used, it is impossible to precisely determine when the car entered the system, which prevents the cost from being calculated correctly.

3. **Determine the shortest path to reach the nearest basket**.
   One of the most crucial things you can do to shorten the system's lifespan is to figure out the shortest route to the empty basket.

4. **Improper parking**.
   To avoid the complete mechanical system collapsing if the car weighs more than the designer's maximum allowed.

1.4. **Method**

The main idea behind the rotary car parking system is that as cars are brought onto the parking platform one at a time, a sensor in the system detects the occupied slot and rotates the car upward. The user should later be able to retrieve the vehicle using a display wherein a number associated with the occupied parking platform is entered, causing the desired vehicle to rotate and locate the shortest route downward. The rotary car parking system prototype requires the following parts: a field programmable gate array (FPGA), strain gauge measuring pressure sensor, DC-motor, sprockets, roller chains, and revolving platforms.

1.5. **Project organization**

The next chapter provides general background to rotary car parking system. Chapter 3 talk about similar works that have been written and the controllers used in these works. Capter4 explains how each part of the program works and presents the test results. Chapter 5 presents a conclusion of this project.

Chapter 2

Background

2.1. History

With the increase in the number of vehicles in high-density urban housing the problem with a limited amount of parking space appeared. The solution to this problem is the automatic parking system. The first such system was created in the United States the automatic parking system attracted attention in the years 1940-1950, and the systems used were Bowser, Pigeon Hole and Roto Park. In the years 1957 to 1974 Bowser and Pigon Hole systems were used [2], but due to frequent mechanical problems and prolonged waiting time for the vehicle, the interest in such a solution significantly decreased. This interest, however, came back since 1990, and already in 2012 there were 25 projects planned or in progress, which yielded about 6,000 parking spaces. While until 1990 the interest in the systems was mediocre in Europe, Asia and Central America, since 1970 in the automatic parking system more advanced technology was use. In Japan, since the early 1990's about 40,000 parking spaces based on the automatic parking system paternoster type were created. It is estimated that in Japan in 2012 there were about 1.6 million parking spaces. A car parking system is a mechanical device that multiplies parking capacity inside a parking lot. Parking systems are generally powered by electric motors or hydraulic pumps that move vehicles into a storage position. There are two types of car parking systems: traditional and automated. In the long term, automated car parking systems are likely to be more cost effective when compared to traditional parking garages. Automatic multi-store automated car park systems are less expensive per parking slot, since they tend to require less building volume and less ground area than a conventional facility with the same capacity. Both automated car parking systems and automated parking garage systems reduce pollution cars are not running or circling around while drivers look for parking spaces [3].

2.2. Literature Review

Automatic parking systems have been around for numerous years and were developed to enhance conventional parking. They are designed to make parking easier for the user and save a lot of space.

The first automated parking system was introduced in Paris in 1905 [4]. The garage on Rue de Ponthieu had an elevator system that could lift vehicles to various floors. Fig. (2.2) displays the system.

Since then, autonomous systems in many fields have developed tremendously and are vital in today's society.

Around 1920-1960, a parking system known as the "Paternoster system" was created. It had a Ferris wheel-like design and could fit eight cars in a place that would normally hold only two [5]. Due to its simplicity of use and compact design, as seen in Fig. (2.1), the structure quickly gained popularity.

Figure 2.5: 'Paternoster system' .

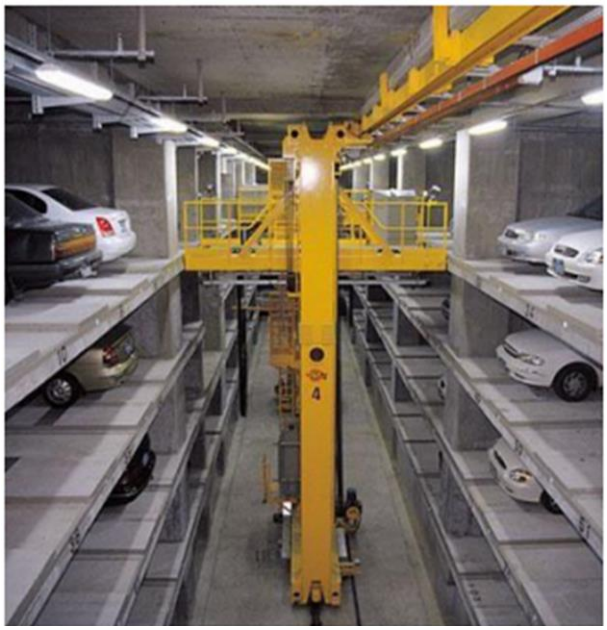

Figure2.6: various floors parking system.

2.3. **Controller**

There are numerous microcontroller boards available for selection. In this research, the rotary car parking system was implemented using a Field Programmable Gate Array (FPGA).

The FPGA board as a controller offers numerous benefits over conventional microcontrollers or processors, including flexibility, high-speed processing, low latency, customization, decreased cost, and reduced power usage.

FPGA boards come in a variety of forms. In this project, we use photographs of the DE2i-150 board.

Chapter 3

Related Work

Rotary parking systems have grown in popularity in the last five years due to its space-saving and effective design. Several studies and research papers have been published on this topic, outlining the merits and downsides of various systems. We mention a few of them:

In (S. Suliman, Y. Ali, 2016) [6], As a controller in this project, an ATmega16 microcontroller was chosen for the RPS. The control algorithm is coded in the C programming language. PROTEUS simulation program is used to draw control circuits .

(M. Alshaer, M. Jondi, M. Samamra, 2018) [7] used the Arduino Mega is a microcontroller board based on the ATmega1280 in their project. It contains 54 digital input/output pins (14 of which can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. As indicated inThe control algorithm is coded in C. PROTEUS simulation program is used to create a control circuit.

The authors (S. Hsajjaj, M. Bin Mahmood, 2018) [8] used the PLC controller was programmed using the Integrated Development Environment (IDE).

The authors (N. Mane, A. Mahajan, J. Jedhe, P. Nawale, 2019) [9] used the IC AT89S51 controller is a low-power, high-performance CMOS 8- bit microcontroller and has 4K bytes.

The authors (R. Alnakar and D. Catovic, 2021) [10] in their project an Arduino UNO board was chosen as a controller for the RPS. The code uploaded to the Arduino board is written in C, and the 8-bit microcontroller used in Arduino boards is known as the ATMEGA328P.

Chapter4

System Implementation and Testing

In this chapter we will discuss how to control the parts of the mechanical system using the FPGA board that programmable by Verilog code.

4.1 Software Part

And the software part will be divided into several parts, which are as follows:

4.1.1 Cars counter

The cars counter code uses to count cars in the system and compute the empty spaces in rotary car parking system.

We will use infrared sensor (I_R_S) to sense whether there is a car inside the room. If the sensor senses a car entering the room the cars counter is increased by one and if it does not sense a car entering the room the cars counter remains the same if the sensor state change from one to zero, then this is the case the case is one subtraction from the cars counter and this means that the car has left the system.

And in the event that the car does not exit and the sensor is working then no one visit the car meter because it has already been counted.

And to calculate the number of empty places inside the rotary car parking we subtract the cars counter from the number of rooms inside the system in this project it is sixteen rooms, and thus we get the number of empty places that will be displayed to the user. following flowchart shows work of code:

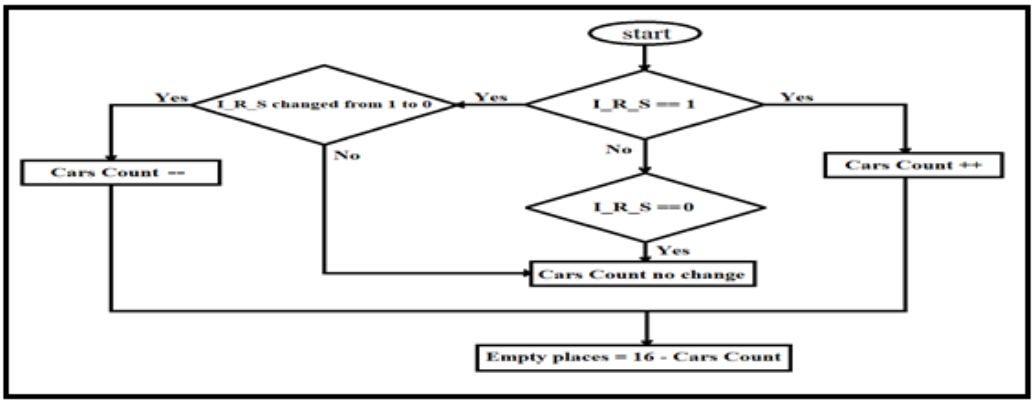

Figure 4.7:Flowchart for cars counter code.

4.1.2 Car lift motor

This code is used when the car entered in rotary car barking system to lift the car from the receiving room to the storage place, and the direction of movement is changed to the right and left, respectively, to distribute the loads within the system.

The benefits of this program prevents the loading of cars on one side of the mechanical system and this prolongs its life span and prevents the occurrence of stresses on the parts of the system.

We used infrared sensor named as receiving basket sensor (R_B_S) to sense if there are car in receiving basket following flowchart shows work of code:

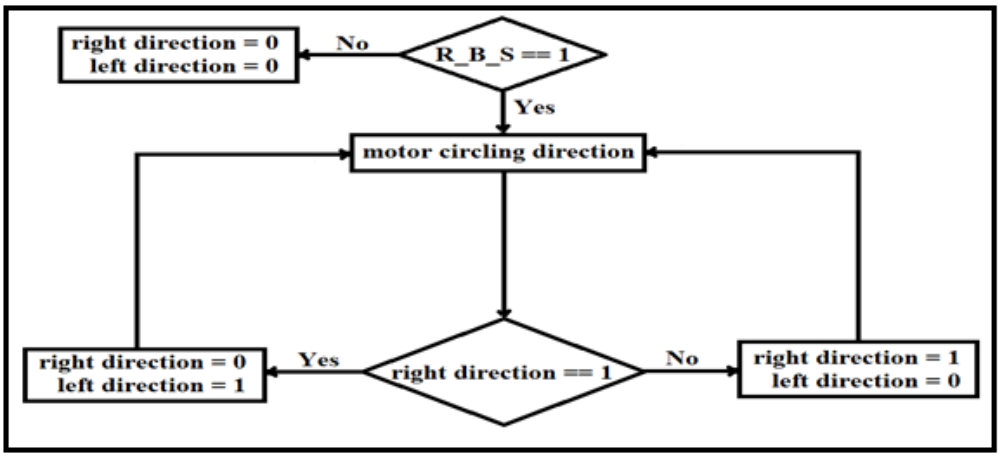

Figure 4.8: Flowchart for car lift motor code.

4.1.3 Car weight

This code function is to determine whether this car is allowed to enter the system or not, by calculating the weight of the car, and it is divided as follows: two-thirds on the front wheels and one-third on the rear wheels.

And to calculate the weight of the car we will use the strain gauge measurement-pressure sensor, and the benefits of this program is to reduce the stresses that lead to the collapse of the mechanical system.

following flowchart shows work of Car weigh code:

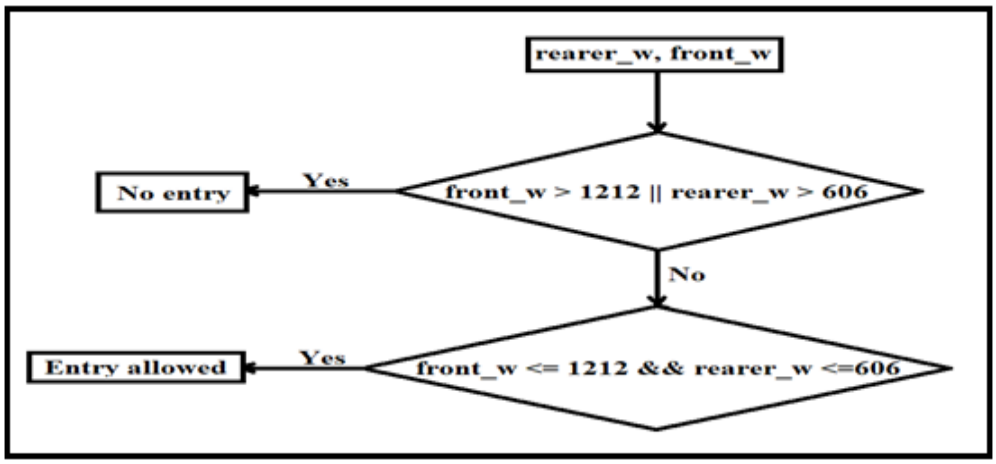

Figure 4.9: Flowchart for Car weight code.

4.1.4 Fire Extinguishing

The function of this code is to extinguish the fire if one of the rooms in rotary car parking system catches fire.

This is done by turning on the pump switch of the room in which the fire caught fire, when a sense higher than expected temperature using the temperature sensor installed inside the room the controller gives a signal to the pump switch to operate the fire extinguishing system and alarm bell and remain the fire extinguishing system and the bell in the operating state until the sensor measures the temperature and finds it in the normal position.

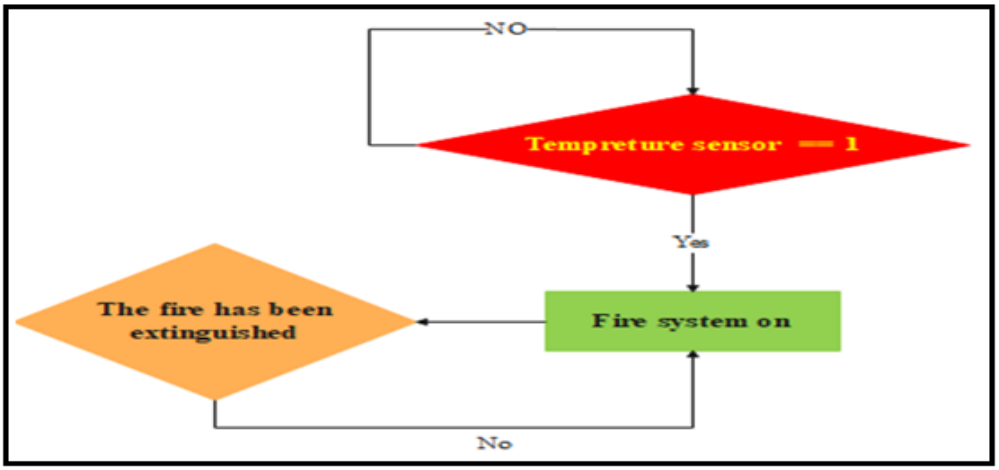

Figure 4.10: Flowchart for Fire Extinguishing code.\

4.1.5 **Shortest Path**

The function of this program is to determine the direction of rotation of the motor responsible for the movement of the system, either left or right, and that depends on the place where the car is stored.

If the parking place between 1 and 8, the motor will turn to the left and if the car is parked between 9 and 15, the motor will turn to the right. The location of the car is known by including sensors inside the system at the location shown in Fig. (4.5).

This program starts working when the user requests that his car leave from rotary car parking system and upon receiving the request the controller locates the car inside the system by means of the sensor and then determines the direction of rotation of the engine either left or right and then it starts rotating until it reaches the receiving room.

For example, if the car is located in room 3 then the engine is turning to the left. following flowchart shows work of Shortest Path code:

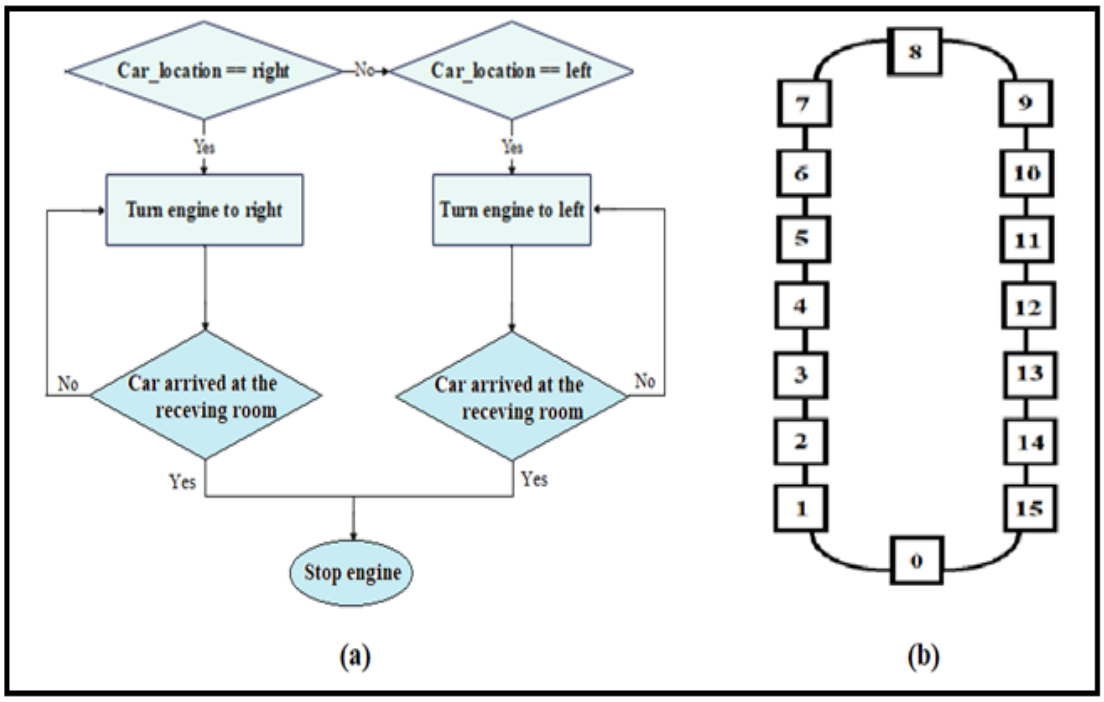

Figure 4.11: (a) Flowchart for Shortest Path code. (b) Basket location in rotary car parking system.

4.1.6 **Time count**

The function of this program is to calculate the time spent by the car inside the system, and upon receiving an exit request for this car, it is confirmed that the password for this room is entered correctly, and then the cost of staying inside the system is calculated. This is done by multiplying the value for each hour, which is specified as a parameter in the program facilitates the process of change it.

If a request to leave the car is received before the complete of the hour, the timer is looked at, and if a quarter of an hour is exceeded it is calculated as hour.

If the room code entered by the user is incorrect the counter completes the count and the exit request is rejected.

following flowchart shows work of Time count code:

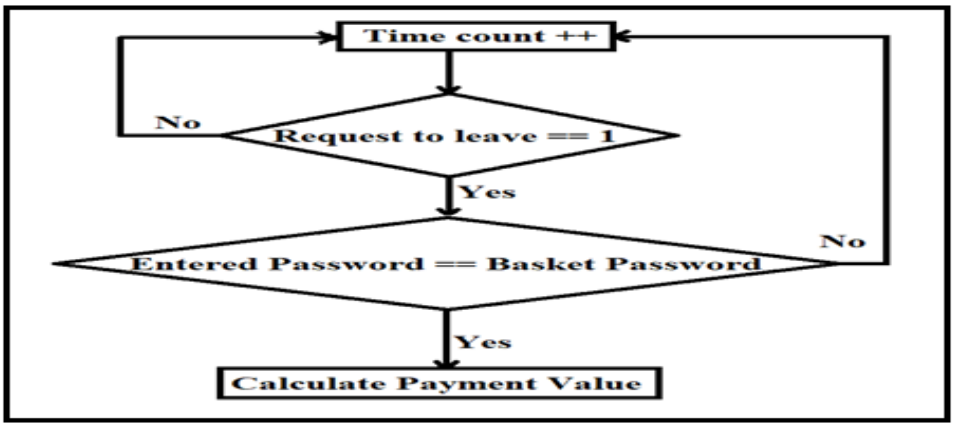

Figure 4.12: Flowchart for time count code.

4.1.7 **Rotary Car Parking**

This program is responsible for collecting all the previous parts to represent the general program and in this program each of the aforementioned parts is called to perform its function and it is also responsible for displaying the results of all previous parts of code as if it were one program and it is the one on which the test is performed and loaded with each the previous parts on the FPGA chip become the controller of the rotary car parking system.

The following flowchart shows the entire program's work, and it is a collection of all the aforementioned parts in one part:

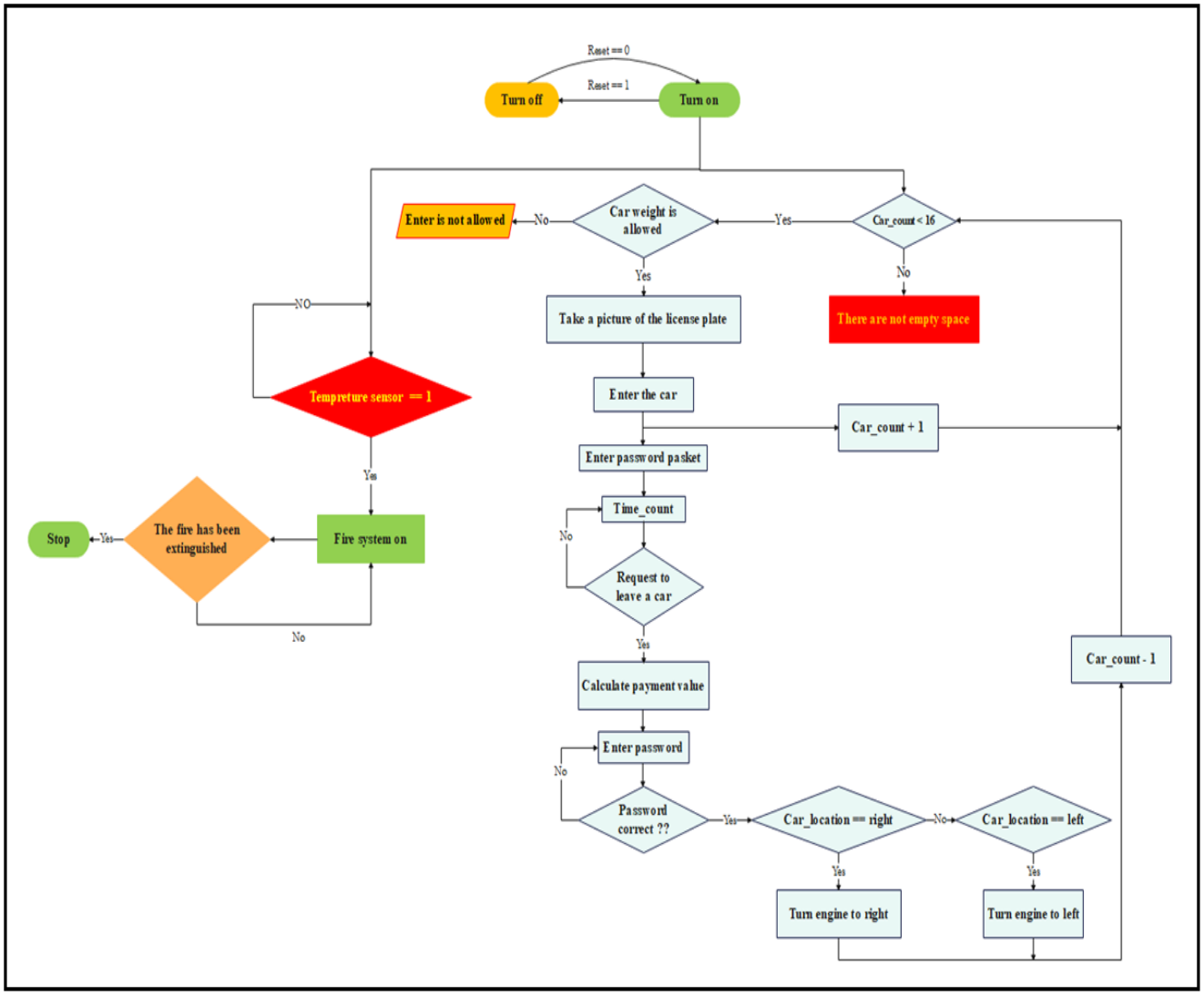

Figure 4.13:flowchart top module.

4.2 **Hardwar Part**

In this part we are going to download the piece of software onto FPGA chip by *Quartus II 12.1 Web Edition (64-Bit*)

In the software part, it has been designed as containing sixteen rooms, but due to the weakness of the capabilities of our FPGA board in terms of the number of its input, so

that you can only try two rooms only, part of the total program has been downloaded as shown in Fig. (4.8)

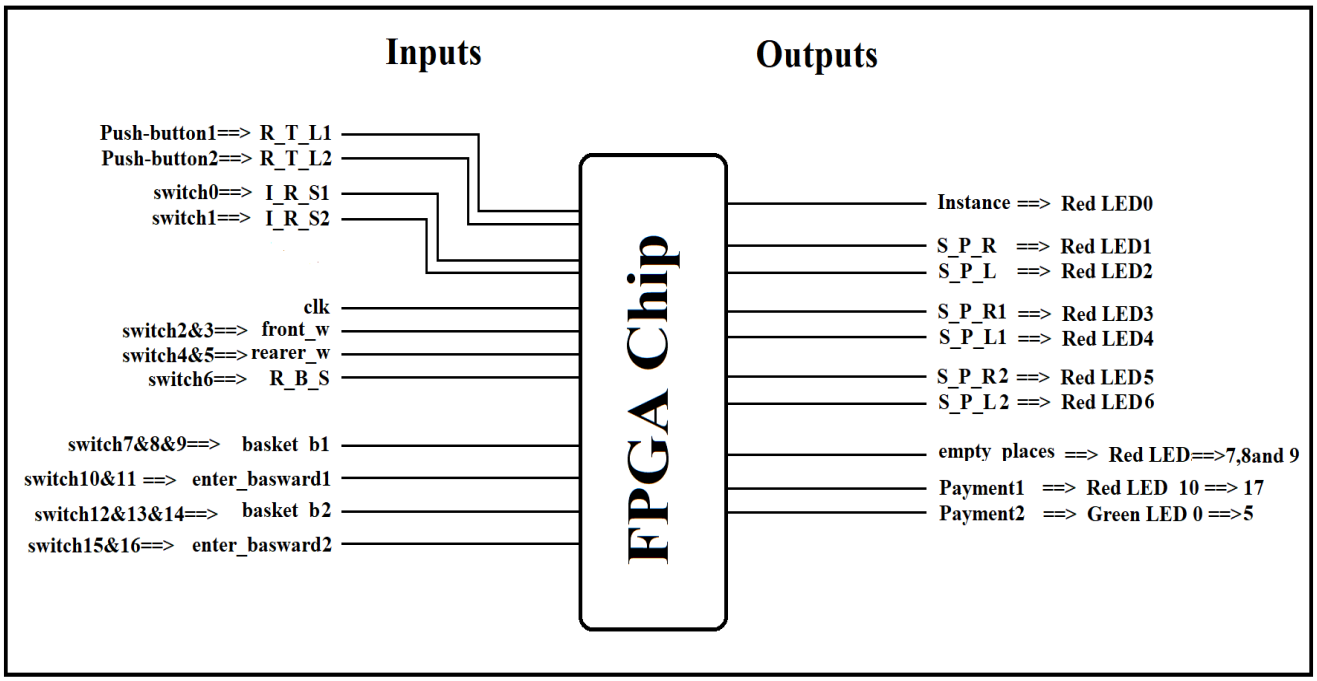

Figure 4.14:Block diagram of connections on FPGA chip.

4.3 **Testing & results**

First part of testing was by "ModelSim Advanced simulation and debugging" to test the design before downloading it to an FPGA chip and it worked successfully as shown in Figures below.

As shown in Fig. (4.9) show result of cars count code. And in it we see that when any of the sixteen infrared sensor (I_R_S) change from one to zero the empty places counter decreases by one, and this indicates that a car entered to rotary car parking system and reserved one of the sixteen rooms.

And we also see when the state of I_R_S changes from zero to one the counter of empty places increases by one and this indicates that the car has exited from rotary car parking system and an empty room is available.

As shown in Fig. (4.10) show result of Fire Extinguishing code and in it we see that when the measured room temperature reaches 50C or higher than it by temperature sensor both the alarm bell and water pump for this room operate and this indicates the presence of a fire inside the room and it remains in this state until the temperature sensor senses a degree less than 50C and then both the water pump and alarm bell are turned off.

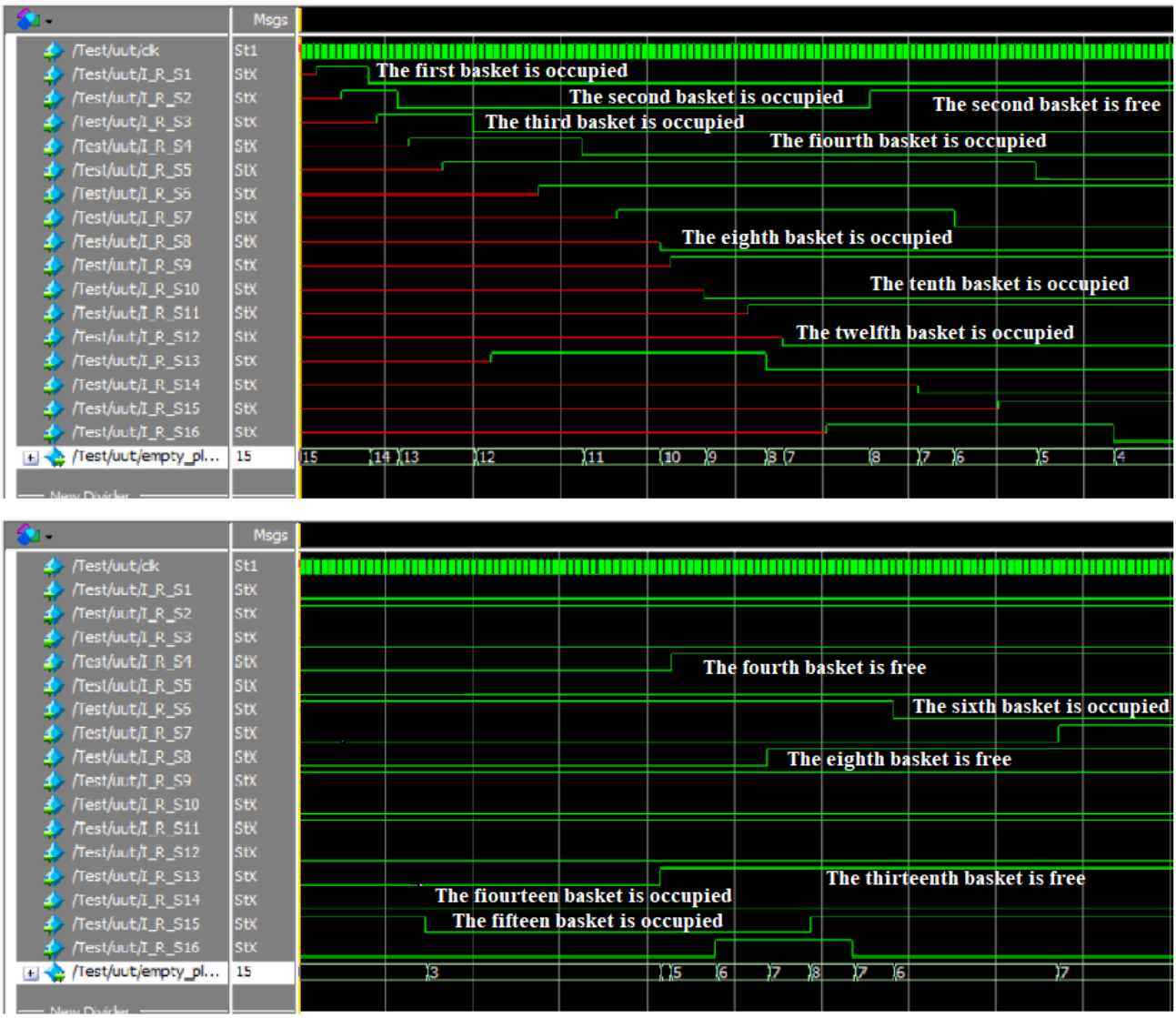

Figure 4.15:Car count test result on Modelsim simulation..

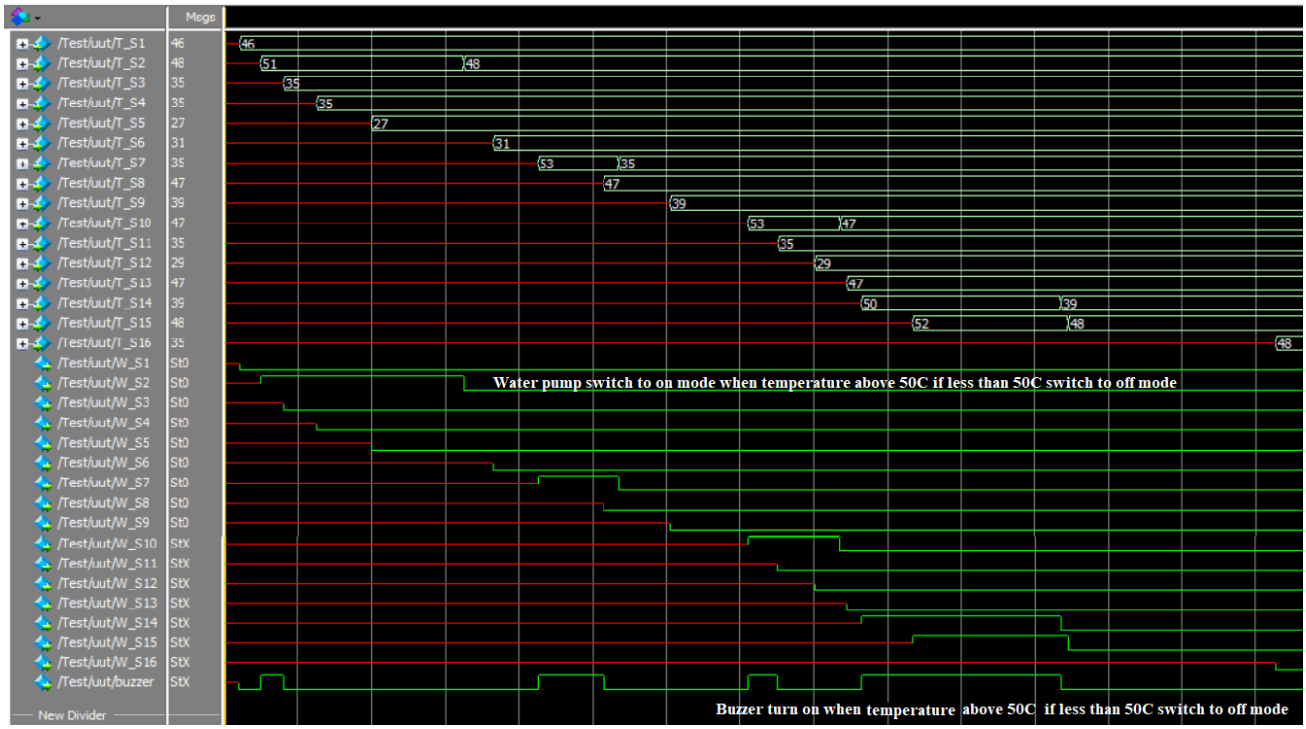

Figure 4.16:Fire Extinguishing result on Modelsim simulation.

As shown in Fig. (4.11) show result of Car weight code and it in see that the car is not accepted to enter the rotary car parking system when the weight measured on the front or rear wheels by strain gauge measurement-pressure sensor is higher than the weight specified by the designer and in our case we made the permissible weight on the front wheels 1212kg and the weight of rear wheels 606kg which is the average weight of sedan cars and in case it is smaller than these values it is accepted to enter in rotary car parking.

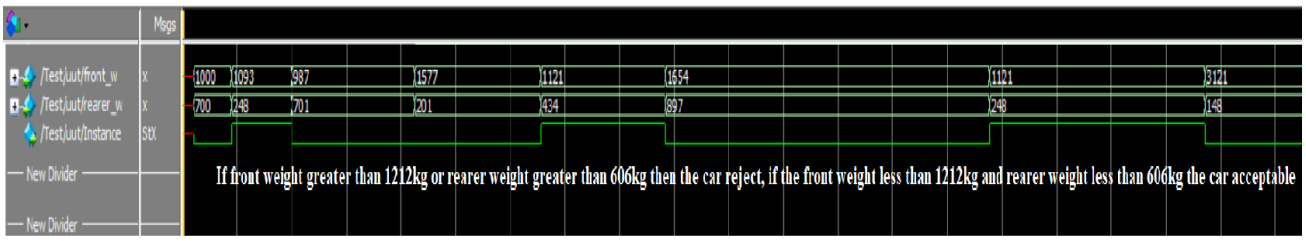

Figure 4.17:Car weight test result on Modelsim simulation.

As shown in Fig. (4.12) show result of Shortest path code and it in see that the engine rotates to the right when the location of room is between 9 and 15 and it rotates to the left if the location of room is between 1 and 8.this is done after verifying the validity of the password entered by the user if the code is incorrect the engine will not work as we can see in the case of room number two, it's location is 12 and a request to leave has been received and the engine is supposed to start on the right side (S_P_R). but the code was incorrect so the engine wouldn't work.

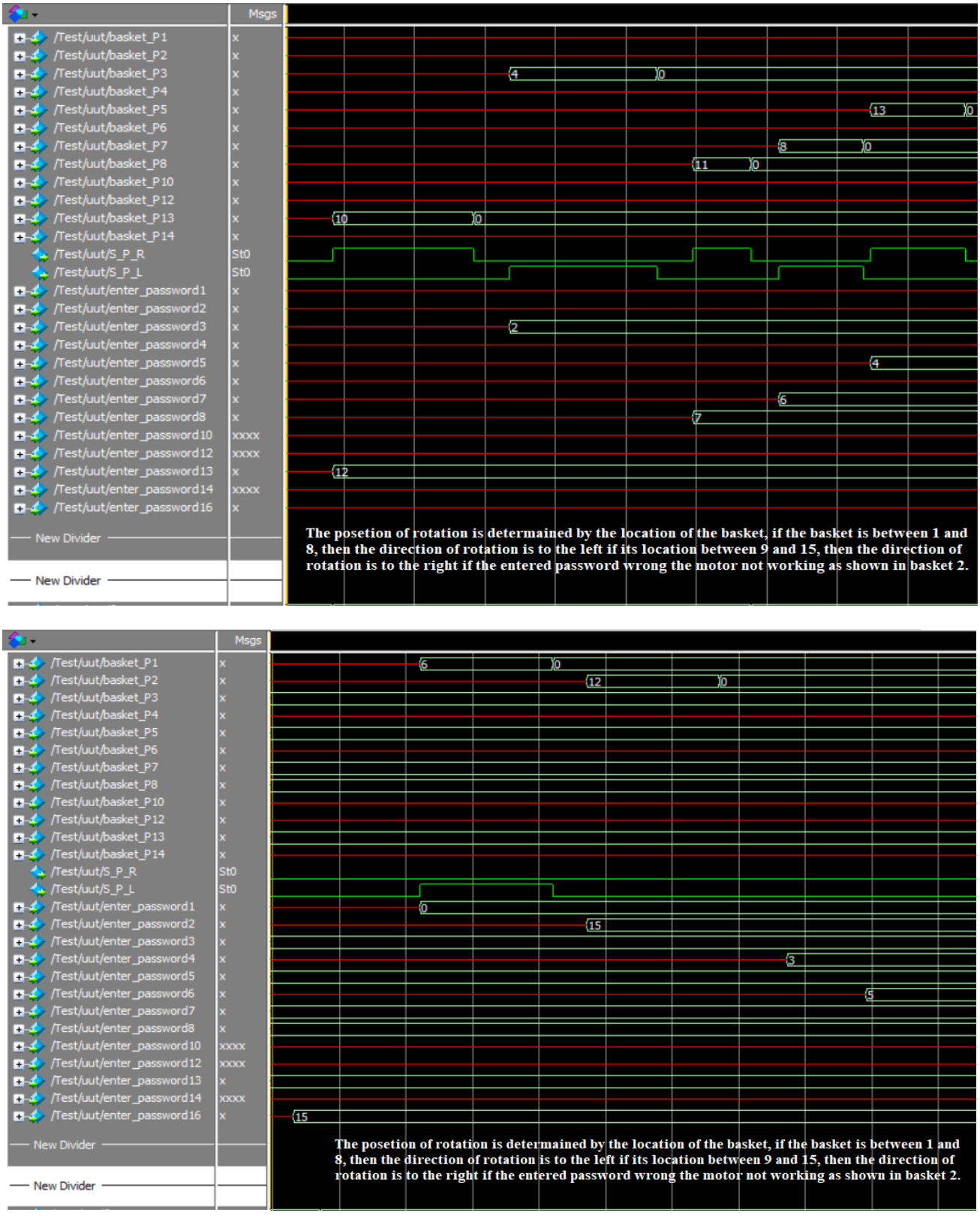

Figure 4.18:Shortest path test result on Modelsim simulation

As shown in Fig. (4.13) show result of Car lift motor and in it see that when a signal comes to enter the car in the receiving room (R_B_S) the engine starts to transfer it to the storage place and the direction of rotation of the engine is alternate between right and left in each entry case.

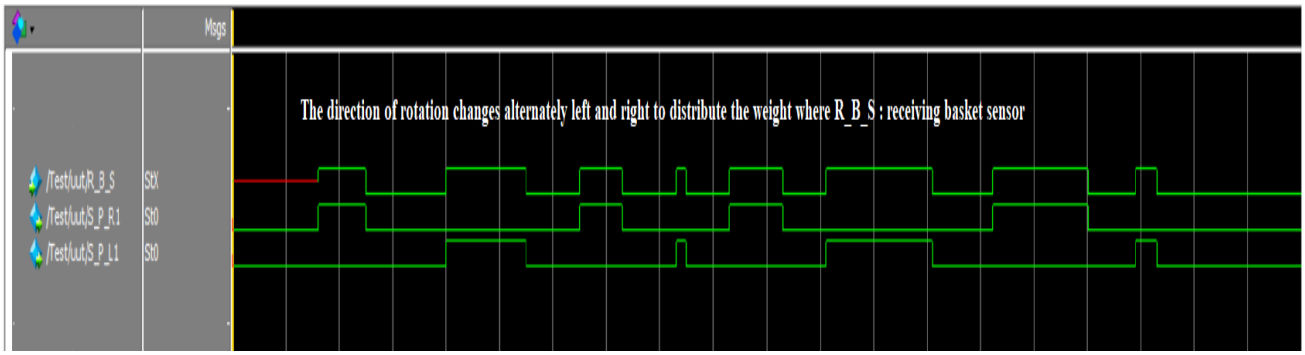

Figure 4.19:Car lift motor test result on Modelsim simulation.

As shown in Fig. (4.14) show result of Time count code and in it we see three cases which are room 1,2 and 3 and to calculate the time the seconds are counted until they reach 59 seconds so that the minutes counter increases by one and also the hours are increased by one to the hours counter when the minutes counter reaches 59 minutes , but to display the results more clearly we will make the minutes counter it increases by one when the seconds counter reaches 10 and we increase one to the hours counter when the minutes counter reaches 5 and when the seconds counter reaches 10 it is reset and the counting starts again as well as the minutes counter when it reaches 5 it is reset and the counting starts again when there is a request to leave the number of hours is multiplied by the value of staying and in our case it was 10 dinars.

The value paid is calculated after verifying the password entered by the user if it is incorrect the counting continues without calculating the payment value as happened in room 2 shown in

Fig. (4.14).

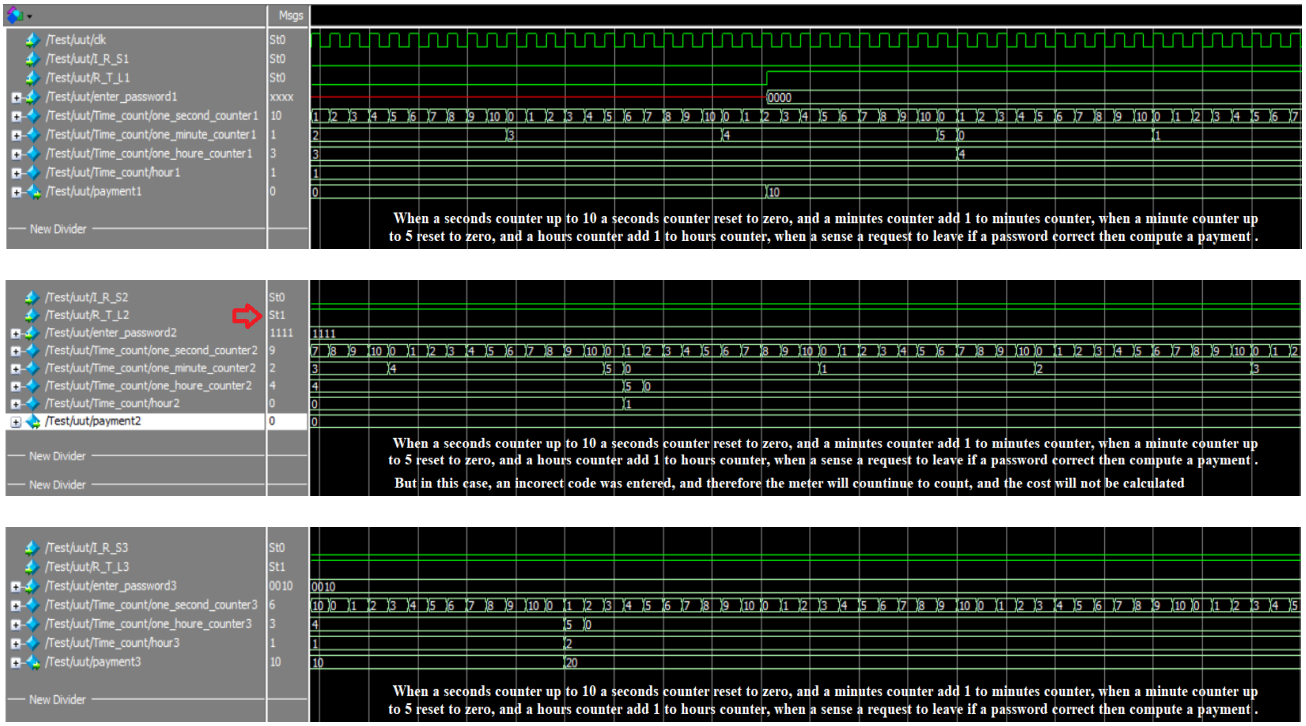

Figure 4.20:Time count test result on Modelsim simulation.

As shown in Fig. (4.15) show commands is displayed when the user entered wrong code for the room and when there is no vacant place inside the rotary car parking system or when an exit request is made from an empty room.

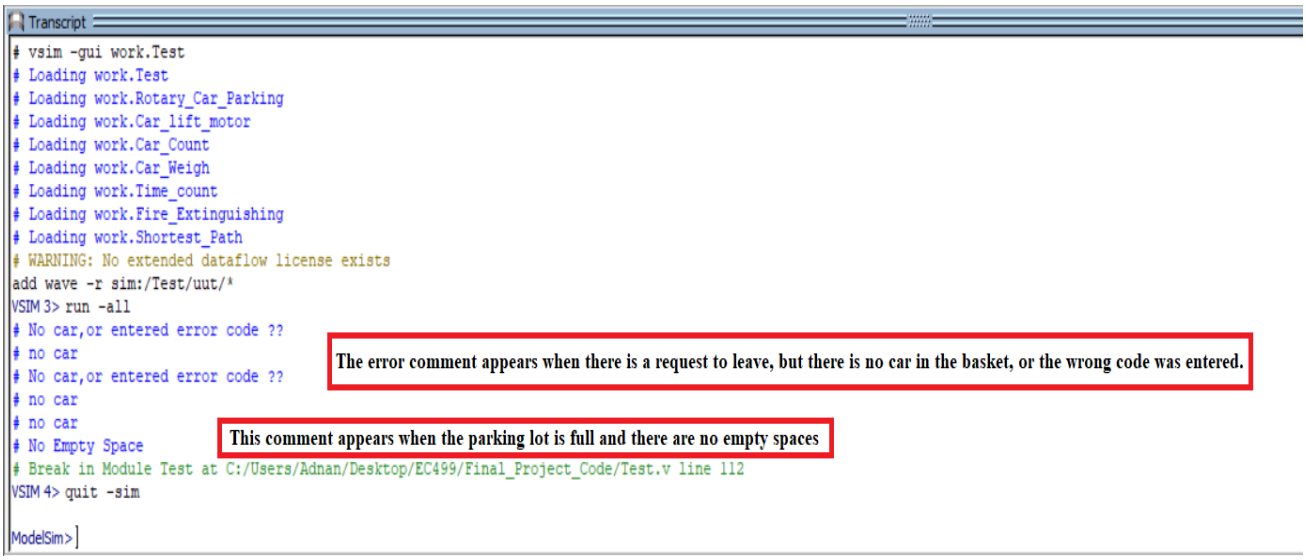

Figure 4.21:Modelsim output comment when an error occurs.

Second part of testing was by field programmable gate array (FPGA) in this case we test car count code, where a key 0 and 1 is an infrared sensor, and result of empty places out on LEDs 7,8 and 9 and, when test the weight of the car if it is allowed or not, where the keys 2 and 3 represent the front weight, the keys 4 and 5 represent the rear weight, and the LED0 represents the state if it is allowed or not, and when, experimenting with the shortest path program, where keys7, 8 and 9 represent the location of the basket from 1 to 7 and LED1 represent the direction of rotation of the motor towards the right, and LED2 represent the direction of rotation of the motor towards the left, and when the motor rotation tested when the car entered the receiving room, and key 6 represent the entry of the car, and the output appears on LED5.

In this case we test a time count code to out the payment of car stays in the rotary car parking system where a LEDs 10,11,12,13,14 and 15 to out payment and key 17 represent a request to leave button

In Fig. (4.20) we show the result (011110) in binary (30) in decimal that is mean the car stays in system 3 hour and the payment for one hour 10 coins

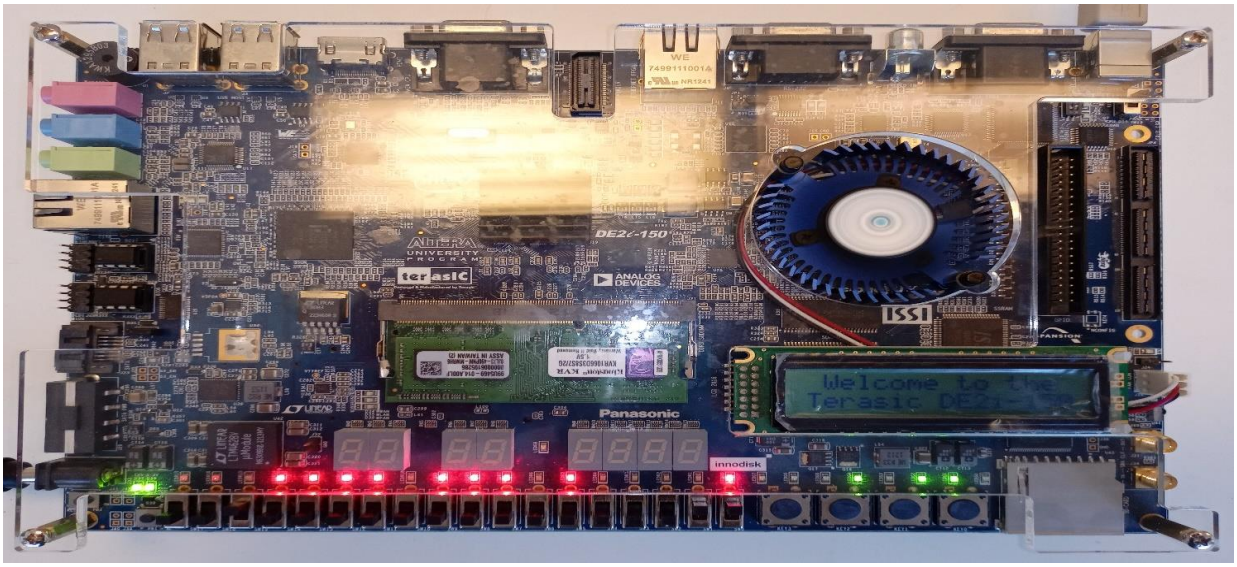

Figure 22: Time count test result on FPGA.

Chapter5

Conclusions and Future work

5.1 Conclusion

To conclude, designing and manufacturing of a rotary parking system is one of the responsibilities towards our country as these types of parking will help to solve the traffic jam issue in Libya. Making a rotary parking system prototype as a graduation project was very helpful issues that can help to use these positions more accurately and easier for the users.

Field Programmable Gate Array (FPGA) have been used to control of Rotary Car Parking System.

In the first part of this project, we designed an electronic system to control the mechanical parts programed by Verilog language, and it was tested by "ModelSim Advanced simulation and debugging" and it worked successfully and gave the required result.

In second part of this project, we wanted to test this design and download it on the FPGA chip, but due to the weak capabilities in terms of inputs, we could not download it on the chip, and for this we tested two rooms out of sixteen rooms, and they worked successfully.

5.2 Future Work

There are many ideas that can be added to this project to facilitate dealing with it and increase its efficiency and safety, and from these ideas:

1. A user-friendly display could be added to the system, to facilitate interaction between the system and the user.
2. Use a solar-powered energy this eco-friendly solution can reduce energy costs and carbon footprint.
3. Include a bank card payment system.
4. Include a face recognition system so that the system will down a car to receiving place without the need for a secret code.

References

[1] A.Saleh, A.Kamoure, "Structure and Mechanical Design of a Smart Car Park ", B.Sc. Project, Tripoli University, 2022.

[2] McDonald S. Shannon, “Cars Parking and Sustainability”, The Transportation Research Forum, 2012.

[3] Beebe Richard, “Automated Parking”, Status in the United States, 2001.

[4] Qurius, Vertical parking (1920-1976). https://www.q-park.com/blogs/newsitem/11761/vertical-parking1920-1976

[5] Wendy. "the car parking machine ", 2015. From the webpage_ https://www.qpark.com/blogs/newsitem/11761/verticalparking1920-1976

[6] S. Suliman, Y. Ali, " Implementation and Simulation of Rotary Automated Car Parking System", M.Sc. thesis, Sudan University of Science and Technology, 2016.

[7] M. Alshaer, M. Jondi, M. Samamra, " Rotary Parking System", B.Sc. Project, Palestine Polytechnic University, 2018.

[8] S. Hsajjaj, M. Bin Mahmood, "Design and Implementation of a Rotary Parking System for a Truly Smart City ", 2018, DOI: 10.1109/ICCSCE.2018.8685011.

[9] N. Mane, A. Mahajan, J. Jedhe, P. Nawale, " Design and Manufacturing of Rotary Automatic Parking ", Journal of Emerging Technologies and Innovative Research (JETIR), April 2019, Volume 6, Issue 4, PP:39-41.

[10] R. Alnakar and D. Catovic, " Rotary Parking System", B.Sc. Project, Kthroyal Institute of Technology, Stockholm Sweden 2021.

[11] FPGA System User Manual, "DE2i-150" , 2003-2013, Terasic Tecnologies Inc.

Appendices A

Field Programmable Gate Array (FPGA)

A photograph of the DE2i-150 board is shown in Fig. (1.6) the layout of the board and indicates the location of the connectors and key components [10].

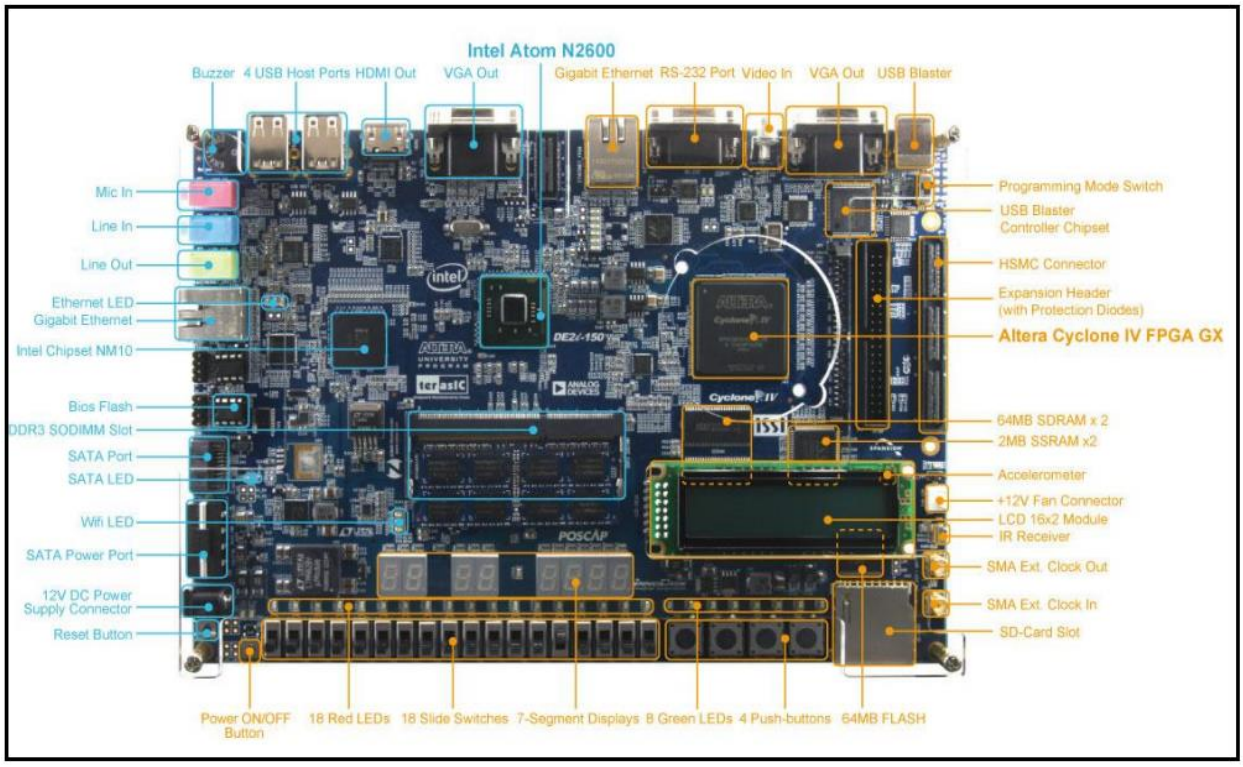

Figure 6.23 : The DE2i-150 board.

There are 27 user-controllable LEDs on the DE2i-150 board. Eighteen red LEDs are situated above the 18 Slide switches, and eight green LEDs are found above the push-button switches (the 9th green LED is in the middle of the 7-segment displays).

But in this project we use LEDs, switches and push-button switches following tables shows a Pin Assignments for Slide Switches, Push-buttons, LEDs.

Table 1 : Pin Assignments for Slide Switches.

Signal Name	FPGA Pin No.	Description	I/O Standard
SW[0]	PIN_V28	Slide Switch[0]	2.5V
SW[1]	PIN_U30	Slide Switch[1]	2.5V
SW[2]	PIN_V21	Slide Switch[2]	2.5V
SW[3]	PIN_C2	Slide Switch[3]	2.5V
SW[4]	PIN_AB30	Slide Switch[4]	2.5V
SW[5]	PIN_U21	Slide Switch[5]	2.5V
SW[6]	PIN_T28	Slide Switch[6]	2.5V
SW[7]	PIN_R30	Slide Switch[7]	2.5V
SW[8]	PIN_P30	Slide Switch[8]	2.5V
SW[9]	PIN_R29	Slide Switch[9]	2.5V
SW[10]	PIN_R26	Slide Switch[10]	2.5V
SW[11]	PIN_N26	Slide Switch[11]	2.5V
SW[12]	PIN_M26	Slide Switch[12]	2.5V
SW[13]	PIN_N25	Slide Switch[13]	2.5V
SW[14]	PIN_J26	Slide Switch[14]	2.5V
SW[15]	PIN_K25	Slide Switch[15]	2.5V
SW[16]	PIN_C30	Slide Switch[16]	2.5V
SW[17]	PIN_H25	Slide Switch[17]	2.5V

Table 2 : Pin Assignments for Push-buttons.

Signal Name	FPGA Pin No.	Description	I/O Standard
KEY[0]	PIN_AA26	Push-button[0]	2.5V
KEY[1]	PIN_AE25	Push-button[1]	2.5V
KEY[2]	PIN_AF30	Push-button[2]	2.5V
KEY[3]	PIN_AE26	Push-button[3]	2.5V

Table 3 : Pin Assignments for LEDs.

Signal Name	FPGA Pin No.	Description	I/O Standard
LEDR[0]	PIN_T23	LED Red[0]	2.5V
LEDR[1]	PIN_T24	LED Red[1]	2.5V
LEDR[2]	PIN_V27	LED Red[2]	2.5V
LEDR[3]	PIN_W25	LED Red[3]	2.5V
LEDR[4]	PIN_T21	LED Red[4]	2.5V
LEDR[5]	PIN_T26	LED Red[5]	2.5V
LEDR[6]	PIN_R25	LED Red[6]	2.5V
LEDR[7]	PIN_T27	LED Red[7]	2.5V
LEDR[8]	PIN_P25	LED Red[8]	2.5V
LEDR[9]	PIN_R24	LED Red[9]	2.5V
LEDR[10]	PIN_P21	LED Red[10]	2.5V
LEDR[11]	PIN_N24	LED Red[11]	2.5V
LEDR[12]	PIN_N21	LED Red[12]	2.5V
LEDR[13]	PIN_M25	LED Red[13]	2.5V
LEDR[14]	PIN_K24	LED Red[14]	2.5V
LEDR[15]	PIN_L25	LED Red[15]	2.5V

LEDR[16]	PIN_M21	LED Red[16]	2.5V
LEDR[17]	PIN_M22	LED Red[17]	2.5V
LEDG[0]	PIN_AA25	LED Green[0]	2.5V
LEDG[1]	PIN_AB25	LED Green[1]	2.5V
LEDG[2]	PIN_F27	LED Green[2]	2.5V
LEDG[3]	PIN_F26	LED Green[3]	2.5V
LEDG[4]	PIN_W26	LED Green[4]	2.5V
LEDG[5]	PIN_Y22	LED Green[5]	2.5V
LEDG[6]	PIN_Y25	LED Green[6]	2.5V
LEDG[7]	PIN_AA22	LED Green[7]	2.5V
LEDG[8]	PIN_J25	LED Green[8]	2.5V

And we use a "*Model Sim Advanced simulation and debugging Starter Edition 10.1b*" to write code.

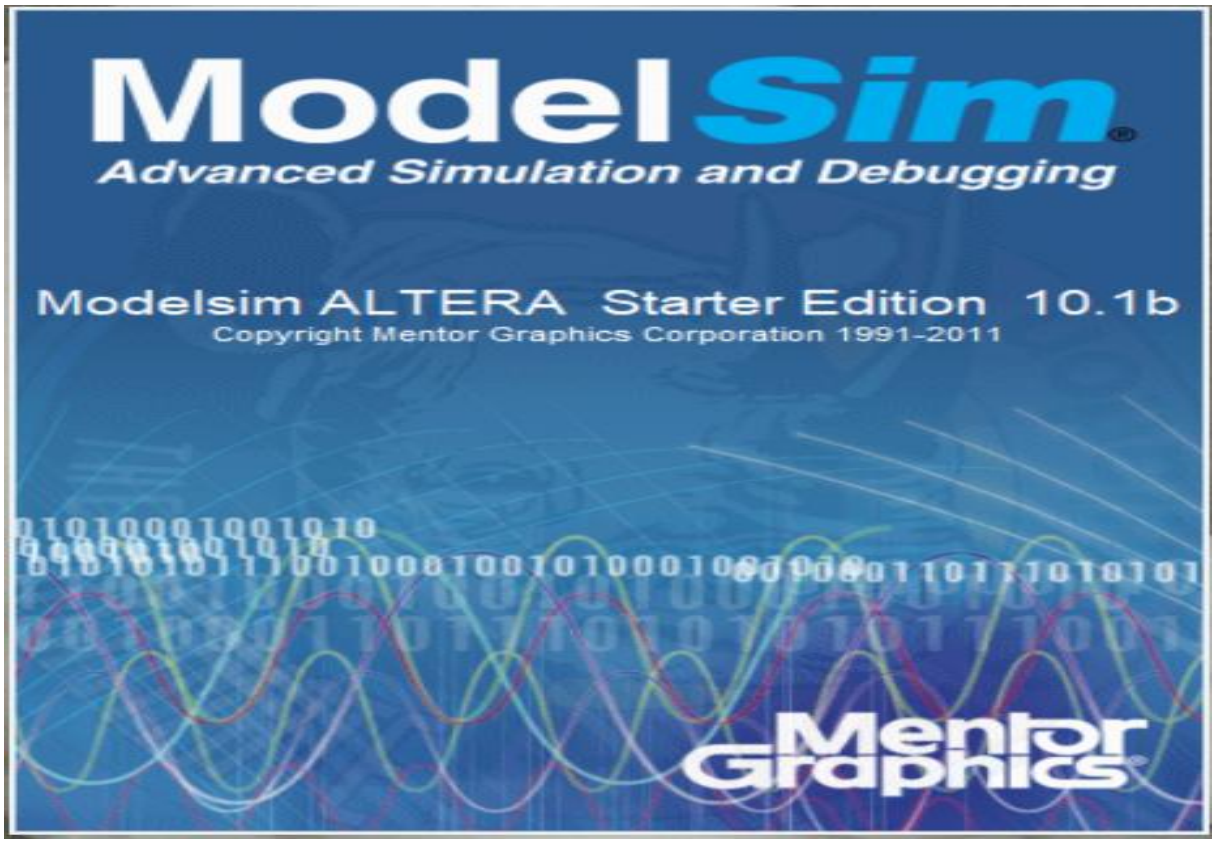

Figure 6.2: ModelSim Advanced simulation and debugging Starter Edition 10.1b.

we are going to download the piece of software onto FPGA chip by *Quartus II 12.1 Web Edition (64-Bit*) .

Figure 6.3: Quartus II 12.1 window.

Appendices B

Top model of project (Rotary_Car_Parking)

module Rotary_Car_Parking(clk,empty_places,R_B_S,I_R_S1,I_R_S2,I_R_S3,I_R_S4,I_R_S5,

I_R_S6,I_R_S7,I_R_S8,I_R_S9,I_R_S10,I_R_S11,I_R_S12,I_R_S13,I_R_S14,I_R_S15,I_R_S16,front_w,rearer_w,Instance,R_T_L1,R_T_L2,R_T_L3,R_T_L4,R_T_L5,R_T_L6,R_T_L7,R_T_L8,

R_T_L9,R_T_L10,R_T_L11,R_T_L12,R_T_L13,R_T_L14,R_T_L15,R_T_L16,enter_password1,enter_password2,enter_password3,enter_password4,enter_password5,enter_password6,enter_password7,enter_password8,enter_password9,enter_password10,enter_password11,enter_password12,enter_password13,enter_password14,enter_password15,enter_password16,payment1,payment2,payment3,payment4,payment5,payment6,payment7,payment8,payment9,payment10,payment11,payment12,payment13,payment14,payment15,payment16,T_S1,T_S2,T_S3,T_S4,T_S5,T_S6,T_S7,T_S8,T_S9,T_S10,T_S11,T_S12,T_S13,T_S14,T_S15,T_S16,W_S1,W_S2,W_S3,W_S4,W_S5,W_S6,W_S7,W_S8,W_S9,W_S10,W_S11,W_S12,W_S13,W_S14,W_S15,W_S16,buzzer,S_P_R1,S_P_L1,S_P_R,S_P_L,basket_P1,basket_P2, basket_P3, basket_P4,basket_P5, basket_P6,basket_P7, basket_P8,basket_P9, basket_P10, basket_P11, basket_P12,basket_P13, basket_P14, basket_P15);

input clk;

input I_R_S1,I_R_S2,I_R_S3,I_R_S4,I_R_S5,I_R_S6,I_R_S7,I_R_S8,I_R_S9,I_R_S10,I_R_S11,I_R_S12,I_R_S13,

I_R_S14,I_R_S15,I_R_S16;//Infrared sensor.

Input R_T_L1,R_T_L2,R_T_L3,R_T_L4,R_T_L5,R_T_L6,R_T_L7,R_T_L8,R_T_L9,R_T_L10,R_T_L11,R_T_L12,R_T_L13,R_T_L14,R_T_L15,R_T_L16;// R_T_L = Recuest to leave.

input [6:0]T_S1,T_S2,T_S3,T_S4,T_S5,T_S6,T_S7,T_S8,T_S9,T_S10,T_S11,T_S12,T_S13,

T_S14,T_S15,T_S16;// Temperature sensor.

input[3:0] basket_P1, basket_P2, basket_P3, basket_P4,basket_P5, basket_P6, basket_P7, basket_P8,basket_P9, basket_P10, basket_P11, basket_P12,basket_P13, basket_P14, basket_P15; //basket position

input[3:0]enter_password1,enter_password2,enter_password3,enter_password4,enter_password5,

enter_password6,enter_password7,enter_password8,enter_password9,enter_password10,enter_password11,enter_password12,enter_password13,enter_password14,enter_password15,enter_password16; //password entered by user to check.

input [12:0] front_w,rearer_w;

input R_B_S; //receiving basket sensor.

output Instance;

output W_S1,W_S2,W_S3,W_S4,W_S5,W_S6,W_S7,W_S8,W_S9,W_S10,W_S11,W_S12,

W_S13,W_S14,W_S15,W_S16,buzzer; // Water switch.

output [3:0]empty_places;

output [9:0] payment1,payment2,payment3,payment4,payment5,payment6,payment7,payment8,

payment9,payment10,payment11,payment12,payment13,payment14,payment15,payment16;

output S_P_R,S_P_R1,S_P_L1,S_P_L;//motor circling direction.

Car_lift_motor Car_lift_motor (.R_B_S(R_B_S),.S_P_R1(S_P_R1),.S_P_L1(S_P_L1));

Car_Count Car_Count(.clk(clk),.I_R_S1(I_R_S1),.I_R_S2(I_R_S2),.I_R_S3(I_R_S3),.

I_R_S4(I_R_S4),.I_R_S5(I_R_S5),.I_R_S6(I_R_S6),.I_R_S7(I_R_S7),.I_R_S8(I_R_S8),.I_R_S9(I_R_S9),.I_R_S10(I_R_S10),.I_R_S11(I_R_S11),.I_R_S12(I_R_S12),.I_R_S13(I_R_S13),.I_R_S14(I_R_S14),.I_R_S15(I_R_S15),.I_R_S16(I_R_S16),.empty_places(empty_places));

Car_Weigh Car_Weigh (.front_w(front_w),.rearer_w(rearer_w),.Instance(Instance));

Time_count Time_count (.clk(clk),.payment1(payment1),.payment2(payment2),.payment3

(payment3),.payment4(payment4),.payment5(payment5),.payment6(payment6),.payment7(payment7),.payment8(payment8),.payment9(payment9),.payment10(payment10),.payment11(payment11),.payment12(payment12),.payment13(payment13),.payment14(payment14),.payment15(payment15),.payment16(payment16),.R_T_L1(R_T_L1),.R_T_L2(R_T_L2),.R_T_L3(R_T_L3),.R_T_L4(R_T_L4),.R_T_L5(R_T_L5),.R_T_L6(R_T_L6),.R_T_L7(R_T_L7),.R_T_L8(R_T_L8),.R_T_L9(R_T_L9),.R_T_L10(R_T_L10),.R_T_L11(R_T_L11),.R_T_L12(R_T_L12),.R_T_L13(R_T_L13),.R_T_L14(R_T_L14),.R_T_L15(R_T_L15),.R_T_L16(R_T_L16),.I_R_S1(I_R_S1),.I_R_S2(I_R_S2),.I_R_S3(I_R_S3),.I_R_S4(I_R_S4),.I_R_S5(I_R_S5),.I_R_S6(I_R_S6),.I_R_S7(I_R_S7),.I_R_S8(I_R_S8),.I_R_S9(I_R_S9),.I_R_S10(I_R_S10),.I_R_S11(I_R_S11),.I_R_S12(I_R_S12),.I_R_S13(I_R_S13),.I_R_S14(I_R_S14),.I_R_S15(I_R_S15),.I_R_S16(I_R_S16),.enter_password1(enter_password1),.enter_password2(enter_password2),.enter_password3(enter_password3),.enter_password4(enter_password4),.enter_password5(enter_password5),.enter_password6(enter_password6),.enter_password7(enter_password7),.enter_password8(enter_password8),.enter_password9(enter_password9),.

enter_password10(enter_password10),.enter_password11(enter_password11),.enter_password12(enter_password12),.enter_password13(enter_password13),.enter_password14(enter_password14),.enter_password15(enter_password15),.enter_password16(enter_password16));

Fire_Extinguishing Fire_Extinguishing(.T_S1(T_S1),.T_S2(T_S2),.T_S3(T_S3),.T_S4(T_S4),.

T_S5(T_S5),.T_S6(T_S6),.T_S7(T_S7),.T_S8(T_S8),.T_S9(T_S9),.T_S10(T_S10),.T_S11(T_S11),.T_S12(T_S12),.T_S13(T_S13),.T_S14(T_S14),.T_S15(T_S15),.T_S16(T_S16),.W_S1(W_S1),.W_

S2(W_S2),.W_S3(W_S3),.W_S4(W_S4),.W_S5(W_S5),.W_S6(W_S6),.W_S7(W_S7),.W_S8(W_S8),.W_S9(W_S9),.W_S10(W_S10),.W_S11(W_S11),.W_S12(W_S12),.W_S13(W_S13),.

W_S14(W_S14),.W_S15(W_S15),.W_S16(W_S16),.buzzer(buzzer));

Shortest_Path Shortest_Path (.clk(clk),.S_P_R(S_P_R),.S_P_L(S_P_L),.R_T_L1(R_T_L1),.

R_T_L2(R_T_L2),.R_T_L3(R_T_L3),.R_T_L4(R_T_L4),.R_T_L5(R_T_L5),.R_T_L6(R_T_L6),.R_T_L7(R_T_L7),.R_T_L8(R_T_L8),.R_T_L9(R_T_L9),.R_T_L10(R_T_L10),.R_T_L11(R_T_L11),.R_T_L12(R_T_L12),.R_T_L13(R_T_L13),.R_T_L14(R_T_L14),.R_T_L15(R_T_L15) ,.I_R_S1(I_R_S1),.I_R_S2(I_R_S2),.I_R_S3(I_R_S3),.I_R_S4(I_R_S4).I_R_S5(I_R_S5),.I_R_S6(I_R_S6),.I_R_S7(I_R_S7),.I_R_S8(I_R_S8),.I_R_S9(I_R_S9),.I_R_S10(I_R_S10),.I_R_S11(I_R_S11),.I_R_S12(I_R_S12),.I_R_S13(I_R_S13),.I_R_S14(I_R_S14),.I_R_S15(I_R_S15)

,.basket_P1(basket_P1),.basket_P2(basket_P2),.basket_P3(basket_P3),.basket_P4(basket_P4),.

basket_P5(basket_P5),.basket_P6(basket_P6),.basket_P7(basket_P7),.basket_P8(basket_P8),.basket_P9(basket_P9),.basket_P10(basket_P10),.basket_P11(basket_P11),.basket_P12(basket_P12),.basket_P13(basket_P13),.basket_P14(basket_P14),.basket_P15(basket_P15),.enter_password1(enter_password1),.enter_password2(enter_password2),.enter_password3(enter_password3),.enter_password4(enter_password4),.enter_password5(enter_password5),.enter_password6(enter_password6),.enter_password7(enter_password7),.enter_password8(enter_password8),.enter_password9(enter_password9),.enter_password10(enter_password10),.enter_password11(enter_password11),.enter_password12(enter_password12),.enter_password13(enter_password13),.enter_password14(enter_password14),.enter_password15(enter_password15));

endmodule

Sub module to count number of cars in system (Car_Count)

/*This code to compute number of cars in the system when a sensor read (0) ther are car in pasket if read (1) the pasket empty*/

module Car_Count(clk,empty_places,I_R_S1,I_R_S2,I_R_S3,I_R_S4,I_R_S5,I_R_S6,I_R_S7,

I_R_S8,I_R_S9,I_R_S10,I_R_S11,I_R_S12,I_R_S13,I_R_S14,I_R_S15,I_R_S16);

input clk;

input I_R_S1,I_R_S2,I_R_S3,I_R_S4,I_R_S5,I_R_S6,I_R_S7,I_R_S8,

I_R_S9,I_R_S10,I_R_S11,I_R_S12,I_R_S13,I_R_S14,I_R_S15,I_R_S16;

output reg[3:0] empty_places;

reg[3:0] Car_count;

```
reg current_state1,current_state2,current_state3,current_state4,current_state5,current_state6,
current_state7,current_state8,current_state9,current_state10,current_state11,current_state12,current_
state13,current_state14,current_state15,current_state16;
initial Car_count=4'b0000;
initial current_state1  = 1;   initial current_state2  = 1;     initial current_state3  = 1;
initial current_state4  = 1;   initial current_state5  = 1;     initial current_state6  = 1;
initial current_state7  = 1;   initial current_state8  = 1;     initial current_state9  = 1;
initial current_state10 = 1;   initial current_state11 = 1;     initial current_state12 = 1;
initial current_state13 = 1;   initial current_state14 = 1;     initial current_state15 = 1;
initial current_state16 = 1;
always @ (posedge clk or Car_count)
        begin
        if (Car_count >= 4'b1110)
           begin
            $display ("No Empty Space");         Car_count<= 4'b0000;
           end   end
always@(posedge clk or I_R_S1)
       begin
                  if (I_R_S1 == 0 && current_state1 == 1)
                              begin
                               Car_count <= Car_count + 1;     current_state1 = I_R_S1;
                              end
                  else if (I_R_S1 == 1 && current_state1 == 0 )
                              begin
                               Car_count <= Car_count-1;      current_state1 = I_R_S1;
                              end
                      else  Car_count <= Car_count;
```

```
                empty_places <= 4'b1111- Car_count;
        end
    ////////////////////////////////////////////////////////////////////
always@(posedge clk or I_R_S2)
        begin
                if (I_R_S2 == 0 && current_state2 == 1)
                        begin
                         Car_count <= Car_count + 1;    current_state2 <= I_R_S2;
                        end
                else if (I_R_S2 == 1 && current_state2 == 0 )
                        begin
                         Car_count <= Car_count-1;   current_state2 <= I_R_S2;
                        end
                   else  Car_count <= Car_count;
                 empty_places <= 4'b1111- Car_count;

        end
    ////////////////////////////////////////////////////////////////////
always@(posedge clk or I_R_S3)
        begin
                if (I_R_S3 == 0 && current_state3 == 1)
                        begin
                         Car_count <= Car_count + 1;    current_state3 <= I_R_S3;
                        end
                else if (I_R_S3 == 1 && current_state3 == 0 )
                        begin
                         Car_count <= Car_count-1;    current_state3 <= I_R_S3;
```

```
                    end
                else  Car_count <= Car_count;
               empty_places <= 4'b1111- Car_count;
    end
  ////////////////////////////////////////////////////////////////
always@(posedge clk or I_R_S4)
    begin
            if (I_R_S4 == 0 && current_state4 == 1)
                    begin
                     Car_count <= Car_count + 1;      current_state4 <= I_R_S4;
                    end
            else if (I_R_S4 == 1 && current_state4 == 0 )
                    begin
                     Car_count <= Car_count-1;     current_state4 <= I_R_S4;
                    end
                else  Car_count <= Car_count;
               empty_places <= 4'b1111- Car_count;
    end
  ////////////////////////////////////////////////////////////////
always@(posedge clk or I_R_S5)
    begin
            if (I_R_S5 == 0 && current_state5 == 1)
                    begin
                     Car_count <= Car_count + 1;      current_state5 <= I_R_S5;
                    end
            else if (I_R_S5 == 1 && current_state5 == 0 )
                    begin
```

```
                    Car_count <= Car_count-1;     current_state5 <= I_R_S5;
                    end
                else  Car_count <= Car_count;
               empty_placs <= 4'b1111- Car_count
        end
        ////////////////////////////////////////////////////////////
always@(posedge clk or I_R_S6)
        begin
                if (I_R_S6 == 0 && current_state6 == 1)
                    begin
                     Car_count <= Car_count + 1;      current_state6 <= I_R_S6;
                    end
                else if (I_R_S6 == 1 && current_state6 == 0 )
                    begin
                     Car_count <= Car_count-1;         current_state6 <= I_R_S6;
                    end
                else  Car_count <= Car_count;
               empty_places <= 4'b1111- Car_count;
        end
       ////////////////////////////////////////////////////////////
always@(posedge clk or I_R_S7)
        begin
                if (I_R_S7 == 0 && current_state7 == 1)
                    begin
                     Car_count <= Car_count + 1;    current_state7 <= I_R_S7;
                    end
                else if (I_R_S7 == 1 && current_state7 == 0 )
```

```
                    begin
                     Car_count <= Car_count-1;       current_state7 <= I_R_S7;
                    end
                else  Car_count <= Car_count;
               empty_places <= 4'b1111- Car_count;
        end
          ////////////////////////////////////////////////////////
always@(posedge clk or I_R_S8)
        begin
                if (I_R_S8 == 0 && current_state8 == 1)
                    begin
                     Car_count <= Car_count + 1;       current_state8 <= I_R_S8;
                    end
                else if (I_R_S8 == 1 && current_state8 == 0 )
                    begin
                     Car_count <= Car_count-1;       current_state8 <= I_R_S8;
                    end
                else  Car_count <= Car_count;
               empty_places <= 4'b1111- Car_count;
        end
          ////////////////////////////////////////////////////////
always@(posedge clk or I_R_S9)
        begin
                if (I_R_S9 == 0 && current_state9 == 1)
                    begin
                     Car_count <= Car_count + 1;    current_state9 <= I_R_S9;
                    end
```

```
                else if (I_R_S9 == 1 && current_state9 == 0 )
                        begin
                         Car_count <= Car_count-1;    current_state9 <= I_R_S9;
                        end
                    else  Car_count <= Car_count;
                 empty_places <= 4'b1111- Car_count;
        end
          ////////////////////////////////////////////////////////
always@(posedge clk or I_R_S10)
        begin
                if (I_R_S10 == 0 && current_state10 == 1)
                        begin
                         Car_count <= Car_count + 1;     current_state10 <= I_R_S10;
                        end
                else if (I_R_S10 == 1 && current_state10 == 0 )
                        begin
                         Car_count <= Car_count-1;     current_state10 <= I_R_S10;
                        end
                    else  Car_count <= Car_count;
                 empty_places <= 4'b1111- Car_count;
        end
          ////////////////////////////////////////////////////////
always@(posedge clk or I_R_S11)
        begin
                if (I_R_S11 == 0 && current_state11 == 1)
                        begin
                         Car_count <= Car_count + 1;        current_state11 <= I_R_S11;
```

```
                end
            else if (I_R_S11 == 1 && current_state11 == 0 )
                begin
                 Car_count <= Car_count-1;       current_state11 <= I_R_S11;
                end
              else  Car_count <= Car_count;
             empty_places <= 4'b1111- Car_count;
        end
          ////////////////////////////////////////////////////////
always@(posedge clk or I_R_S12)
       begin
            if (I_R_S12 == 0 && current_state12 == 1)
                begin
                 Car_count <= Car_count + 1;     current_state12 <= I_R_S12;
                end
            else if (I_R_S12 == 1 && current_state12 == 0 )
                begin
                 Car_count <= Car_count-1;     current_state12 <= I_R_S12;
                end
              else  Car_count <= Car_count;
             empty_places <= 4'b1111- Car_count;

        end
          ////////////////////////////////////////////////////////
always@(posedge clk or I_R_S13)
       begin
            if (I_R_S13 == 0 && current_state13 == 1)
```

```
                    begin
                     Car_count <= Car_count + 1;   current_state13 <= I_R_S13;
                    end
                else if (I_R_S13 == 1 && current_state13 == 0 )
                    begin
                     Car_count <= Car_count-1;       current_state13 <= I_R_S13;
                    end
                  else  Car_count <= Car_count;
                 empty_places <= 4'b1111- Car_count;
        end
            ////////////////////////////////////////////////////////
always@(posedge clk or I_R_S14)
      begin
                if (I_R_S14 == 0 && current_state14 == 1)
                    begin
                     Car_count <= Car_count + 1;       current_state14 <= I_R_S14;
                    end
                else if (I_R_S14 == 1 && current_state14 == 0 )
                    begin
                     Car_count <= Car_count-1;       current_state14 <= I_R_S14;
                    end
                  else  Car_count <= Car_count;
                 empty_places <= 4'b1111- Car_count;
        end
            ////////////////////////////////////////////////////////
always@(posedge clk or I_R_S15)
      begin
```

```
                if (I_R_S15 == 0 && current_state15 == 1)
                        begin
                         Car_count <= Car_count + 1;     current_state15 <= I_R_S15;
                        end
                else if (I_R_S15 == 1 && current_state15 == 0 )
                        begin
                         Car_count <= Car_count-1;     current_state15 <= I_R_S15;
                        end
                   else  Car_count <= Car_count;
                 empty_places <= 4'b1111- Car_count;
         end
            ////////////////////////////////////////////////////////
always@(posedge clk or I_R_S16)
        begin
                if (I_R_S16 == 0 && current_state16 == 1)
                        begin
                         Car_count <= Car_count + 1;      current_state16 <= I_R_S16;
                        end
                else if (I_R_S16 == 1 && current_state16 == 0 )
                        begin
                         Car_count <= Car_count-1;       current_state16 <= I_R_S16;
                        end
                   else  Car_count <= Car_count;
                 empty_places <= 4'b1111- Car_count;
         end
endmodule
```

Sub module to lift car cars when it enters the room (Car_lift_motor)

```
/* this code to turn motor if there are car in receiving basket*/
module Car_lift_motor (R_B_S,S_P_R1,S_P_L1);
input R_B_S; //receiving basket sensor.

output reg S_P_R1,S_P_L1;//motor circling direction.
reg state_direction;
parameter [1:0] left=2'b00,right=2'b01;
initial S_P_R1=0 ;initial S_P_L1=0 ;initial state_direction = right;
always @(R_B_S)
      begin
      if ( R_B_S == 1 && state_direction == right)//if there are car in basket.
         begin
           S_P_R1 =1;       state_direction <= right;
         end
      else if (R_B_S == 0 && state_direction == right)
               begin
                S_P_R1 =0;   state_direction <= left;
               end
          else if ( R_B_S == 1 && state_direction == left)//if there are car in basket.
               begin
                S_P_L1=1;   state_direction <= left;
               end
               else if (R_B_S == 0 && state_direction == left)
                      begin
                       S_P_L1 =0;   state_direction <= right;
```

```
                    end        end
endmodule
```

Sub module to find out if the car is allowed to enter or not (Car_Weigh)

```
/*This code to compute a weigh of car if incepted or not*/
module Car_Weigh(front_w,rearer_w,Instance);
input [12:0]front_w,rearer_w;
output reg Instance;
parameter front_condition =1212;
parameter rearer_condition=606;
always @(front_w or rearer_w)
      begin
        if (front_w > front_condition || rearer_w > rearer_condition)
           begin
            Instance = 0;
           end
        else if (front_w < front_condition && rearer_w < rearer_condition)
               begin
               Instance = 1;
               end
      end
endmodule
```

Sub module to fire extinguishing (Fire_Extinguishing)

```
/*This program works as a fire extinguishing system, as it turn-on the pump
```

```
switch when the temperature sensor read temperature higher or equal to 50C*/

module
Fire_Extinguishing(T_S1,T_S2,T_S3,T_S4,T_S5,T_S6,T_S7,T_S8,T_S9,T_S10,T_S11,T_S12,T_S
13,T_S14,T_S15,T_S16,W_S1,W_S2,W_S3,W_S4,W_S5,W_S6,W_S7,W_S8,W_S9,W_S10,W_S1
1,W_S12,W_S13,W_S14,W_S15,W_S16,buzzer);

parameter temperature_rate =7'b0110010; //temperatur condition.


input [6:0]T_S1,T_S2,T_S3,T_S4,T_S5,T_S6,T_S7,T_S8,T_S9,T_S10,T_S11,T_S12,T_S13,

T_S14,T_S15,T_S16;// Temperature sensor.

output reg W_S1,W_S2,W_S3,W_S4,W_S5,W_S6,W_S7,W_S8,W_S9,W_S10,W_S11,W_S12,

W_S13,W_S14,W_S15,W_S16,buzzer; // Water switch.

always @(T_S1) begin
                if (T_S1 >= temperature_rate)
                   begin
                   W_S1= 1;     buzzer = 1;
                   end
                else
                   begin
                    W_S1= 0;  buzzer = 0;
                   end       end
always @(T_S2)
        begin
                if (T_S2 >= temperature_rate)
                   begin
                   W_S2= 1;      buzzer = 1;
                   end
                else
                   begin
```

```
                W_S2= 0;     buzzer = 0;
                end       end
always @(T_S3)
        begin
                if (T_S3 >= temperature_rate)
                   begin
                   W_S3= 1;       buzzer = 1;
                   end
                else
                   begin
                    W_S3= 0;    buzzer = 0;
                   end
         end
always @(T_S4)
        begin
                if (T_S4 >= temperature_rate)
                   begin
                   W_S4= 1;        buzzer = 1;
                   end
                else
                   begin
                    W_S4= 0;     buzzer = 0;
                   end       end
always @(T_S5)
        begin
                if (T_S5 >= temperature_rate)
                   begin
```

```
            W_S5= 1;      buzzer = 1;
            end
        else
            begin
             W_S5= 0;       buzzer = 0;
            end         end
always @(T_S6)
    begin
        if (T_S6 >= temperature_rate)
            begin
            W_S6= 1;       buzzer = 1;
            end
        else
            begin
             W_S6= 0;       buzzer = 0;
            end
      end
always @(T_S7)
    begin
        if (T_S7 >= temperature_rate)
            begin
            W_S7= 1;       buzzer = 1;
            end
        else
            begin
             W_S7= 0;        buzzer = 0;
            end         end
```

```
always @(T_S8)
        begin
                if (T_S8 >= temperature_rate)
                   begin
                   W_S8= 1;       buzzer = 1;
                   end
                else
                   begin
                    W_S8= 0;    buzzer = 0;
                   end         end
always @(T_S9)
        begin
                if (T_S9 >= temperature_rate)
                   begin
                   W_S9= 1;    buzzer = 1;
                   end
                else
                   begin
                    W_S9= 0;      buzzer = 0;
                   end      end
always @(T_S10)
        begin
                if (T_S10 >= temperature_rate)
                   begin
                   W_S10= 1;    buzzer = 1;
                   end
                else
```

```
                begin
                 W_S10= 0;     buzzer = 0;
                end
        end
always @(T_S11)
        begin
                if (T_S11 >= temperature_rate)
                begin
                W_S11= 1;    buzzer = 1;
                end
                else
                begin
                 W_S11= 0;     buzzer = 0;
                end       end
always @(T_S12)
        begin
                if (T_S12 >= temperature_rate)
                begin
                W_S12= 1;      buzzer = 1;
                end
                else
                begin
                 W_S12= 0;      buzzer = 0;
                end         end
always @(T_S13)
        begin
                if (T_S13 >= temperature_rate)
```

```
                begin
                W_S13= 1;    buzzer = 1;
                end
            else
                begin
                 W_S13= 0;     buzzer = 0;
                end
        end
always @(T_S14)
        begin
            if (T_S14 >= temperature_rate)
                begin
                W_S14= 1;   buzzer = 1;
                end
            else
                begin
                 W_S14= 0;      buzzer = 0;
                end       end
always @(T_S15)
        begin
            if (T_S15 >= temperature_rate)
                begin
                W_S15= 1;       buzzer = 1;
                end
            else
                begin
                 W_S15= 0;      buzzer = 0;
```

```
                end        end
always @(T_S16)
        begin
                if (T_S16 >= temperature_rate)
                   begin
                   W_S16= 1;       buzzer = 1;
                   end
                else
                   begin
                    W_S16= 0;      buzzer = 0;
                   end        end
endmodule
```

Sub module to compute shortest path to receive a car (Shortest_Path)

```
/* following code to compute shortest path to receive a car*/
/*حيكون فيه حساس في كل سلة يحدد مكان السلة والسلات مرقمات بالشكل التالي*/
                    8
                  7    9
                  6    10
                  5    11
                  4    12
                  3    13
                  2    14
                  1    15
                    0
*/

module Shortest_Path (clk,R_T_L1,R_T_L2,R_T_L3,R_T_L4,R_T_L5,R_T_L6,R_T_L7,R_T_L8,
```

```
R_T_L9,R_T_L10,R_T_L11,R_T_L12,R_T_L13,R_T_L14,R_T_L15,basket_P1, basket_P2 , basket_P3, basket_P4,basket_P5, basket_P6, basket_P7, basket_P8,basket_P9, basket_P10, basket_P11, basket_P12, basket_P13,basket_P14 basket_P15,enter_password1,enter_password2,enter_password3,enter_password4,enter_password5,enter_password6,enter_password7,enter_password8,enter_password9,enter_password10,enter_password11,enter_password12,enter_password13,enter_password14,enter_password1,I_R_S1 ,I_R_S2,I_R_S3,I_R_S4,I_R_S5,I_R_S6,I_R_S7,I_R_S8,I_R_S9,I_R_S10,I_R_S11,I_R_S12,I_R_S13,I_R_S14 ,I_R_S15,S_P_R,S_P_L)

parameter password1=4'b0000 ,password2=4'b0001,password3=4'b0010,password4=4'b0011,

password5=4'b0100,password6=4'b0101,password7=4'b0110, //Password for rooms. password8=4'b0111,password9=4'b1000,password10=4'b1001,password11=4'b1010,password12=4'b1011,password13=4'b1100,password14=4'b1101,password15=4'b1110;

input clk,I_R_S1,I_R_S2,I_R_S3,I_R_S4,I_R_S5,I_R_S6,I_R_S7,I_R_S8,I_R_S9,I_R_S10,

I_R_S11,I_R_S12,I_R_S13,I_R_S14,I_R_S15;// Infrared sensors

input R_T_L1,R_T_L2,R_T_L3,R_T_L4,R_T_L5,R_T_L6,R_T_L7,R_T_L8,R_T_L9,

R_T_L10,R_T_L11,R_T_L12,R_T_L13,R_T_L14,R_T_L15;// R_T_L = Recuest to leave.


input[3:0] basket_P1, basket_P2, basket_P3, basket_P4,basket_P5, basket_P6, basket_P7, basket_P8,basket_P9, basket_P10, basket_P11, basket_P12,basket_P13, basket_P14, basket_P15; //basket position

input [3:0] enter_password1,enter_password2,enter_password3,enter_password4,enter_password5,

enter_password6,enter_password7,enter_password8,enter_password9,enter_password10,enter_password11,enter_password12,enter_password13,enter_password14,enter_password15; //password entered by user to check

output reg S_P_R,S_P_L;//motor circling direction.

always @(R_T_L1 or basket_P1)

 begin

  if (R_T_L1 == 1 && I_R_S1 == 0 && ( password1 == enter_password1)&&(basket_P1 >= 1 && basket_P1 <=8)) //if receive recuest to leave and there are car.

      begin

       S_P_L <= 1; //motor circling left direction.      S_P_R <= 0;

      end
```

```
  else if (basket_P1 == 0)// when a basket reraches the receving location,the motor stops.
          begin
           S_P_R <= 0;    S_P_L <= 0;
          end
       else if (R_T_L1 == 1 && I_R_S1 == 0 && ( password1 == enter_password1)&&(basket_P1 > 8 && basket_P1 <= 15))
                  begin
                   S_P_R <= 1; //motor circling right direction.    S_P_L <= 0;
                  end
              else if (basket_P1 == 0) // when a basket reraches the receving location,the motor stops.
                     begin
                      S_P_R <= 0;    S_P_L <= 0;
                     end
  else if (R_T_L1 == 1 &&(I_R_S1 == 1 || ( password1 != enter_password1)))
          begin
            S_P_R <= 0;    S_P_L <= 0;
            $display ("No car,or entered error code ??");
          end
  end
 ////////////////////////////////////////////////////////////////////////////////////////////////////
always @(R_T_L2 or basket_P2)
 begin
  if (R_T_L2 == 1 && I_R_S2 == 0 && ( password2 == enter_password2)&&(basket_P2 >= 1 && basket_P2 <=8)) //if receive recuest to leave and there are car.
      begin
       S_P_L <= 1; //motor circling left direction.    S_P_R <= 0;
      end
  else if (basket_P2 == 0)// when a basket reraches the receving location,the motor stops.
```

```
        begin
         S_P_R <= 0;     S_P_L <= 0;
        end
    else if (R_T_L2 == 1 && I_R_S2 == 0 && ( password2 == enter_password2)&&(basket_P2 >8 && basket_P2 <=15))
            begin
             S_P_R <= 1; //motor circling right direction.       S_P_L <= 0;
            end
        else if (basket_P2 == 0) // when a basket reraches the receving location,the motor stops.
                begin
                 S_P_R <= 0;         S_P_L <= 0;
                end
  else if (R_T_L2 == 1 &&(I_R_S2 == 1 || ( password2 != enter_password2)))
        begin
         S_P_R <= 0;     S_P_L <= 0;
         $display ("No car,or entered error code ??");
        end             end
 ////////////////////////////////////////////////////////////////////////////////////////////////////
 always @( R_T_L3 or basket_P3)
 begin
  if (R_T_L3 == 1 && I_R_S3 == 0 && ( password3 == enter_password3)&&(basket_P3 >= 1 && basket_P3 <=8)) //if receive recuest to leave and there are car.
      begin
       S_P_L <= 1; //motor circling left direction.          S_P_R <= 0;
      end
  else if (basket_P3 == 0)// when a basket reraches the receving location,the motor stops.
         begin
          S_P_L <= 0;      S_P_R <= 0;
```

```
        end
    else if (R_T_L3 == 1 && I_R_S3 == 0 && ( password3 == enter_password3)&&(basket_P3 >8 && basket_P3 <=15))
            begin
             S_P_R <= 1; //motor circling right direction.        S_P_L <= 0;
            end
        else if (basket_P3 == 0) // when a basket reraches the receving location,the motor stops.
                begin
                 S_P_R <= 0;     S_P_L <= 0;
                end
  else if (R_T_L3 == 1 &&(I_R_S3 == 1 || ( password3 != enter_password3)))
      begin
        S_P_R <= 0;      S_P_L <= 0;
       $display ("No car,or entered error code ??");
      end       end
 ////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(R_T_L4 or basket_P4)
 begin
  if (R_T_L4 == 1 && I_R_S4 == 0 && ( password4 == enter_password4)&&(basket_P4 >= 1 && basket_P4 <=8)) //if receive recuest to leave and there are car.
      begin
       S_P_L <= 1; //motor circling left direction.          S_P_R <= 0;
      end
  else if (basket_P4 == 0)// when a basket reraches the receving location,the motor stops.
          begin
           S_P_L <= 0;      S_P_R <= 0;
          end
```

```
      else if (R_T_L4 == 1 && I_R_S4 == 0 && ( password4 == enter_password4)&&(basket_P4 >8
&& basket_P4 <=15))

                begin

                 S_P_R <= 1; //motor circling right direction.    S_P_L <= 0;

                 end

           else if (basket_P4 == 0) // when a basket reraches the receving location,the motor stops.

                    begin

                     S_P_R <= 0;    S_P_L <= 0;

                    end

  else if (R_T_L4 == 1 &&(I_R_S4 == 1 || ( password4 != enter_password4)))

        begin

           S_P_R <= 0;   S_P_L <= 0;

         $display ("No car,or entered error code ??");

        end                                    end

 ////////////////////////////////////////////////////////////////////////////////////////////////////

always @(R_T_L5 or basket_P5)

 begin

  if (R_T_L5 == 1 && I_R_S5 == 0 && ( password5 == enter_password5)&&(basket_P5 >= 1 &&
basket_P5 <=8)) //if receive recuest to leave and there are car.

       begin

        S_P_L <= 1; //motor circling left direction.            S_P_R <= 0;

       end

  else if (basket_P5 == 0)// when a basket reraches the receving location,the motor stops.

           begin

            S_P_R <= 0;       S_P_L <= 0;

           end

       else if (R_T_L5 == 1 && I_R_S5 == 0 && ( password5 == enter_password5)&&(basket_P5 >
8 && basket_P5 <= 15))
```

```
                begin
                 S_P_R <= 1; //motor circling right direction.        S_P_L <= 0;
                end
              else if (basket_P5 == 0) // when a basket reraches the receving location,the motor stops.
                  begin
                   S_P_R <= 0;          S_P_L <= 0;
                  end
  else if (R_T_L5 == 1 &&(I_R_S5 == 1 || ( password5 != enter_password5)))
          begin
            S_P_R <= 0;    S_P_L <= 0;
            $display ("No car,or entered error code ??");
          end             end
 ////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(R_T_L6 or basket_P6)
 begin
  if (R_T_L6 == 1 && I_R_S6 == 0 && ( password6 == enter_password6)&&(basket_P6 >= 1 &&
basket_P6 <=8)) //if receive recuest to leave and there are car.
      begin
       S_P_L <= 1; //motor circling left direction.             S_P_R <= 0;
      end
  else if (basket_P6 == 0)// when a basket reraches the receving location,the motor stops.
          begin
           S_P_R <= 0;     S_P_L <= 0;
          end
       else if (R_T_L6 == 1 && I_R_S6 == 0 && ( password6 == enter_password6)&&(basket_P6 >
8 && basket_P6 <= 15))
                begin
                 S_P_R <= 1; //motor circling right direction.        S_P_L <= 0;
```

```
                end
            else if (basket_P6 == 0) // when a basket reraches the receving location,the motor stops.
                begin
                 S_P_R <= 0;       S_P_L <= 0;
                end
 else if (R_T_L6 == 1 &&(I_R_S6 == 1 || ( password6 != enter_password6)))
        begin
          S_P_R <= 0;    S_P_L <= 0;
          $display ("No car,or entered error code ??");
        end              end
 ////////////////////////////////////////////////////////////////////////////////////////////////////////
 always @(R_T_L7 or basket_P7)
 begin
  if (R_T_L7 == 1 && I_R_S7 == 0 && ( password7 == enter_password7)&&(basket_P7 >= 1 &&
basket_P7 <=8)) //if receive recuest to leave and there are car.
      begin
       S_P_L <= 1; //motor circling left direction.    S_P_R <= 0;
      end
  else if (basket_P7 == 0)// when a basket reraches the receving location,the motor stops.
        begin
         S_P_R <= 0;     S_P_L <= 0;
        end
       else if (R_T_L7 == 1 && I_R_S7 == 0 && ( password7 == enter_password7)&&(basket_P7 >
8 && basket_P7 <= 15))
                begin
                 S_P_R <= 1; //motor circling right direction.       S_P_L <= 0;
                end
            else if (basket_P7 == 0) // when a basket reraches the receving location,the motor stops.
```

```
                begin
                 S_P_R <= 0;      S_P_L <= 0;
                end
  else if (R_T_L7 == 1 &&(I_R_S7 == 1 || ( password7 != enter_password7)))
        begin
          S_P_R <= 0;     S_P_L <= 0;
          $display ("No car,or entered error code ??");
        end             end
 ////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(R_T_L8 or basket_P8)
 begin
  if (R_T_L8 == 1 && I_R_S8 == 0 && ( password8 == enter_password8)&&(basket_P8 >= 1 &&
basket_P8 <=8)) //if receive recuest to leave and there are car.
      begin
       S_P_L <= 1; //motor circling left direction.          S_P_R <= 0;
      end
  else if (basket_P8 == 0)// when a basket reraches the receving location,the motor stops.
        begin
         S_P_R <= 0;    S_P_L <= 0;
        end
       else if (R_T_L8 == 1 && I_R_S8 == 0 && ( password8 == enter_password8)&&(basket_P8 >
8 && basket_P8 <= 15))
              begin
               S_P_R <= 1; //motor circling right direction.         S_P_L <= 0;
              end
            else if (basket_P8 == 0) // when a basket reraches the receving location,the motor stops.
                begin
                 S_P_R <= 0;       S_P_L <= 0;
```

```
                end
  else if (R_T_L8 == 1 &&(I_R_S8 == 1 || ( password8 != enter_password8)))
          begin
            S_P_R <= 0;     S_P_L <= 0;
            $display ("No car,or entered error code ??");
          end              end
 ////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(R_T_L9 or basket_P9)
 begin
  if (R_T_L9 == 1 && I_R_S9 == 0 && ( password9 == enter_password9)&&(basket_P9 >= 1 && basket_P9 <=8)) //if receive recuest to leave and there are car.
      begin
       S_P_L <= 1; //motor circling left direction.            S_P_R <= 0;
      end
  else if (basket_P9 == 0)// when a basket reraches the receving location,the motor stops.
          begin
           S_P_R <= 0;      S_P_L <= 0;
          end
       else if (R_T_L9 == 1 && I_R_S9 == 0 && ( password9 == enter_password9)&&(basket_P9 > 8 && basket_P9 <= 15))
                  begin
                   S_P_R <= 1; //motor circling right direction.       S_P_L <= 0;
                  end
              else if (basket_P9 == 0) // when a basket reraches the receving location,the motor stops.
                     begin
                      S_P_R <= 0;        S_P_L <= 0;
                     end
  else if (R_T_L9 == 1 &&(I_R_S9 == 1 || ( password9 != enter_password9)))
```

```
        begin
          S_P_R <= 0;      S_P_L <= 0;
          $display ("No car,or entered error code ??");
        end                    end
////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(R_T_L10 or basket_P10)
 begin
  if (R_T_L10 == 1 && I_R_S10 == 0 && ( password10 == enter_password10)&&(basket_P10 >= 1 && basket_P10 <=8)) //if receive recuest to leave and there are car.
      begin
       S_P_L <= 1; //motor circling left direction.              S_P_R <= 0;
      end
  else if (basket_P10 == 0)// when a basket reraches the receving location,the motor stops.
          begin
           S_P_R <= 0;      S_P_L <= 0;
          end
      else if (R_T_L10 == 1 && I_R_S10 == 0 && ( password10 == enter_password10)&&(basket_P10 > 8 && basket_P10 <= 15))
                  begin
                   S_P_R <= 1; //motor circling right direction.          S_P_L <= 0;
                  end
              else if (basket_P10 == 0) // when a basket reraches the receving location,the motor stops.
                      begin
                       S_P_R <= 0;      S_P_L <= 0;
                      end
  else if (R_T_L10 == 1 &&(I_R_S10 == 1 || ( password10 != enter_password10)))
          begin
            S_P_R <= 0;             S_P_L <= 0;
```

```
        $display ("No car,or entered error code ??");
      end       end
////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(R_T_L11 or basket_P11)
 begin
  if (R_T_L11 == 1 && I_R_S11 == 0 && ( password11 == enter_password11)&&(basket_P11 >= 1 && basket_P11 <=8)) //if receive recuest to leave and there are car.
     begin
      S_P_L <= 1; //motor circling left direction.        S_P_R <= 0;
     end
  else if (basket_P11 == 0)// when a basket reraches the receving location,the motor stops.
        begin
         S_P_R <= 0;    S_P_L <= 0;
        end
     else if (R_T_L11 == 1 && I_R_S11 == 0 && ( password11 == enter_password11)&&(basket_P11 > 8 && basket_P11 <= 15))
               begin
                S_P_R <= 1; //motor circling right direction.     S_P_L <= 0;
               end
           else if (basket_P11 == 0) // when a basket reraches the receving location,the motor stops.
                  begin
                   S_P_R <= 0;         S_P_L <= 0;
                  end
  else if (R_T_L11 == 1 &&(I_R_S11 == 1 || ( password11 != enter_password11)))
        begin
         S_P_R <= 0;        S_P_L <= 0;
         $display ("No car,or entered error code ??");
        end               end
```

```
////////////////////////////////////////////////////////////////////////////////////////////////////////////

always @(R_T_L12 or basket_P12)

 begin

  if (R_T_L12 == 1 && I_R_S12 == 0 && ( password12 == enter_password12)&&(basket_P12 >= 1 && basket_P12 <=8)) //if receive recuest to leave and there are car.

      begin

       S_P_L <= 1; //motor circling left direction.    S_P_R <= 0;

      end

  else if (basket_P12 == 0)// when a basket reraches the receving location,the motor stops.

          begin

           S_P_R <= 0;     S_P_L <= 0;

          end

      else if (R_T_L12 == 1 && I_R_S12 == 0 && ( password12 == enter_password12)&&(basket_P12 > 8 && basket_P12 <= 15))

                  begin

                   S_P_R <= 1; //motor circling right direction.       S_P_L <= 0;

                  end

              else if (basket_P12 == 0) // when a basket reraches the receving location,the motor stops.

                      begin

                       S_P_R <= 0;       S_P_L <= 0;

                      end

  else if (R_T_L12 == 1 &&(I_R_S12 == 1 || ( password12 != enter_password12)))

          begin

            S_P_R <= 0;       S_P_L <= 0;

            $display ("No car,or entered error code ??");

          end                 end

////////////////////////////////////////////////////////////////////////////////////////////////////////////

always @(R_T_L13 or basket_P13)
```

```
 begin
  if (R_T_L13 == 1 && I_R_S13 == 0 && ( password13 == enter_password13)&&(basket_P13 >= 1 && basket_P13 <=8)) //if receive recuest to leave and there are car.
      begin
       S_P_L <= 1; //motor circling left direction.    S_P_R <= 0;
      end
  else if (basket_P13 == 0)// when a basket reraches the receving location,the motor stops.
          begin
           S_P_R <= 0;           S_P_L <= 0;
          end
      else if (R_T_L13 == 1 && I_R_S13 == 0 && ( password13 == enter_password13)&&(basket_P13 > 8 && basket_P13 <= 15))
                  begin
                   S_P_R <= 1; //motor circling right direction.        S_P_L <= 0;
                  end
              else if (basket_P13 == 0) // when a basket reraches the receving location,the motor stops.
                      begin
                       S_P_R <= 0;        S_P_L <= 0;
                      end
  else if (R_T_L13 == 1 &&(I_R_S13 == 1 || ( password13 != enter_password13)))
          begin
            S_P_R <= 0;       S_P_L <= 0;
            $display ("No car,or entered error code ??");
          end                  end
 ////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(R_T_L14 or basket_P14)
 begin
```

```
 if (R_T_L14 == 1 && I_R_S14 == 0 && ( password14 == enter_password14)&&(basket_P14 >= 1 && basket_P14 <=8)) //if receive recuest to leave and there are car.
      begin
       S_P_L <= 1; //motor circling left direction.        S_P_R <= 0;
      end
  else if (basket_P14 == 0)// when a basket reraches the receving location,the motor stops.
          begin
           S_P_R <= 0;          S_P_L <= 0;
          end
      else if (R_T_L14 == 1 && I_R_S14 == 0 && ( password14 == enter_password14)&&(basket_P14 > 8 && basket_P14 <= 15))
                 begin
                  S_P_R <= 1; //motor circling right direction.              S_P_L <= 0;
                 end
              else if (basket_P14 == 0) // when a basket reraches the receving location,the motor stops.
                     begin
                      S_P_R <= 0;          S_P_L <= 0;
                     end
  else if (R_T_L14 == 1 &&(I_R_S14 == 1 || ( password14 != enter_password14)))
          begin
            S_P_R <= 0;         S_P_L <= 0;
            $display ("No car,or entered error code ??");
          end                    end
 ////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(R_T_L15 or basket_P15)
 begin
  if (R_T_L15 == 1 && I_R_S15 == 0 && ( password15 == enter_password15)&&(basket_P15 >= 1 && basket_P15 <=8)) //if receive recuest to leave and there are car.
```

```
      begin
       S_P_L <= 1; //motor circling left direction.   S_P_R <= 0;
      end
  else if (basket_P15 == 0)// when a basket reraches the receving location,the motor stops.
          begin
           S_P_R <= 0;        S_P_L <= 0;
          end
      else if (R_T_L15 == 1 && I_R_S15 == 0 && ( password15 == enter_password15)&&(basket_P15 > 8 && basket_P15 <= 15))
                  begin
                   S_P_R <= 1; //motor circling right direction.     S_P_L <= 0;
                  end
              else if (basket_P15 == 0) // when a basket reraches the receving location,the motor stops.
                      begin
                       S_P_R <= 0;        S_P_L <= 0;
                      end
  else if (R_T_L15 == 1 &&(I_R_S15 == 1 || ( password15 != enter_password15)))
          begin
            S_P_R <= 0;        S_P_L <= 0;
            $display ("No car,or entered error code ??");
          end          end
endmodule
```

Sub module to compute a payment when a car stay in system (Time_count)

```
/* This code to compute a payment when a car stay in system */

moduleTime_count  (clk,payment1,payment2,payment3,payment4,payment5,payment6,payment7
,payment8,payment9,payment10,payment11,payment12,payment13,payment14,payment15,
```

payment16,R_T_L1,R_T_L2,R_T_L3,R_T_L4,R_T_L5,R_T_L6,R_T_L7,R_T_L8,R_T_L9,R_T_L10,R_T_L11,R_T_L12,R_T_L13,R_T_L14,R_T_L15,R_T_L16,I_R_S1,I_R_S2,I_R_S3,I_R_S4,I_R_S5,I_R_S6,I_R_S7,I_R_S8,I_R_S9,I_R_S10,I_R_S11,I_R_S12,I_R_S13,I_R_S14,I_R_S15,I_R_S16,enter_password1,enter_password2,enter_password3,enter_password4,enter_password5,enter_password6,enter_password7,enter_password8,enter_password9,enter_password10,enter_password11,enter_password12,enter_password13,enter_password14,enter_password15,enter_password16);

parameter payment_rate = 10;// Payment per hour.

Parameter password1=4'b0000,password2=4'b0001,password3=4'b0010,password4=4'b0011,

password5=4'b0100,password6=4'b0101,password7=4'b0110, //Password for rooms.

password8=4'b0111,password9=4'b1000,password10=4'b1001,password11=4'b1010,password12=4'b1011,password13=4'b1100,password14=4'b1101, password15=4'b1110,password16=4'b1111;

input clk;

input I_R_S1,I_R_S2,I_R_S3,I_R_S4,I_R_S5,I_R_S6,I_R_S7,I_R_S8,I_R_S9,I_R_S10,I_R_S11

,I_R_S12,I_R_S13,I_R_S14,I_R_S15,I_R_S16;// Infrared sensors

input R_T_L1,R_T_L2,R_T_L3,R_T_L4,R_T_L5,R_T_L6,R_T_L7,R_T_L8,R_T_L9,R_T_L10

,R_T_L11,R_T_L12,R_T_L13,R_T_L14,R_T_L15,R_T_L16;// R_T_L = Recuest to leave.

input [3:0] enter_password1,enter_password2,enter_password3,enter_password4,enter_password5,

enter_password6,enter_password7,enter_password8,enter_password9,enter_password10,enter_password11,enter_password12,enter_password13,enter_password14,enter_password15,enter_password16; //password entered by user to check.

output reg [5:0]payment1,payment2,payment3,payment4,payment5,payment6,payment7,payment8,

payment9,payment10,payment11,payment12,payment13,payment14,payment15,payment16;

reg [9:0]one_second_counter1,one_second_counter2,one_second_counter3,one_second_counter4

,one_second_counter5,one_second_counter6,one_second_counter7,one_second_counter8,

one_second_counter9,one_second_counter10,one_second_counter11,one_second_counter12,one_second_counter13,one_second_counter14,one_second_counter15,one_second_counter16;

reg[5:0]one_minute_counter1,one_minute_counter2,one_minute_counter3,one_minute_counter4,one_minute_counter5,one_minute_counter6,one_minute_counter7,one_minute_counter8,

one_minute_counter9,one_minute_counter10,one_minute_counter11,one_minute_counter12,one_minute_counter13,one_minute_counter14,one_minute_counter15,one_minute_counter16; //Regester to count time when car stay in parking system.

reg[5:0]one_houre_counter1,one_houre_counter2,one_houre_counter3,one_houre_counter4,one_houre_counter5,one_houre_counter6,one_houre_counter7,one_houre_counter8,

one_houre_counter9,one_houre_counter10,one_houre_counter11,one_houre_counter12,one_houre_counter13,one_houre_counter14,one_houre_counter15,one_houre_counter16;

reg[4:0]hour1,hour2,hour3,hour4,hour5,hour6,hour7,hour8,hour9,hour10,hour11,hour12,hour13,hour14,hour15,hour16;//regester to count number of hour.

initial payment1 =6'b000000; initial payment2 =6'b000000; initial payment3 =6'b000000; initial payment4 =6'b000000; initial payment5 =6'b000000;

initial payment6 =6'b000000; initial payment7 =6'b000000; initial payment8 =6'b000000; initial payment9 =6'b000000; initial payment10 =6'b000000;

initial payment11 =6'b000000; initial payment12 =6'b000000; initial payment13 =6'b000000; initial payment14 =6'b000000; initial payment15 =6'b000000;

initial payment16 =6'b000000;

initial one_second_counter1=10'b0000000000; initial one_second_counter2=10'b0000000000; initial one_second_counter3=10'b0000000000;

initial one_second_counter4=10'b0000000000; initial one_second_counter5=10'b0000000000; initial one_second_counter6=10'b0000000000;

initial one_second_counter7=10'b0000000000; initial one_second_counter8=10'b0000000000; initial one_second_counter9=10'b0000000000;

initial one_second_counter10=10'b0000000000; initial one_second_counter11=10'b0000000000; initial one_second_counter12=10'b0000000000;

initial one_second_counter13=10'b0000000000; initial one_second_counter14=10'b0000000000; initial one_second_counter15=10'b0000000000;

initial one_second_counter16=10'b0000000000;

initial one_minute_counter1=6'b000000; initial one_minute_counter2=6'b000000; initial one_minute_counter3=6'b000000;

initial one_minute_counter4=6'b000000; initial one_minute_counter5=6'b000000; initial one_minute_counter6=6'b000000;

initial one_minute_counter7=6'b000000; initial one_minute_counter8=6'b000000; initial one_minute_counter9=6'b000000;

initial one_minute_counter10=6'b000000; initial one_minute_counter11=6'b000000; initial one_minute_counter12=6'b000000;

initial one_minute_counter13=6'b000000; initial one_minute_counter14=6'b000000; initial one_minute_counter15=6'b000000;

initial one_minute_counter16=6'b000000;

initial one_houre_counter1=6'b000000; initial one_houre_counter2=6'b000000; initial one_houre_counter3=6'b000000;

initial one_houre_counter4=6'b000000; initial one_houre_counter5=6'b000000; initial one_houre_counter6=6'b000000;

initial one_houre_counter7=6'b000000; initial one_houre_counter8=6'b000000; initial one_houre_counter9=6'b000000;

initial one_houre_counter10=6'b000000; initial one_houre_counter11=6'b000000; initial one_houre_counter12=6'b000000;

initial one_houre_counter13=6'b000000; initial one_houre_counter14=6'b000000; initial one_houre_counter15=6'b000000;

initial one_houre_counter16=6'b000000;

initial hour1 = 5'b00000; initial hour2= 5'b00000; initial hour3 = 5'b00000; initial hour4 = 5'b00000; initial hour5 = 5'b00000; initial hour6 = 5'b00000;

initial hour7 = 5'b00000; initial hour8 = 5'b00000; initial hour9 = 5'b00000; initial hour10 = 5'b00000; initial hour11 = 5'b00000; initial hour12 = 5'b00000;

initial hour13 = 5'b00000;initial hour14 = 5'b00000;initial hour15 = 5'b00000;initial hour16 = 5'b00000;

wire one_second_enable1,one_second_enable2,one_second_enable3,one_second_enable4,

one_second_enable5,one_second_enable6,one_second_enable7,one_second_enable8,

one_second_enable9,one_second_enable10,one_second_enable11,one_second_enable12,one_second_enable13,one_second_enable14,one_second_enable15,one_second_enable16;// one second enable for counting numbers

wire one_minute_enable1,one_minute_enable2,one_minute_enable3,one_minute_enable4,

one_minute_enable5,one_minute_enable6,one_minute_enable7,one_minute_enable8,

one_minute_enable9,one_minute_enable10,one_minute_enable11,one_minute_enable12,one_minute_enable13,one_minute_enable14,one_minute_enable15,one_minute_enable16;

wire one_houre_enable1,one_houre_enable2,one_houre_enable3,one_houre_enable4,

one_houre_enable5,one_houre_enable6,one_houre_enable7,one_houre_enable8,

```
one_houre_enable9,one_houre_enable10,one_houre_enable11,one_houre_enable12,one_houre_enable13,one_houre_enable14,one_houre_enable15,one_houre_enable16;
always @(posedge clk )
   begin
     if (I_R_S1 ==0)
         begin
          if (one_second_counter1>=10) one_second_counter1 <= 0;
          else one_second_counter1 <= one_second_counter1 + 1;
         end  end
assign one_second_enable1 = (one_second_counter1==10)?1:0;
always @(posedge clk)
   begin
      if (one_second_enable1==1) one_minute_counter1 <= one_minute_counter1 + 1;
      else if (one_minute_counter1>=5) one_minute_counter1 <= 0;
   end
assign one_minute_enable1 = (one_minute_counter1==5)?1:0;
always @(posedge clk)
   begin
      if (one_minute_enable1==1) one_houre_counter1 <= one_houre_counter1 + 1;
      else if (one_houre_counter1>=5) one_houre_counter1 <= 0;
   end
assign one_houre_enable1 = (one_houre_counter1==5)?1:0;
always @( * )
        begin
        if (one_houre_enable1 == 1) hour1= hour1+1;
         if (R_T_L1 ==1 &&(enter_password1 == password1))
             begin
```

```
            payment1 = hour1 * payment_rate;
            end
        else if (R_T_L1 ==1 &&(enter_password1 != password1))
            begin
            $display ("no car");
            end        end
 ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(posedge clk )
   begin
     if (I_R_S2 ==0)
         begin
          if (one_second_counter2>=10) one_second_counter2 <= 0;
          else one_second_counter2 <= one_second_counter2 + 1;
         end     end
assign one_second_enable2 = (one_second_counter2==10)?1:0;
always @(posedge clk)
   begin
      if (one_second_enable2==1) one_minute_counter2 <= one_minute_counter2 + 1;
      else if (one_minute_counter2>=5) one_minute_counter2 <= 0;
   end
assign one_minute_enable2 = (one_minute_counter2==5)?1:0;
always @(posedge clk)
   begin
      if (one_minute_enable2==1) one_houre_counter2 <= one_houre_counter2 + 1;
      else if (one_houre_counter2>=5) one_houre_counter2 <= 0;
   end
assign one_houre_enable2 = (one_houre_counter2==5)?1:0;
```

```
always @( * )
        begin
        if (one_houre_enable2 == 1) hour2= hour2+1;
         if (R_T_L2 ==1 &&(enter_password2 == password2))
                begin
                payment2 = hour2 * payment_rate;
                end
         else if (R_T_L2 ==1 &&(enter_password2 != password2))
                 begin
                 $display ("no car");
                 end        end
  ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(posedge clk )
   begin
     if (I_R_S3 ==0)
          begin
           if (one_second_counter3>=10) one_second_counter3 <= 0;
           else one_second_counter3 <= one_second_counter3 + 1;
          end  end
assign one_second_enable3 = (one_second_counter3==10)?1:0;
always @(posedge clk)
   begin
       if (one_second_enable3==1) one_minute_counter3 <= one_minute_counter3 + 1;
       else if (one_minute_counter3>=5) one_minute_counter3 <= 0;
   end
assign one_minute_enable3 = (one_minute_counter3==5)?1:0;
always @(posedge clk)
```

```
    begin
        if (one_minute_enable3==1) one_houre_counter3 <= one_houre_counter3 + 1;
        else if (one_houre_counter3>=5) one_houre_counter3 <= 0;
    end
assign one_houre_enable3 = (one_houre_counter3==5)?1:0;
always @( * )
         begin
         if (one_houre_enable3 == 1) hour3= hour3+1;
          if (R_T_L3 ==1 &&(enter_password3 == password3))
                 begin
                 payment3 = hour3 * payment_rate;
                 end
          else if (R_T_L3 ==1 &&(enter_password3 != password3))
                  begin
                  $display ("no car");
                  end       end
  ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(posedge clk )
    begin
      if (I_R_S4 ==0)
           begin
            if (one_second_counter4>=10) one_second_counter4 <= 0;
            else one_second_counter4 <= one_second_counter4 + 1;
           end    end
assign one_second_enable4 = (one_second_counter4==10)?1:0;
always @(posedge clk)
    begin
```

```
        if (one_second_enable4==1) one_minute_counter4 <= one_minute_counter4 + 1;
        else if (one_minute_counter4>=5) one_minute_counter4 <= 0;
    end
assign one_minute_enable4 = (one_minute_counter4==5)?1:0;
always @(posedge clk)
    begin
        if (one_minute_enable4==1) one_houre_counter4 <= one_houre_counter4 + 1;
        else if (one_houre_counter4>=5) one_houre_counter4 <= 0;
    end
assign one_houre_enable4 = (one_houre_counter4==5)?1:0;
always @( * )
         begin
         if (one_houre_enable4 == 1) hour4= hour4+1;
          if (R_T_L4 ==1 &&(enter_password4 == password4))
                 begin
                 payment4 = hour4 * payment_rate;
                 end
          else if (R_T_L4 ==1 &&(enter_password4 != password4))
                  begin
                  $display ("no car");
                  end          end
   ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(posedge clk )
    begin
      if (I_R_S5 ==0)
          begin
           if (one_second_counter5>=10) one_second_counter5 <= 0;
```

```
          else one_second_counter5 <= one_second_counter5 + 1;
         end  end
assign one_second_enable5 = (one_second_counter5==10)?1:0;
always @(posedge clk)
   begin
      if (one_second_enable5==1) one_minute_counter5 <= one_minute_counter5 + 1;
      else if (one_minute_counter5>=5) one_minute_counter5 <= 0;
   end
assign one_minute_enable5 = (one_minute_counter5==5)?1:0;
always @(posedge clk)
   begin
      if (one_minute_enable5==1) one_houre_counter5 <= one_houre_counter5 + 1;
      else if (one_houre_counter5>=5) one_houre_counter5 <= 0;
   end
assign one_houre_enable5 = (one_houre_counter5==5)?1:0;
 always @( * )
        begin
        if (one_houre_enable5 == 1) hour5= hour5+1;
         if (R_T_L5 ==1 &&(enter_password5 == password5))
               begin
               payment5 = hour5 * payment_rate;
               end
        else if (R_T_L5 ==1 &&(enter_password5 != password5))
                begin
                $display ("no car");
                end       end
  ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

```
always @(posedge clk )
   begin
     if (I_R_S6 ==0)
         begin
          if (one_second_counter6>=10) one_second_counter6 <= 0;
          else one_second_counter6 <= one_second_counter6 + 1;
         end       end
assign one_second_enable6 = (one_second_counter6==10)?1:0;
always @(posedge clk)
   begin
       if (one_second_enable6==1) one_minute_counter6 <= one_minute_counter6 + 1;
       else if (one_minute_counter6>=5) one_minute_counter6 <= 0;
   end
assign one_minute_enable6 = (one_minute_counter6==5)?1:0;
  always @(posedge clk)
   begin
       if (one_minute_enable6==1) one_houre_counter6 <= one_houre_counter6 + 1;
       else if (one_houre_counter6>=5) one_houre_counter6 <= 0;
   end
assign one_houre_enable6 = (one_houre_counter6==5)?1:0;
always @( * )
        begin
        if (one_houre_enable6 == 1) hour6= hour6+1;
         if (R_T_L6 ==1 &&(enter_password6 == password6))
               begin
               payment6 = hour6 * payment_rate;
               end
```

```
        else if (R_T_L6 ==1 &&(enter_password6 != password6))
                begin
                $display ("no car");
                end       end
  ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(posedge clk )
   begin
     if (I_R_S7 ==0)
         begin
          if (one_second_counter7>=10) one_second_counter7 <= 0;
          else one_second_counter7 <= one_second_counter7 + 1;
         end       end
assign one_second_enable7 = (one_second_counter7==10)?1:0;
always @(posedge clk)
   begin
      if (one_second_enable7==1) one_minute_counter7 <= one_minute_counter7 + 1;
      else if (one_minute_counter7>=5) one_minute_counter7 <= 0;
   end
assign one_minute_enable7 = (one_minute_counter7==5)?1:0;
always @(posedge clk)
   begin
      if (one_minute_enable7==1) one_houre_counter7 <= one_houre_counter7 + 1;
      else if (one_houre_counter7>=5) one_houre_counter7 <= 0;
   end
assign one_houre_enable7 = (one_houre_counter7==5)?1:0;
always @( * )
        begin
```

```
        if (one_houre_enable7 == 1) hour7= hour7+1;
         if (R_T_L7 ==1 &&(enter_password7 == password7))
                begin
                payment7 = hour7 * payment_rate;
                end
         else if (R_T_L7 ==1 &&(enter_password7 != password7))
                 begin
                 $display ("no car");
                 end          end
  ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(posedge clk )
   begin
     if (I_R_S8 ==0)
         begin
          if (one_second_counter8>=10) one_second_counter8 <= 0;
          else one_second_counter8 <= one_second_counter8 + 1;
         end       end
assign one_second_enable8 = (one_second_counter8==10)?1:0;
always @(posedge clk)
   begin
      if (one_second_enable8==1) one_minute_counter8 <= one_minute_counter8 + 1;
      else if (one_minute_counter8>=5) one_minute_counter8 <= 0;
   end
assign one_minute_enable8 = (one_minute_counter8==5)?1:0;
always @(posedge clk)
   begin
      if (one_minute_enable8==1) one_houre_counter8 <= one_houre_counter8 + 1;
```

```
        else if (one_houre_counter8>=5) one_houre_counter8 <= 0;
    end
assign one_houre_enable8 = (one_houre_counter8==5)?1:0;
always @( * )
        begin
        if (one_houre_enable8 == 1) hour8= hour8+1;
         if (R_T_L8 ==1 &&(enter_password8 == password8))
                begin
                payment8 = hour8 * payment_rate;
                end
         else if (R_T_L8 ==1 &&(enter_password8 != password8))
                 begin
                 $display ("no car");
                 end      end
  ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(posedge clk )
    begin
      if (I_R_S9 ==0)
          begin
           if (one_second_counter9>=10) one_second_counter9 <= 0;
           else one_second_counter9 <= one_second_counter9 + 1;
          end  end
assign one_second_enable9 = (one_second_counter9==10)?1:0;
always @(posedge clk)
    begin
        if (one_second_enable9==1) one_minute_counter9 <= one_minute_counter9 + 1;
        else if (one_minute_counter9>=5) one_minute_counter9 <= 0;
```

```
    end
assign one_minute_enable9 = (one_minute_counter9==5)?1:0;
always @(posedge clk)
    begin
        if (one_minute_enable9==1) one_houre_counter9 <= one_houre_counter9 + 1;
        else if (one_houre_counter9>=5) one_houre_counter9 <= 0;
    end
assign one_houre_enable9 = (one_houre_counter9==5)?1:0;
always @( * )
         begin
         if (one_houre_enable9 == 1) hour9= hour9+1;
          if (R_T_L9 ==1 &&(enter_password9 == password9))
                 begin
                 payment9 = hour9 * payment_rate;
                 end
          else if (R_T_L9 ==1 &&(enter_password9 != password9))
                  begin
                  $display ("no car");
                  end       end
  ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(posedge clk )
    begin
      if (I_R_S10 ==0)
           begin
            if (one_second_counter10>=10) one_second_counter10 <= 0;
            else one_second_counter10 <= one_second_counter10 + 1;
           end
```

```
    end
assign one_second_enable10 = (one_second_counter10==10)?1:0
always @(posedge clk)
    begin
        if (one_second_enable10==1) one_minute_counter10 <= one_minute_counter10 + 1;
        else if (one_minute_counter10>=5) one_minute_counter10 <= 0;
    end
assign one_minute_enable10 = (one_minute_counter10==5)?1:0;
always @(posedge clk)
    begin
        if (one_minute_enable10==1) one_houre_counter10 <= one_houre_counter10 + 1;
        else if (one_houre_counter10>=5) one_houre_counter10 <= 0;
    end
assign one_houre_enable10 = (one_houre_counter10==5)?1:0;
always @( * )
         begin
         if (one_houre_enable10 == 1) hour10= hour10+1;
          if (R_T_L10 ==1 &&(enter_password10 == password10))
                 begin
                 payment10 = hour10 * payment_rate;
                 end
          else if (R_T_L10 ==1 &&(enter_password10 != password10))
                  begin
                  $display ("no car");
                  end       end
   ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(posedge clk )
```

```
    begin
      if (I_R_S11 ==0)
          begin
          if (one_second_counter11>=10) one_second_counter11 <= 0;
           else one_second_counter11 <= one_second_counter11 + 1;
          end     end
assign one_second_enable11 = (one_second_counter11==10)?1:0;
always @(posedge clk)
    begin
       if (one_second_enable11==1) one_minute_counter11 <= one_minute_counter11 + 1;
       else if (one_minute_counter11>=5) one_minute_counter11 <= 0;
    end
assign one_minute_enable11 = (one_minute_counter11==5)?1:0;
always @(posedge clk)
    begin
       if (one_minute_enable11==1) one_houre_counter11 <= one_houre_counter11 + 1;
       else if (one_houre_counter11>=5) one_houre_counter11 <= 0;
    end
assign one_houre_enable11 = (one_houre_counter11==5)?1:0;
always @( * )
         begin
         if (one_houre_enable11 == 1) hour11= hour11+1;
          if (R_T_L11 ==1 &&(enter_password11 == password11))
                begin
                payment11 = hour11 * payment_rate;
                end
          else if (R_T_L11 ==1 &&(enter_password11 != password11))
```

```
            begin
            $display ("no car");
            end       end
  ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(posedge clk )
   begin
     if (I_R_S12 ==0)
         begin
          if (one_second_counter12>=10) one_second_counter12 <= 0;
          else one_second_counter12 <= one_second_counter12 + 1;
         end  end
assign one_second_enable12 = (one_second_counter12==10)?1:0;
always @(posedge clk)
   begin
      if (one_second_enable12==1) one_minute_counter12 <= one_minute_counter12 + 1;
      else if (one_minute_counter12>=5) one_minute_counter12 <= 0;
   end
assign one_minute_enable12 = (one_minute_counter12==5)?1:0;
always @(posedge clk)
   begin
      if (one_minute_enable12==1) one_houre_counter12 <= one_houre_counter12 + 1;
      else if (one_houre_counter12>=5) one_houre_counter12 <= 0;
   end
assign one_houre_enable12 = (one_houre_counter12==5)?1:0;
always @( * )
       begin
       if (one_houre_enable12 == 1) hour12= hour12+1;
```

```
        if (R_T_L12 ==1 &&(enter_password12 == password12))
              begin
              payment12 = hour12 * payment_rate;
              end
        else if (R_T_L12 ==1 &&(enter_password12 != password12))
               begin
               $display ("no car");
               end       end
  ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(posedge clk )
   begin
     if (I_R_S13 ==0)
         begin
          if (one_second_counter13>=10) one_second_counter13 <= 0;
          else one_second_counter13 <= one_second_counter13 + 1;
         end       end
assign one_second_enable13 = (one_second_counter13==10)?1:0;
always @(posedge clk)
   begin
      if (one_second_enable13==1) one_minute_counter13 <= one_minute_counter13 + 1;
      else if (one_minute_counter13>=5) one_minute_counter13 <= 0;
   end
assign one_minute_enable13 = (one_minute_counter13==5)?1:0;
always @(posedge clk)
   begin
      if (one_minute_enable13==1) one_houre_counter13 <= one_houre_counter13 + 1;
      else if (one_houre_counter13>=5) one_houre_counter13 <= 0;
```

```
    end
assign one_houre_enable13 = (one_houre_counter13==5)?1:0;
always @( * )
        begin
        if (one_houre_enable13 == 1) hour13= hour13+1;
         if (R_T_L13 ==1 &&(enter_password13 == password13))
                begin
                payment13 = hour13 * payment_rate;
                end
         else if (R_T_L13 ==1 &&(enter_password13 != password13))
                 begin
                 $display ("no car");
                 end        end
  ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(posedge clk )
    begin
      if (I_R_S14 ==0)
          begin
           if (one_second_counter14>=10) one_second_counter14 <= 0;
           else one_second_counter14 <= one_second_counter14 + 1;
          end       end
assign one_second_enable14 = (one_second_counter14==10)?1:0;
always @(posedge clk)
    begin
       if (one_second_enable14==1) one_minute_counter14 <= one_minute_counter14 + 1;
       else if (one_minute_counter14>=5) one_minute_counter14 <= 0;
    end
```

```
assign one_minute_enable14 = (one_minute_counter14==5)?1:0;
always @(posedge clk)
   begin
      if (one_minute_enable14==1) one_houre_counter14 <= one_houre_counter14 + 1;
      else if (one_houre_counter14>=5) one_houre_counter14 <= 0;
   end
assign one_houre_enable14 = (one_houre_counter14==5)?1:0;
always @( * )
       begin
       if (one_houre_enable14 == 1) hour14= hour14+1;
        if (R_T_L14 ==1 &&(enter_password14 == password14))
              begin
              payment14 = hour14 * payment_rate;
              end
            else if (R_T_L14 ==1 &&(enter_password14 != password14))
               begin
               $display ("no car");
               end      end
  ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(posedge clk )
   begin
     if (I_R_S15 ==0)
         begin
          if (one_second_counter15>=10) one_second_counter15 <= 0;
          else one_second_counter15 <= one_second_counter15 + 1;
         end end
assign one_second_enable15 = (one_second_counter15==10)?1:0;
```

```
always @(posedge clk)
   begin
      if (one_second_enable15==1) one_minute_counter15 <= one_minute_counter15 + 1;
      else if (one_minute_counter15>=5) one_minute_counter15 <= 0;
   end
assign one_minute_enable15 = (one_minute_counter15==5)?1:0;
always @(posedge clk)
   begin
      if (one_minute_enable15==1) one_houre_counter15 <= one_houre_counter15 + 1;
      else if (one_houre_counter15>=5) one_houre_counter15 <= 0;
   end
assign one_houre_enable15 = (one_houre_counter15==5)?1:0;
always @( * )
       begin
       if (one_houre_enable15 == 1) hour15= hour15+1;
        if (R_T_L15 ==1 &&(enter_password15 == password15))
          begin
              payment15 = hour15 * payment_rate;
              end
        else if (R_T_L15 ==1 &&(enter_password15 != password15))
              begin
              $display ("no car");
              end       end
  ////////////////////////////////////////////////////////////////////////////////////////
always @(posedge clk )
   begin
```

```
    if (I_R_S16 ==0)
        begin
         if (one_second_counter16>=10) one_second_counter16 <= 0;
         else one_second_counter16 <= one_second_counter16 + 1;
        end        end
assign one_second_enable16 = (one_second_counter16==10)?1:0;
always @(posedge clk)
  begin
    if (one_second_enable16==1) one_minute_counter16 <= one_minute_counter16 + 1;
    else if (one_minute_counter16>=5) one_minute_counter16 <= 0;
  end
assign one_minute_enable16 = (one_minute_counter16==5)?1:0;
always @(posedge clk)
  begin
    if (one_minute_enable16==1) one_houre_counter16 <= one_houre_counter16 + 1;
    else if (one_houre_counter1>=5) one_houre_counter1 <= 0;
  end
assign one_houre_enable16 = (one_houre_counter16==5)?1:0;
always @( * )
       begin
       if (one_houre_enable16 == 1) hour16= hour16+1;
        if (R_T_L16 ==1 &&(enter_password16 == password16))
              begin
              payment16 = hour16 * payment_rate;
              end
       else if (R_T_L16 ==1 &&(enter_password16 != password16))
              begin
```

```
            $display ("no car");
            end      end
endmodule
```

Test bench code

```
`timescale 1ns/1ns
module Test;
reg clk;
reg I_R_S1,I_R_S2,I_R_S3,I_R_S4,I_R_S5,I_R_S6,I_R_S7,I_R_S8,I_R_S9,I_R_S10,I_R_S11,
I_R_S12,I_R_S13,I_R_S14,I_R_S15,I_R_S16;
reg R_T_L1,R_T_L2,R_T_L3,R_T_L4,R_T_L5,R_T_L6,R_T_L7,R_T_L8,R_T_L9,R_T_L10,
R_T_L11,R_T_L12,R_T_L13,R_T_L14,R_T_L15,R_T_L16;// R_T_L = Recuest to leave.
reg [12:0]front_w,rearer_w;
reg [6:0]T_S1,T_S2,T_S3,T_S4,T_S5,T_S6,T_S7,T_S8,T_S9,T_S10,T_S11,T_S12,T_S13,
T_S14,T_S15,T_S16;// Temperature sensor.
reg [3:0] enter_password1,enter_password2,enter_password3,enter_password4,enter_password5,
enter_password6,enter_password7,enter_password8,enter_password9,enter_password10,enter_password11,enter_password12,enter_password13,enter_password14,enter_password15,enter_password16;
reg [3:0] basket_P1, basket_P2, basket_P3, basket_P4,basket_P5, basket_P6, basket_P7, basket_P8,basket_P9, basket_P10, basket_P11, basket_P12,basket_P13, basket_P14, basket_P15; //basket position
reg R_B_S;
wire S_P_R1,S_P_L1;//motor circling direction.
wire S_P_R,S_P_L;//motor circling direction.
wire Instance;
wire [3:0] empty_places;
wire W_S1,W_S2,W_S3,W_S4,W_S5,W_S6,W_S7,W_S8,W_S9,W_S10,W_S11,W_S12,
W_S13,W_S14,W_S15,W_S16,buzzer; // Water switch.
```

```
wire [9:0]payment1,payment2,payment3,payment4,payment5,payment6,payment7,payment8,

payment9,payment10,payment11,payment12,payment13,payment14,payment15,payment16;

Rotary_Car_Parkinguut(.clk(clk),.R_B_S(R_B_S),.I_R_S1(I_R_S1),.I_R_S2(I_R_S2),.

I_R_S3(I_R_S3),.I_R_S4(I_R_S4),.I_R_S5(I_R_S5),.I_R_S6(I_R_S6),.I_R_S7(I_R_S7) ,.I_R_S8(I
_R_S8),.I_R_S9(I_R_S9),.I_R_S10(I_R_S10),.I_R_S11(I_R_S11),.I_R_S12(I_R_S12),.I_R_S13(I_
R_S13),.I_R_S14(I_R_S14),.I_R_S15(I_R_S15),.I_R_S16(I_R_S16),.empty_places(empty_places),.
front_w(front_w),.rearer_w(rearer_w),.Instance(Instance),.payment1(payment1) ,.payment2(paymen
t2),.payment3(payment3),.payment4(payment4),.payment5(payment5),.payment6(payment6),.payme
nt7(payment7),.payment8(payment8),.payment9(payment9),.payment10(payment10),.payment11(pa
yment11),.payment12(payment12),.payment13(payment13),.payment14(payment14),.payment15(pa
yment15),.payment16(payment16),.R_T_L1(R_T_L1),.R_T_L2(R_T_L2),.R_T_L3(R_T_L3),.R_T_
L4(R_T_L4),.R_T_L5(R_T_L5),.R_T_L6(R_T_L6),.R_T_L7(R_T_L7),.R_T_L8(R_T_L8),.R_T_L
9(R_T_L9),.R_T_L10(R_T_L10),.R_T_L11(R_T_L11),.R_T_L12(R_T_L12),.R_T_L13(R_T_L13),
.R_T_L14(R_T_L14),.R_T_L15(R_T_L15),.R_T_L16(R_T_L16) ,.enter_password1(enter_passwor
d1),.enter_password2(enter_password2),.enter_password3(enter_password3),.enter_password4(enter
_password4),.enter_password5(enter_password5) ,.enter_password6(enter_password6),.enter_passw
ord7(enter_password7),.enter_password8(enter_password8),.enter_password9(enter_password9),.ent
er_password10(enter_password10) ,.enter_password11(enter_password11),.enter_password12(enter_
password12),.enter_password13(enter_password13),.enter_password14(enter_password14),.enter_pa
ssword15(enter_password15 ,.enter_password16(enter_password16),.T_S1(T_S1),.T_S2(T_S2),.T_S
3(T_S3),.T_S4(T_S4),.T_S5(T_S5),.T_S6(T_S6),.T_S7(T_S7),.T_S8(T_S8),.T_S9(T_S9),.T_S10(T
_S10),.T_S11(T_S11),.T_S12(T_S12),.T_S13(T_S13),.T_S14(T_S14),.T_S15(T_S15),.T_S16(T_S1
6),.W_S1(W_S1),.W_S2(W_S2),.W_S3(W_S3),.W_S4(W_S4),.W_S5(W_S5),.W_S6(W_S6),.W_S
7(W_S7),.W_S8(W_S8),.W_S9(W_S9),.W_S10(W_S10),.W_S11(W_S11),.W_S12(W_S12),.W_S1
3(W_S13),.W_S14(W_S14),.W_S15(W_S15),.W_S16(W_S16),.buzzer(buzzer),.S_P_R(S_P_R),.S_
P_L(S_P_L),.S_P_R1(S_P_R1),.S_P_L1(S_P_L1),.basket_P1(basket_P1),.basket_P2(basket_P2),.b
asket_P3(basket_P3),.basket_P4(basket_P4),.basket_P5(basket_P5),.basket_P6(basket_P6),.basket_
P7(basket_P7),.basket_P8(basket_P8),.basket_P9(basket_P9),.basket_P10(basket_P10),.basket_P11(
basket_P11),.basket_P12(basket_P12),.basket_P13(basket_P13),.basket_P14(basket_P14),.basket_P
15(basket_P15));

 always #2 clk=~clk;

 initial begin   clk=1;

 #20   I_R_S1  = 1; front_w = 1000;   rearer_w = 700;/*false*/  T_S1=7'b0101110; /*temperature
46C*/

 #30   I_R_S2  = 1; T_S2=7'b0110011; /*temperature 51C*/

 #30   I_R_S1  = 0;  R_T_L1      T_S3=7'b0100011; /*temperature 35C*/    R_B_S = 1;

 #10   I_R_S3  = 1;  R_T_L2=0;     front_w = 1093;   rearer_w = 248;/*ok*/
```

#25 I_R_S2 = 0; R_T_L2=0;

#10I_R_S4 = T_S4=7'b0100011; /*temperature 35C*/ R_B_S = 0;

#40 I_R_S5 = 1;

#35 I_R_S3 = 0; R_T_L3=0 front_w = 987; rearer_w = 701;/*false*/ T_S5=7'b0011011; /*temperature 27C*/ R_B_S = 1;

#20 I_R_S13 = 1;

#35 I_R_S3 = 0; R_T_L3=0;

#20 I_R_S6 = 1; R_B_S = 0;

#50 I_R_S4 = 0; R_T_L4=0; T_S2=7'b0110000; /*temperature 48C*/ R_B_S = 1;

#40 I_R_S7 = 1; T_S6=7'b0011111; /*temperature 35C*/ R_B_S = 0;

#50 I_R_S8 = 0; R_T_L8=0; R_B_S = 1;

#10 I_R_S9 = 1; R_T_L9=1; enter_password9=4'b1000; front_w = 1577; rearer_w = 201;/*false*/ T_S7=7'b0110101; /*temperature 53C*/ R_B_S = 0;

#40 I_R_S10 = 0; R_T_L10=0; R_B_S = 1;

#50 I_R_S11 = 1; T_S8=7'b0101111; /*temperature 47C*/ R_B_S = 0;

#20 I_R_S13 = 0; R_T_L13= T_S7=7'b0100011; /*temperature 35C*/ R_B_S = 1;

#20 I_R_S12 = 0; R_T_L12=0; R_B_S = 1;

#50 I_R_S16 = 1; T_S9=7'b0100111; /*temperature 39C*/

#50 I_R_S2 = 1; front_w = 1121; rearer_w = 434;/*ok*/ R_B_S = 0;

#56 I_R_S14 = 0; R_T_L14=0; T_S10=7'b0110101; /*temperature 53C*/ R_B_S = 1;

#40 I_R_S7 = 0; R_T_L7=0; T_S11=7'b0100011; /*temperature 35C*/ R_B_S = 1;

#50 I_R_S15 = 1; T_S12=7'b0011101; /*temperature 29C*/ R_B_S = 0;

#34 I_R_S6 = 1; T_S10=7'b0101111; /*temperature 47C*/

#10 I_R_S5 = 0; R_T_L5=0; T_S13=7'b0101111; /*temperature 47C*/ R_B_S = 1;

#20 I_R_S15 = 1; T_S14=7'b0110010; /*temperature 50C*/ R_B_S = 0;

#20 I_R_S13 = 0; R_T_L13=1; enter_password13=4'b1100; front_w = 1654; rearer_w = 897; /*alse*/ basket_P13 = 10;

#50 I_R_S16 = 0; R_T_L16=0; T_S15=7'b0110100; /*temperature 52C*/

#150 basket_P13 = 0;

#50 I_R_S3 = 0; R_T_L3=1; enter_password3=4'b00 T_S14=7'b0100111; /*temperature 39C*/ basket_P3 = 4;

#10 I_R_S15 = 0; R_T_L15=0; T_S15=7'b0110000; /*temperature 48C*/

#200 basket_P3 = 0;

#50I_R_S8=0; R_T_L8=1; enter_password8=4'b0111; basket_P8 = 11;

#20 I_R_S13 = 1;

#12 I_R_S4 = 1; T_S16=7'b0110000; /*temperature 48C*/

#50 I_R_S16 = 1; front_w = 1121; rearer_w = 248; /*ok*/ T_S3=7'b0110101; /*temperature 53C*/ basket_P8 = 0;

#40 I_R_S7 = 0; R_T_L7=1; enter_password7=4'b0110; R_B_S = 1; basket_P7 = 8;

#20 I_R_S8 = 1; T_S16=7'b0110011; /*temperature 51C*/

#50 I_R_S15 = 1;

#50 I_R_S16 = 0; T_S3=7'b0100011; /*temperature 35C*/ R_B_S = 0; basket_P7 = 0;

#10 I_R_S5 = 0; R_T_L5=1; enter_password5=4'b0100; T_S16=7'b0100011; /*temperature 35C*/ basket_P5 = 13;

#34 I_R_S6 = 0; R_T_L6=0;

#100 basket_P5 = 0;

#50 I_R_S16 = 0; R_T_L16=1; enter_password16=4'b1111;

#40 I_R_S7 = 1; front_w = 3121; rearer_w = 148;/*false*/

#100

#50I_R_S1=0R_T_L1=1;enter_password1=4'b0000; basket_P1 = 6;

#200 basket_P1 = 0;

#50I_R_S2=0;R_T_L2=1;enter_password2=4'b1111; basket_P2 = 12;

#50 I_R_S3 = 0;

```
 #150                                                                basket_P2  = 0;
 #100I_R_S4=0;R_T_L4=1enter_password4=4'b0011;                        R_B_S = 1;
 #70   I_R_S5  = 0;
 #50   I_R_S6  = 0; R_T_L6=1;   enter_password6=4'b0101;
 #15   I_R_S7  = 0;
 #80   I_R_S8  = 0;                          R_B_S = 0;
 #50   I_R_S9  = 0; R_T_L9=0;
 #40   I_R_S10 = 0; R_T_L10=1;  enter_password10=4'b1001;
 #38   I_R_S11 = 0; R_T_L11=0;
 #70   I_R_S12 = 0; R_T_L12=1;  enter_password12=4'b1011;
 #50   I_R_S13 = 0;
 #50   I_R_S14 = 0;
 #50   I_R_S15 = 0;
 #100  I_R_S16 = 0;                                                  R_B_S = 1;
 #100  I_R_S14 = 0; R_T_L14 =1;   enter_password14=4'b1101;
 #100  I_R_S15 = 0; R_T_L15 =1;   enter_password15=4'b1110;    basket_P15  = 14;
 #100                                          basket_P15  = 0;          R_B_S = 0;
 #100  I_R_S11 = 0; R_T_L11 =1;   enter_password11=4'b1010;    basket_P11  = 2;
 #100                                          basket_P11  = 0;
 #100 $stop ;
 end
endmodule
```

Appendices C

This code for download on FPGA chip two baskets only

Top model of project (Rotary_Car_Parking)

```
module Rotary_Car_Parking1 (clk,empty_places,R_B_S,I_R_S1,I_R_S2,front_w,rearer_w,Instance,
R_T_L1,R_T_L2, enter_password1,enter_password2,payment1,payment2 /*,T_S1,T_S2, W_S1,

W_S2,buzzer1,buzzer2*/,S_P_R2,S_P_L2,S_P_R1,S_P_L1,S_P_R,S_P_L,basket_P1, basket_P2);

input clk;

input I_R_S1,I_R_S2;//Infrared sensor.          input R_T_L1,R_T_L2;// R_T_L = Recuest to leave.

//input [2:0]T_S1,T_S2;// Temperature sensor.     input[2:0] basket_P1, basket_P2; //basket position

input [1:0] enter_password1,enter_password2; //password entered by user to check.

input [1:0]front_w,rearer_w;

input R_B_S; //receiving basket sensor.

output Instance;

//output W_S1,W_S2,buzzer1,buzzer2; // Water switch.

output [2:0]empty_places;

output [5:0]payment1,payment2;

output S_P_R,S_P_R1,S_P_R2,S_P_L2,S_P_L1,S_P_L;//motor circling direction.

Car_lift_motor1 Car_lift_motor1 (.R_B_S(R_B_S),.S_P_R2(S_P_R2),.S_P_L2(S_P_L2));

Car_Count1 Car_Count1(.clk(clk),.I_R_S1(I_R_S1),.I_R_S2(I_R_S2),.empty_plaes(empty_places));

Car_Weigh1 Car_Weigh1 (.front_w(front_w),.rearer_w(rearer_w),.Instance(Instance));

Time_count1Time_count1(.clk(clk),.payment1(payment1),.payment2(payment2),.R_T_L1(R_T_L1)
,.R_T_L2(R_T_L2),.I_R_S1(I_R_S1),.I_R_S2(I_R_S2)   ,.enter_password1(enter_password1),.enter
_password2(enter_password2));

//Fire_Extinguishing1
Fire_Extinguishing1(.T_S1(T_S1),.T_S2(T_S2),.W_S1(W_S1),.W_S2(W_S2),.buzzer1(buzzer1),.b
uzzer2(buzzer2));

Shortest_Path1Shortest_Path1(.S_P_R(S_P_R),.S_P_L(S_P_L),.S_P_R1(S_P_R1),.

S_P_L1(S_P_L1),.R_T_L1(R_T_L1),.R_T_L2(R_T_L2),.I_R_S1(I_R_S1),.I_R_S2(I_R_S2),.

basket_P1(basket_P1)
```

```
                    ,.basket_P2(basket_P2),.enter_password1(enter_password1),.enter_password2(ent
er_password2));

endmodule
```

Sub module to count number of cars in system (Car_Count)

```
/*This code to compute number of cars in the system when a sensor read (0) ther are car in pasket if
read (1) the pasket empty*/

module Car_Count1(clk,empty_places,I_R_S1,I_R_S2);

input I_R_S1,I_R_S2;              input clk;

output reg[2:0] empty_places;

reg[1:0] Car_count;

reg current_state1,current_state2;

initial Car_count=2'b00;

initial current_state1  = 1;   initial current_state2  = 1;

always @ (negedge clk )
        begin
        if (Car_count >= 2'b11)
           begin
            $display ("No Empty Space");      Car_count = 2'b00;
           end
 ////////////////////////////////////////////////
        else if (I_R_S1 == 0 && current_state1 == 1)
                begin
                 Car_count = Car_count + 1'b1;      current_state1 = I_R_S1;
                end
            else if (I_R_S1 == 1 && current_state1 == 0 )
                    begin
                     Car_count = Car_count- 1'b1;    current_state1 = I_R_S1;
                    end
```

```
//////////////////////////////////////////////////////////////////
            else if (I_R_S2 == 0 && current_state2 == 1)
                begin
                 Car_count = Car_count + 1'b1;        current_state2 <= I_R_S2;
                end
              else if (I_R_S2 == 1 && current_state2 == 0 )
                  begin
                   Car_count = Car_count- 1'b1;        current_state2 <= I_R_S2;
                  end
                else  Car_count = Car_count;
               empty_places = 3'b111- Car_count;
      end
endmodule
```

Sub module to lift car cars when it enters the room (Car_lift_motor)

```
/* this code to turn motor if there are car in receiving basket*/
module Car_lift_motor1 (R_B_S,S_P_R2,S_P_L2);
input R_B_S; //receiving basket sensor.
output reg S_P_R2,S_P_L2;//motor circling direction.
reg [1:0]state_direction;
parameter [1:0] left=2'b00,right=2'b01;
initial S_P_R2=0 ;initial S_P_L2=0 ;initial state_direction = right;
always @(posedge R_B_S)
    begin
    if ( R_B_S == 1 && state_direction == right)//if there are car in basket.
       begin
```

```
          S_P_R2 =1;    state_direction <= right;
        end
    else if (R_B_S == 0 && state_direction == right)
          begin
           S_P_R2 =0;     state_direction <= left;
          end
       else if ( R_B_S == 1 && state_direction == left)//if there are car in basket.
          begin
           S_P_L2=1;     state_direction <= left;
          end
          else if (R_B_S == 0 && state_direction == left)
                begin
                 S_P_L2 =0;       state_direction <= right;
                 end         end
endmodule
```

Sub module to find out if the car is allowed to enter or not (Car_Weigh)

```
/*This code to compute a weigh of car if incepted or not*/
module Car_Weigh1(front_w,rearer_w,Instance);
input [1:0]front_w,rearer_w;
output reg Instance;
parameter front_condition =2;
parameter rearer_condition=1;
always @(*)
    begin
      if (front_w > front_condition || rearer_w > rearer_condition)
```

```
            begin
             Instance = 0;
            end
         else if (front_w <= front_condition && rearer_w <= rearer_condition)
                begin
                Instance = 1;
                end
             else Instance <= Instance;
       end
endmodule
```

Sub module to fire extinguishing (Fire_Extinguishing)

```
/*This program works as a fire extinguishing system, as it turn-on the pump
switch when the temperature sensor read temperature higher or equal to 50C*/
module Fire_Extinguishing1(T_S1,T_S2,W_S1,W_S2,buzzer1,buzzer2);
parameter temperature_rate =3'b110; //temperatur condition.
input [2:0]T_S1,T_S2;// Temperature sensor.
output reg W_S1,W_S2,buzzer1,buzzer2; // Water switch.
always @(T_S1)
begin
                if (T_S1 >= temperature_rate)
                   begin
                   W_S1= 1;   buzzer1 = 1;
                   end
                else
```

```
                begin
                 W_S1= 0;        buzzer1 = 0;
                end     end
always @(T_S2)
begin
            if (T_S2 >= temperature_rate)
                begin
                W_S2= 1;     buzzer2 = 1;
                end
            else
                begin
                 W_S2= 0;       buzzer2 = 0;
                end       end
endmodule
```

Sub module to compute shortest path to receive a car (Shortest_Path)

```
/* following code to compute shortest path to receive a car*/
/* حيكون فيه حساس في كل سلة يحدد مكان السلة والسلات مرقمات بالشكل التالي*/
                  8
                7    9
                6    10
                5    11
                4    12
                3    13
                2    14
                1    15
                  0
```

```
*/


moduleShortest_Path1(R_T_L1,R_T_L2,basket_P1,basket_P2,enter_password1,enter
_password2,I_R_S1 ,I_R_S2,S_P_R,S_P_L,S_P_R1,S_P_L1);

parameter password1=2'b00 ,password2=2'b01;

input I_R_S1,I_R_S2;// Infrared sensors

input R_T_L1,R_T_L2;// R_T_L = Recuest to leave.

input[2:0] basket_P1, basket_P2; //basket position

input [1:0] enter_password1,enter_password2; //password entered by user to check

 output reg S_P_R,S_P_L,S_P_R1,S_P_L1;//motor circling direction.

always @(*)

 begin

 if (R_T_L1 == 1 && I_R_S1 == 0 && ( password1 == enter_password1)&&(basket_P1 >= 1 && basket_P1 <=4)) //if receive recuest to leave and there are car.

        begin

          S_P_L <= 1; //motor circling left direction.     S_P_R <= 0;

        end

  else if (basket_P1 == 0)// when a basket reraches the receving location,the motor stops.

            begin

             S_P_L <= 0;        S_P_R <= 0;

            end

         else if (R_T_L1 == 1 && I_R_S1 == 0 && ( password1 == enter_password1)&&(basket_P1 > 4 && basket_P1 <= 7))

                begin
```

```
                S_P_R <= 1; //motor circling right direction.        S_P_L <= 0;
              end
  else if (basket_P1 == 0) // when a basket reraches the receving location,the motor stops.
                begin
                 S_P_R <= 0;         S_P_L <= 0;
                end
  else if (R_T_L1 == 1 &&(I_R_S1 == 1 || ( password1 != enter_password1)))
        begin
          S_P_R <= 0;      S_P_L <= 0;
          $display ("No car,or entered error code ??");
        end              end
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
always @(*)
 begin
  if (R_T_L2 == 1 && I_R_S2 == 0 && ( password2 == enter_password2)&&(basket_P2 >= 1 && basket_P2 <=4)) //if receive recuest to leave and there are car.
        begin
          S_P_L1 <= 1; //motor circling left direction.      S_P_R1 <= 0;
        end
  else if (basket_P2 == 0)// when a basket reraches the receving location,the motor stops.
            begin
             S_P_L1 <= 0;         S_P_R1 <= 0;
            end
```

```
        else if (R_T_L2 == 1 && I_R_S2 == 0 && ( password2 == enter_password2)&&(basket_P2 > 4 && basket_P2 <= 7))
                begin
                 S_P_R1 <= 1; //motor circling right direction.     S_P_L1 <= 0;
                end
  else if (basket_P2 == 0) // when a basket reraches the receving location,the motor stops.
                    begin
                     S_P_R1 <= 0;          S_P_L1 <= 0;
                    end
  else if (R_T_L2 == 1 &&(I_R_S2 == 1 || ( password2 != enter_password2)))
          begin
            S_P_R1 <= 0;         S_P_L1 <= 0;
            $display ("No car,or entered error code ??");
          end          end
endmodule
```

Sub module to compute a payment when a car stay in system (Time_count)

```
/* This code to compute a payment when a car stay in system */
/*
  I_R_S        R_T_L       state
    0          0        T_count++
    0          1        compute payment
    1          0        dont care
    1          1        there is no car
*/
```

```
module Time_count1 (clk,payment1,payment2,I_R_S1,I_R_S2,R_T_L1,R_T_L2,enter_password1,enter_password2);

parameter payment_rate = 4'b1010;// Payment per hour.

parameter password1=2'b00 ,password2=2'b01;

input clk;  input I_R_S1,I_R_S2;// Infrared sensors

input R_T_L1,R_T_L2;// R_T_L = Recuest to leave.

input [1:0] enter_password1,enter_password2; //password entered by user to check.

output reg [5:0]payment1,payment2;

reg[5:0]one_minute_counter1,one_houre_counter1,one_minute_counter2,one_houre_counter2;
//Regester to count time when car stay in parking system.

reg[3:0]houre1,houre2;//regester to count number of hour.

reg [26:0]one_second_counter1,one_second_counter2;

initial payment1 =6'b000000; initial one_houre_counter1=6'b000000; initial one_second_counter1=27'b000000000000000000000000000;

initial one_minute_counter1=6'b000000;  initial houre1= 4'b0000;

initial payment2 =6'b000000; initial one_houre_counter2=6'b000000; initial one_second_counter2=27'b000000000000000000000000000;

initial one_minute_counter2=6'b000000;  initial houre2= 4'b0000;

wire one_second_enable1;// one second enable for counting numbers

wire one_minute_enable1;

wire one_houre_enable1;

wire one_second_enable2;// one second enable for counting numbers

wire one_minute_enable2;

wire one_houre_enable2;

always @(posedge clk )

   begin

    if (I_R_S1 ==0)

        begin
```

```
          if (one_second_counter1>=10) one_second_counter1 <= 0;
          else one_second_counter1 <= one_second_counter1 + 1;
         end      end
assign one_second_enable1 = (one_second_counter1==10)?1:0;
always @(posedge clk)
   begin
      if (one_second_enable1==1) one_minute_counter1 <= one_minute_counter1 + 1;
      else if (one_minute_counter1>=5) one_minute_counter1 <= 0;
   end
assign one_minute_enable1 = (one_minute_counter1==5)?1:0;
always @(posedge clk)
   begin
      if (one_minute_enable1==1) one_houre_counter1 <= one_houre_counter1 + 1;
      else if (one_houre_counter1>=5) one_houre_counter1 <= 0;
   end
assign one_houre_enable1 = (one_houre_counter1==5)?1:0;
always @( * )
        begin
        if (one_houre_enable1 == 1) houre1= houre1+1;
         if (R_T_L1 ==1 &&(enter_password1 == password1))
               begin
               payment1 = houre1 * payment_rate;
               end

         else if (R_T_L1 ==1 &&(enter_password1 != password1))
                begin
                $display ("no car");
```

```
                end         end
////////////////////////////////////////////////////////////////////////////////////////////////
always @(posedge clk )
   begin
     if (I_R_S2 ==0)
         begin
          if (one_second_counter2>=10) one_second_counter2 <= 0;
          else one_second_counter2 <= one_second_counter2 + 1;
         end      end
assign one_second_enable2 = (one_second_counter2==10)?1:0;
always @(posedge clk)
   begin
       if (one_second_enable2==1) one_minute_counter2 <= one_minute_counter2 + 1;
       else if (one_minute_counter2>=5) one_minute_counter2 <= 0;
   end
assign one_minute_enable2 = (one_minute_counter2==5)?1:0;
always @(posedge clk)
   begin
       if (one_minute_enable2==1) one_houre_counter2 <= one_houre_counter2 + 1;
       else if (one_houre_counter2>=5) one_houre_counter2 <= 0;
   end
assign one_houre_enable2 = (one_houre_counter2==5)?1:0;
always @( * )
        begin
        if (one_houre_enable2 == 1) houre2= houre2+1;
         if (R_T_L2 ==1 &&(enter_password2 == password2))
               begin
```

```
            payment2 = houre2 * payment_rate;
            end
        else if (R_T_L2 ==1 &&(enter_password2 != password2))
            begin
            $display ("no car");
            end
        end
endmodule
```

Test bench code

```
`timescale 1ns/1ns
module Test1;
reg clk;            reg I_R_S1,I_R_S2;
reg R_T_L1,R_T_L2;// R_T_L = Recuest to leave.
reg [1:0]front_w,rearer_w;
//reg [2:0]T_S1,T_S2;// Temperature sensor.
reg [1:0] enter_password1,enter_password2;
reg [2:0] basket_P1, basket_P2; //basket position
reg R_B_S;
wire S_P_R1,S_P_L1,S_P_R2,S_P_L2;//motor circling direction.
wire S_P_R,S_P_L;//motor circling direction.
wire Instance;
wire [2:0] empty_places;
//wire W_S1,W_S2,buzzer; // Water switch.
wire [5:0]payment1,payment2;
Rotary_Car_Parking1 uut(.clk(clk),.R_B_S(R_B_S),.I_R_S1(I_R_S1),.I_R_S2(I_R_S2),.
```

```
empty_places(empty_places),.front_w(front_w),.rearer_w(rearer_w),.Instance(Instance),.payment1(payment1),.payment2(payment2),.R_T_L1(R_T_L1),.R_T_L2(R_T_L2),.enter_password1(enter_password1),.enter_password2(enter_password2)/*,.T_S1(T_S1),.T_S2(T_S2),.W_S1(W_S1),.W_S2(W_S2),.buzzer(buzzer)*/,.S_P_R(S_P_R),.S_P_L(S_P_L),.S_P_R1(S_P_R1),.S_P_L1(S_P_L1),.

S_P_R2(S_P_R2),.S_P_L2(S_P_L2),.basket_P1(basket_P1),.basket_P2(basket_P2));

 always #2 clk=~clk;

  initial  begin

 clk=1;

 #100  I_R_S1  =  1;                                front_w  =  2;      rearer_w  =  2;/*false*/ //T_S1=3'b010;

 #100  I_R_S2  = 0;  R_T_L2=0;                                                                  //T_S1=3'b111;

 #150  I_R_S1  =  0;  R_T_L1=0;                                                                  R_B_S  =  1; //T_S1=3'b101;

 #100  I_R_S2  = 0;            front_w = 1;   rearer_w = 1;/*ok*/

 #100  I_R_S1  = 0;  R_T_L1=1;  enter_password1=2'b00;                                  basket_P1 = 7;

 #100  I_R_S1  = 0;                                R_B_S = 0;                  //T_S2=3'b011;

 #100  I_R_S1  = 0;

 #100  I_R_S1  = 1;  R_T_L1=0;   front_w = 3;   rearer_w = 2;/*false*/  R_B_S = 0;  basket_P1 = 0;

 #100  I_R_S2  = 0;

 #100  I_R_S2  = 0;  R_T_L2=0;

 #100  I_R_S1  = 1;                                R_B_S = 0;

 #100  I_R_S2  = 0;  R_T_L2=1;  enter_password2=2'b01;                  R_B_S = 1;  basket_P2 = 3;

 #100  I_R_S1  = 0;                                R_B_S = 0;                  //T_S2=3'b110;

 #100  I_R_S2  = 0;                                R_B_S = 1;                  //T_S1=3'b011;

 #100  I_R_S1  = 0;                  front_w  =  2;      rearer_w  =  3;/*false*/      R_B_S  =  0; //T_S2=3'b011;

 #140  I_R_S2  = 1;  R_T_L2=0;                                 R_B_S = 0;  basket_P2 = 0;

 #100  I_R_S2  = 1;                                R_B_S = 0;

 #120  I_R_S1  =  0;      R_B_S=1;
```

```
 #100  I_R_S1  = 0;  R_T_L1=0;                                                R_B_S = 1;
//T_S1=3'b101;

 #150  I_R_S1  = 1;                                                          //T_S1=3'b111;

 #100  I_R_S2  = 1;          front_w = 2;   rearer_w = 1;/*ok*/     R_B_S = 0;

 #100  I_R_S2  = 0;  R_T_L2=1;  enter_password2=2'b11;                       R_B_S = 1;
//T_S2=3'b101;

 #140  I_R_S1  = 0;  R_T_L1=1;  enter_password1=2'b00;    R_B_S = 1;  basket_P1 = 3;
//T_S1=3'b011;

 #100  I_R_S1  = 0;                                  R_B_S = 0;          //T_S2=3'b011;

 #100  I_R_S1  = 0;

 #100  I_R_S1  = 1;  R_T_L1=0;   front_w = 3;   rearer_w = 2;/*false*/  R_B_S = 0;  basket_P1 =
0;

 #100  I_R_S2  = 1;                                  R_B_S = 0;          //T_S1=3'b001;

 #134  I_R_S2  = 0;  R_T_L2=1;  enter_password2=2'b01;                       basket_P2 = 5;
//T_S2=3'b111;

 #100  I_R_S1  = 0;                                  R_B_S = 0;

 #120  I_R_S1  = 0;

 #140  I_R_S2  = 1;  R_T_L2=0;                          R_B_S = 0;  basket_P2 = 0;

 #100 $stop ;

 end

endmodule
```