Project on

Automated Car Parking System

Spring 2023

Course Code: CSE 326, Sec: 3

Course Title: Computer Peripherals and Interfacing Lab

Submitted to
Tazen Tasneem

Assistant professor of CSE Dept.

Eastern University

Submitted By
Name: Mst Esrat Jahan Refa
Id: 201400021
Name : Md Riduan Sakib
Id: 201400023
Name: Mst Iftari Khatun
Id: 211410004

Department of Computer Science and Engineering
Faculty of Engineering and Technology
Eastern University

Table of Content

Content		Page
Chapter 1: Introduction		
1.1	Introduction --	
1.3	Objective ---	
1.5	Expected Outcome ---	
Chapter 2: Methodology and tools		
2.1	Methodology---	
2.2	Tools--	
Chapter 3: Design and Implementation		
3.1	Use Case/ Block Diagram of the System --	
3.2	Implementation (Code) ---	
Chapter 4: Result and Conclusion		
4.1	Result (screen shoot)--	
4.2	Conclusion ---	

Chapter 1
Introduction

1.1 Introduction: The Automated Car Parking System is a project that aims to streamline and automate the process of parking and retrieving cars in a parking facility. Traditional parking systems often involve manual ticketing, searching for parking spaces, and potential human errors. By implementing an automated system, the project aims to enhance efficiency, optimize space utilization, and improve the overall parking experience. This project involves the use of various technologies such as sensors, cameras, automation systems, and software algorithms to manage and control the parking process. The system tracks available parking spaces, guides drivers to vacant spots, and automates the entry and exit procedures. It also provides real-time information about parking availability to drivers, reducing congestion and saving time.

1.2 Objectives:

1. Efficient Space Utilization: The automated car parking system aims to optimize the utilization of available parking spaces by accurately tracking occupancy and efficiently allocating parking slots.

2. Enhanced User Experience: The project aims to provide a seamless and user-friendly experience for drivers. It includes features such as automated entry and exit procedures, clear signage, real-time information on available parking spaces, and convenient payment options.

3. Improved Security and Safety: The automated system enhances security by monitoring parking areas using cameras and sensors. It can detect unauthorized access, suspicious activities, or parking violations, thereby ensuring a safer environment for vehicles and individuals.

4. Time and Cost Efficiency: By automating the parking process, the system reduces the time spent in finding parking spaces, eliminates the need for manual ticketing, and minimizes human errors. This leads to cost savings, increased operational efficiency, and improved traffic flow.

5. Integration and Scalability: The project aims to design a system that can be easily integrated with existing parking facilities and scaled up to accommodate larger parking areas or multiple locations. It should provide a modular and adaptable solution.

1.3 Expected Outcome:

1. Efficient Parking Space Utilization: The automated system should effectively utilize available parking spaces by accurately detecting and allocating parking slots. It should optimize space allocation to maximize the number of vehicles that can be accommodated.

2. Smooth Entry and Exit Process: The system should provide a seamless and user-friendly experience for drivers entering and exiting the parking facility. It should automate the process, eliminating the need for manual ticketing or validation.

3. Real-time Parking Availability Information: The system should provide real-time information to drivers about the availability of parking spaces. This can be displayed on electronic signage or communicated through mobile applications or on-site kiosks.

4. Secure Vehicle Storage: The automated system should ensure the security and safety of parked vehicles. It should incorporate measures such as surveillance cameras, access control systems, and alarms to prevent theft, vandalism, or unauthorized access.

5. Efficient Traffic Flow: The system should optimize the flow of traffic within the parking facility. It should provide clear signage, guidance systems, and designated paths to direct drivers to available parking spaces and minimize congestion.

6. Integration with Payment Systems: The automated system should seamlessly integrate with payment systems to facilitate convenient and secure transactions for parking fees. This can include options such as ticketless payments, cashless methods, or integration with mobile payment applications.

7. Scalability and Expansion: The hardware project should be designed to be scalable and easily expandable to accommodate larger parking facilities or future expansions. The system should have the capability to handle increased traffic and additional parking spaces.

8. Maintenance and Reliability: The hardware components of the system should be reliable, durable, and require minimal maintenance. They should be able to withstand environmental factors such as weather conditions and heavy usage.

9. Data Analytics and Reporting: The automated system should capture relevant data, such as parking occupancy rates, usage patterns, and revenue. It should provide analytics and reporting capabilities to help parking facility operators make informed decisions for optimization and planning.

10. Energy Efficiency: The hardware components should be designed to minimize energy consumption, contributing to sustainability and cost savings.

By achieving these expected outcomes, the Automated Car Parking System hardware project can successfully deliver an efficient, user-friendly, and secure parking solution that enhances the overall parking experience for users while maximizing space utilization and operational efficiency.

Chapter 2
Methodology and Tools

2.1 Methodology:

Developing a peripheral hardware project for an automated car parking system requires careful planning, implementation, and testing. Here is a suggested methodology to guide you through the process:

1. **Define Project Requirements: Clearly define the requirements of the automated car parking system. Consider factors such as the number of parking spaces, types of vehicles to accommodate, entry and exit processes, security measures, scalability, and integration with existing systems.**

2. **Conduct Feasibility Study: Assess the feasibility of the project by evaluating technical, economic, and operational factors. Identify potential challenges and constraints and determine the viability of implementing the automated car parking system.**

3. **Design Hardware Architecture: Develop the hardware architecture for the automated car parking system. Determine the components needed, such as sensors, cameras, entry and exit gates, signage, control systems, and communication infrastructure. Consider factors like power requirements, connectivity options, and physical layout.**

4. **Select and Procure Hardware Components: Identify and select appropriate hardware components based on the project requirements and design. Consider factors like reliability, compatibility, durability, and cost-effectiveness. Procure the necessary hardware components from reliable suppliers.**

5. **Prototype Development: Create a prototype of the automated car parking system using the selected hardware components. This includes assembling the hardware, integrating sensors and cameras, developing control mechanisms, and connecting the various components.**

6. **Software Integration: Integrate the hardware components with the software systems required for the automated car parking system. This may include developing firmware or drivers to enable communication between the hardware and software components. Ensure compatibility and seamless integration.**

7. **Testing and Validation: Conduct thorough testing and validation of the hardware system. This involves checking the functionality of each component, verifying the accuracy of sensors and cameras, testing the entry and exit processes, and ensuring proper communication between hardware and software. Address any issues or bugs identified during testing.**

8. **Integration with Overall System: Integrate the peripheral hardware project with the overall automated car parking system. Ensure that the hardware interacts seamlessly with the central control system, payment systems, data analytics, and other subsystems.**

9. **User Acceptance Testing: Engage end-users or stakeholders in user acceptance testing to validate the functionality and usability of the automated car parking system. Gather feedback and make necessary improvements or adjustments based on user inputs.**

10. **Documentation and Maintenance: Create comprehensive documentation that includes hardware specifications, wiring diagrams, installation procedures, user manuals, troubleshooting guides, and maintenance instructions. Regularly monitor and maintain the hardware system to ensure its continued efficiency and reliability.**

Following this methodology will help you navigate the various stages of developing a peripheral hardware project for an automated car parking system. It ensures a systematic approach, enhances the chances of success, and facilitates a smooth implementation of the project.

2.1 Tools:

To develop a peripheral hardware project for an automated car parking system, I needed a combination of some hardware and software tools. Here are some commonly used tools for such projects:

1. **Microcontrollers: Microcontrollers are essential components for controlling and interfacing with hardware devices in the automated car parking system. Popular microcontroller platforms include Arduino, Raspberry Pi, and STM32.**

2. **Sensors and Actuators: Various sensors and actuators are required for the automated car parking system, such as proximity sensors, ultrasonic sensors, infrared sensors, motor drivers, and solenoids.**

Depending on the specific requirements of your project, you will need to select and integrate appropriate sensors and actuators.

3. **Integrated Development Environment (IDE): An IDE is necessary for programming and debugging the microcontroller. The choice of IDE depends on the microcontroller platform you are using. For Arduino, the Arduino IDE is commonly used, while platforms like Raspberry Pi may use Python development environments like Thonny or PyCharm.**

4. **Circuit Design and Simulation Tools: To design the electronic circuits for the hardware components, tools like Autodesk EAGLE, KiCad, or Altium Designer can be used. These tools allow you to create schematics, design PCB layouts, and simulate the circuit to ensure proper functionality.**

5. **Communication Protocols: Depending on the requirements, you may need to implement communication protocols such as UART, I2C, SPI, or CAN bus for inter-device communication. Tools like PuTTY, Serial Monitor, or Logic Analyzer help monitor and debug communication between devices.**

6. **3D Modeling and Design Tools: If your project involves designing custom enclosures or mounts for the hardware components, 3D modeling and design tools like AutoCAD, SolidWorks, or Fusion 360 can be used to create accurate 3D models.**

7. **Power Supply Design: Designing an efficient power supply system is crucial for the hardware project. Tools like LTspice or PSpice can help simulate and optimize power supply circuits.**

8. **Testing and Debugging Equipment: Basic tools like a multimeter, oscilloscope, logic analyzer, and soldering equipment are necessary for testing, troubleshooting, and debugging hardware components.**

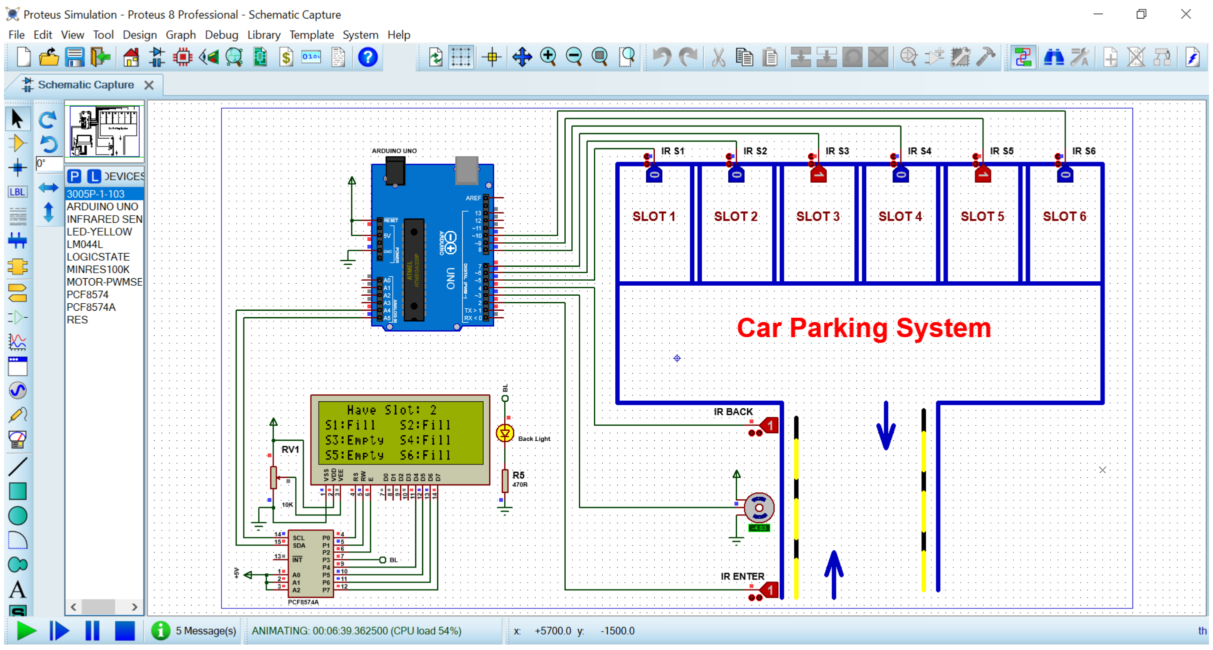
9. **Version Control System: Using a version control system like Git enables proper management of source code, allowing for collaboration, versioning, and tracking changes made to the software part of the project.**

10. **Project Management and Collaboration Tools: Tools like Jira, Trello, or Asana can help manage and track project tasks, assign responsibilities, and monitor progress. Collaboration tools like Slack or Microsoft Teams facilitate communication among team members.**

Remember to choose tools that align with specific project requirements and hardware platforms. The selection of tools may vary based on the complexity and scope of automated car parking system project.

Chapter 3
Design and Implementation

3.1 Use Case/ Block Diagram of the System:



```
  Have Slot: 2
S1:Fill   S2:Fill
S3:Empty  S4:Fill
S5:Empty  S6:Fill
```

3.2 Implementation:

The implementation of the Automated Car Parking System project involves several components and technologies. It typically includes:

1.Sensor and Camera Infrastructure: Sensors and cameras are deployed throughout the parking area to monitor occupancy, detect vehicles, and capture relevant data. They provide real-time information about available parking spaces.

2.Automated Entry and Exit Gates: The system incorporates automated entry and exit gates that are controlled by software algorithms. These gates allow authorized vehicles to enter or exit the parking facility smoothly.

3.Central Control System: A central control system manages and monitors the entire parking operation. It collects data from sensors and cameras, analyzes parking availability, and controls the automated gates. It may include a user interface for operators to monitor and manage the system.

4.Payment and Ticketing Integration: The project may involve integrating payment systems, such as ticket dispensers or cashless payment options, to facilitate easy and secure transactions for parking fees.

5.Software Algorithms and Data Analytics: The system utilizes software algorithms to process sensor data, track occupancy, optimize parking space allocation, and generate real-time reports on parking availability. Data analytics techniques can be applied to gain insights into usage patterns, demand forecasting, and operational efficiency.

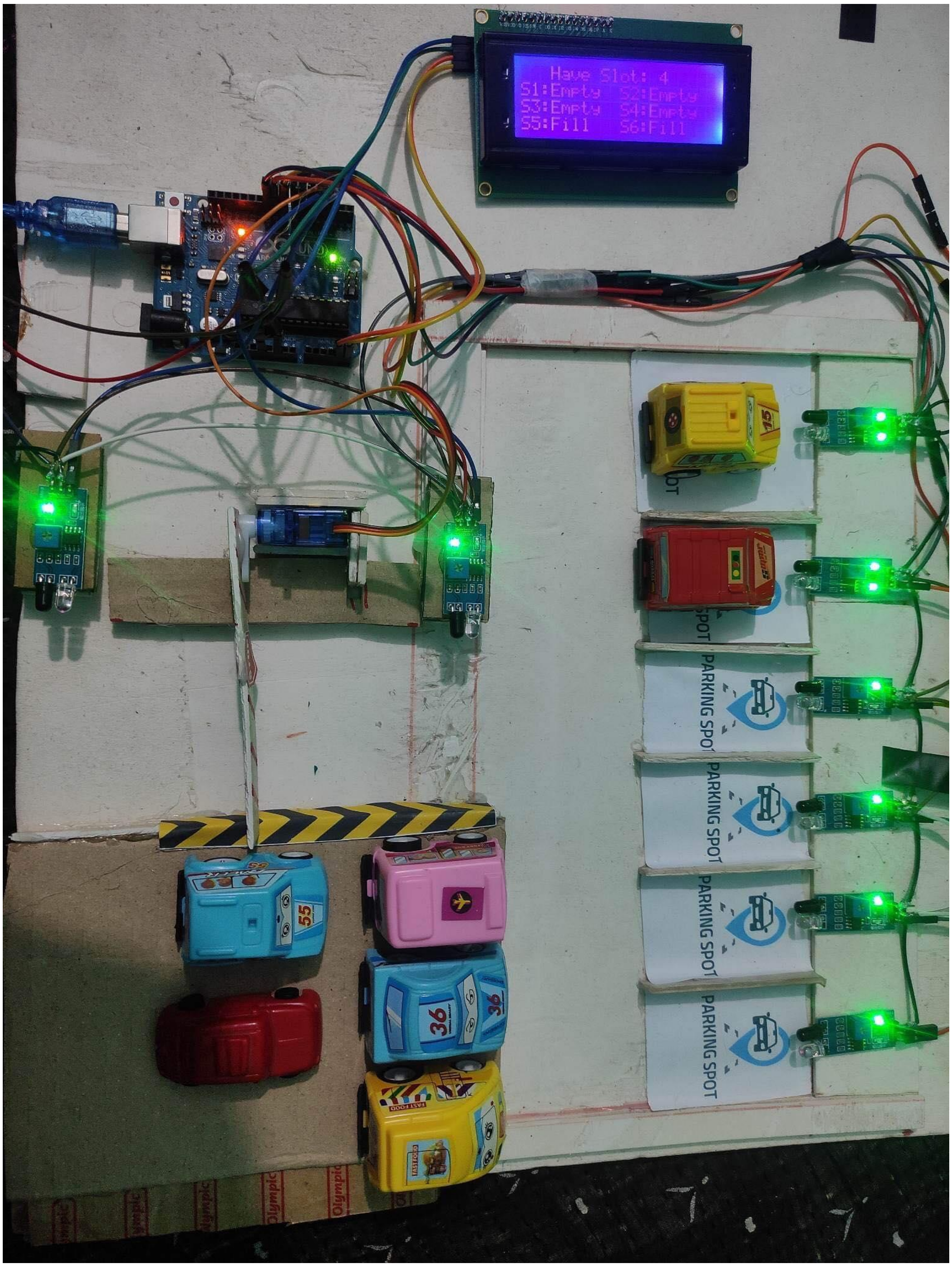
CODE:

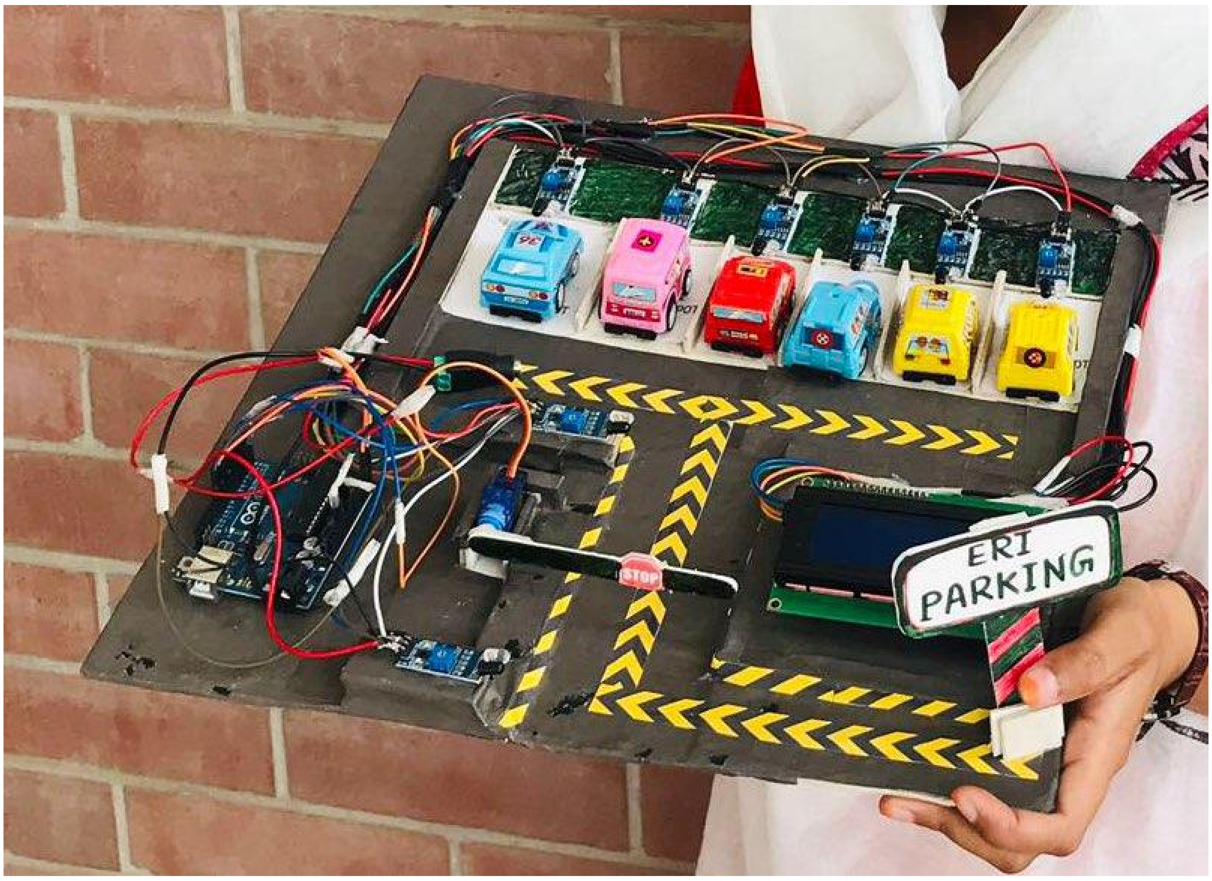
```
#include <LCD.h>
#include <LiquidCrystal.h>
#include <LiquidCrystal_I2C.h>

#include <LiquidCrystal_SI2C.h>


#include <Servo.h> //includes the servo library
#include <Wire.h>

LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);

Servo myservo;

#define ir_enter 2
#define ir_back  4

#define ir_car1 5
#define ir_car2 6
#define ir_car3 7
#define ir_car4 8
#define ir_car5 9
#define ir_car6 10

int S1=0, S2=0, S3=0, S4=0, S5=0, S6=0;
int flag1=0, flag2=0;
int slot = 6;

void setup(){
Serial.begin(9600);

pinMode(ir_car1, INPUT);
pinMode(ir_car2, INPUT);
pinMode(ir_car3, INPUT);
pinMode(ir_car4, INPUT);
pinMode(ir_car5, INPUT);
pinMode(ir_car6, INPUT);

pinMode(ir_enter, INPUT);
pinMode(ir_back, INPUT);

myservo.attach(3);
myservo.write(90);

lcd.begin(20, 4);
lcd.setCursor (0,1);
lcd.print("   Refa's Car  parking  ");
lcd.setCursor (0,2);
lcd.print("      System     ");
delay (2000);
lcd.clear();

Read_Sensor();

int total = S1+S2+S3+S4+S5+S6;
slot = slot-total;
}

void loop(){

Read_Sensor();

lcd.setCursor (0,0);
lcd.print("   Have Slot: ");
lcd.print(slot);
lcd.print("    ");

lcd.setCursor (0,1);
if(S1==1){lcd.print("S1:Fill ");}
    else{lcd.print("S1:Empty");}

lcd.setCursor (10,1);
if(S2==1){lcd.print("S2:Fill ");}
    else{lcd.print("S2:Empty");}

lcd.setCursor (0,2);
if(S3==1){lcd.print("S3:Fill ");}
    else{lcd.print("S3:Empty");}

lcd.setCursor (10,2);
if(S4==1){lcd.print("S4:Fill ");}
    else{lcd.print("S4:Empty");}

 lcd.setCursor (0,3);
if(S5==1){lcd.print("S5:Fill ");}
    else{lcd.print("S5:Empty");}

lcd.setCursor (10,3);
if(S6==1){lcd.print("S6:Fill ");}
    else{lcd.print("S6:Empty");}

if(digitalRead (ir_enter) == 0 && flag1==0){
if(slot>0){flag1=1;
if(flag2==0){myservo.write(180); slot = slot-1;}
}else{
lcd.setCursor (0,0);
lcd.print(" NO SPACE FOR PARK ");
delay(1500);
}
}

if(digitalRead (ir_back) == 0 && flag2==0){flag2=1;
if(flag1==0){myservo.write(180); slot = slot+1;}
}
```

```
if(flag1==1 && flag2==1){
delay (1000);
myservo.write(90);
flag1=0, flag2=0;
}

delay(1);
}

void Read_Sensor(){
S1=0, S2=0, S3=0, S4=0, S5=0, S6=0;

if(digitalRead(ir_car1) == 0){S1=1;}
if(digitalRead(ir_car2) == 0){S2=1;}
if(digitalRead(ir_car3) == 0){S3=1;}
if(digitalRead(ir_car4) == 0){S4=1;}
if(digitalRead(ir_car5) == 0){S5=1;}
if(digitalRead(ir_car6) == 0){S6=1;}
```

Chapter 4
Result and Conclusion

4.1 Result:


Have Slot: 4
S1:Empty S2:Empty
S3:Empty S4:Empty
S5:Fill S6:Fill
PARKING SPOT
PARKING SPOT
PARKING SPOT
PARKING SPOT



4.2 Conclusion:

The implementation of an automated car parking system offers numerous advantages, including efficient space utilization, enhanced user experience, improved security, and cost savings. By integrating hardware components such as sensors, cameras, entry and exit gates, and software algorithms, the system streamlines the parking process, reduces congestion, and provides real-time information on parking availability.

Automated car parking systems provide convenience for drivers by automating entry and exit procedures, guiding them to available parking spaces, and offering seamless payment options. They also enhance security through surveillance cameras, access control systems, and alarms to ensure the safety of parked vehicles.
Additionally, the scalability and integration capabilities of these systems enable easy expansion and integration with existing parking facilities or future developments. They can be customized to accommodate different types of vehicles, handle increased traffic, and provide flexible payment options. The implementation of an automated car parking system requires careful planning, including defining project requirements, conducting feasibility studies, designing hardware architecture, selecting appropriate components, and integrating software systems. Thorough testing, user

acceptance testing, and documentation ensure the system's functionality, reliability, and ease of use.
Overall, an automated car parking system significantly improves the parking experience for both drivers and parking facility operators. It maximizes parking space utilization, reduces congestion, enhances security, and simplifies the payment process. As technology advances, automated car parking systems continue to evolve, offering innovative features and contributing to the efficient management of parking facilities.