

GOVERNMENT COLLEGE OF ENGINEERING, ERODE



அரசினர் பொறியியல் கல்லூரி, ஈரோடு
Government College of Engineering, Erode
(Approved by AICTE, New Delhi and Affiliated to Anna University, Chennai)



B.E Electronics and Communication Engineering

SMART PARKING

Done By

TEAM LEADER: HARIKUMAR S	731121106305
TEAM MEMBER: SADAI ESWARAN M	731121106039
TEAM MEMBER: MEYYARASAN B	731121106307
TEAM MEMBER: SIVABALAN R	731121106308

Under the mentor of **Dr.M.SATHYAKALA**

Department of Information Technology (IT)

Department of Electronics and Communication Engineering.

Government College of Engineering

Erode ,PO ,near Vasavi College,TamilNadu-638316, Affiliated to Anna University ,Chennai.





Introduction

In these modern days finding car parking is a big issue in congested cities. There are too many vehicles on the road but not enough parking spaces. One of the biggest problems is when we enter a parking area then we realize that there are no empty parking slots to park our cars. Important time. Another biggest problem is after entering in a big parking area we confused to find the empty parking slot to park our car. Sometimes maybe we all have been facing these two problems that wasted our important time. That's why we need efficient parking management systems in all parking areas that will provide confusion-free and easy parking.

In this tutorial, we will design a “Smart Parking System Project” to overcome this problem. This project helps the car's driver to park their car with minimum wastage of time with accurate information of the availability of the space to park.

Smart Parking System Project Concept:

This smart parking system project consists of

-  Arduino
-  Six IR sensor
-  One servo motor and
-  One LCD display

Where the Arduino is the main microcontroller that controls the whole system. Two IR sensors are used at the entry and exit gates to detect vehicle entry and exit in the parking area. And other four IR sensors are used to detect the parking slot availability. The servo motor is placed at the entry and exit gate that is used to open and close the gates. Also, an LCD display is placed at the entrance, which is used to show the availability of parking slots in the parking area.

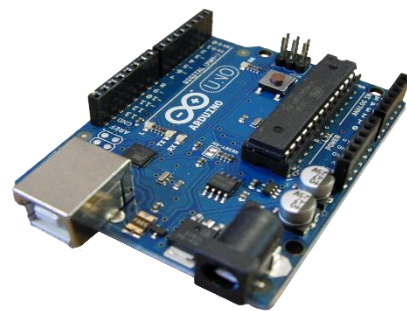
When a vehicle arrives at the gate of the parking area, the display continuously shows the number of empty slots. If there have any empty slots then the system opens the entry gate by the servo motor. After entering the car into the parking area, when it will occupy a slot, then the display shows this slot is full

Components in this project

- Arduino Uno
- IR Sensor
- Sg90 Servo Motor
- I2C LCD (LiquidCrystal_I2C.h)

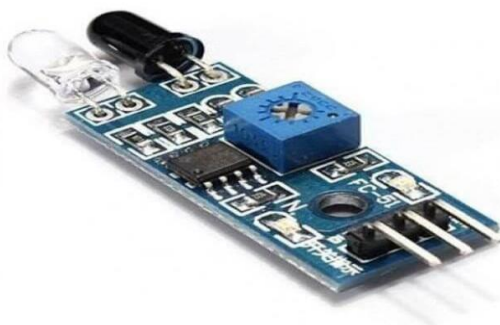
Arduino uno:

The heart of the project is an Arduino board (e.g., Arduino Uno or similar) which serves as the microcontroller to control and coordinate all other components.



IR Sensor:

IR sensors are used for detecting the presence of cars in parking slots. In this project, there are four IR sensors (ir_car1, ir_car2, ir_car3, and ir_car4) for monitoring individual parking slots, and two additional IR sensors (IR enter and IR back) for detecting cars entering and leaving the parking area.



Sg90 Servo Motor:



A servo motor (connected to pin D9 in this sketch) is used to simulate the gate/barrier in the parking system. It can be positioned at different angles to represent an open or closed gate.

I2C LCD:

Liquid Crystal Display (LCD) with I2C Interface A 20x4 character LCD is used to display information about the parking system's status. An I2C interface is employed to simplify the connection and control of the LCD.



Arduino code

```
#include <Servo.h> //includes the servo library
#include <Wire.h>
#include <LiquidCrystal_I2C.h> //includes LiquidCrystal_I2C library
LiquidCrystal_I2C lcd (0x27, 20, 4);
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
int S1=0, S2=0, S3=0, S4=0;
int flag1=0, flag2=0;
int slot = 6;
void setup ()
{
  Serial.begin(9600);
  // initialize digital pins as input.
  pinMode (ir_car1, INPUT);
  pinMode (ir_car2, INPUT);
  pinMode (ir_car3, INPUT);
  pinMode (ir_car4, INPUT);
  pinMode (ir_enter, INPUT);
  pinMode (ir_back, INPUT);
  myservo. Attach (9); // Servo motor pin connected to D9
  myservo. Write (90); // sets the servo at 0 degree position
  // Print text on display
  lcd.begin(20, 4);
  lcd.setCursor (0,1);
  lcd.print("  Smart Car  ");
  lcd.setCursor (0,2);
  lcd.print(" Parking System ");
  delay (2000);
  lcd.clear();
  Read_Sensor();
  int total = S1+S2+S3+S4;
  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");
}

/* Servo Motor Control
*****/

if(digitalRead (ir_enter) == 0 && flag1==0) // read degital data from IR
sensor1
{
    if(slot>0)
    {
        flag1=1;
        if(flag2==0)
        {
            myservo.write(180);
            slot = slot-1;
        }
    }
}

```

```

    }
else
{
    lcd.setCursor (0,0);
    lcd. Print(" Sorry Parking Full ");
    delay(1500);
}
}
if(digitalRead (ir_back) == 0 && flag2==0) // read degital data from IR
sensor2
{
    flag2=1;
    if(flag1==0)
    {
        myservo.write(180); // sets the servo at 180 degree position
        slot = slot+1;
    }
}
if(flag1==1 && flag2==1)
{
    delay (1000);
    myservo.write(90); // sets the servo at 90 degree position
    flag1=0, flag2=0;
}
delay(1);
}
void Read_Sensor()
{
    S1=0, S2=0, S3=0, S4=0;

```



```
if(digitalRead(ir_car1) == 0){S1=1;} // read digital data from IR sensor3
if(digitalRead(ir_car2) == 0){S2=1;} // read digital data from IR sensor4
if(digitalRead(ir_car3) == 0){S3=1;} // read digital data from IR sensor5
if(digitalRead(ir_car4) == 0){S4=1;} // read digital data from IR sensor6
}
```

Explanation of the program:

1. *Libraries*:

- Servo.h: This library is used to control the Servo motor.
- Wire.h: A library for I2C communication.
- LiquidCrystal_I2C.h: This library is used to interface with the I2C-based Liquid Crystal Display (LCD).

2. *Global Variables*:

- LiquidCrystal_I2C lcd(0x27, 20, 4): Initializes the LCD object with the I2C address 0x27 and a 20x4 character display.
- Servo myservo: Defines a Servo object for controlling a servo motor.
- Defines various integer variables (S1, S2, S3, S4, flag1, flag2, and slot) for managing parking slots and sensor states
- Defines pin numbers for IR sensors (ir_enter, ir_back, ir_car1, ir_car2, ir_car3, ir_car4).

3. *Setup Function*:

- Initializes serial communication at 9600 baud.
- Configures pin modes for IR sensors and the Servo motor.
- Initializes the Servo motor at a 90-degree position.
- Initializes the LCD display and shows a startup message.
- Calls Read_Sensor function to read the initial state of parking slots.

4. *Loop Function*:

- Continuously updates the state of parking slots and displays it on the LCD.
- Checks the status of IR sensors and adjusts parking slots accordingly.
- If the "Enter" IR sensor is triggered, and there's an available slot, it will increment the slot count and adjust the Servo motor.
- If the "Back" IR sensor is triggered, it decrements the slot count and adjusts the Servo motor.
- The Servo motor is controlled to simulate the barrier at the parking entrance.

5. *Read_Sensor Function*:

- Reads the state of the IR sensors (IR sensor1 to IR sensor4) and updates the corresponding variables (S1, S2, S3, S4) to indicate whether a parking slot is occupied (1) or empty (0).

The system uses IR sensors to detect the presence of a car in each parking slot and a Servo motor to simulate a barrier. The LCD displays information about available slots and the status of each slot. The program continuously updates this information based on the sensor inputs.

Python code:

```
class SmartParking:
    def __init__(self, capacity):
        self.capacity = capacity
        self.available_spaces = capacity

    def park_vehicle(self):
        if self.available_spaces > 0:
            self.available_spaces -= 1
            print("Vehicle parked. Available spaces:", self.available_spaces)
        else:
```

```
print("Parking is full. Cannot park the vehicle.")
```

```
def leave_parking(self):
```

```
    if self.available_spaces < self.capacity:
```

```
        self.available_spaces += 1
```

```
        print("Vehicle left. Available spaces:", self.available_spaces)
```

```
    else:
```

```
        print("Parking is empty. No vehicle to leave.")
```

```
def main():
```

```
    capacity = 10 # Adjust the capacity as needed
```

```
    parking_lot = SmartParking(capacity)
```

```
    while True:
```

```
        print("1. Park Vehicle")
```

```
        print("2. Leave Parking")
```

```
        print("3. Exit")
```

```
        choice = input("Enter your choice: ")
```

```
        if choice == "1":
```

```
            parking_lot.park_vehicle()
```

```
        elif choice == "2":
```

```
            parking_lot.leave_parking()
```

```
        elif choice == "3":
```

```
            break
```

```
        else:
```

```
            print("Invalid choice. Please try again.")
```

```
if __name__ == "__main__":  
    main()
```

CONCLUSION:

Our Smart Parking Project is set to revolutionize urban parking with innovation in real-time insights, user-friendliness, resource optimization, enforcement, and data analytics. It's a leap towards more sustainable and efficient urban living, promising reduced congestion and enhanced convenience for all. This project represents the future of intelligent and eco-friendly cities.