WESTERN NEW ENGLAND UNIVERSITY
SPRINGFIELD, MASSACHUSETTS

DEPARTMENT OF MECHANICAL ENGINEERING

ME 455-41 APPLIED MECHATRONIC SYSTEMS
ME 656-41 ADVANCED MECHATRONICS SYSTEMS

Instructor: Dr. Vedang Chauhan

Project 1 Wheeled Mobile Motion Control using IR Remote Control

Objective: This project involves programming a mobile robot to be controlled via an infrared (IR) remote control for precise motion control. You will start by understanding the basics of infrared communication and interfacing an IR receiver with the microcontroller used in the robot. You will then write code to decode signals from an IR remote, allowing you to control the robot's movement—such as forward, backward, left, and right turns—based on the received commands. This project not only enhances your programming skills but also provides hands-on experience with embedded systems, sensor integration, and robotics, fostering a deeper understanding of control systems and real-world applications.

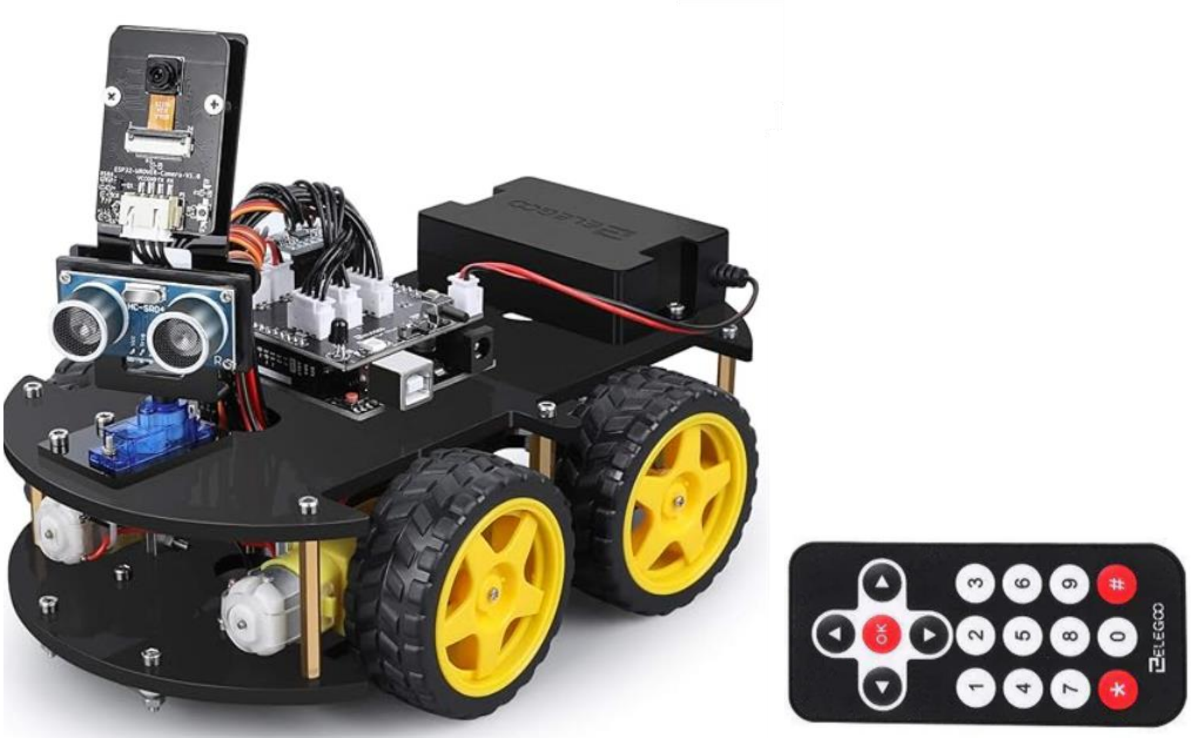


Figure 1 Smart Mobile Robot with an IR Remote (image course: *ELEGOO UNO R3 Smart Robot Car Kit* Amazon.com)

1: IR Remote and Serial Print

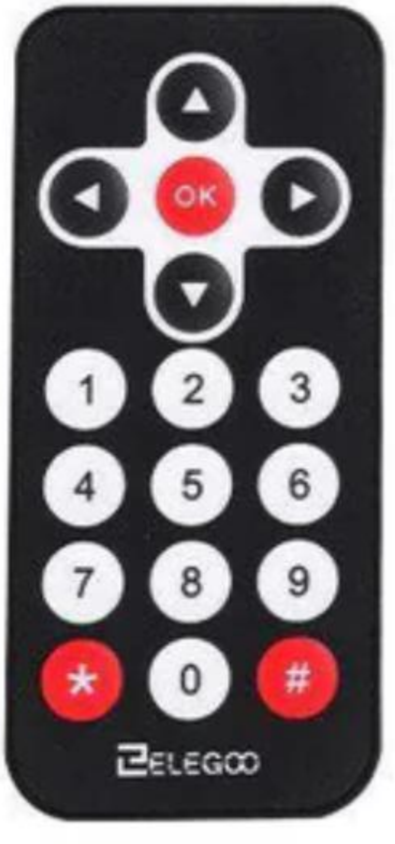


Figure 2 IR Remote

In this part of the project, program your robot control using Arduino IDE that can read signals from an IR remote and print the message on the serial window as per the following table. The robot doesn't move during this part.

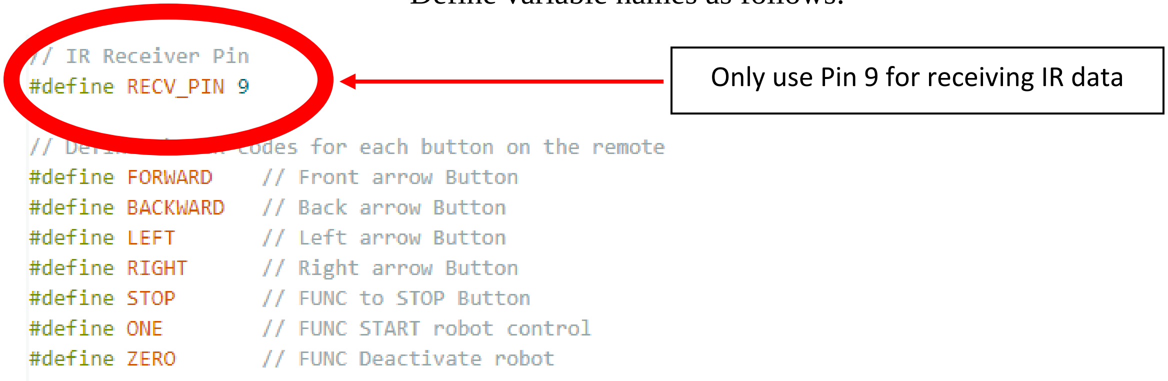
Use Switch-Case statement for this activity. You will find HEX codes for the following buttons and add the table to your report.

Table 1 IR Buttons and Activities on Serial Monitor

IR Remote Button	Hex Code	Serial Print
1		Robot ON
0		Robot STOP
UP		Robot moves forward at a set speed
DOWN		Robot moves backward at a set speed
LEFT		Robot turns left
RIGHT		Robot turns right

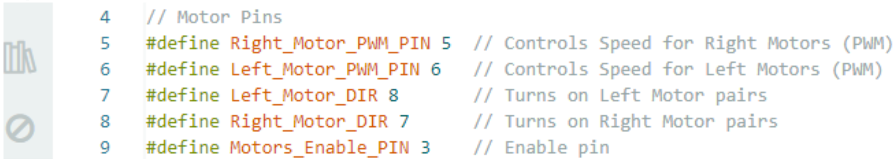
2: Robot Motion (without IR Remote)

Define variable names as follows:



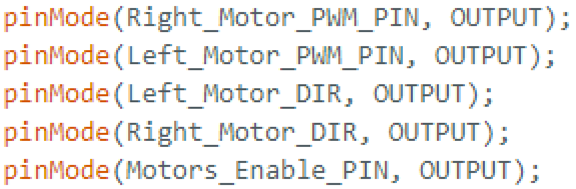
```
// IR Receiver Pin
#define RECV_PIN 9

// Define the IR codes for each button on the remote
#define FORWARD    // Front arrow Button
#define BACKWARD   // Back arrow Button
#define LEFT       // Left arrow Button
#define RIGHT      // Right arrow Button
#define STOP       // FUNC to STOP Button
#define ONE        // FUNC START robot control
#define ZERO       // FUNC Deactivate robot
```

Only use Pin 9 for receiving IR data



```
// Motor Pins
#define Right_Motor_PWM_PIN 5  // Controls Speed for Right Motors (PWM)
#define Left_Motor_PWM_PIN 6   // Controls Speed for Left Motors (PWM)
#define Left_Motor_DIR 8       // Turns on Left Motor pairs
#define Right_Motor_DIR 7      // Turns on Right Motor pairs
#define Motors_Enable_PIN 3    // Enable pin
```

Figure 3: Motor Pins defined



```
pinMode(Right_Motor_PWM_PIN, OUTPUT);
pinMode(Left_Motor_PWM_PIN, OUTPUT);
pinMode(Left_Motor_DIR, OUTPUT);
pinMode(Right_Motor_DIR, OUTPUT);
pinMode(Motors_Enable_PIN, OUTPUT);
```

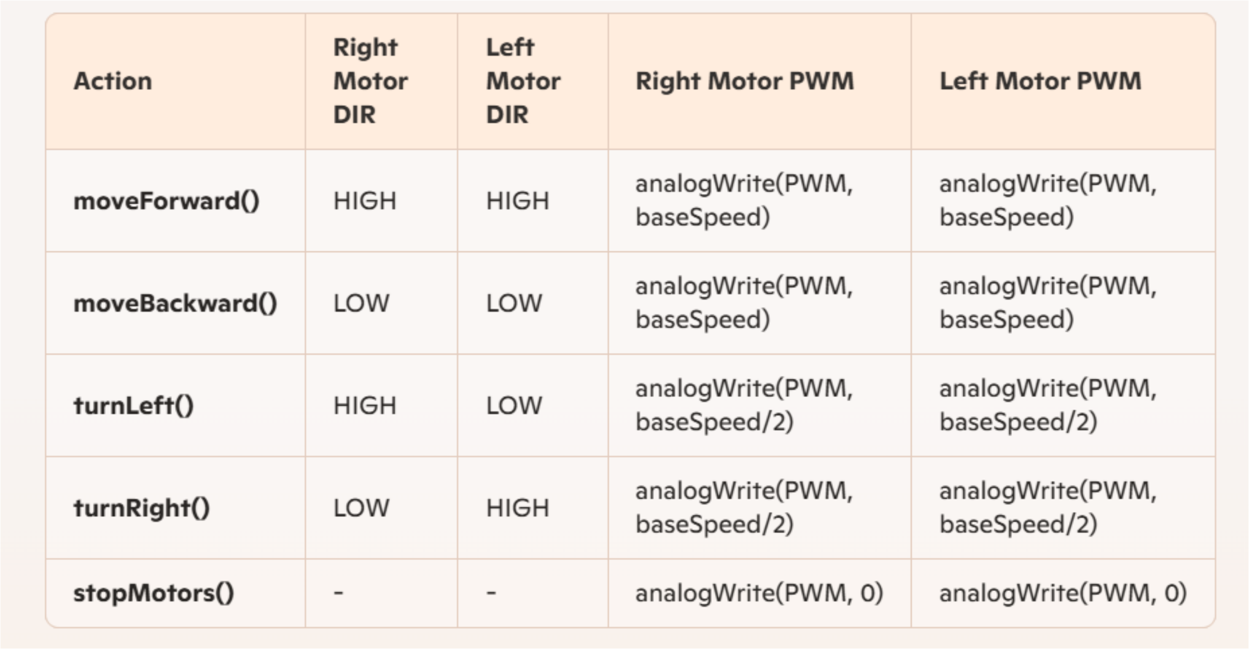
Figure 4: Motor Pins set as outputs in setup

Motor Operation

1. **DIR Pins (Direction Pins)**:
   - These pins control the direction of the motor.
   - **HIGH** on the DIR pin makes the motor rotate in one direction.
   - **LOW** on the DIR pin makes the motor rotate in the opposite direction.
2. **PWM Pins (Pulse Width Modulation Pins)**:
   - These pins control the speed of the motor.
   - The **analogWrite()** function is used to send a PWM signal to these pins.
   - The value sent (0-255) determines the speed:
     - **255** means full speed.
     - **0** means stop.

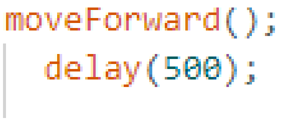
Define subroutine as follows:

Table 2 Subroutines for motor control



Action	Right Motor DIR	Left Motor DIR	Right Motor PWM	Left Motor PWM
moveForward()	HIGH	HIGH	analogWrite(PWM, baseSpeed)	analogWrite(PWM, baseSpeed)
moveBackward()	LOW	LOW	analogWrite(PWM, baseSpeed)	analogWrite(PWM, baseSpeed)
turnLeft()	HIGH	LOW	analogWrite(PWM, baseSpeed/2)	analogWrite(PWM, baseSpeed/2)
turnRight()	LOW	HIGH	analogWrite(PWM, baseSpeed/2)	analogWrite(PWM, baseSpeed/2)
stopMotors()	-	-	analogWrite(PWM, 0)	analogWrite(PWM, 0)

Call subroutine as follows:



```
moveForward();
  delay(500);
```

Figure 5: Calling a subroutine in loop

In this part you write a new program that should move the robot as per the following cycle.

Table 3 Robot Motion without IR Remote Control

Robot Motion Sequence
• Robot waits for 5 sec
• Robot moves forward for 3 sec
• Robot stops and moves backward for 3 sec
• Robot stops and turns left for 3 sec
• Robot stops and turns right for 3 sec
• Robot stops forever until reset
All stops except the last one are momentary (for 50 ms)

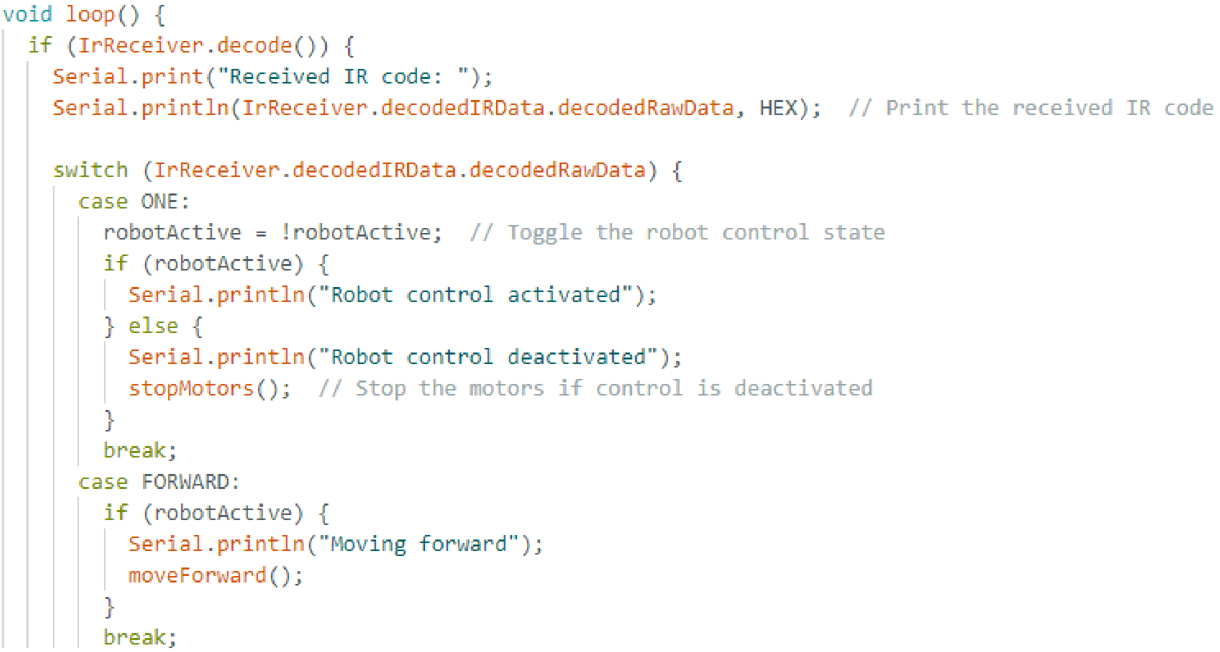
3: Robot Motion Control with IR Remote

Create a new program that is the combination of Part 1 and Part 2. Your program should do the following motion control of the robot.

Table 4 IR Remote Buttons and Robot Motion

IR Remote Button	Robot Status
1	Robot ON
0	Robot STOP
UP	Robot moves forward at a set speed
DOWN	Robot moves backward at a set speed
LEFT	Robot turns left
RIGHT	Robot turns right

4: Motor Controller



```
void loop() {
  if (IrReceiver.decode()) {
    Serial.print("Received IR code: ");
    Serial.println(IrReceiver.decodedIRData.decodedRawData, HEX);  // Print the received IR code

    switch (IrReceiver.decodedIRData.decodedRawData) {
      case ONE:
        robotActive = !robotActive;  // Toggle the robot control state
        if (robotActive) {
          Serial.println("Robot control activated");
        } else {
          Serial.println("Robot control deactivated");
          stopMotors();  // Stop the motors if control is deactivated
        }
        break;
      case FORWARD:
        if (robotActive) {
          Serial.println("Moving forward");
          moveForward();
        }
        break;
```

Figure 6: Reading IR data from the remote to control the RC car

Deliverables and assessment rubric:

Submit onto Kodiak.

Final PDF report (file name: Project1_YourGroupNumber.PDF") that contains • Table 1 with the IR Hex Codes. • Arduino Program • Link of a video of robot's motion for Part 3.	**90**
Quality and professionalism of the report.	**10**

***