

**III<sup>rd</sup> Semester B. Tech - Computer and Communication  
Engineering**

**19CCE201 Microcontroller and Interfacing  
Term Work**

**Line Follower Robot**

**Prepared By:**

**Niranjan K (CB.EN.U4CCE21039)**

**Nithin G.D (CB.EN.U4CCE21040)**

**Sree Hari Nathan K S (CB.EN.U4CCE21070)**

**Krishnamoorthy V (CB.EN.U4CCE21077)**

**Department of Electronics and Communication Engineering**

**Amrita School of Engineering, Coimbatore – 641112**

**2022-23(odd)**

## Motivation:

We took this project because generally Line follower robot can be implemented using Raspberry Pi module but we want to implement this in LPC 2148, so we took this project.

We saw this project as an interesting one and we want to learn more coding in keil software to implement this project and that's why we took this project.

## Theory:

### 1. LPC 2148

The LPC2148 microcontroller is based on a 16-bit/32-bit ARM7TDMI-S CPU with real-time emulation and embedded trace support, that combine the microcontroller with embedded high-speed flash memory ranging from 32 kB to 512 kB. A 128-bit wide memory interface and an accelerator architecture enable 32-bit code execution at the maximum clock rate. For critical code size applications, the alternative 16-bit Thumb mode reduces code by more than 30 % with minimal performance penalty.

Due to their tiny size and low power consumption, LPC2148 is ideal for applications where miniaturization is a key requirement, such as access control. Serial communications interfaces ranging from a USB 2.0 Full-speed device, multiple UARTs, SPI, on-chip SRAM of 8 kB up to 40 kB, make these devices very well suited for communication gateways, voice recognition and low-end imaging, providing both large buffer size and high processing power.

- Up to 21 external interrupt pins available.
- CPU operating voltage range of 3.0 V to 3.6 V ( $3.3\text{ V} \pm 10\%$ ) with 5 V tolerant I/O pads.

### 2. L293D Motor Driver IC

The L293D is characterized for operation from 0°C to 70°C.

The L293D is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5 V to 36 V. Both devices are designed to drive inductive loads such as relays,

solenoids, DC and bipolar stepping motors, as well as other high-current/high-voltage loads in positive- supply applications.

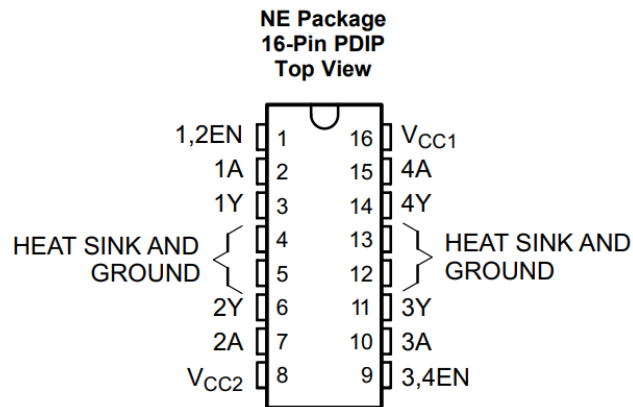
Features:

- Wide Supply-Voltage Range: 4.5 V to 36 V
- Separate Input-Logic Supply
- Internal ESD Protection
- Output Current 600 mA for L293D
- Peak Output Current 1.2 A for L293D
- Output Clamp Diodes for Inductive Transient Suppression

Applications:

- Stepper Motor Drivers
- DC Motor Drivers
- Latching Relay Drivers

### Pin Configuration and Functions



### 3. DC Motor

A DC motor or direct current motor is an electrical machine that transforms electrical energy into mechanical energy by creating a magnetic field that is powered by direct current. When a DC motor is powered, a magnetic field is created in its stator.

The motor can be rotated at a certain speed by applying a fixed voltage to it. If the voltage varies, the speed of the motor varies.

Thus, the DC motor speed can be controlled by applying varying DC voltage; whereas the direction of rotation of the motor can be changed by reversing the direction of current through it.

Features:

- Runs on DC power or AC line voltage with a rectifier
- Operating speeds of 1,000 to 5,000 rpm
- 60-75% efficiency rate
- High starting torque
- Low no-load speeds

### 4. IR Sensor

**Line Follower Robot (LFR)** follows a line, and in order to follow a line, robot must detect the line first. The reflection of light on the white surface is maximum and minimum on the black surface because the black surface absorbs maximum amount of light. So, in order to use this property of light to detect the line, we need IR sensors.

Infrared sensors consist of two elements, a transmitter and a receiver. The transmitter is basically an IR LED, which produces the signal and the IR receiver is a photodiode, which

senses the signal produced by the transmitter. The IR sensor emits the infrared light on an object, the light hitting the black part gets absorbed thus giving a low output but the light hitting the white part reflects back to the transmitter which is then detected by the infrared receiver.

IR Sensors are more accurate, cheap, small sensors often used in robots, and Arduino project to detect objects near the sensor.

The IR sensor has **Power**, **Ground**, **Signal**, and **Enable** pins.

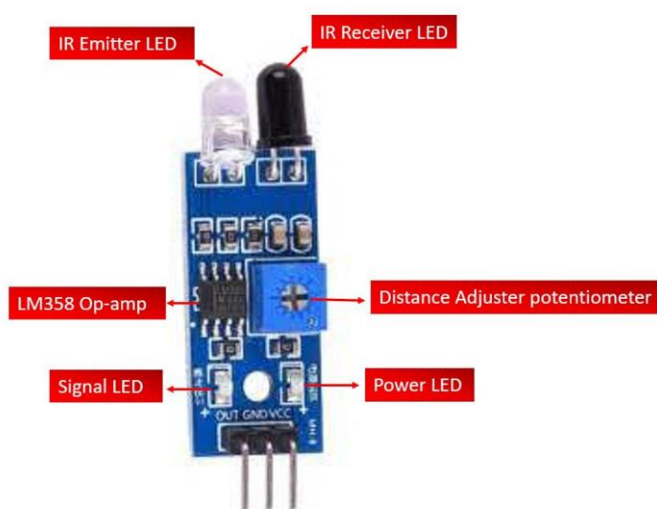
There are also **2 potentiometers**, and **one jumper** on the board.

Features:

- Easy to assemble and use
- Onboard detection indication
- The effective distance range of 2cm to 80cm
- A preset knob to fine-tune distance range
- If there is an obstacle, the indicator lights on the circuit board.
- It can easily detect the object from 2 to 30 cm area and its detection angle is 35°.

Applications:

- Object detection
- Motion detection
- Obstacle avoidance robot
- Gas leakage detection
- Smoke detection
- Distance measurement
- Robotics



The diagram illustrates the hardware setup for an IR obstacle sensor using an LPC2138 microcontroller. Two IR sensor modules, labeled IR1 and IR2, are connected to the microcontroller. The sensors are connected to the microcontroller's VCC, GND, and TestPin. The sensors are also connected to a 5V supply and ground. The diagram includes a pin list for the LPC2138, showing the connection of various pins to the sensors and power supply.

**Pin List for LPC2138:**

Pin	Function
1	P0.0/IO0
2	P0.1/IO1
3	P0.2/IO2
4	P0.3/IO3
5	P0.4/IO4
6	P0.5/IO5
7	P0.6/IO6
8	P0.7/IO7
9	P0.8/IO8
10	P0.9/IO9
11	P0.10/IO10
12	P0.11/IO11
13	P0.12/IO12
14	P0.13/IO13
15	P0.14/IO14
16	P0.15/IO15
17	P0.16/IO16
18	P0.17/IO17
19	P0.18/IO18
20	P0.19/IO19
21	P0.20/IO20
22	P0.21/IO21
23	P0.22/IO22
24	P0.23/IO23
25	P0.24/IO24
26	P0.25/IO25
27	P0.26/IO26
28	P0.27/IO27
29	P0.28/IO28
30	P0.29/IO29
31	P0.30/IO30
32	P0.31/IO31
33	P0.32/IO32
34	P0.33/IO33
35	P0.34/IO34
36	P0.35/IO35
37	P0.36/IO36
38	P0.37/IO37
39	P0.38/IO38
40	P0.39/IO39
41	P0.40/IO40
42	P0.41/IO41
43	P0.42/IO42
44	P0.43/IO43
45	P0.44/IO44
46	P0.45/IO45
47	P0.46/IO46
48	P0.47/IO47
49	P0.48/IO48
50	P0.49/IO49
51	P0.50/IO50
52	P0.51/IO51
53	P0.52/IO52
54	P0.53/IO53
55	P0.54/IO54
56	P0.55/IO55
57	P0.56/IO56
58	P0.57/IO57
59	P0.58/IO58
60	P0.59/IO59
61	P0.60/IO60
62	P0.61/IO61
63	P0.62/IO62
64	P0.63/IO63
65	P0.64/IO64
66	P0.65/IO65
67	P0.66/IO66
68	P0.67/IO67
69	P0.68/IO68
70	P0.69/IO69
71	P0.70/IO70
72	P0.71/IO71
73	P0.72/IO72
74	P0.73/IO73
75	P0.74/IO74
76	P0.75/IO75
77	P0.76/IO76
78	P0.77/IO77
79	P0.78/IO78
80	P0.79/IO79
81	P0.80/IO80
82	P0.81/IO81
83	P0.82/IO82
84	P0.83/IO83
85	P0.84/IO84
86	P0.85/IO85
87	P0.86/IO86
88	P0.87/IO87
89	P0.88/IO88
90	P0.89/IO89
91	P0.90/IO90
92	P0.91/IO91
93	P0.92/IO92
94	P0.93/IO93
95	P0.94/IO94
96	P0.95/IO95
97	P0.96/IO96
98	P0.97/IO97
99	P0.98/IO98
100	P0.99/IO99

```
#include <lpc214x.h>
#define LEFT_SENSOR_THRESHOLD 500
#define RIGHT_SENSOR_THRESHOLD 500
```

```
void init() {
    // Set P0.0 and P0.1 as digital outputs for left motor
    IO0DIR |= (1<<0) | (1<<1);

    // Set P0.2 and P0.3 as digital outputs for right motor
    IO0DIR |= (1<<2) | (1<<3);

    // Set P0.4 and P0.5 as digital inputs for IR sensors
    IO0DIR &= ~(1<<4) | (1<<5));
}

void set_motor_speed(int left_speed, int right_speed) {
    // Set left motor speed
    if (left_speed > 0) {
        // Forward
        IO0SET = (1<<0);
        IO0CLR = (1<<1);
    } else if (left_speed < 0) {
        // Reverse
        IO0SET = (1<<1);
```

```

    IO0CLR = (1<<0);
} else {
    // Stop
    IO0CLR = (1<<0);
    IO0CLR = (1<<1);
}

// Set right motor speed
if (right_speed > 0) {
    // Forward
    IO0SET = (1<<2);
    IO0CLR = (1<<3);
} else if (right_speed < 0) {
    // Reverse
    IO0SET = (1<<3);
    IO0CLR = (1<<2);
} else {
    // Stop
    IO0CLR = (1<<2);
    IO0CLR = (1<<3);
}
}

int main() {
    init();

    while (1) {
        int left_sensor = (IO0PIN & (1<<4)) >> 4;
        int right_sensor = (IO0PIN & (1<<5)) >> 5;

        if (left_sensor > LEFT_SENSOR_THRESHOLD && right_sensor > RIGHT_SENSOR_THRESHOLD) {
            // Both sensors detect line, go straight
            set_motor_speed(50, 50);
        } else if (left_sensor > LEFT_SENSOR_THRESHOLD) {
            // Only left sensor detects line, turn right
            set_motor_speed(-50, 50);
        } else if (right_sensor > RIGHT_SENSOR_THRESHOLD) {
            // Only right sensor detects line, turn left
            set_motor_speed(50, -50);
        } else {
            // No sensors detect line, stop
            set_motor_speed(0, 0);
        }
    }

    return 0;
}

```

## Results:

The design of the line follower robot has been done using proteus with the hardwares LPC 2148(LPC 2138 in proteus), L293D Motor Driver, IR Sensors, DC motors with the required power supply of 3.3V. The simulation has also been done with the code using c programming language.

**Inference:**

The robot is able to detect and follow a line with the help of infrared sensors and motors. The robot is able to respond to changes in the line's position and adjust its direction accordingly. The robot is able to maintain a consistent speed while following the line. The robot is able to avoid deviations from the line.