Gesture Controlled Bot

Tired of controlling your robot with a joystick? Ever thought of controlling it with your hands?

Harsimar Singh

M.V.S.S Raju

15.03.2017

INTRODUCTION

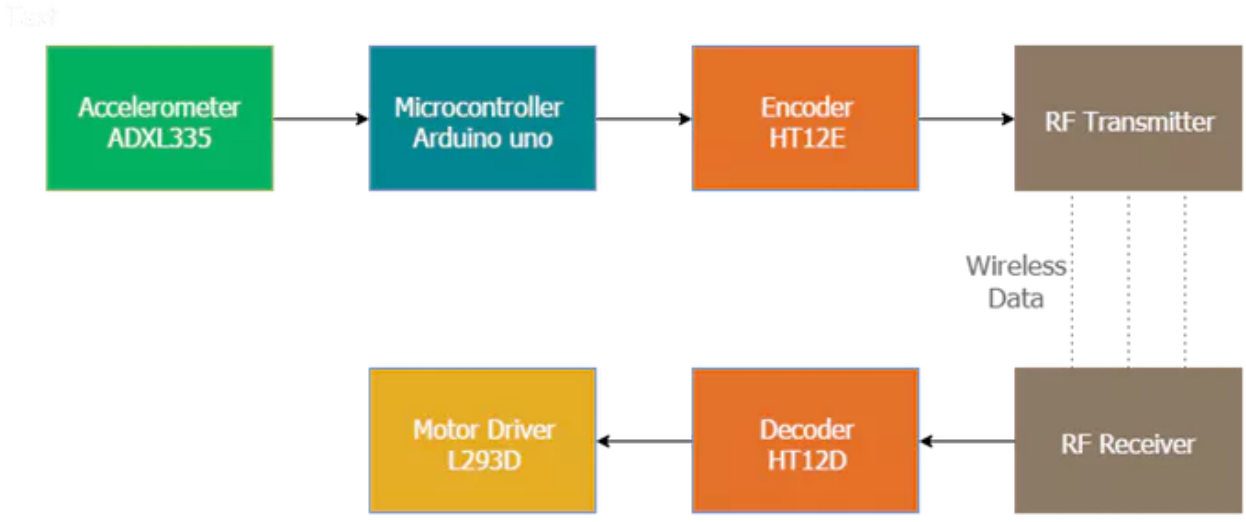
Everyone wish to control thing sitting at your place with hand movements, So finally we decided to build a bot that responds to Gestures of the hand.

The project is based on wireless communication, where the data from the hand gestures is transmitted to the robot over RF link (RF transmitter -Receiver pair). The project is divided into transmitter and receiver section. A motor driver is used which controls and drives the motor to sufficient voltage under the instruction of receiver.

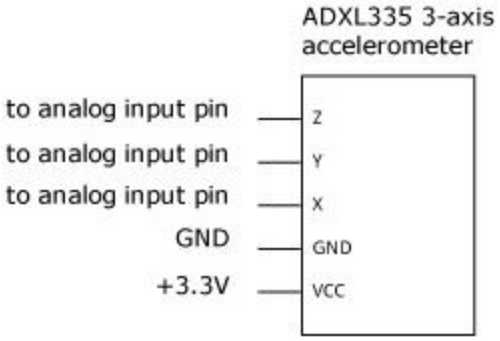
COMPONENTS USED

1. Arduino Uno - 1
2. Accelerometer ADXL-335 -1
3. RF Module 434 MHZ -1
4. Encoder HT12E -1
5. Decoder HT12D -1
6. Motor Driver L293D -1
7. Voltage Regulator IC-7805 -1
8. Motors 12V 60RPM Vega Motors -2

BIOCK DIAGRAM



ACCELEROMETER ADXL-335

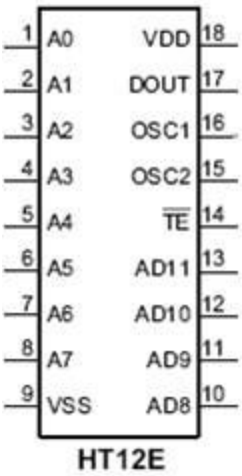


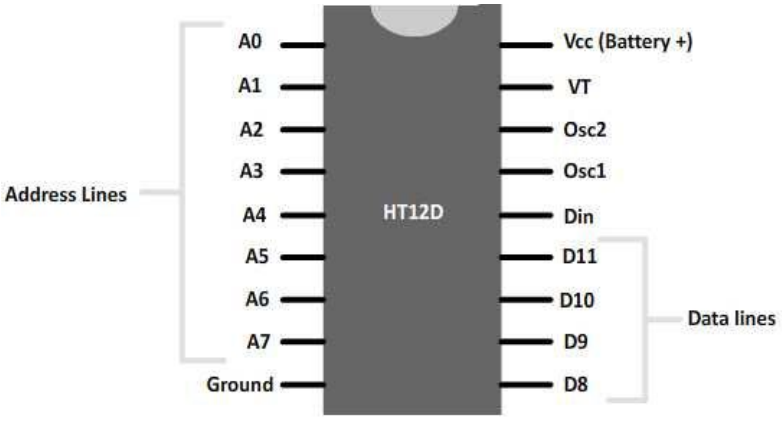
The ADXL335 is a triple axis MEMS accelerometer with extremely low noise and power.It measures accelerations in the X,Y,Z directions the full sensing range is |3G|.

Ever played motion games on your mobile? This is possible because of accelerometer sensor in the mobile.

To find the exact angle and orientation in which our hand is we need one more sensor called Gyroscope, however we limited our project to accelerometer because not that accuracy is required.

Encoder And Decoder (HT12E and HT12D)



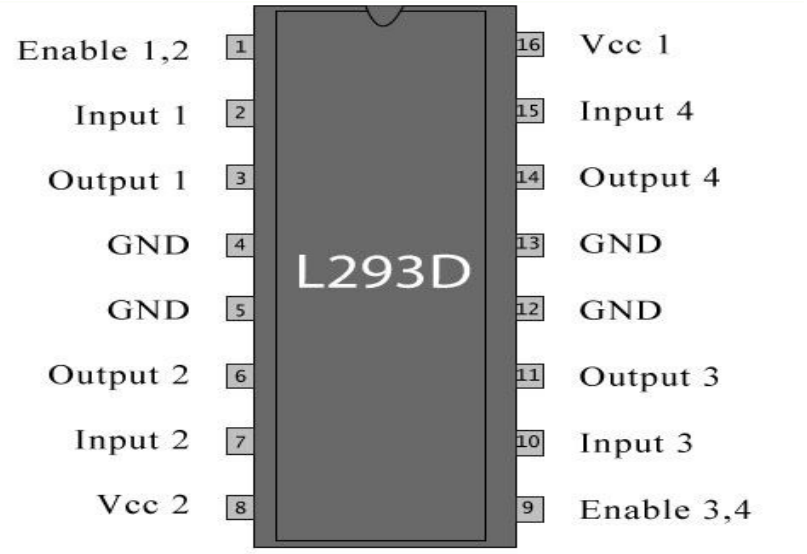


HT12E and HT1D are an encoder integrated circuit and decoder integrated circuit of 2^12 series of encoders respectively . They are paired for use in remote control system applications. It is mainly used to interface RF and infrared circuits. The chosen pair of encoder/decoder should have same number of addresses and data format.

Simply put, HT12E converts parallel inputs into serial output, it encodes the 12 bit parallel data into serial for transmission through an RF transmitter. These 12 bits are divided into 8 address bits and 4 data bits.

These 12 bits of data are decoded by HT12D and it converts the serial input to parallel output which is fed to the motor driver circuit.

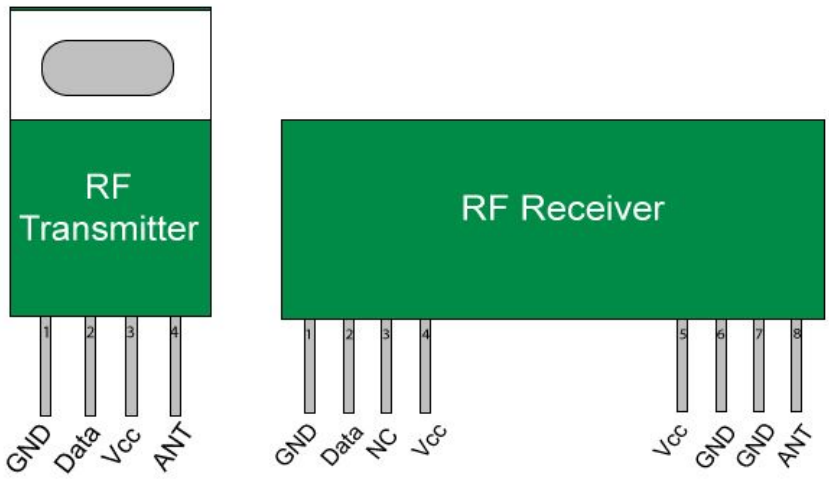
Motor Driver L293D



L293D is a dual H-bridge motor driver integrated circuit. Motor drivers act as current amplifiers since they take a low-current control signal and provide a high current-signal. This higher current signal is used to drive the motors.

L293D contains two inbuilt H-bridge driver circuits. In its common mode of operation,two DC motors can be driven simultaneously, both in forward and reverse direction. The motor operations of two motors can be controlled by input logic at input pins. A input logic of 00 or 11 will stop the corresponding motor. Logic 01 and 10 will rotate it in clockwise and anti-clock directions.

RF Receiver and Transmitter



The RF module, as the name suggests, operates at Radio Frequency. The corresponding frequency range varies between 30 kHz & 300 GHz. In this RF system, the digital data is represented as variations in the amplitude of carrier wave. This kind of modulation is known as Amplitude Shift Keying (ASK).

Transmission through RF is better than IR (infrared) because of many reasons. Firstly, signals through RF can travel through larger distances making it suitable for long range applications. Also, while IR mostly operates in line-of-sight mode, RF signals can travel even when there is an obstruction between transmitter & receiver. Next, RF transmission is more strong and reliable than IR transmission. RF communication uses a specific frequency unlike IR signals which are affected by other IR emitting sources.

This RF module comprises of an RF Transmitter and an RF Receiver. The transmitter/receiver (Tx/Rx) pair operates at a frequency of 434 MHz. An RF transmitter receives serial data and transmits it wirelessly through RF through its antenna connected

at pin4. The transmission occurs at the rate of 1Kbps - 10Kbps.The transmitted data is received by an RF receiver operating at the same frequency as that of the transmitter.

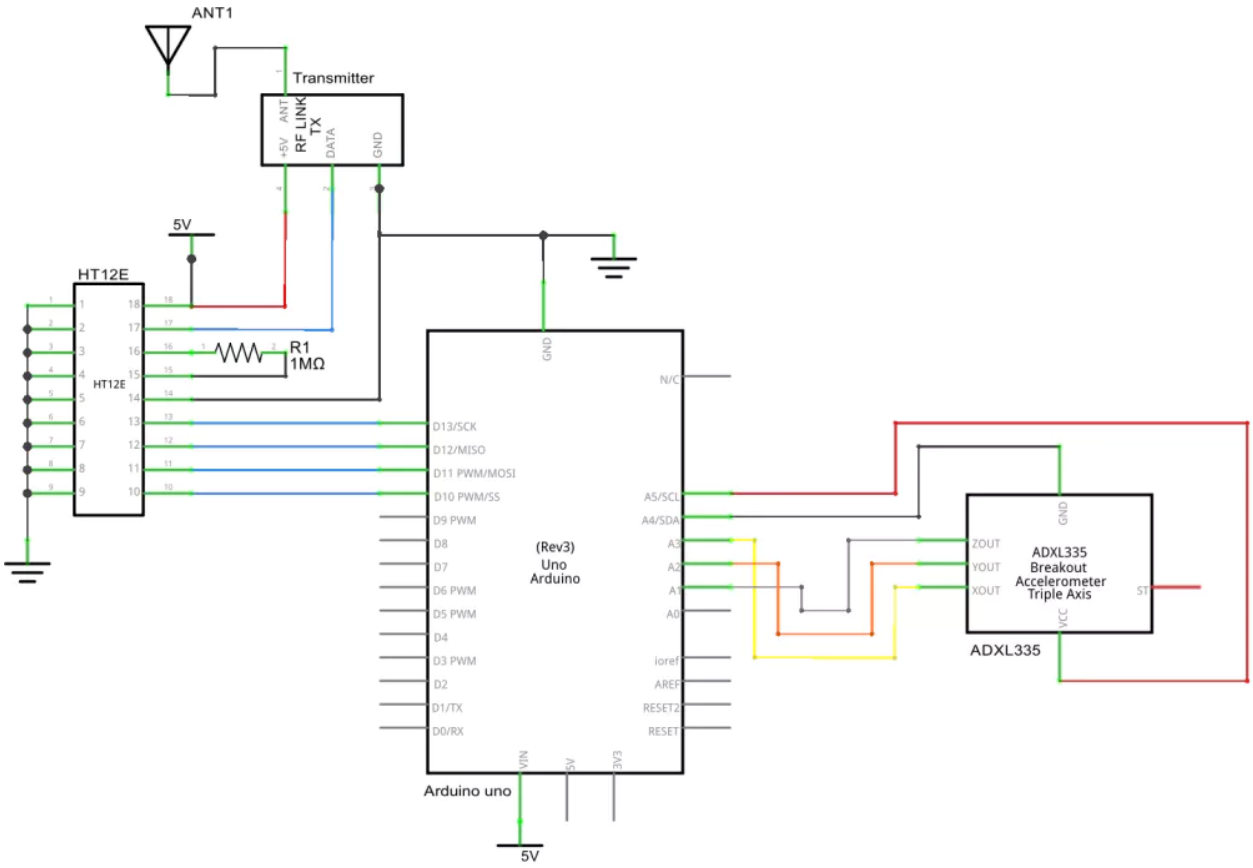
The RF module is often used along with a pair of encoder/decoder. The encoder is used for encoding parallel data for transmission feed while reception is decoded by a decoder.

CIRCUIT DIAGRAM

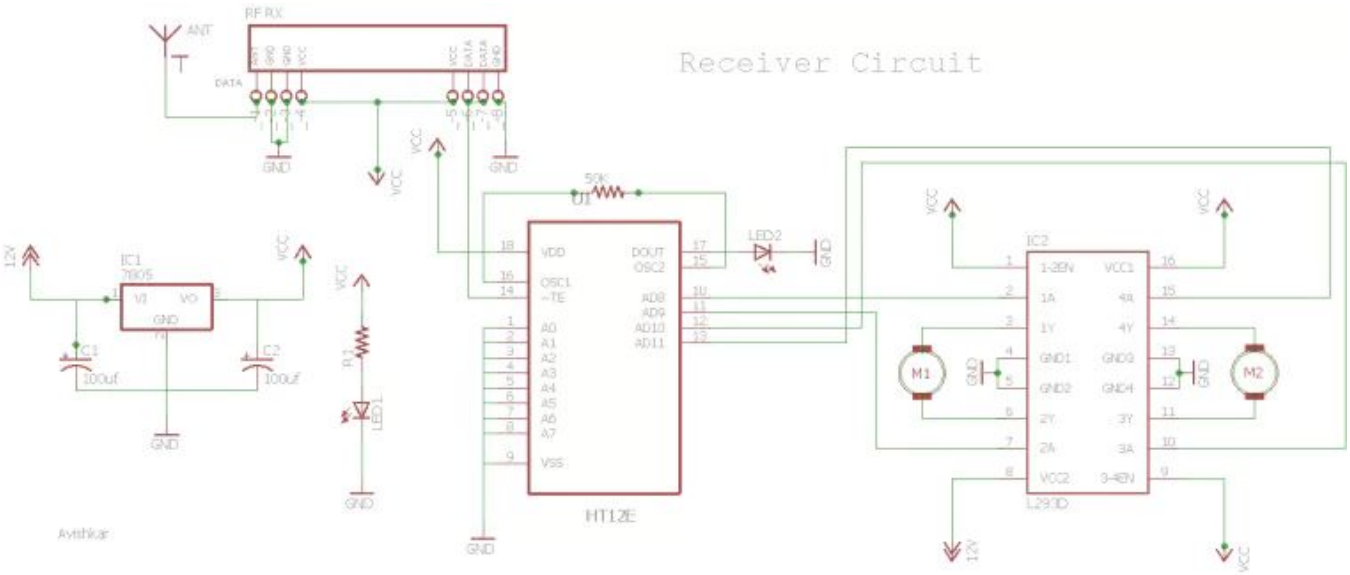
Divided into two stages:

1. Transmitter Section
2. Receiver Section

Transmitter Section



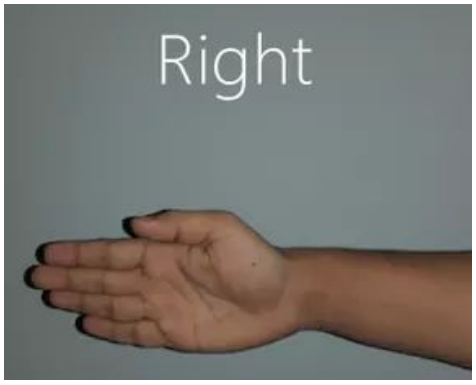
Receiver Section



*it should have decoder ic not encoder ic !

GESTURES THE BOT WILL RESPOND TO

This bot is designed to respond to following gestures.



ARDUINO CODE

```
int GNDPin=A4; //Set Analog pin 4 as GND
int VccPin=A5; //Set Analog pin 5 as VCC
int xPin=A3; //X axis input
int yPin=A2; //Y axis input
int zPin=A1; //Z axis input(not used)
int Q1=10,Q2=11,Q3=12,Q4=13; //Output pins to be connected to 10, 11, 12, 13 of Decoder IC

long x; //Variabe for storing X coordinates
```

```
long y; //Variabe for storing Y coordinates
long z; //Variabe for storing Z coordinates

void setup()
{
  Serial.begin(9600);
  pinMode(Q1,OUTPUT);
  pinMode(Q2,OUTPUT);
  pinMode(Q3,OUTPUT);
  pinMode(Q4,OUTPUT);
  pinMode(GNDPin, OUTPUT);
  pinMode(VccPin, OUTPUT);
  digitalWrite(GNDPin, LOW); //Set A4 pin LOW
  digitalWrite(VccPin, HIGH); //Set A5 pin HIGH
}

void loop()
{
  x = analogRead(xPin); //Reads X coordinates
  y = analogRead(yPin); //Reads Y coordinates
  z = analogRead(zPin); //Reads Z coordinates (Not Used)

    if(x<340)      // Change the value for adjusting sensitivity
      forward();
    else if(x>400) // Change the value for adjusting sensitivity
      backward();
    else if(y>400) // Change the value for adjusting sensitivity
      right();
    else if(y<340) // Change the value for adjusting sensitivity
      left();
    else
      stop_();
}

void stop_()
{
```

```
  Serial.println("");
  Serial.println("STOP");
  digitalWrite(Q1,LOW);
  digitalWrite(Q2,LOW);
  digitalWrite(Q3,LOW);
  digitalWrite(Q4,LOW);
}

void forward()
{
  Serial.println("");
  Serial.println("Forward");
  digitalWrite(Q1,HIGH);
  digitalWrite(Q2,LOW);
  digitalWrite(Q3,HIGH);
  digitalWrite(Q4,LOW);
}

void backward()
{
  Serial.println("");
  Serial.println("Backward");
  digitalWrite(Q1,LOW);
  digitalWrite(Q2,HIGH);
  digitalWrite(Q3,LOW);
  digitalWrite(Q4,HIGH);
}

void left()
{
  Serial.println("");
  Serial.println("Left");
  digitalWrite(Q1,LOW);
  digitalWrite(Q2,HIGH);
  digitalWrite(Q3,HIGH);
  digitalWrite(Q4,LOW);
```

```
}

void right()
{
  Serial.println("");
  Serial.println("Right");
  digitalWrite(Q1,HIGH);
  digitalWrite(Q2,LOW);
  digitalWrite(Q3,LOW);
  digitalWrite(Q4,HIGH);
}
```

REFERENCES

1. Electronics Hub
2. Arduino CC
3. Circuit Digest