Alexandria University
Faculty of Engineering
Computer and Systems Engineering
Department

CS331: Microprocessor Systems

Project
Gesture controlled robot

Team members:
1. Aya Sameh Ismael (01)
2. Mira Samir (75)
3. Mohab Mosaad (74)
4. Passent Mostafa Saad Zaghloul (21)
5. Saeed Hamdy (30)
6. Salma Hesham Mohammed Ragab (33)

1. Project idea and description

Gesture controlled robot:

- It is controlled by using hand instead of buttons or joysticks.
- The robot is controlled by only moving the hand.
- A transmitting device is used in the hand which contains RF transmitter and accelerometer.
- This transmitting device will transmit commands to the robot.
- Commands are like: moving forward, moving backward, turning right or turning left.

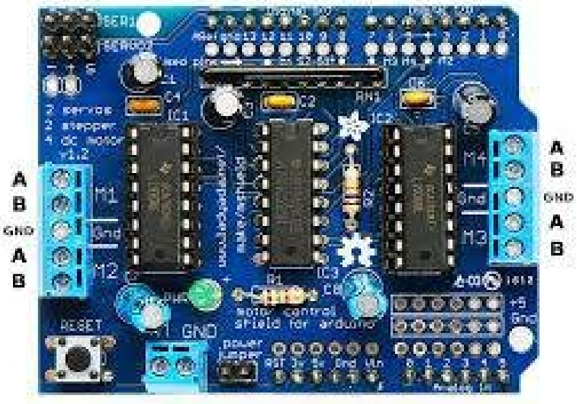
2. List of components used

- Arduino Uno
- Arduino Mega

- 2xTransceiver module (NRF24L01)

- MPU 6050
(Accelerometer)

- Motor shield (Adafruit l293d)

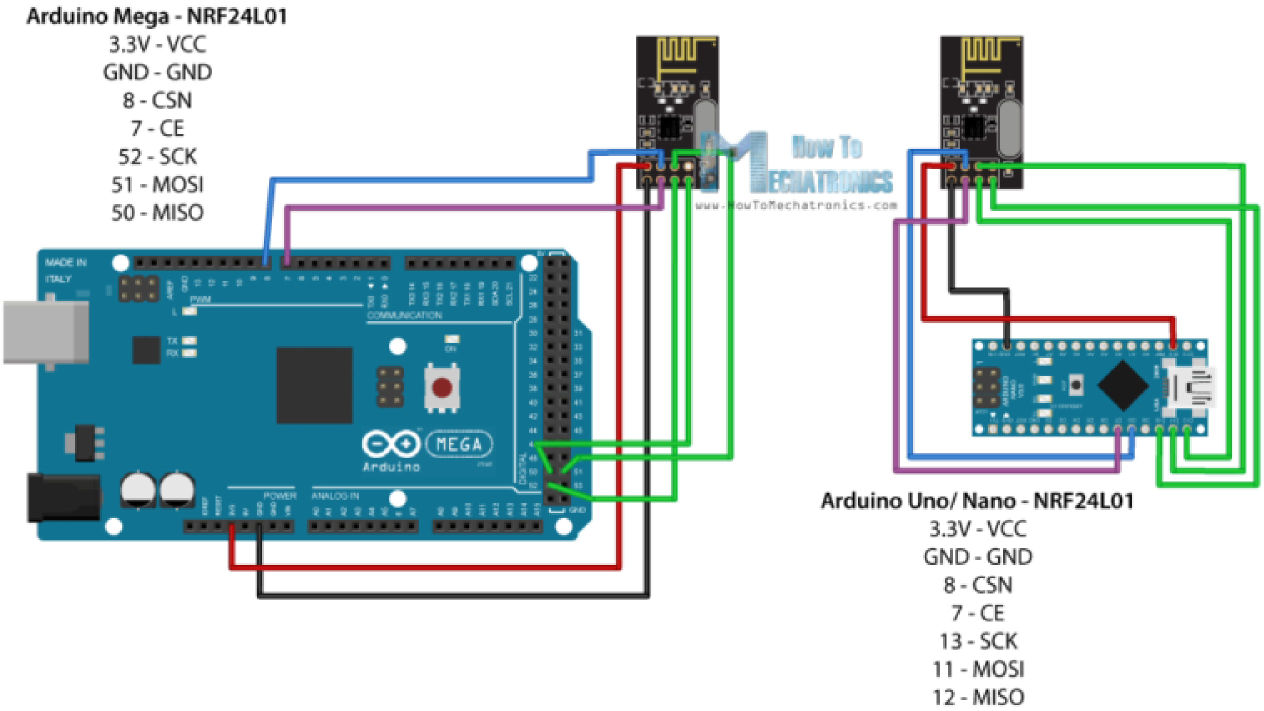


- Robot car frame
- Batteries
- Voltage regulator (lD1117)

- Wires

3. Circuit diagram and connections

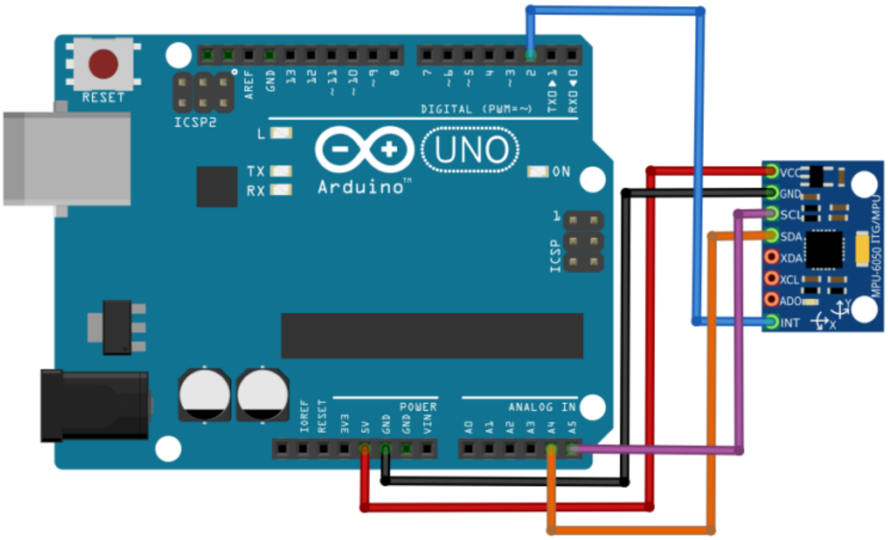
- Connecting the NRF24L01 module to the Arduino boards (Uno and Mega).



Arduino Mega - NRF24L01
3.3V - VCC
GND - GND
8 - CSN
7 - CE
52 - SCK
51 - MOSI
50 - MISO

Arduino Uno/ Nano - NRF24L01
3.3V - VCC
GND - GND
8 - CSN
7 - CE
13 - SCK
11 - MOSI
12 - MISO

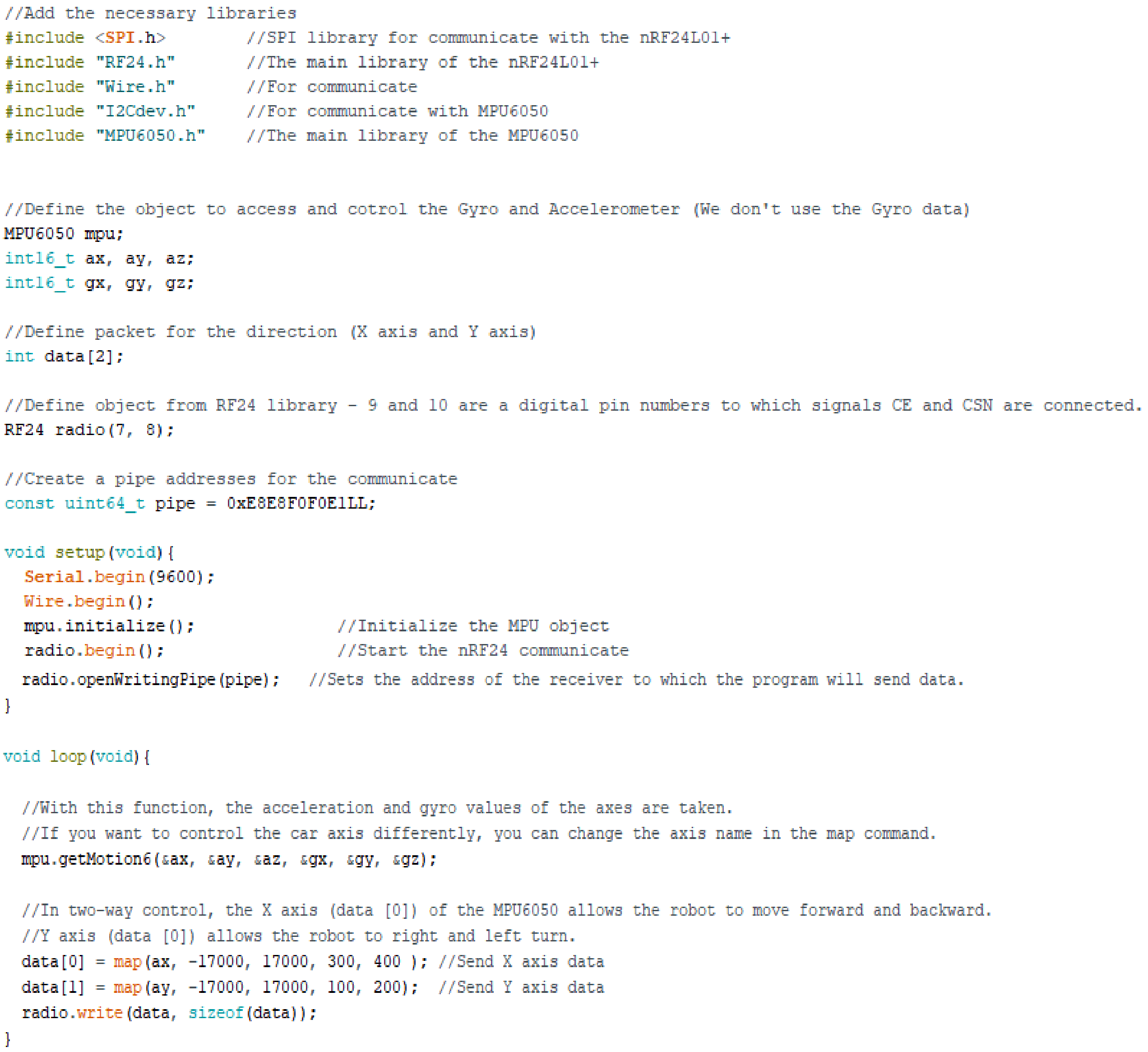
- Connecting the Arduino with the MPU 6050 Sensor.



- Connecting the motor driver shield L293D with the arduino mega.

4. Code

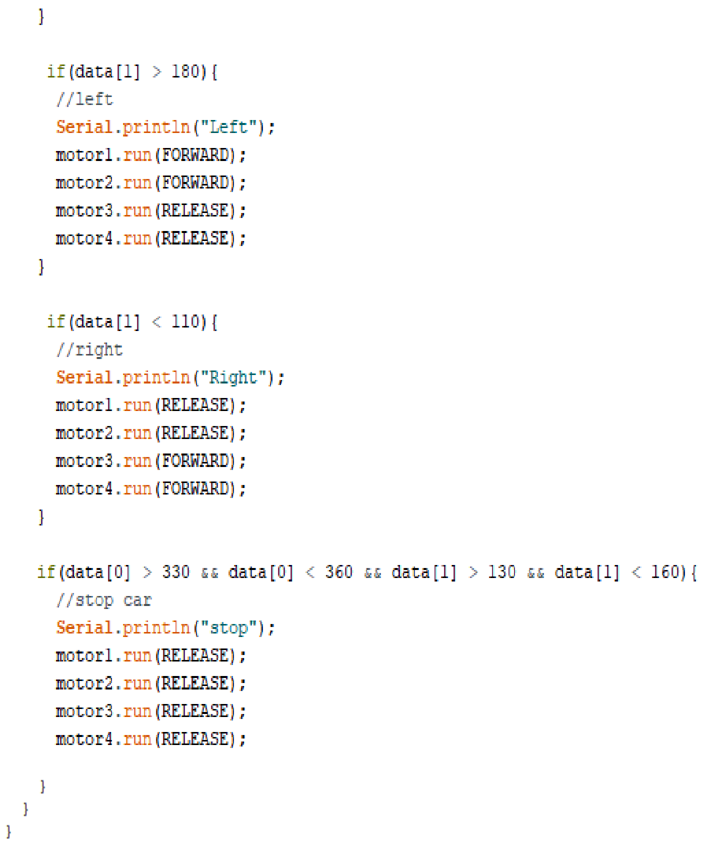
- Transmitter:



```
//Add the necessary libraries
#include <SPI.h>        //SPI library for communicate with the nRF24L01+
#include "RF24.h"       //The main library of the nRF24L01+
#include "Wire.h"       //For communicate
#include "I2Cdev.h"     //For communicate with MPU6050
#include "MPU6050.h"    //The main library of the MPU6050


//Define the object to access and cotrol the Gyro and Accelerometer (We don't use the Gyro data)
MPU6050 mpu;
int16_t ax, ay, az;
int16_t gx, gy, gz;

//Define packet for the direction (X axis and Y axis)
int data[2];

//Define object from RF24 library - 9 and 10 are a digital pin numbers to which signals CE and CSN are connected.
RF24 radio(7, 8);

//Create a pipe addresses for the communicate
const uint64_t pipe = 0xE8E8F0F0E1LL;

void setup(void){
  Serial.begin(9600);
  Wire.begin();
  mpu.initialize();              //Initialize the MPU object
  radio.begin();                 //Start the nRF24 communicate
  radio.openWritingPipe(pipe);   //Sets the address of the receiver to which the program will send data.
}

void loop(void){

  //With this function, the acceleration and gyro values of the axes are taken.
  //If you want to control the car axis differently, you can change the axis name in the map command.
  mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);

  //In two-way control, the X axis (data [0]) of the MPU6050 allows the robot to move forward and backward.
  //Y axis (data [0]) allows the robot to right and left turn.
  data[0] = map(ax, -17000, 17000, 300, 400 ); //Send X axis data
  data[1] = map(ay, -17000, 17000, 100, 200);  //Send Y axis data
  radio.write(data, sizeof(data));
}
```

- Reciever:

```
//Add the necessary libraries
#include <SPI.h>      //SPI library for communicate with the nRF24L01+
#include "RF24.h"     //The main library of the nRF24L01+
#include <AFMotor.h>

//Define packet for the direction (X axis and Y axis)
int data[2];

// motors
AF_DCMotor motor1(1, MOTOR12_64KHZ); // create motor #2, 64KHz pwm
AF_DCMotor motor2(2, MOTOR12_64KHZ); // create motor #2, 64KHz pwm
AF_DCMotor motor3(3, MOTOR12_64KHZ); // create motor #2, 64KHz pwm
AF_DCMotor motor4(4, MOTOR12_64KHZ); // create motor #2, 64KHz pwm

//Define object from RF24 library - 22 and 24 are a digital pin numbers to which signals CE and CSN are connected
RF24 radio(22,24);

//Create a pipe addresses for the communicate
const uint64_t pipe = 0xE8E8F0F0E1LL;

void setup(){

  Serial.begin(9600);
  radio.begin();                    //Start the nRF24 communicate
  radio.openReadingPipe(1, pipe);   //Sets the address of the transmitter to which the program will receive data.
  radio.startListening();
  motor1.setSpeed(255);
  motor2.setSpeed(255);
  motor3.setSpeed(255);
  motor4.setSpeed(255);

  }

void loop(){
  if (radio.available()){
    radio.read(data, sizeof(data));

    if(data[0] > 380){
      //forward
      Serial.println("Forward");
      motor1.run(FORWARD);
      motor2.run(FORWARD);
      motor3.run(FORWARD);
      motor4.run(FORWARD);

    }

     if(data[0] < 310){
      //backward
      Serial.println("Backward");
      motor1.run(BACKWARD);
      motor2.run(BACKWARD);
      motor3.run(BACKWARD);
      motor4.run(BACKWARD);
```



```
    }

     if(data[1] > 180){
      //left
      Serial.println("Left");
      motor1.run(FORWARD);
      motor2.run(FORWARD);
      motor3.run(RELEASE);
      motor4.run(RELEASE);
    }

     if(data[1] < 110){
      //right
      Serial.println("Right");
      motor1.run(RELEASE);
      motor2.run(RELEASE);
      motor3.run(FORWARD);
      motor4.run(FORWARD);
    }

    if(data[0] > 330 && data[0] < 360 && data[1] > 130 && data[1] < 160){
      //stop car
      Serial.println("stop");
      motor1.run(RELEASE);
      motor2.run(RELEASE);
      motor3.run(RELEASE);
      motor4.run(RELEASE);

    }
  }
}
```

5. Challenges faced and any notes

- The pins of the arduino uno weren't enough for both the shield and the transceiver, so arduino mega was used.
- A 3.3 volts were needed by the receiver, so a voltage regualtor was used from the 5 volts that are the output of the arduino mega to change them to 3.3 volts to be used by the reciever.

6. References and any libraries used

Libraries:
- RF24 library for the transceiver module
- I2Cdev library MPU6050 – library
- AF_Motor Arduino library

Resources:
- Learn.adafruit.com
- mertarduinotutorial.blogspot