UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

Documentatie proiect PMP

Robot controlat prin WIFI

Student: Florea Gabriel-Alin

Grupa: 30236

Facultate: Calculatoare, Romana

An: 3

Cuprins

1. Prezentare generala

Acest proiect este reprezentat de un robot controlat prin WIFI, cu detectie de obstacole pe directia inainte.

Robotul este construit pe platformele de experimentare ale catedrei de PMP a Universitatii Tehnice Cluj-Napoca.

O astfel de platforma standard este constituita din urmatoarele componente:

- placa microcontroller compatibila Arduino Uno
- driver motoare l298N Dual H-Bridge
- doua motoare DC
- carcasa baterii 4xAA
- doua roti conectate la motoare si o roata support
- suport Plexiglas
- doua placi de prototipizare
- un senzor sonar
- doi senzori de masurare a turatiei la roti.

Alaturi de componentele amintite mai sus, pentru realizarea proiectului de fata s-au folosit urmatoarele:

- placa microcontroller compatibila Arduino Mega 2560
- modul WIFI ESP 8266.

Programarea componentelor s-a realizat in mediul de programare Arduino 1.8.7, folosind ca limbaj de programare de baza C.

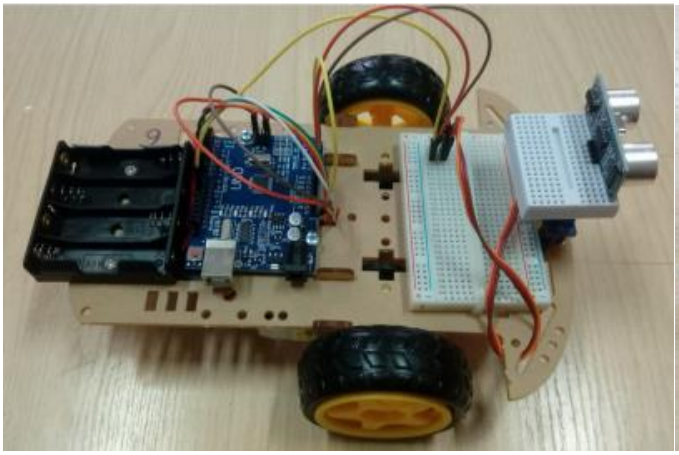
Aplicatia pentru telefonul mobil, pentru a putea controla prin WIFI robotul a fost realizata in mediul MIT App Inventor, care este un mediu intuitive pentru realizarea aplicatiilor Android.

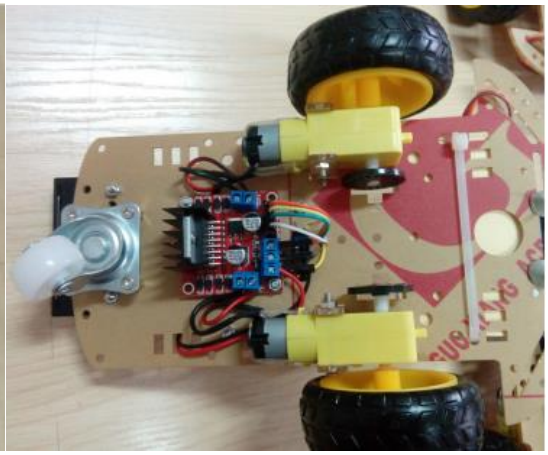
Pentru a putea conecta placile Arduino Uno si Arduino Mega, am folosti protocolul Inter-Integrated Circuit (I2C). Rolul placii Arduino Mega este de Master si anume realizeaza comunicarea cu modulul WIFI si trimite un caracter la Slave, in functie de comanda primita de la modulul WIFI, care interpreteaza comanda primita din aplicatia Android. Rolul placii Arduino Uno este de Slave si anume primeste caracterul transmis de Master si in functie de acesta realizeaza un anumit tip de rotatie pentru motoare. Asadar, placa Arduino uno are rolul de a facilita functionarea motoarelor, cat si interpretarea semnalelor sosite de la senzorul de determinare a distantei.

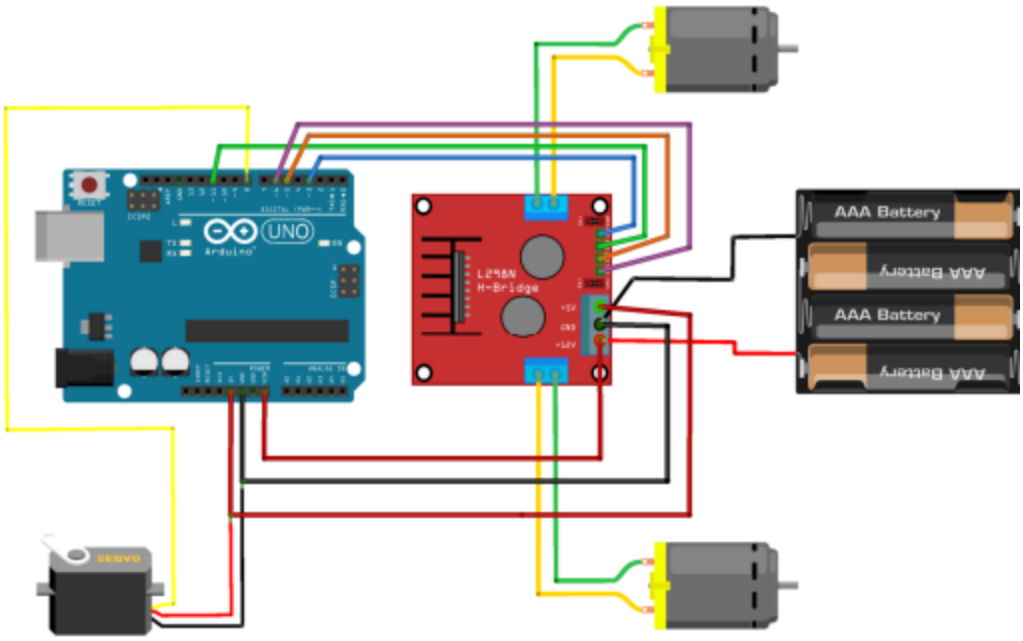
Montajul efectiv poate prezenta mici modificari in ceea ce priveste pinii utilizati, fata de montajul prezentat aici, in functie de modul in care se prezinta componentele hardware la fiecare sedinta de laborator.

2. Platforma si montajul realizat

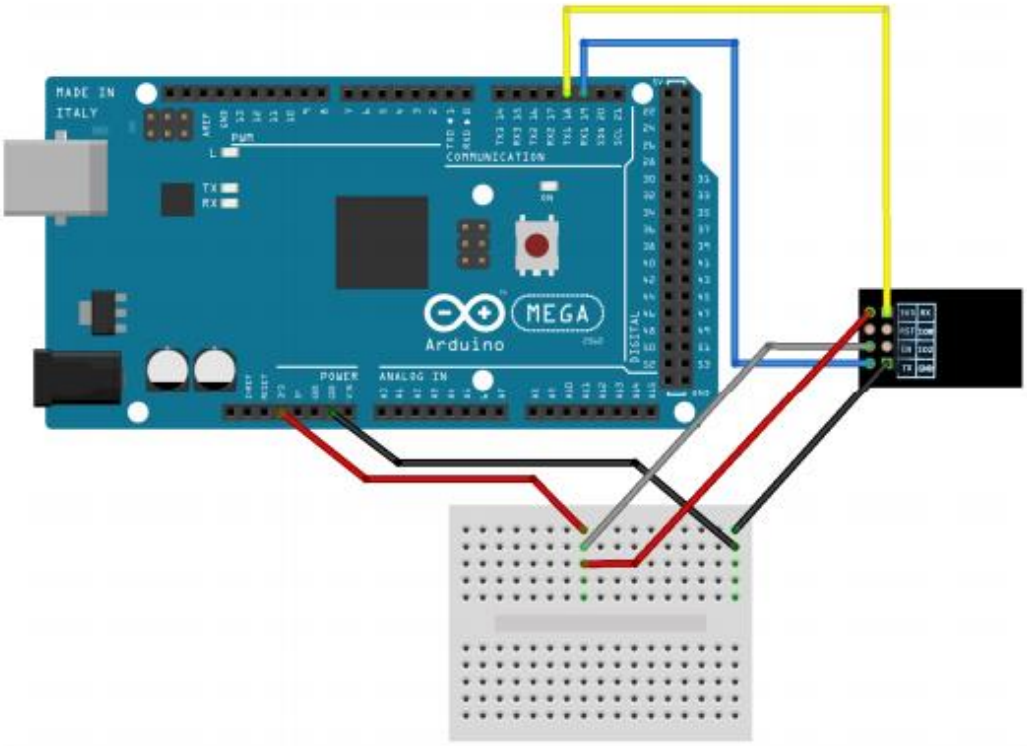
-Platforma de experimentare



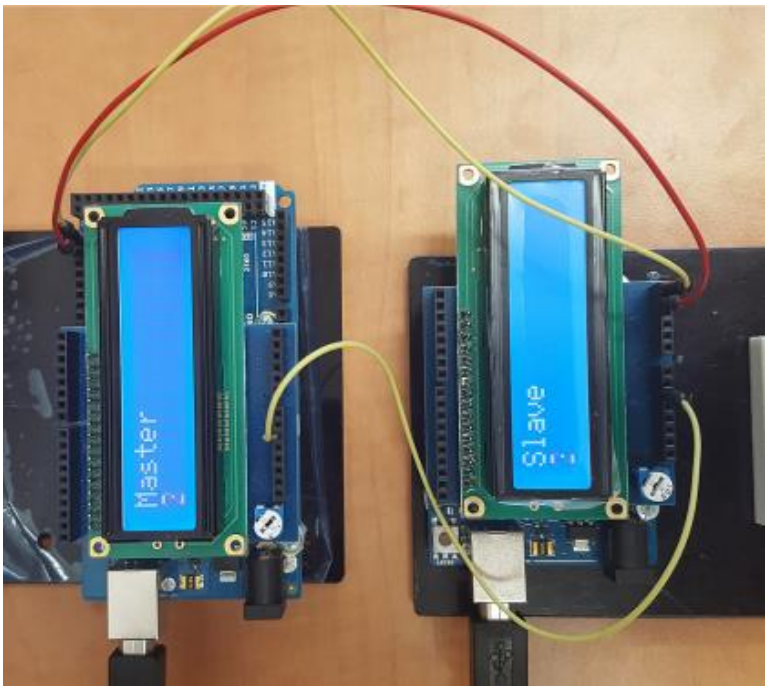




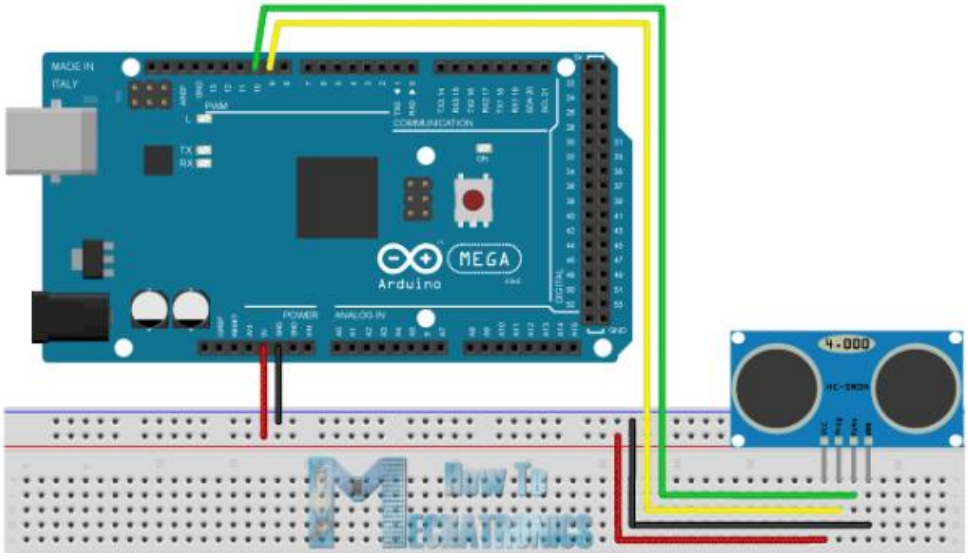
- Conectarea modulului WIFI



-Conectarea placilor Arduino Mega 2560 si Arduino Uno

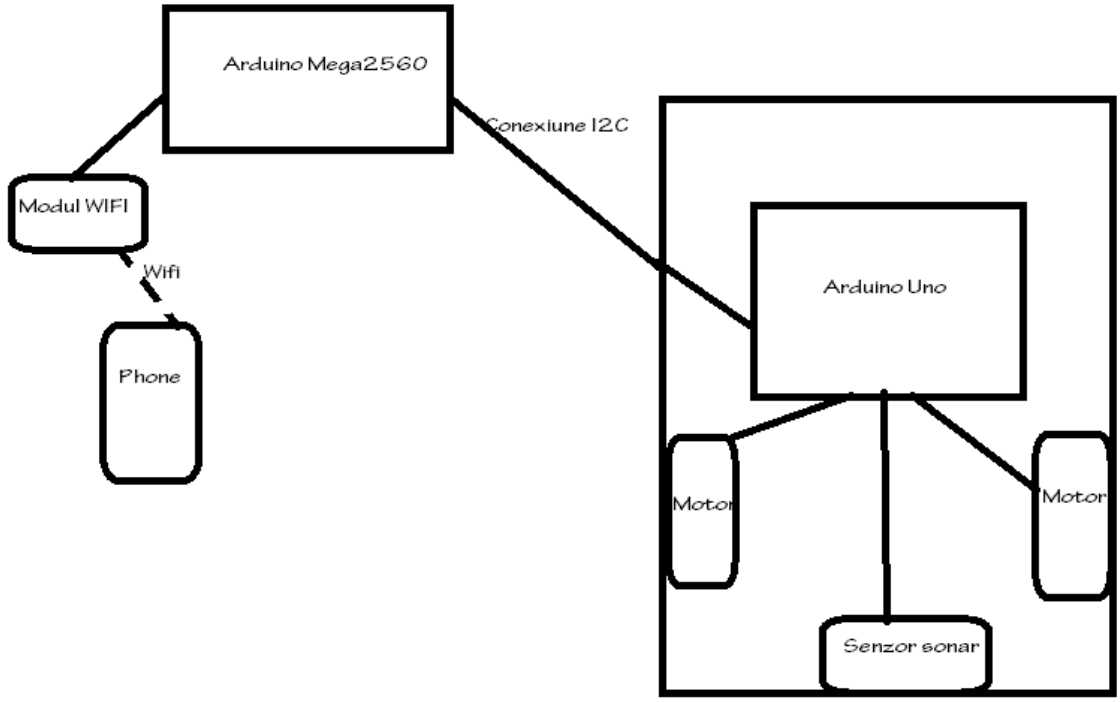


- Conectarea senzorului sonar



3. Constitutia generala a proiectului

Schema bloc a proiectului este prezentata in continuare:



Proiectul functioneaza astfel:

-Placa Arduino Mega e conectata cu modulul WIFI, care prin intremediul unei aplicatii Android, receptioneaza comenzile robotului. Printr-o conexiune I2C, transmite datele necesare miscarii placii Arduino Uno.

-Placa Arduino Uno e responsabila de functionarea motoarelor, de receptionarea comenzilor prin I2C si de receptionarea semnalelor de la senzorul ultrasonic.

Astfel, robotul se misca pe baza comenzilor din telefon, dar in cazul in care distanta fata de un obstacol pe directia inainte e mai mica de 10 cm, se opreste, pentru a evita coliziunile.

4. Aplicatia Android pentru controlul WIFI al robotului

Dupa conectarea la reteaua WIFI indicata in Serial Monitor, se porneste aplicatia si se introduce adresa serverului web prin care modulul wifi functioneaza, si anume: 192.168.4.1. Utilizand butoanele aplicatiei se poate face controlul robotului.

5. Codul Arduino pentru proiect

a.) Codul pentru programul Master (asociat pentru placa Arduino Mega 2560, impreuna cu modulul WIFI, si partea de I2C, specifica programului Master).

```
#define DEBUG true

#include <Wire.h>
```

```
void setup() {
 // Dechidem magistrala I2C ca master
Wire.begin();
// Starts the serial communications
Serial.begin(115200);
Serial1.begin(115200);
//Function calls for wifi module
sendData("AT+RST\r\n", 2000, false); // resetare modul
sendData("AT+CWMODE=2\r\n", 1000, false); // configurare ca access point
sendData("AT+CIFSR\r\n", 1000, DEBUG); // citeste adresa IP
sendData("AT+CWSAP?\r\n", 2000, DEBUG); // citeste informatia SSID (nume retea)
sendData("AT+CIPMUX=1\r\n", 1000, false); // configurare conexiuni multiple
sendData("AT+CIPSERVER=1,80\r\n", 1000, false); // pornire server pe port 80
}

String sendData(String command, const int timeout, boolean debug) {

 String response = "";
 Serial1.print(command); // trimite comanda la esp8266
 long int time = millis();
 while ((time + timeout) > millis()) {
  while (Serial1.available()) {
  char c = Serial1.read(); // citeste caracterul urmator
  response += c;
  }
 }
 Wire.beginTransmission(9);
 if (response.indexOf("/f") != -1) {
 Wire.write('f');
 }
  if (response.indexOf("/s") != -1) {
```

```
 Wire.write('s');
 }
 if (response.indexOf("/b") != -1) {
 Wire.write('b');
 }
  if (response.indexOf("/l") != -1) {
 Wire.write('l');
 }
  if (response.indexOf("/r") != -1) {
 Wire.write('r');
 }
 Wire.endTransmission(); // oprim transmisia
 if (debug) {
 Serial.print(response);
 }
 return response;
}

void loop() {
 if (Serial1.available()) {
   if (Serial1.find("+IPD,")) {
     delay(500);
     int connectionId = Serial1.read() - 48;
     String webpage = "<h1>Gabi's car!</h1>";
     String cipSend = "AT+CIPSEND=";
     cipSend += connectionId;
     cipSend += ",";
     cipSend += webpage.length();
     cipSend += "\r\n";
     sendData(cipSend, 100, DEBUG);
     sendData(webpage, 150, DEBUG);
```

```
    String closeCommand = "AT+CIPCLOSE=";
    closeCommand += connectionId; //se adauga identificatorul conexiunii
    closeCommand += "\r\n";
    sendData(closeCommand, 300, DEBUG);
  }
 }
}
```

b.) Codul pentru programul Slave (asociat pentru placa Arduino Uno, impreuna cu platforma robotica experimentala, alaturi de partea de I2C specifica pentru programul Slave).

```
#include <Wire.h>
// Pinii motor 1
#define mpin00 5
#define mpin01 6
// Pinii motor 2
#define mpin10 10
#define mpin11 11
const int trigPin = 9;
const int echoPin = 12;
// defines variables
long duration;
int distance;
char x;

void setup() {
 Wire.begin(9);
 // Atasam o functie care sa se declanseze atunci cand primim ceva
 Wire.onReceive(receiveEvent);
 analogReference(DEFAULT);
 digitalWrite(mpin00, 0);
```

```
 digitalWrite(mpin01, 0);
 digitalWrite(mpin10, 0);
 digitalWrite(mpin11, 0);
 pinMode (mpin00, OUTPUT);
 pinMode (mpin01, OUTPUT);
 pinMode (mpin10, OUTPUT);
 pinMode (mpin11, OUTPUT);
pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
pinMode(echoPin, INPUT); // Sets the echoPin as an Input
pinMode(A5, INPUT);
digitalWrite(A5,HIGH);
pinMode(A4, INPUT);
digitalWrite(A4,HIGH);
Serial.begin(115200); // Starts the serial communication
}

void receiveEvent(int bytes) {
 x = Wire.read(); // citim un character din I2C
}

void loop() {
 Serial.print(x);
 if (x=='f') {
myForward(mpin00,mpin01,mpin10, mpin11);
 }
  if (x=='s') {
myStop(mpin00,mpin01,mpin10, mpin11);
 }
 if (x=='b') {
myBack(mpin00,mpin01,mpin10, mpin11);
 }
```

```
  if (x=='l') {
myLeft(mpin00,mpin01,mpin10, mpin11);
 }
 if (x=='r') {
myRight(mpin00,mpin01,mpin10, mpin11);
 }
 readDistance();
 if(distance < 10){
   myStop(mpin00,mpin01,mpin10, mpin11);
 }
}

void readDistance(){
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculating the distance
  distance= duration*0.034/2;
  // Prints the distance on the Serial Monitor
  Serial.print("Distance: ");
  Serial.println(distance);
}

void myForward(int m1, int m2, int m3, int m4){
 digitalWrite(m2, 0);
 analogWrite (m1, 100);
```

```
 digitalWrite(m4, 0);
 analogWrite (m3, 100);
}

void myBack(int m1, int m2, int m3, int m4){
 digitalWrite(m1, 0);
 analogWrite (m2, 100);
 digitalWrite(m3, 0);
 analogWrite (m4, 100);
}

void myStop(int m1, int m2, int m3, int m4){
 digitalWrite(m2, 0);
 digitalWrite(m1, 0);
 digitalWrite(m3, 0);
 digitalWrite(m4, 0);
}

void myLeft(int m1, int m2, int m3, int m4){
 digitalWrite(m1, 0);
 analogWrite (m2, 0);
 digitalWrite(m4, 0);
 digitalWrite(m3, 80);
}

void myRight(int m1, int m2, int m3, int m4){
 digitalWrite(m2, 0);
 digitalWrite (m1, 80);
 digitalWrite(m3, 0);
 analogWrite (m4, 0);
}
```

6. Referinte

- http://users.utcluj.ro/~negrum/src/html/dmp.html
- http://ai2.appinventor.mit.edu
- https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf
- https://www.instructables.com/id/ESP8266-Wifi-Controlled-Robot/