2021

Station for checking soil moisture

STUDENT: STIOIU DENIS ADRIAN

Cuprins

1. Concept

The project uses a soil moisture sensor to measure the amount of water inside the soil, and provide the moisture level (or the level of dryness) of the soil. The output of the sensor is sent through the I2C communication protocol to another Arduino board, which sends the value to a local server using an Ethernet shield, and displays the result on a simple HTML page.

Hardware required:

- Arduino UNO development board
- Arduino MEGA 2560 development board
- HM Sensor Series moisture sensor
- Ethernet Shield
- Ethernet Cable
- 2 x USB cable to connect the boards to the computer and power source
- 9 x Jump Wires to connect the components

The I2C communication is used to share the data between the development boards, Arduino UNO board acting as the slave sender, and Arduino MEGA acting as the master reader. I used the Wire.h library's functions to help me send the information. The slave send 6 bytes of information to the master which sends them on a server. Once the moisture percent is received it will be displayed on the page. The I2C protocol involves using two lines to send and receive data: a serial clock pin SCL (corresponding to pin A5 on Arduino UNO and pin D21 on MEGA), and a serial data pin SDA (corresponding to A4 on Arduino UNO and pin D20 on MEGA), and connecting the grounds of the two boards.

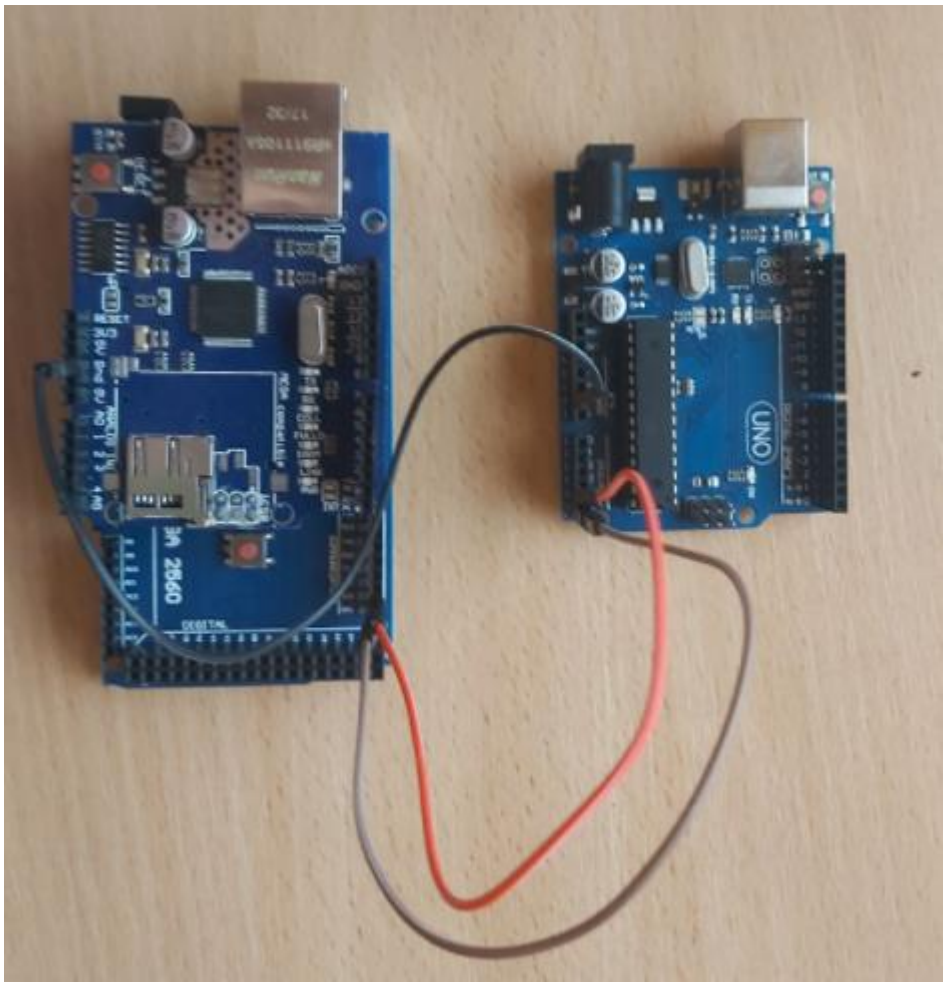
The Arduino MEGA has the Ethernet Shield over it, and an Ethernet cable connected to it in order to send the information to the server. We need to set up a MAC address for the shield(it can be any address as long as it is not used anywhere else in the network) and add the ip address of the network where the page will be loaded. Then we will write the server in the Arduino IDE and create the HTML page.

The sensor has one U-shaped part that is introduced in the soil to measure the moisture and the main part that is connected to the Arduino UNO. We connect the sensor as it follows: the analog pin A0 of the sensor to the A0 pin of the board, the digital pin D0 of the sensor to the D0 pin of the board, and the ground and Vcc to their counterparts on the board.

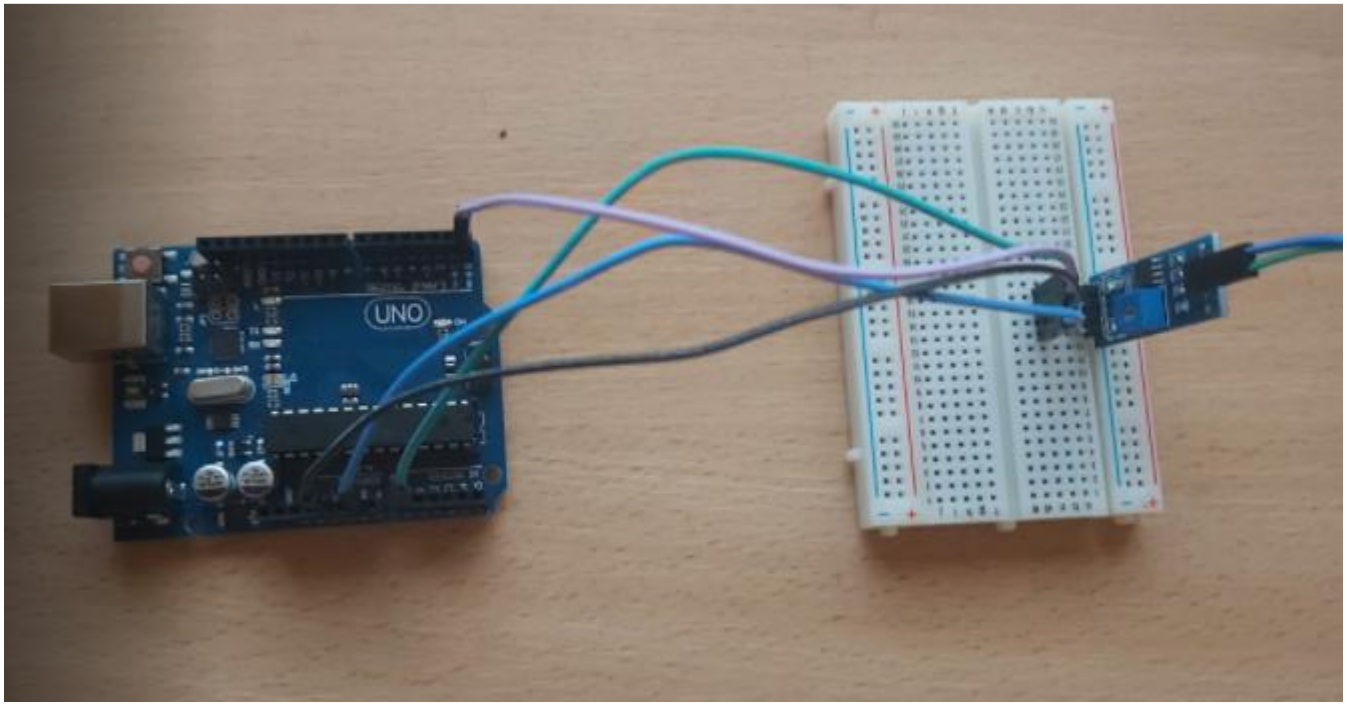
The information regarding the code and the diagrams are to be seen in the following two chapters.

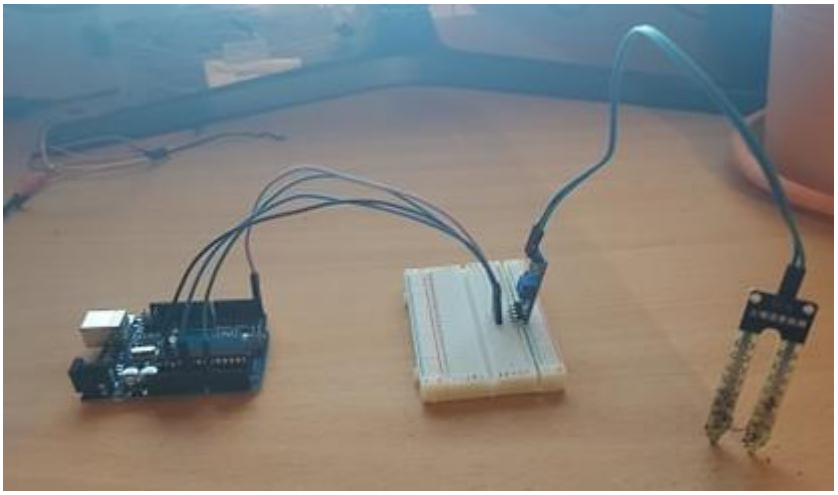
2. Diagrams and connections

The connection between the two boards using the I2C communication protocol:

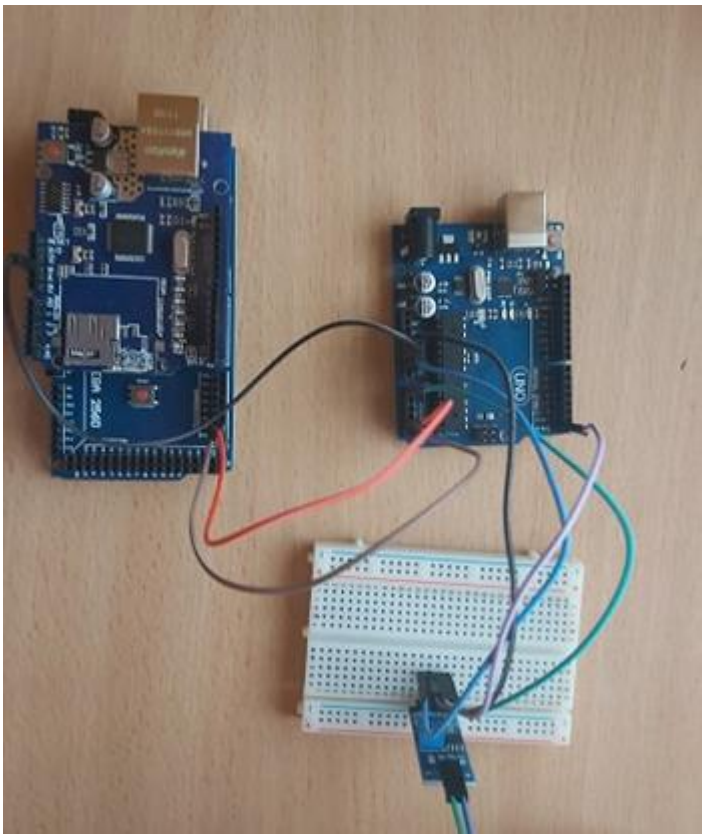


The connection between Arduino UNO and the moisture sensor:

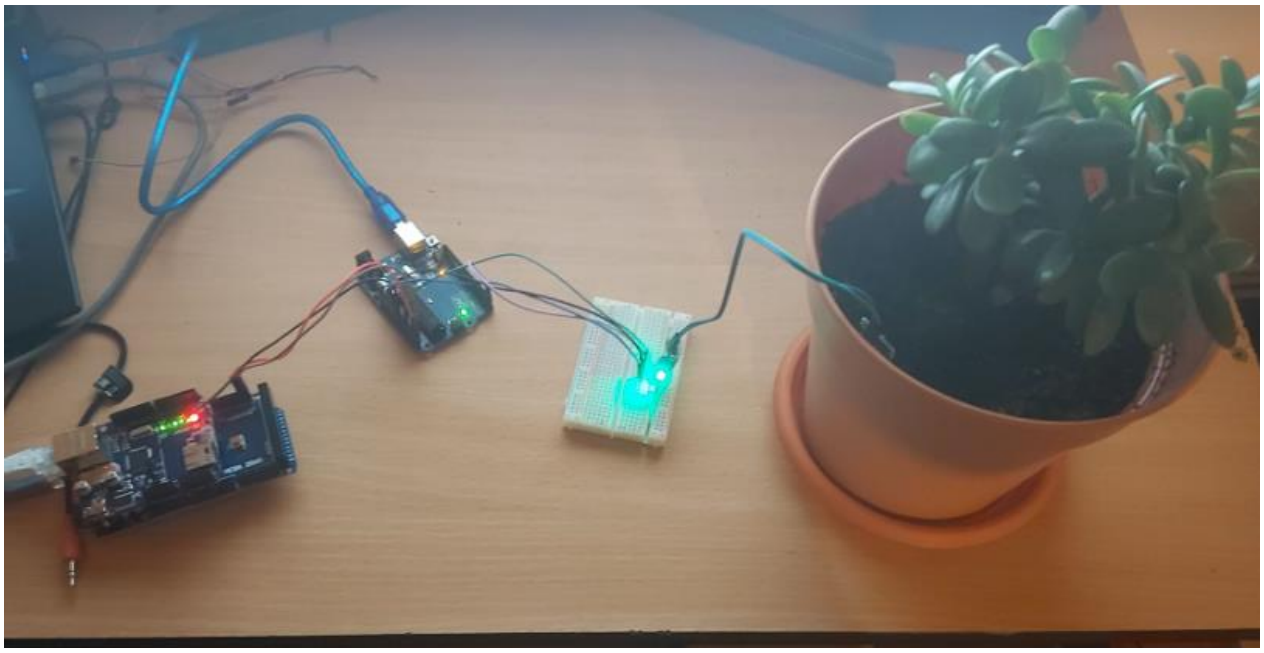




The whole project:



The project working:



3. Code

The slave sender:

```
#include <Wire.h>

//The pin from where we read the value
#define sensorPin A0


float sensorValue = 0;
char result[8];

void setup()
{
  Wire.begin(4);  // the address of the master board where the information will be sent
  Wire.onRequest(requestEvent);//the register event
}

void loop()
{
  delay(100);
}

//The events sends data to the master whenever it is requested
void requestEvent(){
  // We read the value from the pin
  sensorValue = analogRead(sensorPin);
  //The values of the HM sensor come from 0 to 1023 so we map them in order to use them as percents
  sensorValue = map(sensorValue, 0,1023,0,100);
  //We convert the value to a string so it is easier to send
  dtostrf(sensorValue,6,2,result);
  //Send the data through the bus
  Wire.write(result);
}
```

The master reader and server:

```
#include<Wire.h>
#include <SPI.h>
#include <Ethernet.h>

char moisture[6];
// The mac address of the shield
byte mac[] = {
 0x00, 0xAA, 0xBB, 0xCC, 0xDE, 0x02
};

// The ip address of the server
IPAddress ip(169,254,10,222);
// We initialise the server on port 80

EthernetServer server(80);
void setup(){
  //We start the connection with the other board
  Wire.begin();
  Serial.begin(9600);


  Serial.println("Project initialization");

  // We start the ethernet connection and initialize the server
  Ethernet.begin(mac, ip);

  // We check for the ethernet shield to be connected
  if (Ethernet.hardwareStatus() == EthernetNoHardware) {
    Serial.println("Ethernet shield was not found.");
    while (true) {
      delay(1);
    }
  }

  // We check for the ethernet cable to be connected
  if (Ethernet.linkStatus() == LinkOFF) {
    Serial.println("Ethernet cable is not connected.");
  }

  // We start the server and display the address to access it
  server.begin();
  Serial.print("Server is at the address: ");
  Serial.println(Ethernet.localIP());
}
```

```
void loop(){
  //We request 6 bytes from the slave board(maximum in the case the humidity is 100.00)
  Wire.requestFrom(4,6);
  //There can be fewer bytes sent though and we store them inside a string
  while(Wire.available()){
   for(int i = 0;i<6;i++){
    moisture[i] = Wire.read();
   }
  }
  delay(500);
  // We start the client only if the server is available
  EthernetClient client = server.available();
  if (client) {
    Serial.println("Client connected");
    //A http request ends with a blank line
    boolean currentLineIsBlank = true;
    // If the client is connected then we can send the data
    while (client.connected()) {
      if (client.available()) {
        char c = client.read();
        Serial.write(c);
        //If we have gotten to the end of the line and the line is blank, than the http request has ended so we can send a reply to the server
        if (c == '\n' && currentLineIsBlank) {
          // Create the simple HTML page
          client.println("HTTP/1.1 200 OK");
          client.println("Content-Type: text/html");
          client.println("Connection: close");  // After the response the connection will be closed
          client.println("Refresh: 5");  // Refresh the page every 5 seconds
          client.println();
          client.println("<!DOCTYPE HTML>");
          client.println("<html>");
          //Display the moisture level
          client.print("<b>Moisture level: ");
          client.print(moisture);
          client.println("%</b><br />");
          client.println("</html>");
          break;
        }
        if (c == '\n') {
          // We start a new line
          currentLineIsBlank = true;
        } else if (c != '\r') {
          //There is a character on the new line
          currentLineIsBlank = false;
```

```
        }
      }
    }
    // We delay so the server can recieve the data
    delay(1);
    // We close the connection
    client.stop();
    Serial.println("Client disconnected");
  }
}
```

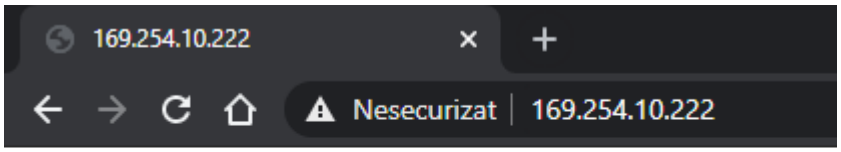
4. Results from the simulation

The message from the serial monitor that the server has started and a client connected:

```
Project initialization
Server is at the address: 169.254.10.222
Client connected
GET / HTTP/1.1
Host: 169.254.10.222
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: ro-RO,ro;q=0.9,en-US;q=0.8,en;q=0.7

client disconnected
```

The page displaying the result:



Moisture level: 80.00%