Co-funded by the Erasmus+ Programme of the European Union

BUILDING A ROBOTIC ARM WITH A TEMPERATURE PROBE

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Robotic Arm with a Temperature Probe

Equipment List:

Arduino Uno

5 V Power Supply

Groove RGB LCD Screen

220-ohm resistor

Jumper Wires

4 x Servo

4 x Rotary Potentiometer

DS18B20 1-Wire Digital Temperature Sensor

Methodology

This project can be separated into three subsections, where then we will combine it at the end to show how the elements can work together. For the purpose of this explanation, we are going to use an open-source project for designing the robotic arm, called uFactory but in reality, any other one can be used since they use the same principle.

1. Hardware for the Robotic Arm

The most critical part for this project to be successful is the implementation of a robotic arm. For the purpose of this project, we are going to use an open-source design, which can be 3D printed to simplify matters.

Below is a link to an open source Robotic Arm Designs. This is to cover the physical design of the robotic arm.

UFactory: https://www.ufactory.cc/?gclid=CjwKCAjwsJ3ZBRBJEiwAtuvtlAy4GE087c-t4VmAOyyVVlXL7j3ZOg9c7-LNJ1x_lcBRGOlYuTCQ_BoCHFoQAvD_BwE#/en/uarmswift

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

2. Connecting a Servo

Servos are quite power consuming components and depending upon the servo used, the power consumption is different for each one so depending upon the one used, caution needs to be done to know which one to use. The best method to avoid problems is to use a separate power supply 5-6V DC is good (as a hint, since these are problematic to find or else quite expensive, you can use the power supply from a computer since most are powerful enough to power several servos at the same time). Figure 1 shows the configuration on how to connect it.

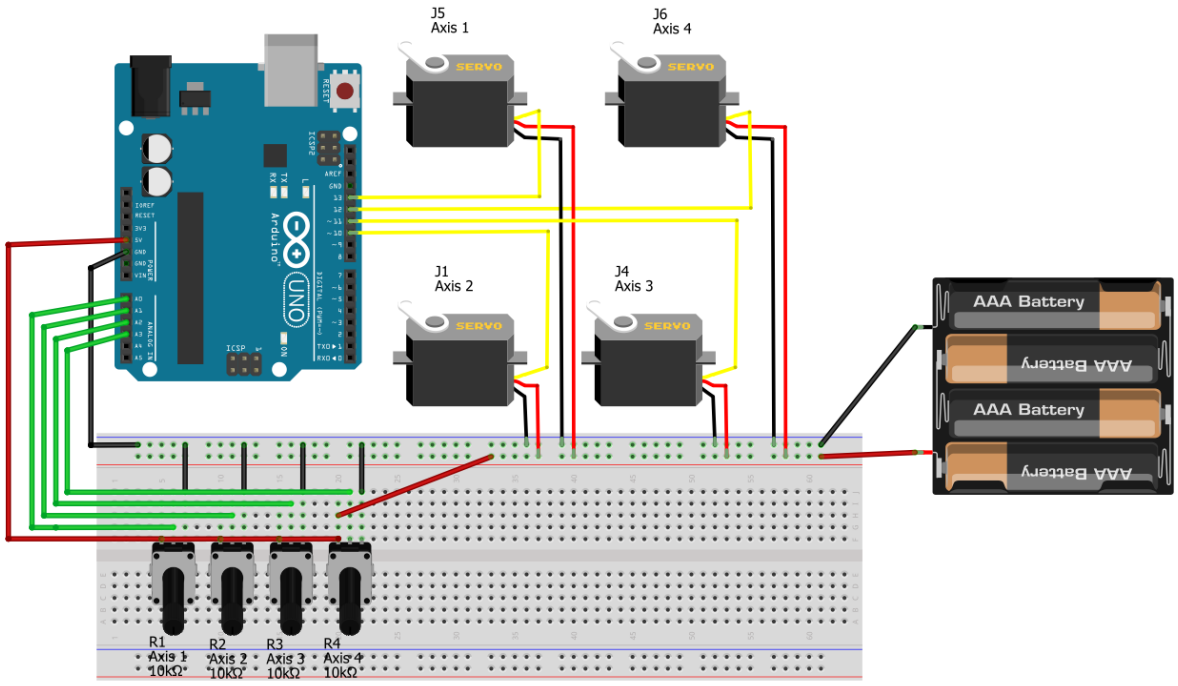


Figure 1 - Servo and Potentiometers

After configuring the hardware above, the servos are powered on but the Arduino still needs to be set up with the software. To check the configuration, load the file “Servo Controllers Only”.

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Below is an explanation of what the code does with the comments added in explaining what each part does what.

```
#include <Servo.h> //Library for Servo

Servo Axis1; //Define a Servo
Servo Axis2;
Servo Axis3;
Servo Axis4;



//Define Register for Integer. This is used to contain the value of the potentiometers
int Axis1Pot;
int Axis1val;
int Axis2Pot;
int Axis2val;
int Axis3Pot;
int Axis3val;
int Axis4Pot;
int Axis4val;

void setup() {
  Axis1.attach(8); //Define which servo is attached to which pin upon the Arduino Board
  Axis2.attach(9);
  Axis3.attach(10);
  Axis4.attach(11);
}

void loop() {
   Axis1val = analogRead(A0); //Read values from the Arduino Pin
   Axis2val = analogRead(A1);
   Axis3val = analogRead(A2);
   Axis4val = analogRead(A3);

   Axis1Pot = map(Axis1val,0,1023,0,180); //Map values from "0to1024" to "0to180"
   Axis2Pot = map(Axis2val,0,1023,15,165);
   Axis3Pot = map(Axis3val,0,1023,0,180);
   Axis4Pot = map(Axis4val,0,1023,0,180);

   Axis1.write(Axis1Pot); //Output value upon Servos
   Axis2.write(Axis2Pot);
   Axis3.write(Axis3Pot);
   Axis4.write(Axis4Pot);
}
```

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

3. Connecting the LCD & Temperature Sensor

The temperature sensor is used to give real-time feedback of the medium being probed. In this scenario we are going to use a DS18B20 temperature sensor. For this sensor, we will use a pull-up resistor (Resistor which is connected in parallel with the sensor line to the VDD, this is to stabilize the signal).

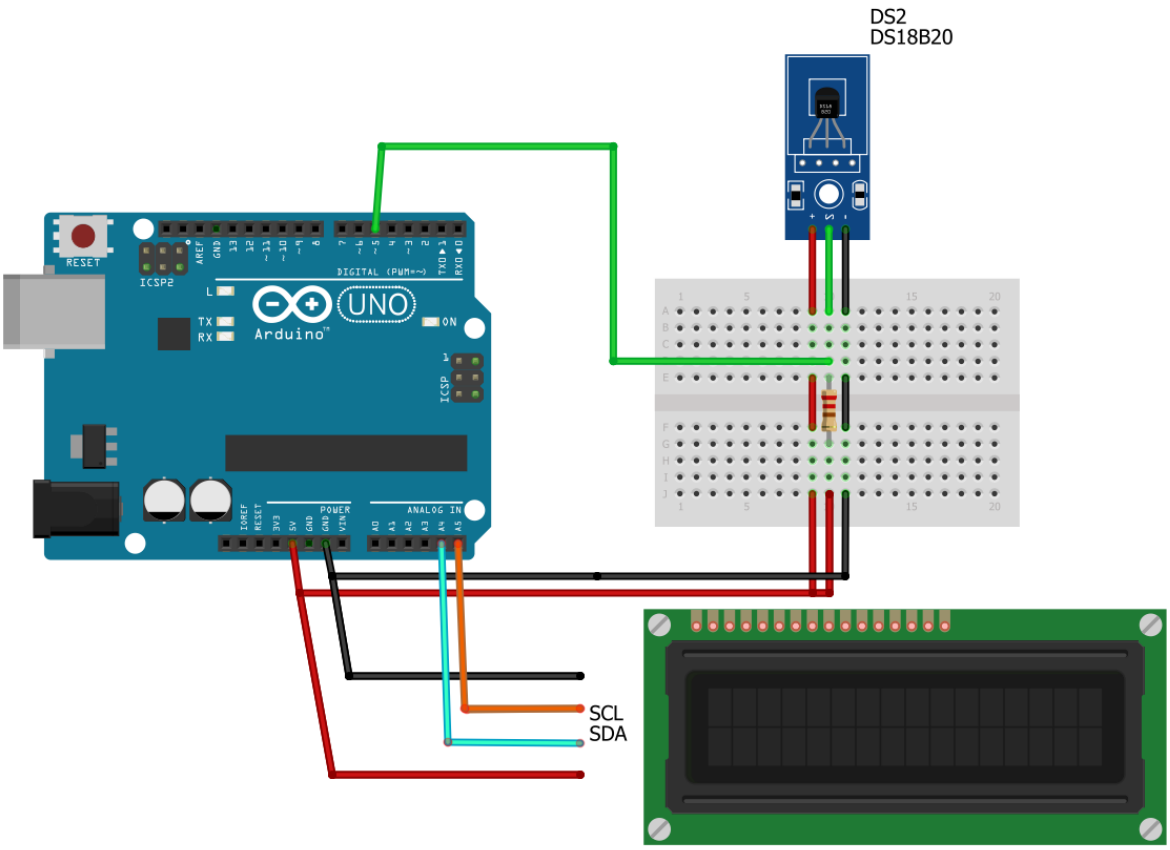


Figure 2 LCD & Temperature Sensor

Since the configuration is different, we will use the following code to make it operational.

```
#include <OneWire.h>
#include <DallasTemperature.h>
#include <Wire.h>
#include "rgb_lcd.h"

#define ONE_WIRE_BUS 5
//define that temperature sensor will use pinout 5

OneWire oneWire(ONE_WIRE_BUS);

DallasTemperature sensors(&oneWire);

float Celcius=0;
float Fahrenheit=0;
```

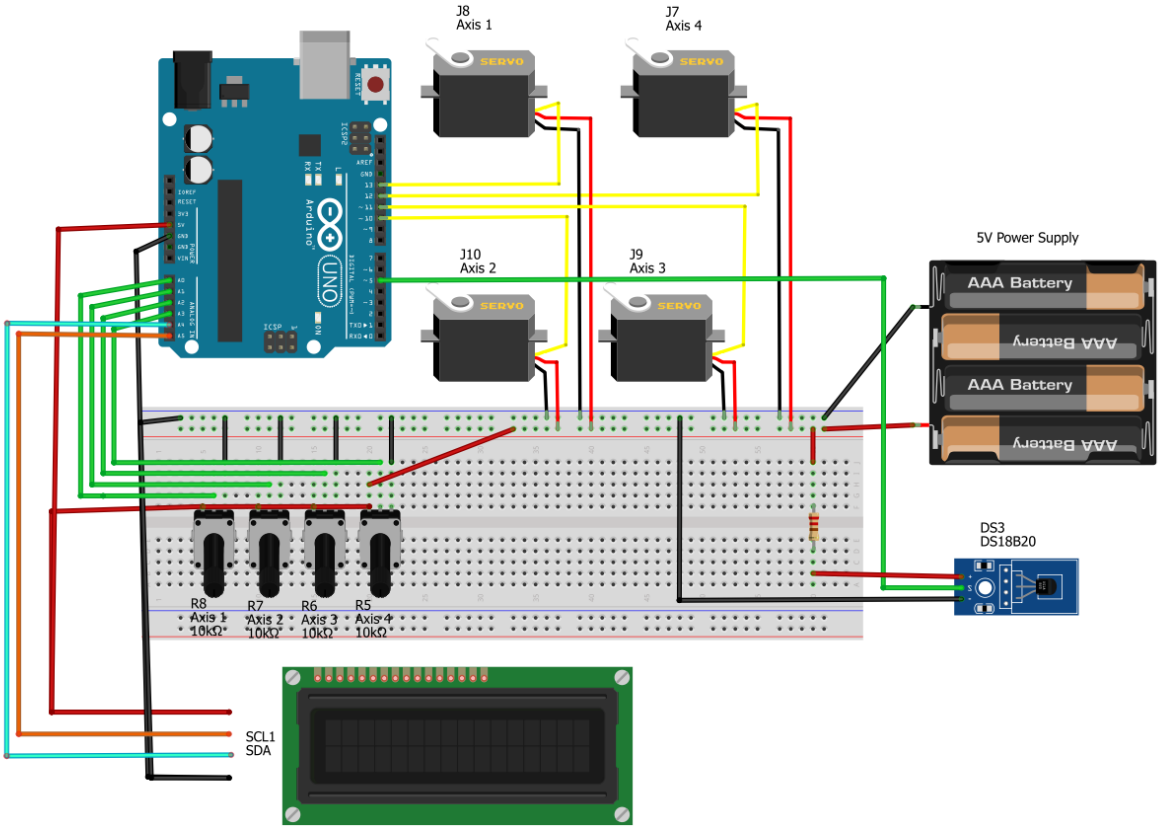
This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
rgb_lcd lcd;

const int colorR = 255;
const int colorG = 0;
const int colorB = 0;

void setup(void)
{
  Serial.begin(9600);
  sensors.begin();

  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);

  lcd.setRGB(colorR, colorG, colorB);

  delay(1000);
}


void loop(void)
{
  sensors.requestTemperatures();
  Celcius=sensors.getTempCByIndex(0);
  Fahrenheit=sensors.toFahrenheit(Celcius);
  Serial.print(" C  ");
  Serial.print(Celcius);
  Serial.print(" F  ");
  Serial.println(Fahrenheit);

  // set the cursor to column 0, line 0
  lcd.setCursor(0, 0);
  lcd.print(" C: ");
  lcd.print(Celcius);

  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(0, 1);
  lcd.print(" F: ");
  lcd.print(Fahrenheit);

  delay(100);
}
```

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

4. Putting all together

The sections above were shown separately so that it is easier for fault finding since these are quite complex circuits and mistakes are bound to happen. The sections above use the same configuration as shown as the image below. The image below is just shown to show that it can be combined into one circuit.



The code is also the same as the other sections but needs to be combined into one. The following is the code which needs to be used, it is also in the file “Case Study 1”.

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
#include <Servo.h> //Library for Servo
#include <OneWire.h>
#include <DallasTemperature.h>
#include <Wire.h>
#include "rgb_lcd.h"

#define ONE_WIRE_BUS 5

OneWire oneWire(ONE_WIRE_BUS);

DallasTemperature sensors(&oneWire);

float Celcius=0;
float Fahrenheit=0;

rgb_lcd lcd;

const int colorR = 138;
const int colorG = 43;
const int colorB = 226;

Servo Axis1; //Define a Servo
Servo Axis2;
Servo Axis3;
Servo Axis4;

int Axis1Pot; //Define Register for Integer
int Axis1val;
int Axis2Pot;
int Axis2val;
int Axis3Pot;
int Axis3val;
int Axis4Pot;
int Axis4val;


void setup() {
  Serial.begin(9600);
  sensors.begin();

  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);

  lcd.setRGB(colorR, colorG, colorB);

  Axis1.attach(8); //Define which servo is attached to which pin upon the Arduino Board
  Axis2.attach(9);
  Axis3.attach(10);
  Axis4.attach(11);
}
```

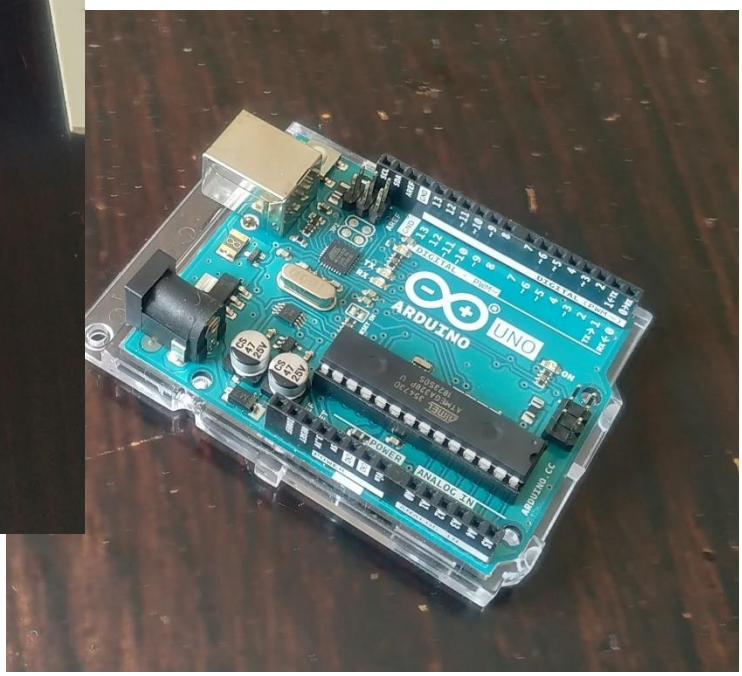
This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
void loop() {
  sensors.requestTemperatures();
  Celcius=sensors.getTempCByIndex(0);
  Fahrenheit=sensors.toFahrenheit(Celcius);
  Serial.print(" C  ");
  Serial.print(Celcius);
  Serial.print(" F  ");
  Serial.println(Fahrenheit);

  // set the cursor to column 0, line 0
  lcd.setCursor(0, 0);
  lcd.print(" C: ");
  lcd.print(Celcius);

  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(0, 1);
  lcd.print(" F: ");
  lcd.print(Fahrenheit);

  Axis1val = analogRead(A0); //Read values from the Arduino Pin
  Axis2val = analogRead(A1);
  Axis3val = analogRead(A2);
  Axis4val = analogRead(A3);

  Axis1Pot = map(Axis1val,0,1023,0,180); //Map values from "0to1024" to "0to180"
  Axis2Pot = map(Axis2val,0,1023,15,165);
  Axis3Pot = map(Axis3val,0,1023,0,180);
  Axis4Pot = map(Axis4val,0,1023,0,180);

  Axis1.write(Axis1Pot); //Output value upon Servos
  Axis2.write(Axis2Pot);
  Axis3.write(Axis3Pot);
  Axis4.write(Axis4Pot);
}
```

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

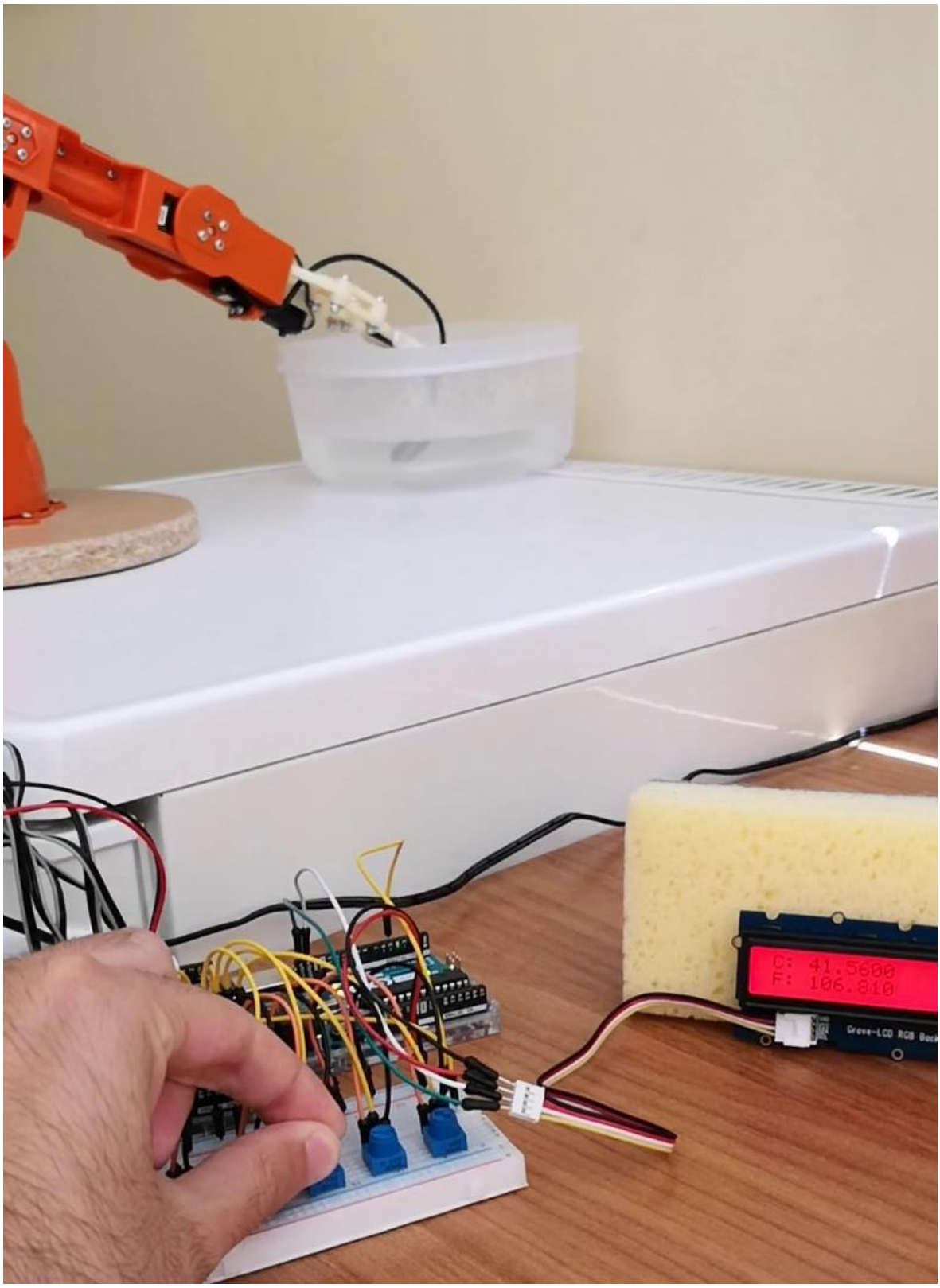
5. Finished Project

Below are some images showing the project how its working and to showcase the work involved.



This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

The Image below shows the robotic arm working. The robot arm is holding a temperature probe and is lowering it into the water where it is then displayed upon the LCD



This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.