Testing process for components in the Chemical plant project
Embedded and Ubiquitous systems
Task part of the second sprint
UDL - MINF - 2021 - Team 1

Content:

1. Objective
2. List of components
3. Tests done

Objective

The main objective of the task is to ensure that every component of the project is working, doing simple tests.

List of components

1x Raspberry Pi
1x Arduino UNO
3x ESP-01
1x LCD
1x DHT11 sensor
1x Ultrasonic sensor (HC-SR04)
1x PCF
1x I2c module

Components already tested while developing: Arduino and Raspberry, so they won't need a specific test method.

Update: components left to test is PCF and I2c module.

Tests done

1. **LCD**

First you need to include the library for working with the LCD <SparkFunSerialGraphicLCD.h>. (via the Arduino IDE)

The test will be taken using the arduino as controller with the following code:

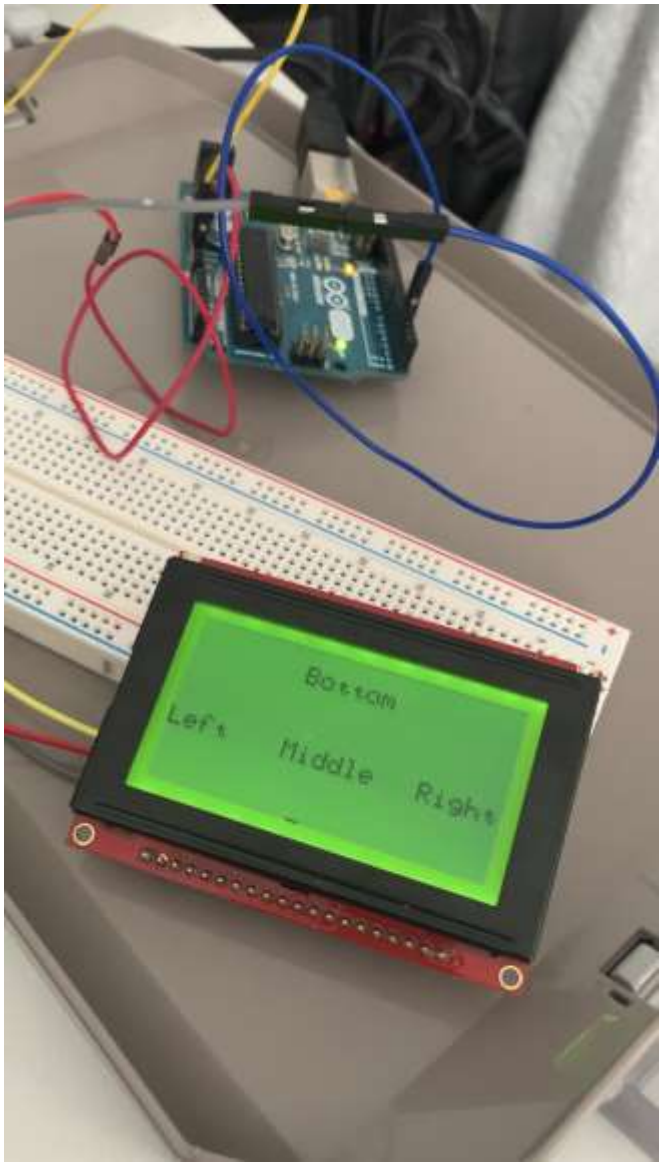
```
#include <SparkFunSerialGraphicLCD.h> //include  the Serial Graphic LCD library
#include <SoftwareSerial.h>

#define maxX 127//159
#define maxY 63 //127

LCD LCD;

void setup() {
 // put your setup code here, to run once:

 LCD.setHome();//set the cursor back to 0,0.
 LCD.clearScreen();//clear anything that may have been previously printed ot the screen.

 LCD.printStr("MINF UDL");
 delay(5000);

 LCD.setX((maxX/2)-18);
 LCD.setY((maxY/2)-4);
 LCD.printStr("Master Informatic Engineering");
}

void loop() {
 // put your main code here, to run repeatedly:

}
```

Electrical wiring:

Arduino	Graphic LCD
5V	Vin
GND	GND
Pin 3	RX

It will write some things based on the position (X,Y).

Photo:



Conclusion: This test was enough to start knowing the LCD behaviour and also ensuring it's working.

2. **ESP-01**

In this test we will program the ESP-01 to act as a wifi server. (We must repeat this test for the 3 ESP-01).

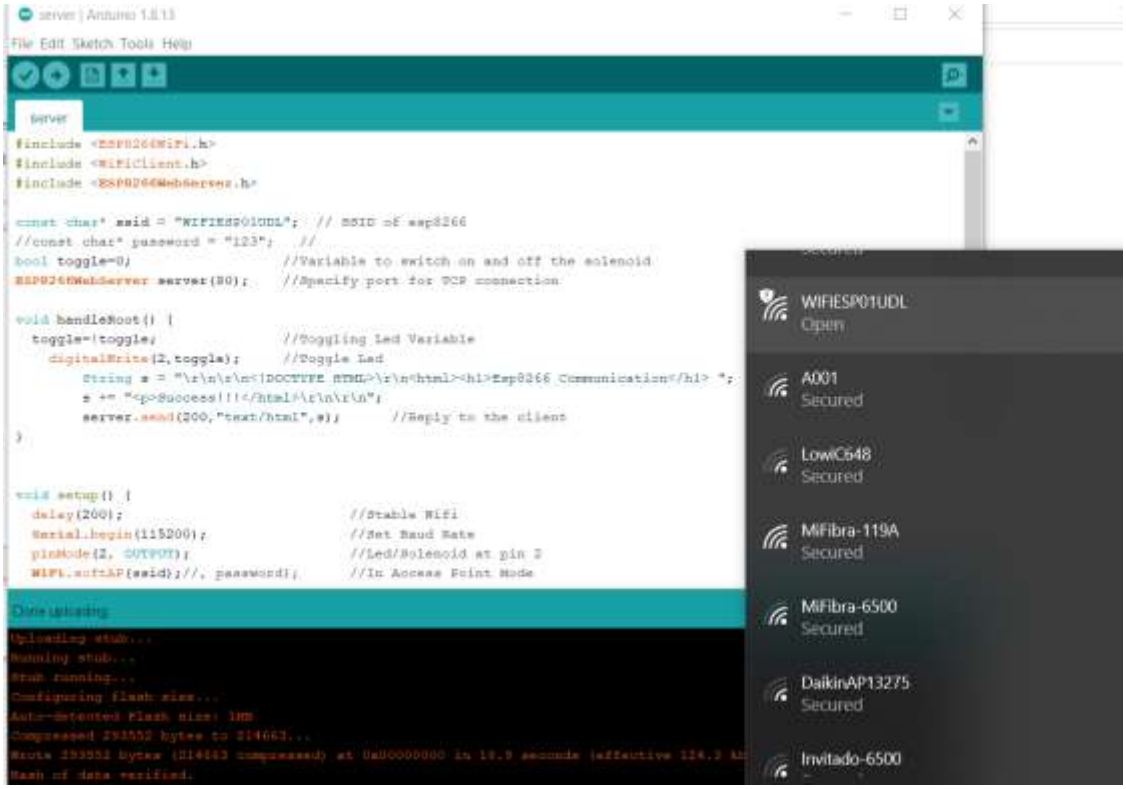
1. On the adapter, switch to the PROG mode (the other is UART)
2. Connect the adapter on computer and install its drivers:
   USB Adapter Driver: http://www.wch.cn/download/CH341SER_EXE.html
3. After that, you can connect your ESP-01 on it and plug on the computer
4. In the Arduino IDE go Tools > Board > ESP8266 boards and then select Generic ESP8266 Module.
5. Tools > Port > Select the port the USB is in (check in the devices page of windows)
6. Now you can Verify and then compile your code.

a. Here you have a simple code for blinking the led (blue) of the ESP8266:
b. https://learn.sparkfun.com/tutorials/esp8266-thing-hookup-guide/example-sketch-blink

Code for the testing:

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>

const char* ssid = "WIFIESP01UDL";  // SSID of esp8266
//const char* password = "123";  //
bool toggle=0;                  //Variable to switch on and off the solenoid
ESP8266WebServer server(80);                  //Specify port for TCP connection

void handleRoot() {
 String s = "\r\n\r\n<!DOCTYPE HTML>\r\n<html><h1>Esp8266 Communication</h1> ";
 s += "<p>Success!!!</html>\r\n\r\n";
 server.send(200,"text/html",s);                //Reply to the client
}


void setup() {
 delay(200);                    //Stable Wifi
 Serial.begin(115200);          //Set Baud Rate
 //pinMode(2, OUTPUT);                       //Led/Solenoid at pin 2
 WiFi.softAP(ssid);//, password);              //In Access Point Mode

 IPAddress myIP = WiFi.softAPIP();             //Check the IP assigned. Put this Ip in the client host.
 Serial.print("AP IP address: ");
 Serial.println(myIP);          //Print the esp8266-01 IP(Client must also be on the save IP series)
 server.on("/Led", handleRoot);                //Checking client connection
 server.begin();                // Start the server
 Serial.println("Server started");
}

void loop() {
 // Check if a client has connected. On first connection switch on the Solenoid on next switch off.
 server.handleClient();
}
```

Photos:



Conclusion: This test was enough to ensure the functionality of the ESP-01.

3. DHT11

In this test we will see if our temperature sensor component (DHT11 is working).

First, we need to add the following libraries on the Arduino IDE: https://github.com/adafruit/DHT-sensor-library
and
https://github.com/adafruit/Adafruit_Sensor

You can do that manually or going into the Arduino IDE > Tools > Manage Libraries > and then search and install the libraries.

Code:

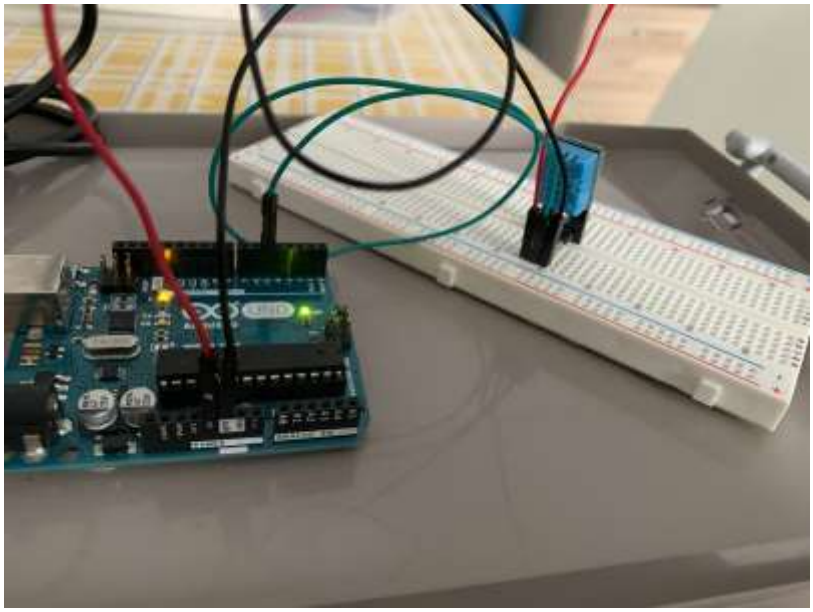
```
#include "DHT.h"

// Uncomment whatever type you're using!
#define DHTTYPE DHT11   // DHT 11
//#define DHTTYPE DHT22  // DHT 22  (AM2302), AM2321
//#define DHTTYPE DHT21  // DHT 21 (AM2301)

// Connect pin 1 (on the left) of the sensor to +5V
// NOTE: If using a board with 3.3V logic like an Arduino Due connect pin 1
// to 3.3V instead of 5V!
// Connect pin 2 of the sensor to whatever your DHTPIN is
// Connect pin 4 (on the right) of the sensor to GROUND
// Connect a 10K resistor from pin 2 (data) to pin 1 (power) of the sensor

const int DHTPin = 5;           // what digital pin we're connected to
```

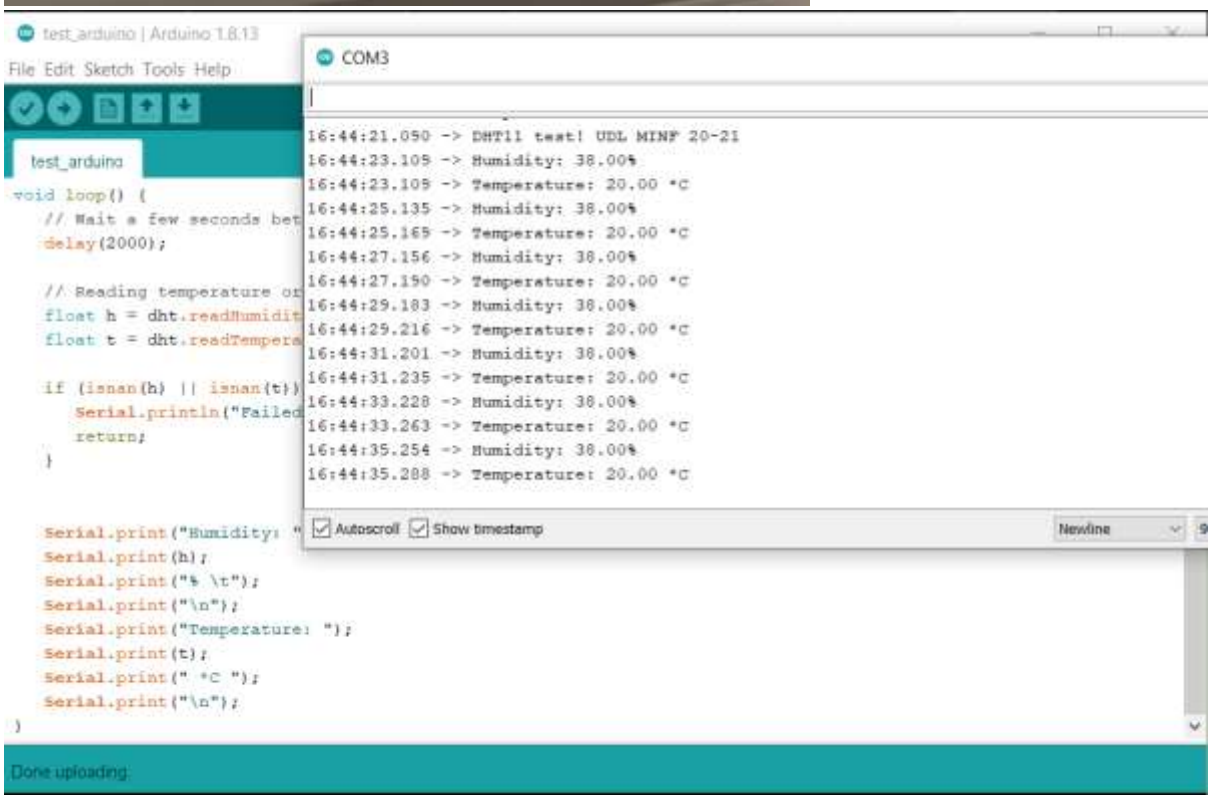
```
DHT dht(DHTPin, DHTTYPE);

void setup() {
  Serial.begin(9600);
  Serial.println("DHT11 test! UDL MINF 20-21");

  dht.begin();
}

void loop() {
  // Wait a few seconds between measurements.
  delay(2000);

  // Reading temperature or humidity takes about 250 milliseconds!
  float h = dht.readHumidity();
  float t = dht.readTemperature();

  if (isnan(h) || isnan(t)) {
                  Serial.println("Failed to read from DHT sensor!");
                  return;
  }


  Serial.print("Humidity: ");
  Serial.print(h);
  Serial.print("% \t");
  Serial.print("\n");
  Serial.print("Temperature: ");
  Serial.print(t);
  Serial.print(" *C ");
  Serial.print("\n");
}
```

Electrical Wiring:

DHT11 Vcc -> Arduino 5V
DHT11 Output ->one of the digital pins of Arduino (5 per example)
DHT11 GND ->Arduino GND (its on the side of the 5V)

Photos:





Conclusion: This test showed us that the sensor is functioning correctly and showing the values as it should.

4. HC-SR04

The testing was done through the data producer 2 development, all the process is in its respective documentation.

5. PCF

Test pending.

6. I2c module

Test pending.