University of Lleida

Master's Degree in Informatics Engineering

Higher Polythecnic School

Data Producer 2

Ubiquitous Computing and Embedded Systems

Francesc Contreras

Albert Pérez

Marc Visa

November 10, 2021

Table of contents

List of Figures

1 Introduction

The purpose of this document is to explain the Data Producer 2 development for the Sprint 1 and 2 of the project.

We shall introduce all the steps and software we have used when developing it and showing images as a real example for detailed explanation.

2 Environment

At first, these are the steps for installing the libraries and setting up the Arduino IDE to get started for the Data Producer 2 development:

1. Install Arduino IDE.
2. Install driver for USB adapter to ESP-01 in PROG mode.
3. Install ESP8266 Board from Boards Manager from Arduino IDE.
4. Install the following libraries from the Manage Libraries of Arduino IDE:
    (a) SparkFun ADXL345
    (b) Adafruit Unified Sensor by Adafruit.
5. Install the following libraries from GitHub: (These libraries should be added to the Arduino/libraries folder):
    (a) LiquidCrystal I2C[1]
    (b) PubSubClient[2]
6. Connect the ESP-01[3] along with the USB adapter to your machine.
7. Code the program for the ESP-01.

[1]We use this custom library, because it correct the bugs related to ESP-01 using Arduino IDE v 1.8.16
[2]MQTT Library
[3]Low-cost WiFi microchip with built-in TCP/IP networking software, and microcontroller capability

2.1 Material required

Subsequently, you will find all the required components for the Data Producer 2, and a short description for those not so common.

- **ESP-01 (3.3 V)**: is a Wi-Fi module that allows micro-controllers access to a Wi-Fi network.
- **Breadboard**: A board, having a matrix of small holes to which components may be attached without solder.
- **LCD Display (5V)**: is a flat-panel display or other electronically modulated optical device that uses the light-modulating properties of liquid crystals combined with polarizers.
- **ADXL345 (3.3V)**: is an component that measures the acceleratometer, it allows us to use I2C or SPI .
- **VCC Protoboard adapter (3.3V, 5V)**
- **ESP Programmer module**
- **I2C Bus**: is an component that extends I2C Bus and allows us to use with another components such as LCD Display.
- **Wires**

3 Development

Regards the development, this is based on two parts, the first one realated to work directly with the electronic components, and the second by programming the arduino sketch.

The following sections shows you how the electronic schema is in real, as well as how finally we got the visualization of the acceleration.

Moreover, there is the code shown below, but also you may consult the GitHub repository of the project, the one is specified subsequently.

3.1 Hardware

3.1.1 Diagram schema

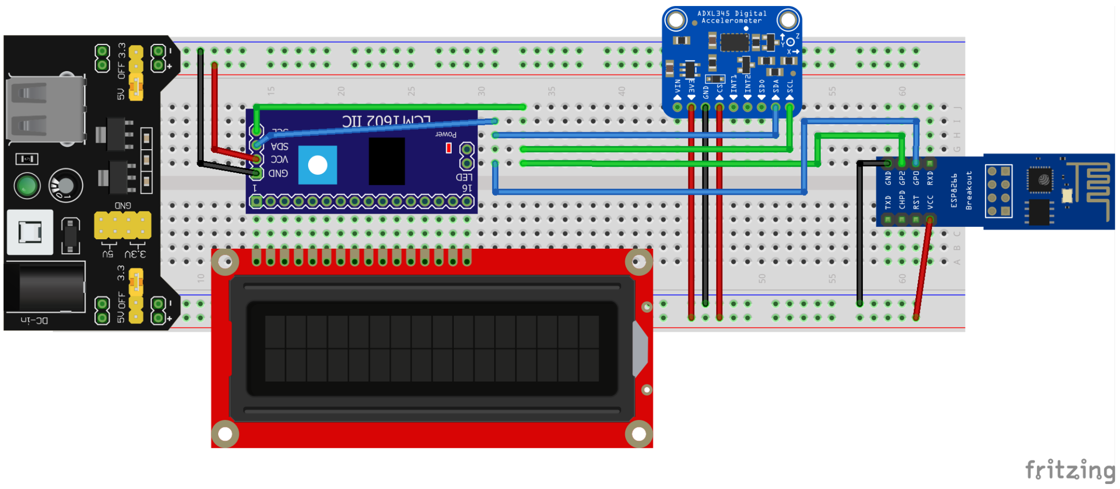

Figure 1

3.1.2 Assembly

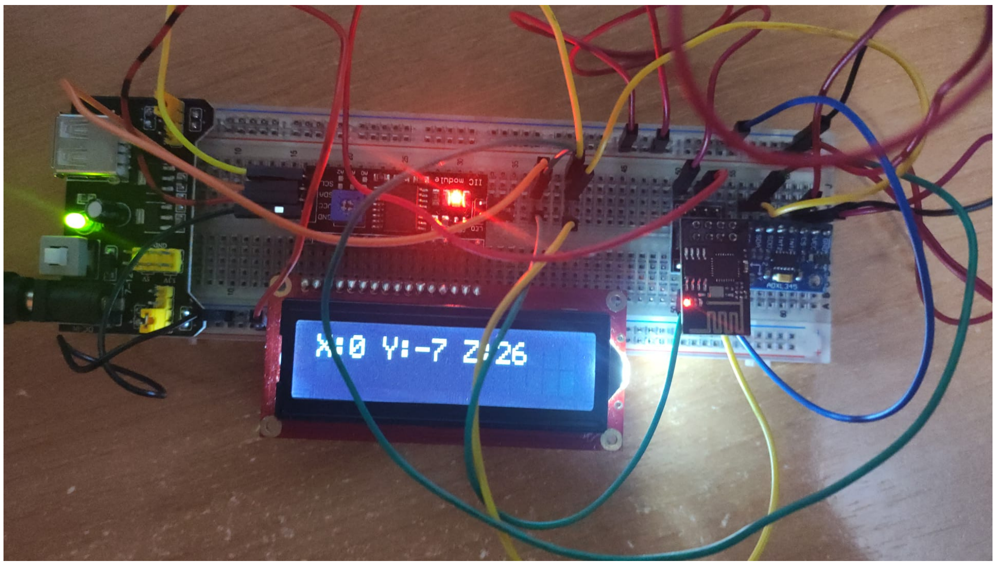

Figure 2: Data Producer 2 Assembly

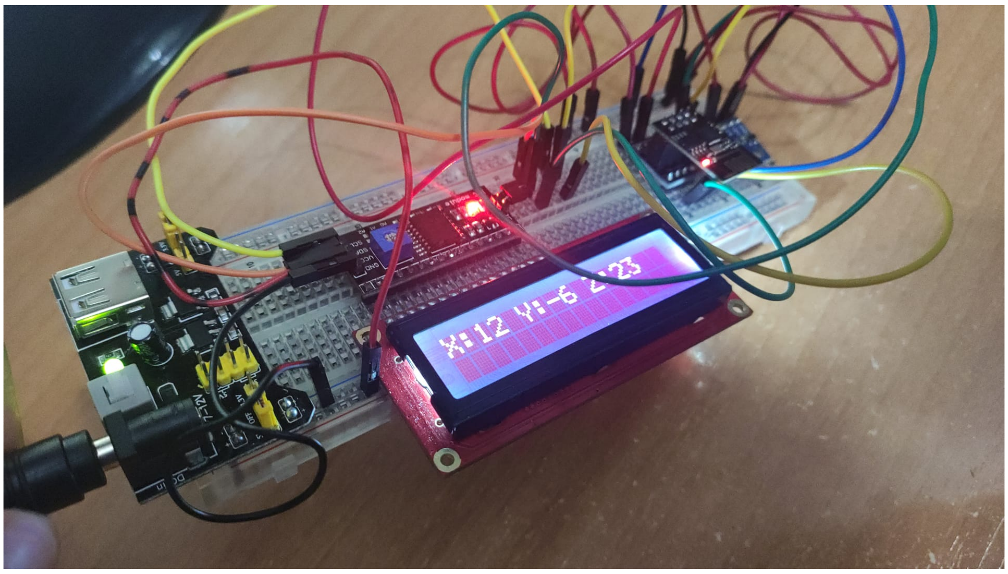

Figure 3: Data Producer 2 in movement

3.2 Software

```
// Import required libraries
#include <Arduino.h>
#include <ESP8266WiFi.h>

//#include <SoftwareSerial.h>

#include <Wire.h>
#include <LiquidCrystal_I2C.h>

#include <SPI.h>
#include <SparkFun_ADXL345.h>

//MQTT Pub
#include <PubSubClient.h>

// Replace with your network credentials
const char* ssid = "-";
const char* password = "-";

LiquidCrystal_I2C lcd(0x27,16,2);

ADXL345 adxl = ADXL345();

// MQTT

const char* mqtt_server = "192.168.1.163"; //MQTT BROKER IP
WiFiClient espClient;
PubSubClient client(espClient);

#define MSG_BUFFER_SIZE (50)
char msg[MSG_BUFFER_SIZE];

void callback(char* topic, byte* payload, unsigned int length) {
  /*Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
  for (int i=0;i<length;i++) {
    Serial.print((char)payload[i]);
  }
  Serial.println();*/
}

void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Create a random client ID
    String clientId = "DataProducer2";
    clientId += String(random(0xffff), HEX);
    // Attempt to connect
    if (client.connect(clientId.c_str())) {
      Serial.println("connected");
      // Once connected, publish an announcement...
      client.publish("outTopic", "connected");
      // ... and resubscribe
      client.subscribe("#");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}

void setup(){
  // Serial port for debugging purposes
  Serial.begin(115200);

  // WIFI
  // Connect to Wi-Fi
  WiFi.begin(ssid, password);

  Serial.println("Connecting to WiFi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println(".");
  }

  // Print ESP8266 Local IP Address
  Serial.println(WiFi.localIP());

  Wire.begin(0, 2); // SDA and SCL
  // ACCELEROMETER
  adxl.powerOn();

  adxl.setRangeSetting(16);    //Definir el rango, valores 2, 4, 8 o 16

  // DISPLAY
  lcd.init();
  lcd.backlight();

  // MQTT
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}

void loop(){
    // MQTT
    if (!client.connected()) {
      reconnect();
    }
    client.loop();

    int x, y, z;
    adxl.readAccel(&x, &y, &z);
    lcd.clear();

    lcd.print((String)"X:"+x+" Y:"+y+" Z:"+z);

    snprintf (msg, MSG_BUFFER_SIZE, "%f m/s",sqrt(x^2+y^2+z^2));
    client.publish("/acceleration", msg);
    delay(5000);
}
```

4 Code repository

Wind Turbine Generator Github repository

References

[1] ADXL345 Tutorial