Dublin City University

Electronic and Computer Engineering

EE445 Bioelectronics IoT Group Project

Authors

Michael Lenehan	michael.lenehan4@mail.dcu.ie
Student Number:	15410402
Kiran Chandrath Ravi	kiran.chandrathravi2@mail.dcu.ie
Student Number:	18212432
Hanzhu Zhang	hanzhu.zhang@mail.dcu.ie
Student Number:	

08/04/2019

Contents

List of Tables

List of Figures

1 Introduction

1.1 Aims

The aim of this project is to implement a connected ECG sensing system, utilising the Texas Instruments MSP432 Microcontroller, and IoT analytics platform ThingSpeak. The system must take heart rate measurements, via a designed bioamplifier circuit using the standard lead configurations, using the MSP432 Microcontroller. These measurements must then be transmit to ThingSpeak via a HTTP POST command, at which point they can be analysed using MATLAB scripts on ThingSpeak. Finally, results must be compared to that of another group.

This project aims to improve the students familiarity with bioamplifier circuits, measurements of biopotential values, and with connected services, in the ThingSpeak implementation.

2 Hardware

2.1 Components

2.2 Amplifier Design

3 Software

There are threen main aspects to the software portion of this project. These are the ECG Sampling, the transfer of sampled data to ThingSpeak, and the MATLAB analysis of the data available on ThingSpeak.

3.1 ECG Sampling

The ECG sampling code implementaion, as used within this project, was performed at a frequecny of 50Hz (one sample every 20ms). This frequency was chosen in part due to limitations imposed by ThingSpeak, which could not allow for a higher frequency to be chosen for single data entries.

In order to take measurements from the bioamplifier circuit, an input pin must first be defined. An integer variable must be defined in order to store the measured value.

Listing 3.1: Analog Input Pin Definition

```
int AnalogPin = A6; //P4.7
int reading = 0;
```

In order to take the reading, the analogRead() method must be called. This method allows the MSP to read in analog values from 0 to 5V, assigning them a value between 0 and 1023. An if statement is used to ensure that the readings are taken at the appropriate timing.

Listing 3.2: Analog Read

```
if(millis() - lastReadingTime > updateThingSpeakInterval){
    reading = analogRead(AnalogPin);
}
```

3.2 ThingSpeak Integration

ThingSpeak integration is performed using the ThingSpeakClient.ino code example found in the Energia git repository [1]. This example works much in the same way as the Energia WifiWebClient example code, however, it has code included to update ThingSpeak input fields, in the required format.

In order to correctly utilize this code, the ThingSpeak write API key must be included in the code, as a String value. An update interval value is used in order to set how often data is sent to ThingSpeak from the client. For this implementation, this has been set to 20ms, as this was the lowest accepted value.

Listing 3.3: ThingSpeakClient.ino write API key and update interval

```
String writeAPIkey = ""; //Enter channel's write API key
                         //between the ""
const int updateThingSpeakInterval = 20; //20ms time interval
```

The startWifi() method is called from within the codes setup, and connects to the network of the input ssid and password, printing a success message to the serial monitor once connected.

Listing 3.4: ThingSpeakClient.ino startWifi()

```
char ssid[] = ""; // your network SSID (name)
char pass[] = ""; // your network password
WiFiClient client;
void startWiFi(){
    WiFi.disconnect();
    client.stop();
    ...
    if(WiFi.begin(ssid, pass) == 0){
    ...
    } else {
    Serial.println("LaunchPad connected to network using DHCP");
    ...
    }
}
```

3.3 MATLAB Script

Bibliography

[1] R. Inant, "ThingSpeakClient.ino," 2013. [Online]. Available: https://github.com/energia/Energia/blob/master/libraries/SimplelinkWifi/examples/ThingSpeakClient/ThingSpeakClient.ino