Smart Shoe

Internet of Things
EVD520
B. Tech ICT
Semester - 7
Group No: 8

Group Members
Falgun Prajapati – 1401080
Kedar Acharya – 1401081
Shyam Khakhar – 1401094
Shloak Agarwal – 1401105

Motivation

Now-a-days people are more concern about their health and people are keen towards checking their loss of calories due to their regular walking. Even though gadgets like smart watches are available. The problem with the smart watch is that they are less accurate and are costly, thus most people cannot afford it. So, we came with an idea of the smart shoe which can count one's steps and can calculate the number of calories burnt. As it will be an extension to shoe, anyone will be able to fit these on any shoe.

Description

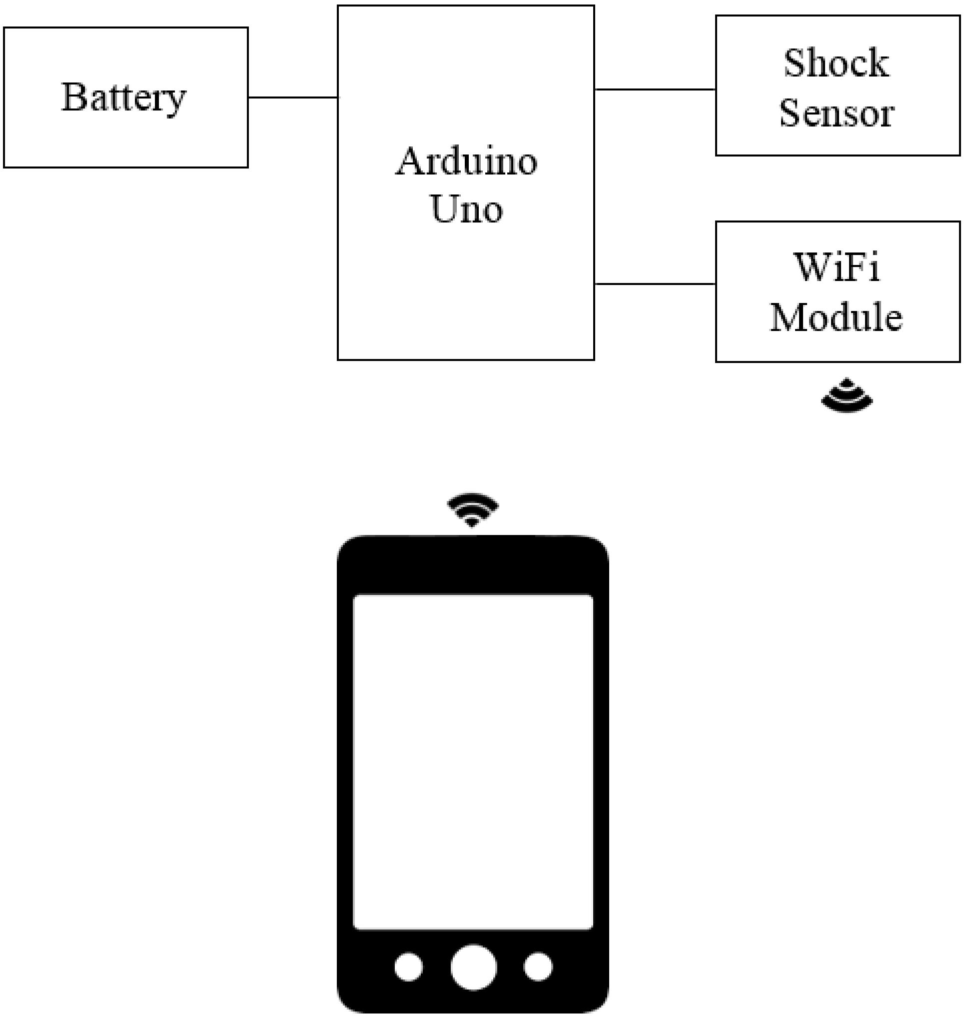
As described above, the smart shoes are just an extension to simple shoes. What makes this smart shoe 'SMART' is the extension we are attaching to the simple ones. The extension is mainly driven by Arduino Uno development board which with the help of accelerometer attached to shoes counts the number of steps taken by one while wearing this shoe. The number is displayed on one's smartphone's application. In this application, one can find the number of steps taken by him and also the application gives you the number of calories burnt due to the number of steps one has took.

Final Outcome

It displays

- Total number of steps and skipping.
- Overall calories burnt.

Block Diagram



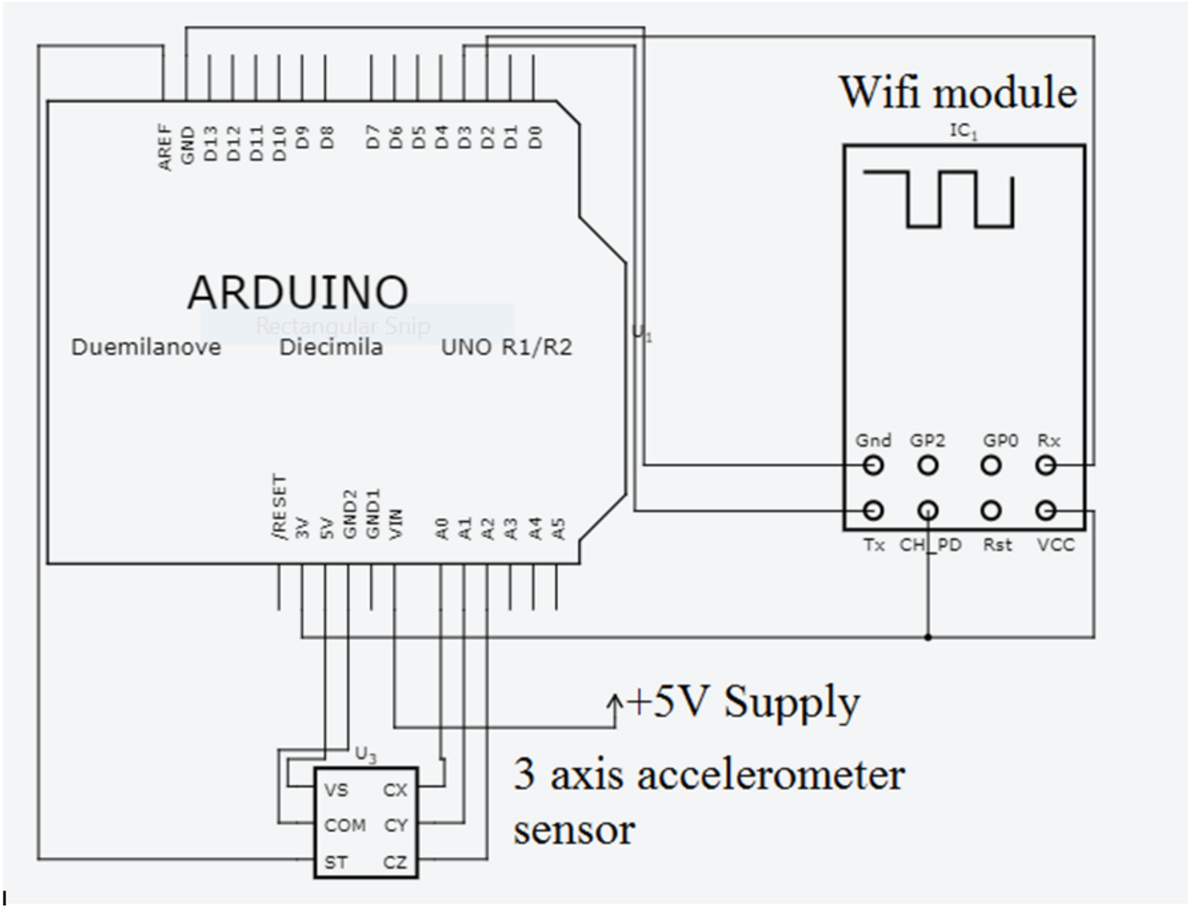
Components Needed

- Battery (here Power Bank)
- Arduino Uno Board
- Accelerometer – ADXL 335 Triple Axis Accelerometer
- Wireless fidelity module – ESP8266
- Smart phone or laptop with internet access
- Shoe

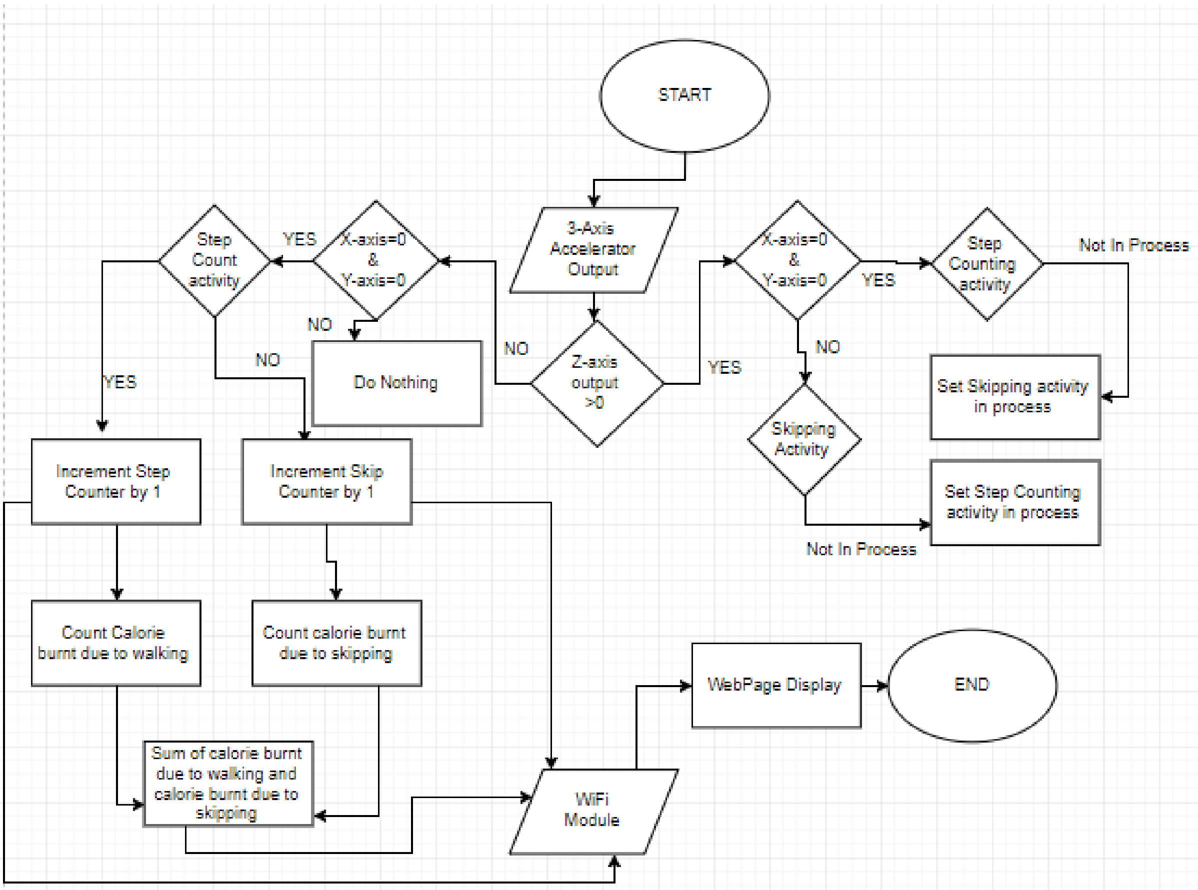
Selection Criteria of Components

- Battery: to provide power to Arduino Uno
- Accelerometer: to count the number of steps and skipping
- Wireless Fidelity module: to provide data to smart phone via WiFi
- Smart phone: to display the outcome

Circuit Diagram



Flow Chart of Program



Datasheets of Major Components

- Arduino Uno

Arduino Uno

Arduino Uno R3 Front *Arduino Uno R3 Back*

Arduino Uno R2 Front *Arduino Uno SMD* *Arduino Uno Front* *Arduino Uno Back*

Overview

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.
The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.
Revision 2 of the Uno board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode.
Revision 3 of the board has the following new features:

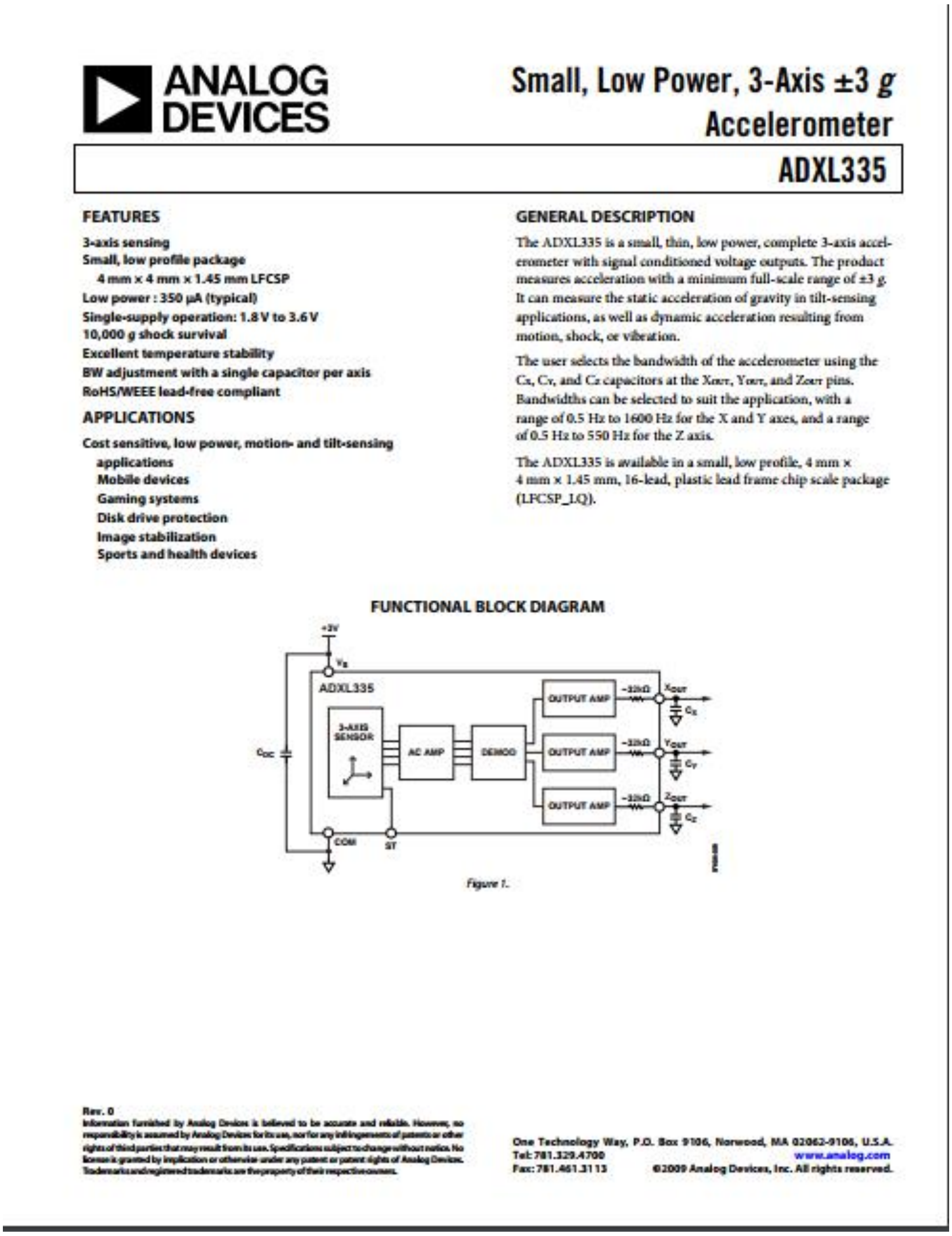
- 1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Arduino Due that operate with 3.3V. The second one is a not connected pin, that is reserved for future purposes.
- Stronger RESET circuit.
- Atmega 16U2 replace the 8U2.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V

- Accelerometer



ANALOG DEVICES

Small, Low Power, 3-Axis ±3 *g* Accelerometer

ADXL335

FEATURES

3-axis sensing
Small, low profile package
4 mm × 4 mm × 1.45 mm LFCSP
Low power : 350 μA (typical)
Single-supply operation: 1.8 V to 3.6 V
10,000 *g* shock survival
Excellent temperature stability
BW adjustment with a single capacitor per axis
RoHS/WEEE lead-free compliant

APPLICATIONS

Cost sensitive, low power, motion- and tilt-sensing applications
Mobile devices
Gaming systems
Disk drive protection
Image stabilization
Sports and health devices

GENERAL DESCRIPTION

The ADXL335 is a small, thin, low power, complete 3-axis accelerometer with signal conditioned voltage outputs. The product measures acceleration with a minimum full-scale range of ±3 *g*. It can measure the static acceleration of gravity in tilt-sensing applications, as well as dynamic acceleration resulting from motion, shock, or vibration.

The user selects the bandwidth of the accelerometer using the C_X, C_Y, and C_Z capacitors at the X_{OUT}, Y_{OUT}, and Z_{OUT} pins. Bandwidths can be selected to suit the application, with a range of 0.5 Hz to 1600 Hz for the X and Y axes, and a range of 0.5 Hz to 550 Hz for the Z axis.

The ADXL335 is available in a small, low profile, 4 mm × 4 mm × 1.45 mm, 16-lead, plastic lead frame chip scale package (LFCSP_LQ).

FUNCTIONAL BLOCK DIAGRAM

Figure 1.

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781.329.4700 www.analog.com
Fax: 781.461.3113

- WiFi Module

zero G
CONNECTING THE INTERNET OF THINGS

Preliminary
ZG2100M/ZG2101M Wi-Fi Module Datasheet

2.4GHz 802.11b Low Power Transceiver Module

Utility and Smart Energy - Thermostats - Smart Meters - White Goods - HVAC	***Consumer Electronics*** - Remote Control - Internet Radio - Home Security - Toys	***Industrial Controls*** - Chemical Sensors - HVAC - Security Systems - M2M Communication
Remote Device Management - Location and Asset Tracking - Automotive - Code Update	***Retail*** - POS Terminals - Wireless Price Tags - Digital Remote Signage	***Medical, Fitness, and Healthcare*** - Glucose Meters - *Fitness Equipment* - *Patient Asset Tracking*

Features

- Single-chip 802.11B including MAC, baseband, RF and power amplifier
- Data Rate: 1 & 2 Mbps
- 802.11B/G and 802.11n draft 2.0 compatible
- Low power operation
- API for embedded markets, no OS required
- PCB or external antenna options
- Hardware support for AES and RC4 based ciphers (WEP, WPA, WPA2 security)
- SPI slave interface with interrupt
- Single 3.3V supply, operates from 2.7V to 3.6V (see section 5)
- 21mm x 31mm 36-pin Dual Flatpack PCB SM Package
- FCC Certified (USA, FCC ID: W7O-ZG2100-ZG2101)
- IC Certified (IC: 8248A-G21ZEROG)
- Wi-Fi Certified
- RoHS and CE compliant
- Fully compliant with EU; meets Japan R&TTE Directive for Radio Spectrum

2 / 28

Code

```
/*  Group 8:
 *  Title: Smart Shoe
 *  SEAS,AU  Sem-7
 *  It calculate calories by calculating the
 *  number of steps and skipping
 */

#include <Arduino.h>
#include <SoftwareSerial.h>
#include <stdlib.h>

const int xpin = A3;
const int ypin = A2;
const int zpin = A1;

SoftwareSerial ser(2, 3);

int dist=0;
int flag=0;

int skipflag=0;
int stepcount=0;
int skipcount=0;
float calorie_burnt=0;
float c1=0;
float c2=0;

void setup() {
  // enable debug serial
  Serial.begin(115200);
  // enable software serial
  ser.begin(115200);
}

void loop() {

  float x,y,z;

  if(abs(analogRead(xpin) - 323)>10 && abs(analogRead(xpin) - 323)<55)
  {
    x=float(abs(analogRead(xpin) - 323));
  }
```

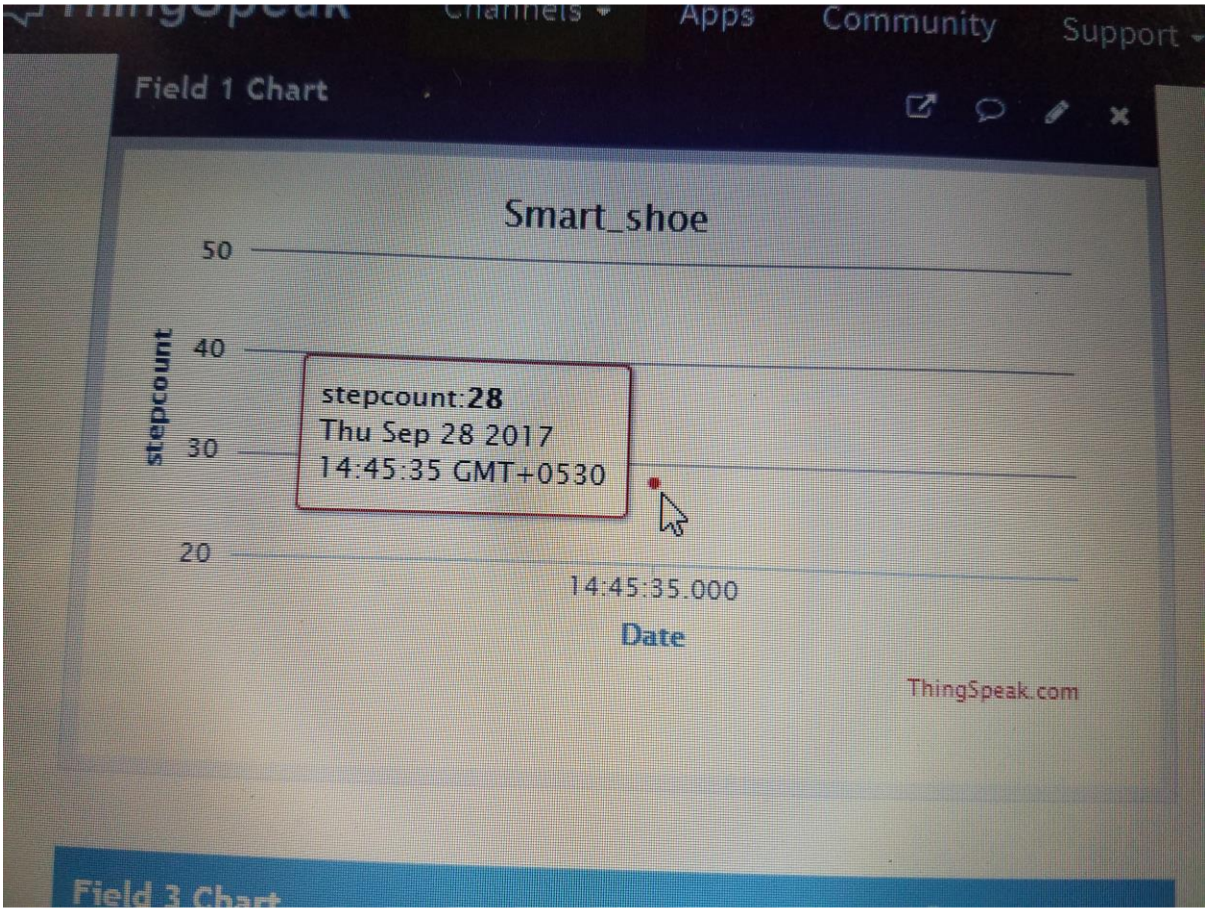
```
else
{
  x=0;
}
if(abs(analogRead(ypin) - 323)>10 && (abs(analogRead(ypin) - 323)<55))
{
  y=float(abs(analogRead(ypin) - 323));
}
else
{
  y=0;
}
if(abs(analogRead(zpin)-397)>10)
{
  z=float(abs(analogRead(zpin) - 397));
}
else
{
  z=0;
}

if(z>0)
{
  if(x==0 && y==0)
  {
    if(flag!=1 && z>12)
    {
      skipflag=1;
    }
  }
  else if(x>0 || y>0)
  {
    if(skipflag!=0)
    {
      flag=1;
    }
  }
}
else if(z==0)
{
  if(x==0 && y==0)
  {
    if(flag==1)
```

```
    {
      stepcount++;
      skipcount--;
      flag=0;
    }
    else if(skipflag==1)
    {
      skipcount++;
      skipflag=0;
    }
  }
}

c1=(stepcount/20.0);
c2=(skipcount/5.0);
calorie_burnt=c1+c2;

String state1=String(stepcount);
String state2=String(skipcount);
String state3=String(calorie_burnt);

// TCP connection
String cmd = "AT+CIPSTART=\"TCP\",\"";
cmd += "184.106.153.149";
//api of thingspeak server api.thingspeak.com
cmd += "\",80";
ser.println(cmd);

// prepare GET string
//Write key of API form Thingspeak
String getStr = "GET /update?api_key=2C1LFAUHU0BN86UD";
getStr +="&field1=";
getStr += String(state1);
getStr +="&field2=";
getStr += String(state2);
getStr +="&field3=";
getStr += String(state3);
getStr += "\r\n\r\n";
// send data length
cmd = "AT+CIPSEND=";
cmd += String(getStr.length());

ser.println(cmd);
```

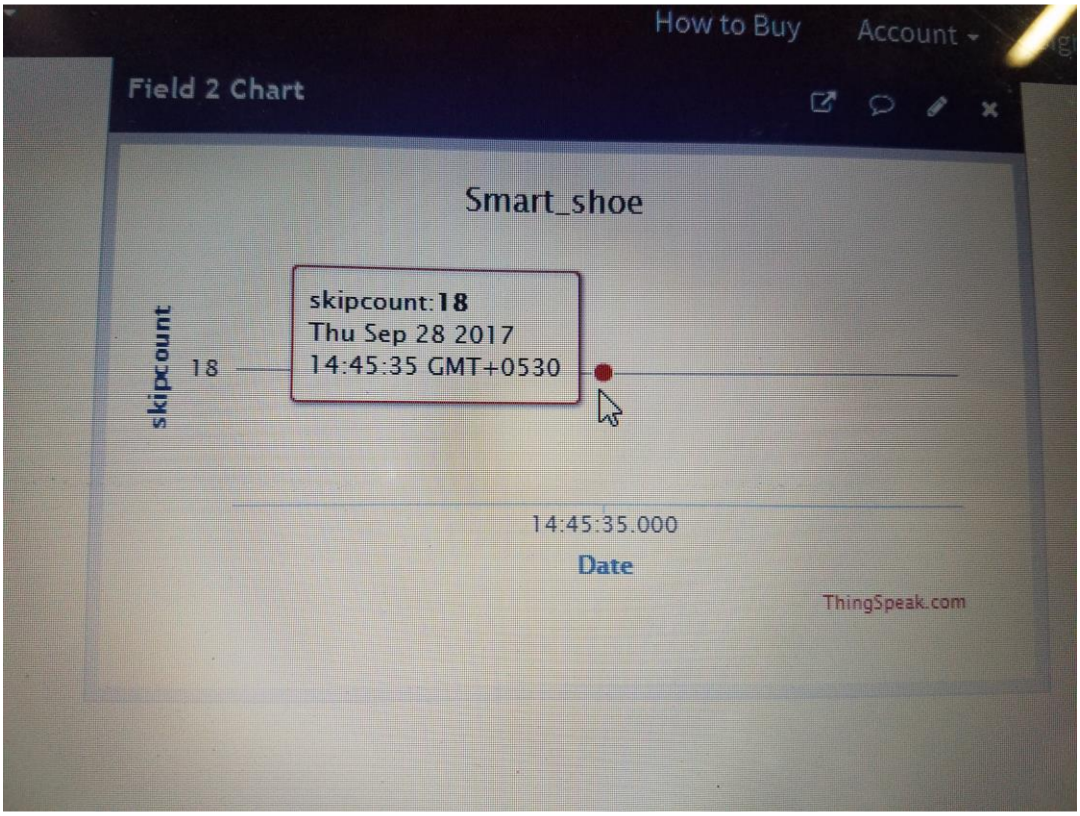
```
 ser.print(getStr);
 // Serial.print(getStr);
 // thingspeak needs 10 sec delay between updates
// delay(100);
}
```

Outcomes

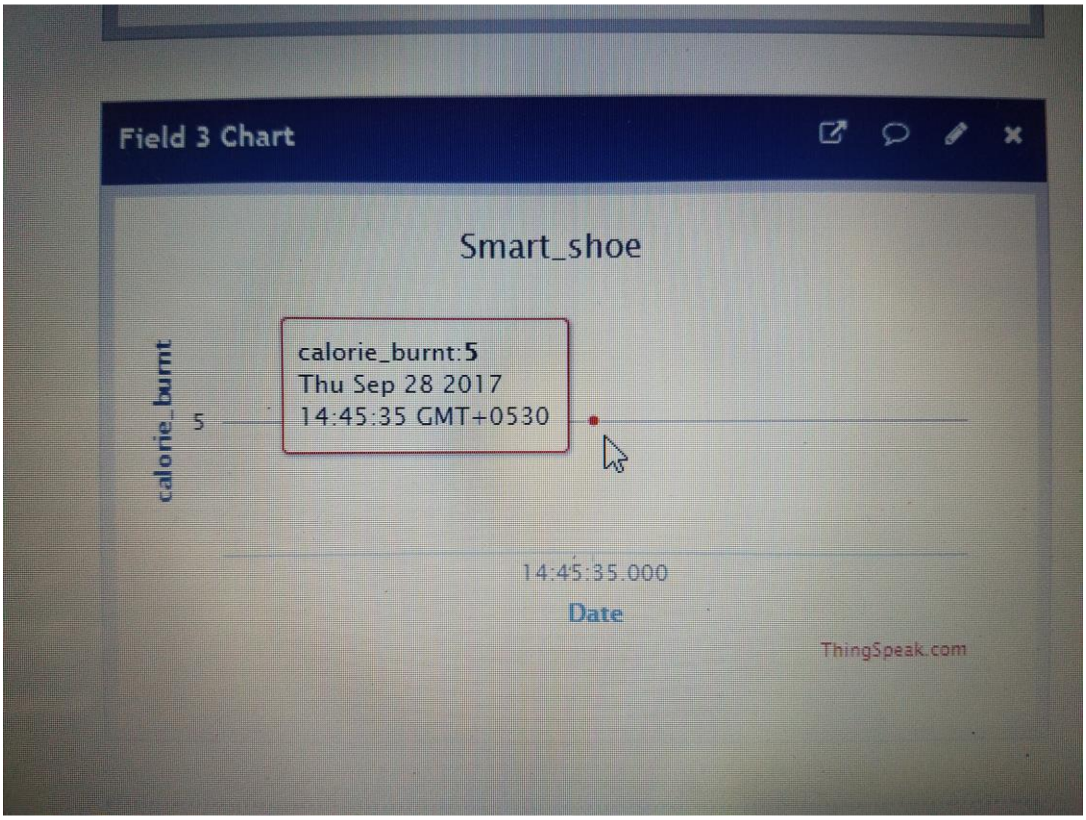
Step Count:



Skips Count:



Overall Calories Burnt:



Challenges and Solutions

- Tilted accelerometer
  - Problem: Unwanted values were displayed when there was no activity.
  - Solution: Took some threshold values; beyond that sensor will detect if it's a false value or true value.

- Measurement of acceleration due to the other human activities
  - Problem: Acceleration would be measured when the person is driving automotive.
  - Solution: A range is set according to the minimum and maximum acceleration produced by human walking.

- Automatic skipping and step detection
  - Problem: While measuring the steps that were some similarities of measurement for skipping or jumping
  - Solution: Used a 'flag' variable to detect if it is a step or a jump.

Time line of project

	08/08/2017	16/08/2017	21/08/2017	28/08/2017	31/08/2017
Assignment 1	✓				
Circuit Diagram		✓	✓		
uC connection to sensor			✓		
Program completion			✓	✓	
Testing and demo				✓	✓