**LOW COST ENERGY METER USING**

**WIFI MODULE**

**A PROJECT REPORT**

*By*

**ASHUTOSH EMBHE 18BEE1025**

**HARSH RAJKAMAL 18BEE1045**

**MAYANK RAJE 18BEE1082**

**SIVA KUMAR 18BEE1019**

**ADARSH KUMAR 18BEE1041**

*under the guidance of*

**Dr.A.Peer Fathima,SELECT**

****

**SCHOOL OF ELECTRICAL ENGINEERING**

**VIT UNIVERSITY**

***November 2019***

**ABSTRACT**

# Objective:

* To design and implement a smart energy meter which notifies the user about the electricity consumption.
* To make the energy meter IOT based.
* To make it low cost and effective.
* To make it accessible anywhere and anytime.

# 1. INTRODUCTION

## INTRODUCTION

In this project, we have built an IOT based Smart energy Meter which informs the user through an Email/SMS about his/her electricity consumption on a daily or monthly basis.

The user also receives a message when the electricity consumption reaches a particular threshold value set by the user.

# 2. PROJECT DESCRIPTION

**2.1 DESIGN APPROACH/DESIGN SPECIFICATION.**

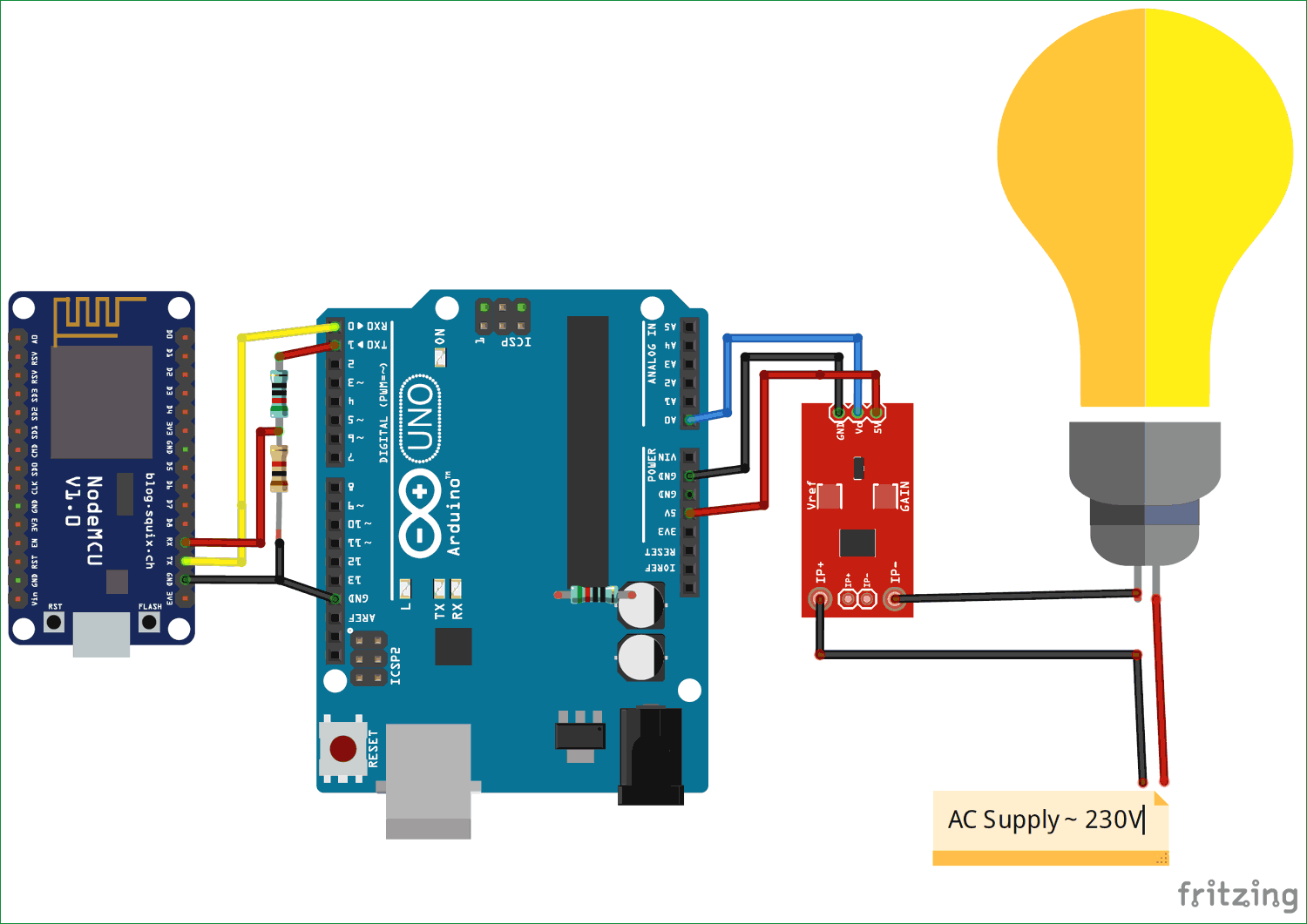


Figure 1:CIRCUIT DIAGRAM

There is one analog pin available in NodeMCU (ESP12), we could use that pin but ESP series can take upto 3.3 volts on their pins. As we are using current sensor which can give upto 5 Volts so, it can damage our Wi-Fi module that’s why we are not using standalone NodeMCU.

To **make output of current sensor 3.3V instead of 5V**, we cannot use voltage divider circuit between Current sensor and analog pin of NodeMCU because as we discussed above about the current sensor that at 2.5Volts output, current is 0Amp.

So, Arduino will read the current sensor value through analog pin and send it to the Wi-Fi module ESP12 using Serial communication. Use voltage divider circuit at receiver pin of NodeMCU so that receiver pin can get upto 3.3 Voltage level.

**2.2 CODE AND SOFTWARE EXPLANATION.**

We have used Arduino IDE for coding this energy meter and wifi module the code has been written and configured in such a way that it reads the value from current sensor and wifi module sends the values

**2.2.1 SETTING UP CURRENT SENSOR TO READ VALUES**.

Library for current sensor as:

**#include "ACS712.h"**

 Array to store power for sending it to NodeMCU.

**char watt[5];**

Create an instance to use ACS712-30Amp at PIN A0. Change First argument if you are using 20Amp or 5 Amp variant.

**ACS712 sensor(ACS712\_30A, A0);**

In loop function, we will call sensor.getCurrentAC();function to get the current value and store in the float variable **I.**After getting current, calculate power using P=V\*I formula.

**void loop() {**

**float V= 230;**

**float I = sensor.getCurrentAC();**

**float P = V \* I;**

**FINAL CODE FOR CURRENT SENSOR.**

#include "ACS712.h"  
char watt[5];  
ACS712 sensor(ACS712\_30A, A0);  
unsigned long last\_time =0;  
unsigned long current\_time =0;  
float Wh =0 ;    
void setup() {  
  Serial.begin(115200);  
  sensor.calibrate();  
}

void loop() {  
  float V = 230;  
  float I = sensor.getCurrentAC();  
// Serial.println(I);  
  float P = V \* I;  
  last\_time = current\_time;  
  current\_time = millis();      
  Wh = Wh+  P \*(( current\_time -last\_time) /3600000.0) ;   
   dtostrf(Wh, 4, 2, watt);          
Serial.write(watt);  
  delay(10000);  
}

**2.2.2 SETTING UP NODEMCU OR WIFI MODULE.**

We have to include all the libraries for ESP12 Wi-Fi Module and AdaFruit MQTT.

**#include <ESP8266WiFi.h>**

**#include "Adafruit\_MQTT.h"**

**#include "Adafruit\_MQTT\_Client.h"**

We define the SSID and Password for your Wi-Fi, from which you want to connect your ESp-12e.Here I have used my phone’s hotspot from phone to link the wifi module.

**#define WLAN\_SSID "ashutosh"**

**#define WLAN\_PASS "88888888"**

**2.2.3 SETTING UP ADAFRUIT SERVER FOR VALUE STORAGE.**

We have given the link of AdaFruit server and server port which is fixed as “io.adafruit.com” and “1883” respectively.

**#define AIO\_SERVER "io.adafruit.com"**

**#define AIO\_SERVERPORT 1883**

Replace these fields with your username and AIO keys which you have copied from AdaFruit site while making the Feed. User name and password are for authentication.

**#define AIO\_USERNAME "ashutosh2000"**

**#define AIO\_KEY "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"**

**FINAL CODE FOR NODEMCU OR ESP8266**

#include <ESP8266WiFi.h>  
#include "Adafruit\_MQTT.h"  
#include "Adafruit\_MQTT\_Client.h"  
#define WLAN\_SSID       "ashutosh"  
#define WLAN\_PASS       "88888888"  
char watt[5];  
#define AIO\_SERVER      "io.adafruit.com"  
#define AIO\_SERVERPORT  1883                     
#define AIO\_USERNAME    "ashutosh2000"  
#define AIO\_KEY         "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

WiFiClient client;  
int bill\_amount = 0;     
unsigned int energyTariff = 8.0;   
Adafruit\_MQTT\_Client mqtt(&client, AIO\_SERVER, AIO\_SERVERPORT, AIO\_USERNAME, AIO\_KEY);  
Adafruit\_MQTT\_Publish Power = Adafruit\_MQTT\_Publish(&mqtt, AIO\_USERNAME "/feeds/Power");  
Adafruit\_MQTT\_Publish bill = Adafruit\_MQTT\_Publish(&mqtt, AIO\_USERNAME "/feeds/bill");  
void MQTT\_connect();

void setup() {  
  Serial.begin(115200);  
  delay(10);  
  Serial.println(F("Adafruit MQTT demo"));

  // Connect to WiFi access point.  
  Serial.println(); Serial.println();  
  Serial.print("Connecting to ");  
  Serial.println(WLAN\_SSID);

  WiFi.begin(WLAN\_SSID, WLAN\_PASS);  
  while (WiFi.status() != WL\_CONNECTED) {  
    delay(500);  
    Serial.print(".");  
  }  
  Serial.println();

  Serial.println("WiFi connected");  
  Serial.println("IP address: "); Serial.println(WiFi.localIP());  
}

void loop() {  
  // Ensure the connection to the MQTT server is alive (this will make the first  
  // connection and automatically reconnect when disconnected).  See the MQTT\_connect  
  // function definition further below.  
  MQTT\_connect();

 int i=0;  
 float watt1;  
 if(Serial.available() > 0 ){  
   delay(100); //allows all serial sent to be received together  
    while(Serial.available() && i<5) {  
     watt[i++] = Serial.read();  
    }  
    watt[i++]='\0';  
  }

 watt1 = atof(watt);  
  bill\_amount = watt1 \* (energyTariff/1000);      // 1unit = 1kwH  
  Serial.print(F("\nSending Power val "));  
  Serial.println(watt1);  
  Serial.print("...");

  if (! Power.publish(watt1)) {  
    Serial.println(F("Failed"));  
  } else {  
    Serial.println(F("OK!"));  
  }

   if (! bill.publish(bill\_amount)) {  
    Serial.println(F("Failed"));  
  } else {  
    Serial.println(F("OK!"));  
  }

if (bill\_amount==4){  
for (int i =0; i<=2; i++)  
{  
  bill.publish(bill\_amount);  
delay(5000);  
}  
bill\_amount =6;  
}  
    
delay(5000);  
}

// Function to connect and reconnect as necessary to the MQTT server.  
// Should be called in the loop function and it will take care if connecting.  
void MQTT\_connect() {  
  int8\_t ret;

  // Stop if already connected.  
  if (mqtt.connected()) {  
    return;  
  }

  Serial.print("Connecting to MQTT... ");

  uint8\_t retries = 3;  
  while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected  
       Serial.println(mqtt.connectErrorString(ret));  
       Serial.println("Retrying MQTT connection in 5 seconds...");  
       mqtt.disconnect();  
       delay(5000);  // wait 5 seconds  
       retries--;  
       if (retries == 0) {  
         // basically die and wait for WDT to reset me  
         while (1);  
       }  
  }  
  Serial.println("MQTT Connected!");  
}

# 2.3 HARWARE MODEL AND HARDWARE EXPLANATION

### 1) **ACS712 Current Sensor**

Asc712 sensor is a key component of this project. This sensor measures the AC

current in the circuit and eliminates any noise coupled with it due to improper

Isolation.

This module works on the principle of HALL EFFECT, according to this principle, when a current carrying conductor is placed into a magnetic field, a voltage is generated across its edges perpendicular to the directions of both the current and the magnetic field. Simply put we use the sensor to measure the magnetic field around a current carrying conductor. This measurement will be in terms of millivolts which we called as the hall voltage. This measured hall voltage is proportional to the current that was flowing through the conductor.

The major advantage of using **ACS712 Current Sensor** is that is can measure both AC and DC current and it also provides isolation between the Load (AC/DC load) and Measuring Unit (Microcontroller part). As shown in the picture we have three pins on the module which are V cc, V out and Ground respectively.



Figure 2-ACS712 current sensor.

2) Arduino Uno

Arduino uno is an open source micro controller board , this board is equipped with sets of digital and analog input and output pins on each sides that can be connected to other circuits. This board has 14 digital pins , 6 analog pins. It is programmable with this arduino IDE via a type b USB cable. It can be powered by a USB cable or by an external 9 volt battery, although it accepts voltages between 7 and 20 volts. In short this component converts analog inputs to digital form.

### General pin functions

* **LED**: There is a built-in LED driven by digital pin 13. When the pin is high value, the LED is on, when the pin is low, it's off.
* **VIN**: The input voltage to the Arduino/Genuino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
* **5V**: This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 20V), the USB connector (5V), or the VIN pin of the board (7-20V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board.
* **3V3**: A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
* **GND**: Ground pins.
* **IOREF**: This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.
* **Reset**: Typically used to add a reset button to shields which block the one on the board.[[7]](https://en.wikipedia.org/wiki/Arduino_Uno#cite_note-website-7)

The Arduino/Genuino Uno has a number of facilities for communicating with a computer, another Arduino/Genuino board, or other microcontrollers.it provides serial communication, which is available on digital pins 0 (RX) and 1 (TX). The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A SoftwareSerial library allows serial communication on any of the Uno's digital pins.

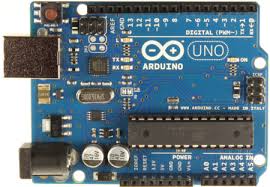


Figure 3-Arduino board

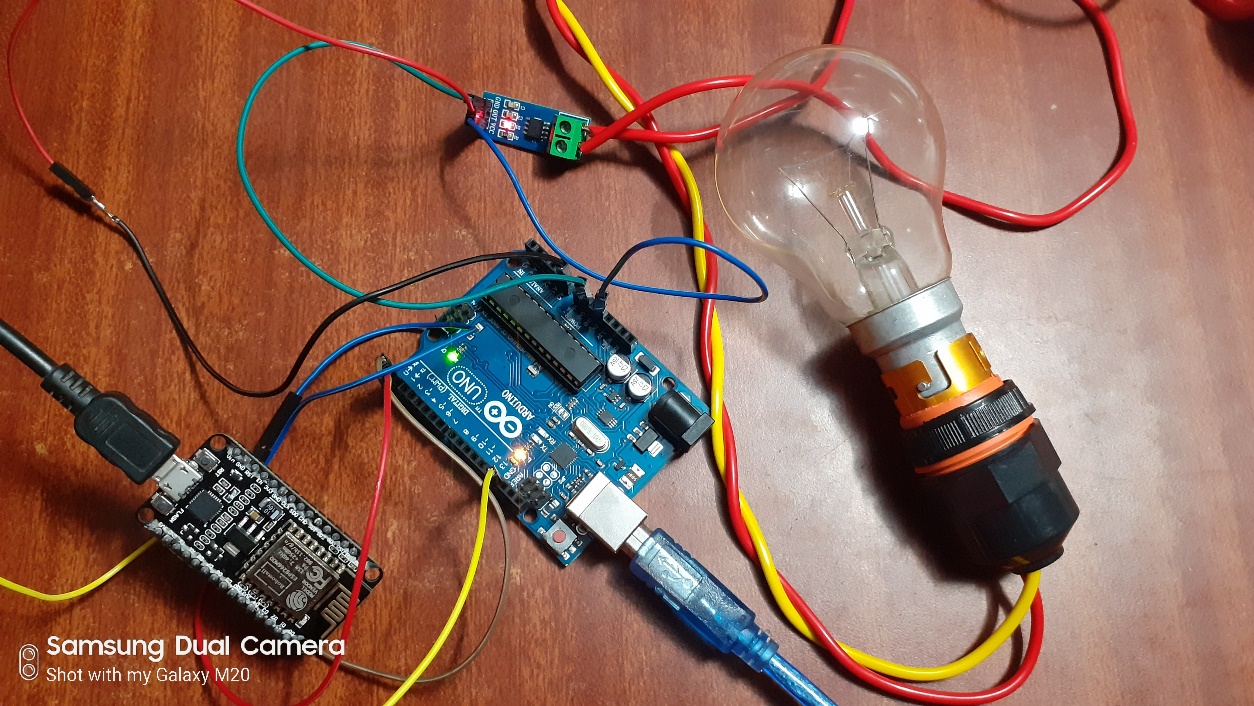
Nodemcu is an open souce IOT platform, whose firmware runs on Espressif's SoC Wi-Fi **ESP8266**,Its hardware is based on the ESP-12 module.one of its main components is the wifi module, in this project we use the Wi-Fi module built in the circuit to send our data from the arduino and upload it to the internet from where we can get the readings.  Most of the I/O pins are led out to the pin headers on both sides for easy interfacing. We can connect these pins to peripherals as needed. Standard headers also make development easy and convenient when using a breadboard.

ESP32 provides 2.4 GHz Wi-Fi.

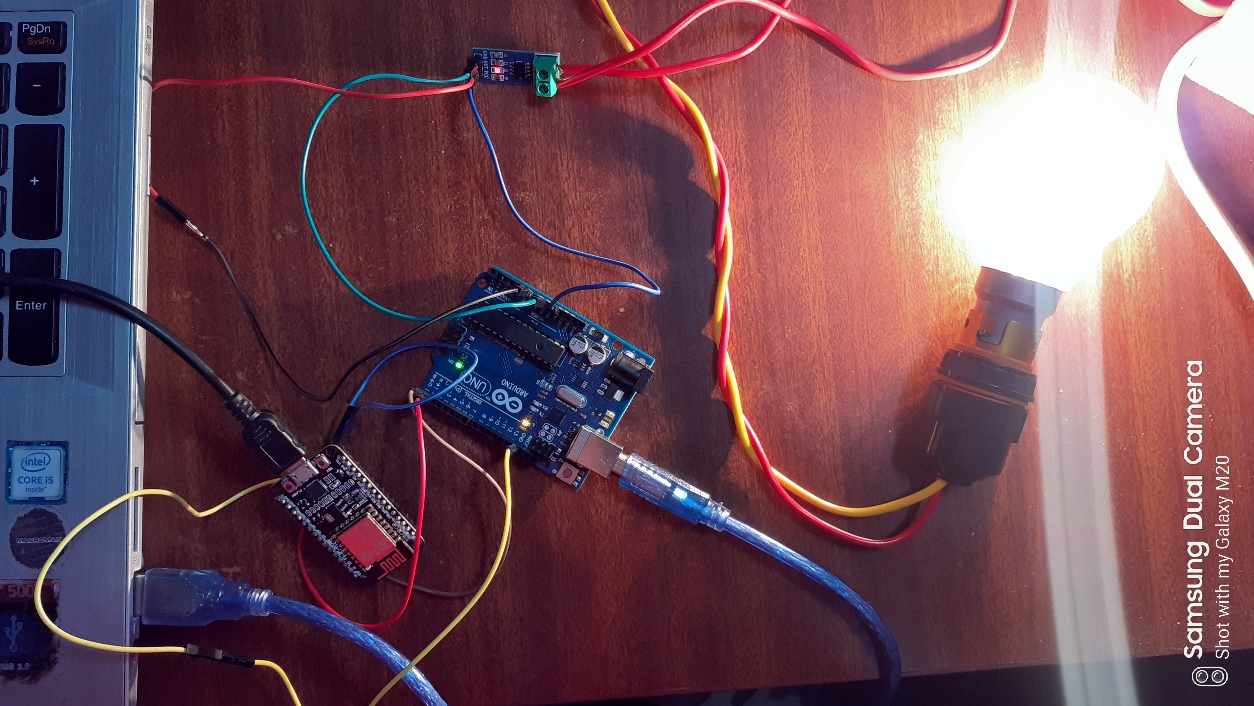


Figure 4-ESP8266 wifi module.

**FULL ASSEMBLED PROJECT.**

****

**Figure 5-WHEN SWITCH IS OFF**

****

**Figure 6-WHEN SWITCHED IS ON.**

**4.WORKING OF PROJECT**

# 3. RESULTS AND DISCUSSION

We have tested the smart energy meter for the bulb which calculated the power consumed and current drawn by the bulb and sent us an email of the bill generated by the use of bulb.

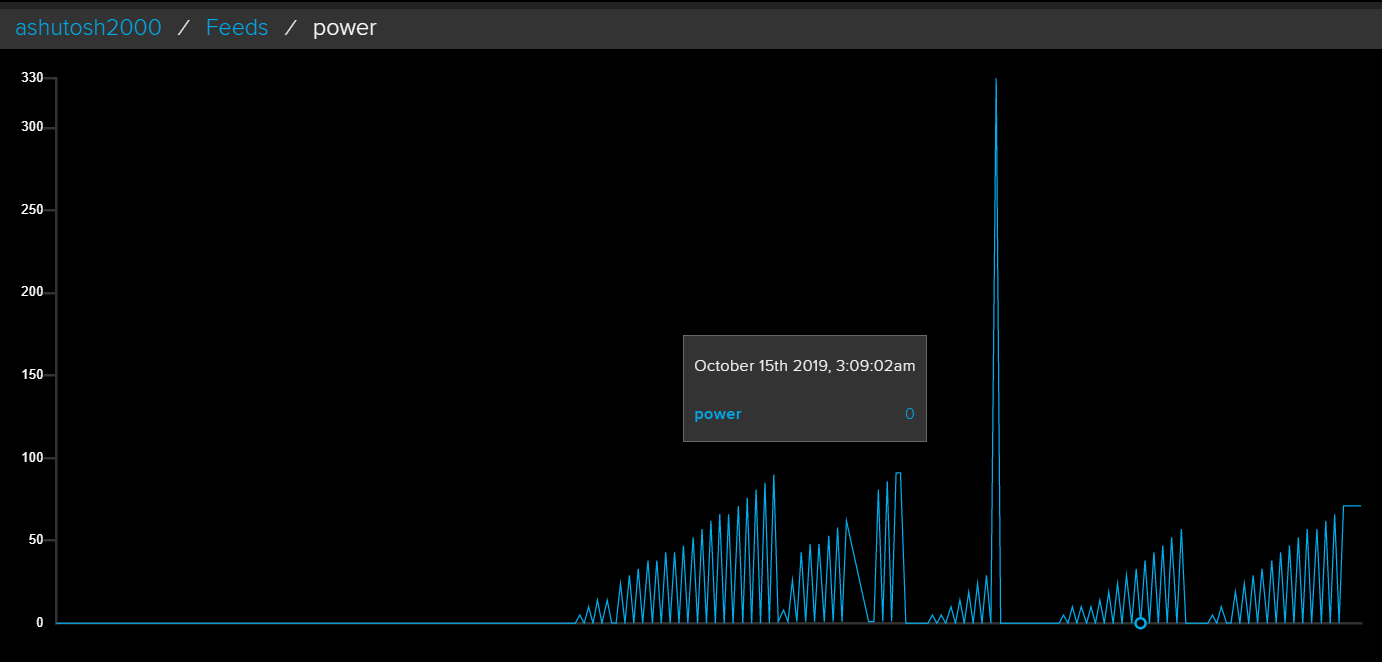


Figure 7-data stored on adafruit server.



Figure 8-Values stored in adafruit of power consumption.

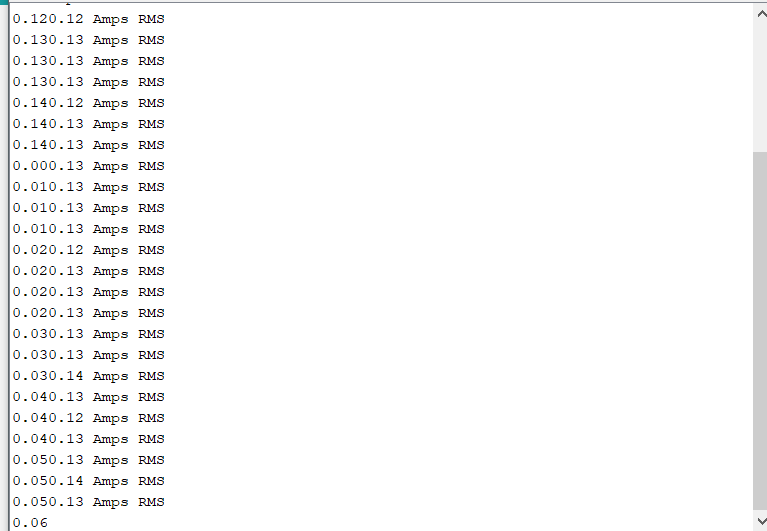


Figure 9-Current value in Arduino serial monitor.

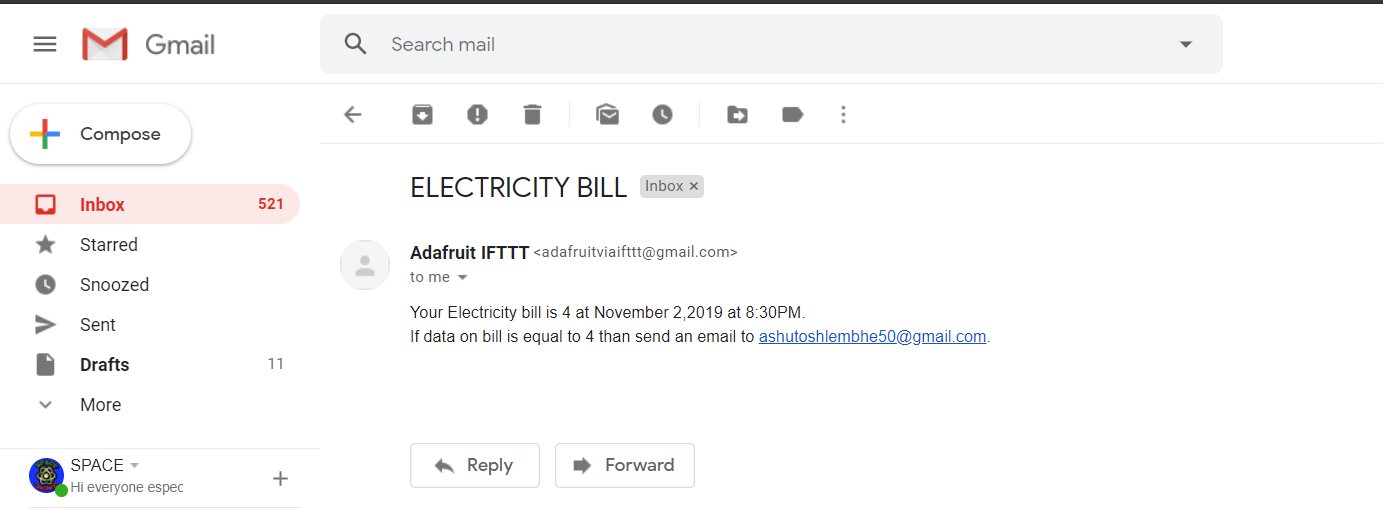


Figure 10-Email of electricity bill from Adafruit server.

# CONCLUSION AND FUTURE WORK

We can conclude that effective measurement of electricity consumption can be performed by using Smart MetersIn this project we have studied and analysed the power consumption of a simple bulb this energy meter can be used on different devices as an add on kit for seeing power consumptions. This smart energy meter will help us see our power consumption anywhere and anytime. It will also encourage people to for going towards paper less bills hence it is a step closer towards environment.

FUTURE WORK

Future case studies can be conducted on Artificial Neural Network.We can connect this device to other household electronic items and their data can be analysed for power consumption which will be monitored over the machine learning algorithm and the pattern of consumption will be effectively maintain the power consumption over machine learning algorithms which will predict which item is consuming more and which is consuming less.

# REFERENCES

1. <https://www.diva-portal.org/smash/get/diva2:829754/FULLTEXT01.pdf>
2. <https://circuitdigest.com/>
3. <https://ieeexplore.ieee.org/document/8521650>
4. <https://ieeexplore.ieee.org/document/6860868/>