POWER METER DESIGN

Burak Güven Özat burakozat@sabanciuniv.edu
Electronics Engineering Program / Junior Student

Oğuz Kağan Yavuz oyavuz@sabanciuniv.edu
Electronics Engineering Program / Junior Student

Mahir Burak Usta mahirusta@sabanciuniv.edu
Electronics Engineering Program / Sophomore Student

Project Supervisor's

Yaşar Gürbüz
Electronics Engineering

Ömer Ceylan
Electronics Engineering

Melik Yazıcı
Electronics Engineering

Abstract

High energy consumption in households is one of the today's energy problems. As such, some measures must be taken to reduce high energy consumption. Currently, users can only see monthly energy usage on the bill. Furthermore, the bill provided by the energy distributor is not detailed enough to track power consumptions in specific time intervals. In this study, main aim was developing a device that provides detailed data for users. Thus, users can track immediate power consumption, consumption data loggings and estimated cost. In this way, the device would create awareness about power consumption and help users to optimize and minimize their power usage.

KEY WORDS: *Power Consumption, Cost, ESP32, Micro-controller, Mobile Application.*

1 Introduction

As the technology improves in years, the amount of energy consumed by the humans has increased dramatically. This rapid rise in the consumption has drove people to seek for ways to reduce the adverse impacts of the massive energy consumption. One of the adverse impacts is the cost result of high energy consumption in the houses. Consumers can see only monthly energy usage on the bill at the end of month. Furthermore, the bill provided by the energy distributor is not detailed enough to track power consumptions in real-time and specific time intervals. Therefore, the consumers don't have any idea about which devices make the most contribution to the total consumption. In addition, they don't know when the most of total power consumption is consumed.

Thus, main objectives of the research were as follows;

- Collecting household's power consumption data with at most +/- 10 % margin of error.
- Monitoring immediate, hourly, daily, weekly, monthly power consumption data and estimation of its corresponding cost.
- Developing easily installable, hand-held device such that user can access the device through mobile application from anywhere.

In line with those objectives, a device that meets these necessities was developed in this project. This device can show the data mentioned above to consumers through a mobile application or browser. By looking at the data logging, user can see the estimated bill. Thus, users can track immediate power consumption, consumption data loggings and estimated cost. In this way, the device would create awareness about power consumption and help users to optimize and minimize their power usage.

2 System Design

2.1 Hardware

Hardware of the system consists of 3 main parts: sensor, microcontroller and the circuit which is designed to manipulate the signal. Sensor is the current transducer, microcontroller that is being used is ESP32 and the designed circuit, which consists of resistors, capacitors, a charge pump and precision rectifier; is complementary to the sensor and has the purpose of a bridge between the sensor and the microcontroller.

2.1.1 Transducer

A transducer is a sensor that converts energy from one form to another. The transducer used in this system is a current transducer which senses current flowing through a hot wire and transforms it to voltage. This process is necessary as the microcontroller board only takes voltage as input. The amplitude of input voltage is calculated by the given formula below where I is the RMS AC current enclosed by the transducer and R being the resistance value of burden resistor used for conversion:

$$V = \frac{I * R}{3100}$$

Figure 1. CR-3110 current transformer.

A burden resistor with 130 Ω resistance is used to cap the voltage at 3.3 V. Burden resistance is selected to be a low number to achieve a larger range and resolution compared to previous studies.[1] The converted voltage signal is going to be fed into the bridge circuit, and to the microcontroller through the circuit.

2.1.2 ESP-32 Micro-controller

ESP32 is a low-power microcontroller which can be programmed with Arduino IDE.

In the project, 3 pins of ESP32 are utilized. Two of them are used for supplying voltage to the circuit and connecting it with a common ground. The other pin is used for receiving analog input from the burden resistor. ESP32 also has a micro USB port which is used for uploading the code from IDE and, to charge up the device.

Figure 2. NodeMCU ESP-32S development board.

The upsides of ESP32, compared to other Arduino units are higher resolution and Wi-Fi connection. Resolution refers to the resolution of ADC inside ESP32, which is the number of bits used by microcontroller. In analog to digital converters (ADCs), analog signal is taken as input and it is converted into a digital signal through sampling, which is then used by microcontroller to perform operations. ESP32 has 12-bits of ADC compared to 10-bits of ADC found in ESP8266, predecessor of ESP32, and other Arduino boards. This higher resolution means ESP32 has a digital signal output that is more accurate and resembling of its analog input signal.

ESP32 comes with its own Wi-Fi module and as a result, it is not necessary to use extra items such as Arduino Wi-Fi Shield.

ESP32, however, has some shortcomings that have to be accounted for. As the input taken from burden resistor is AC voltage, it is bound to have a negative half cycle. However, ESP32 cannot handle negative voltage and this could result in malfunction. As a result, some approaches have to be taken. Similarly, ESP32 also cannot deliver negative voltage. In the next section, the importance of this matter is going to be discussed.

[1] Christopher McNally and Bruce Land, Arduino Based Wireless Power Meter, report, Master of Engineering (Electrical), Cornell University (2010).

2.1.3 Circuit Design

The designed circuit has 2 main components, precision rectifier and charge pump. This circuit is a complementary part of the system. There are numerous ways to comply with ESP32's aforementioned shortcomings and this circuit design is one of those ways.

As mentioned earlier, ESP32 cannot receive negative voltage, as a result; the burden resistor cannot be directly connected to ESP32's analog input pin. Precision rectifier is used to cut off the negative half cycle of input voltage signal and outputs a half cycle signal with only positive values.

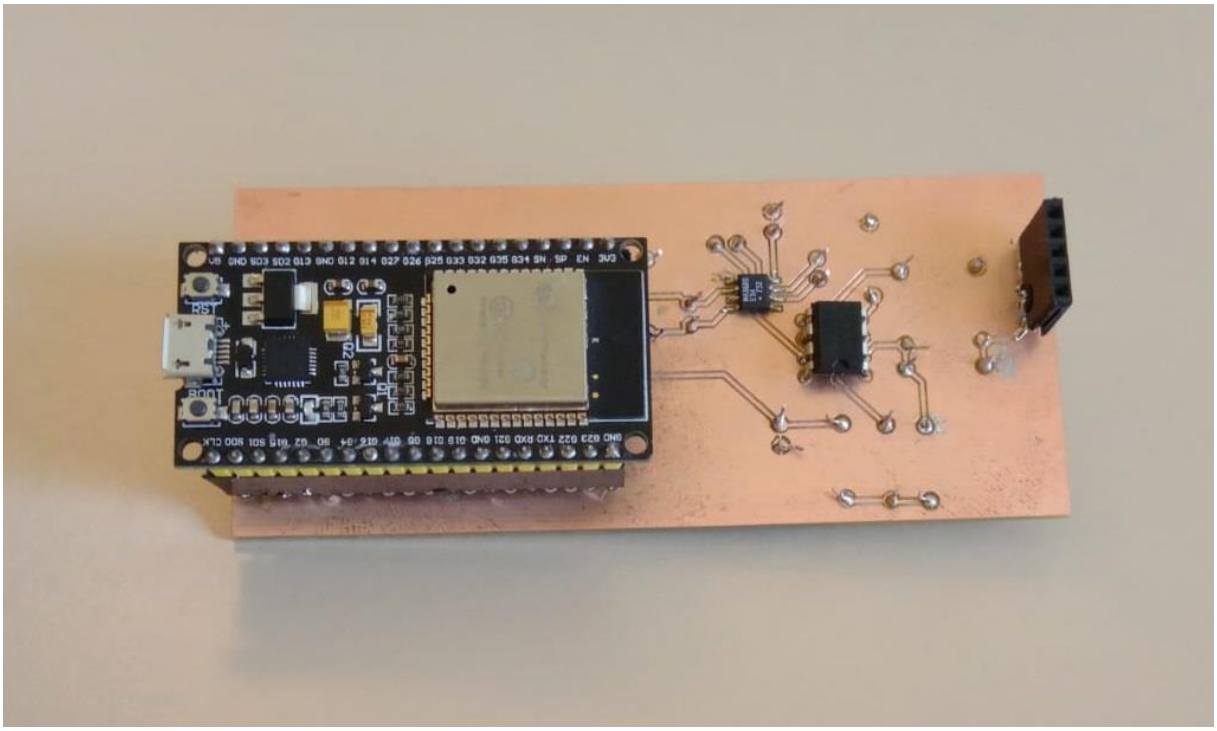


Figure 3. Completed circuit board.

As the input of op-amp is an AC signal with positive and negative values, op-amp should be supplied with positive and negative DC voltages both. This is where ESP32's shortcoming comes into play. As ESP32 cannot supply negative voltage, this precision rectifier circuit cannot work. To get negative voltage out of ESP32, a charge pump is used in this project.

Charge pump is a DC-to-DC converter which is used in different ways such as doubling, tripling, halving or inverting voltages. As ESP32 can only supply positive DC voltage, the op-amp used in precision rectifier would not work, as negative voltage is also required to supply the op-amp. The charge pump IC takes input voltage from ESP32's supply pin, and then it scales the supply voltage to a higher value and produces 2 outputs, a positive DC and a negative DC output. After that, these 2 output pins of charge pump have to be connected with the supply pins of the op-amp.

In the end, the output signal of precision rectifier is fed into the analog input pin of ESP32, ready to be converted to a digital signal and to have calculations be done with the given data.

2.1.4 PCB Design

Last of all, a printed circuit board (PCB) is designed in Altium Designer, which is a program for designing circuits, to reduce noise to signal ratio. Previous experiments with this circuit have been done with the help of a breadboard and jumper cables. However, that design is scrapped in favor of PCB design as jumper cables acting as antennas add extra noise to the signal which is an undesirable quantity.

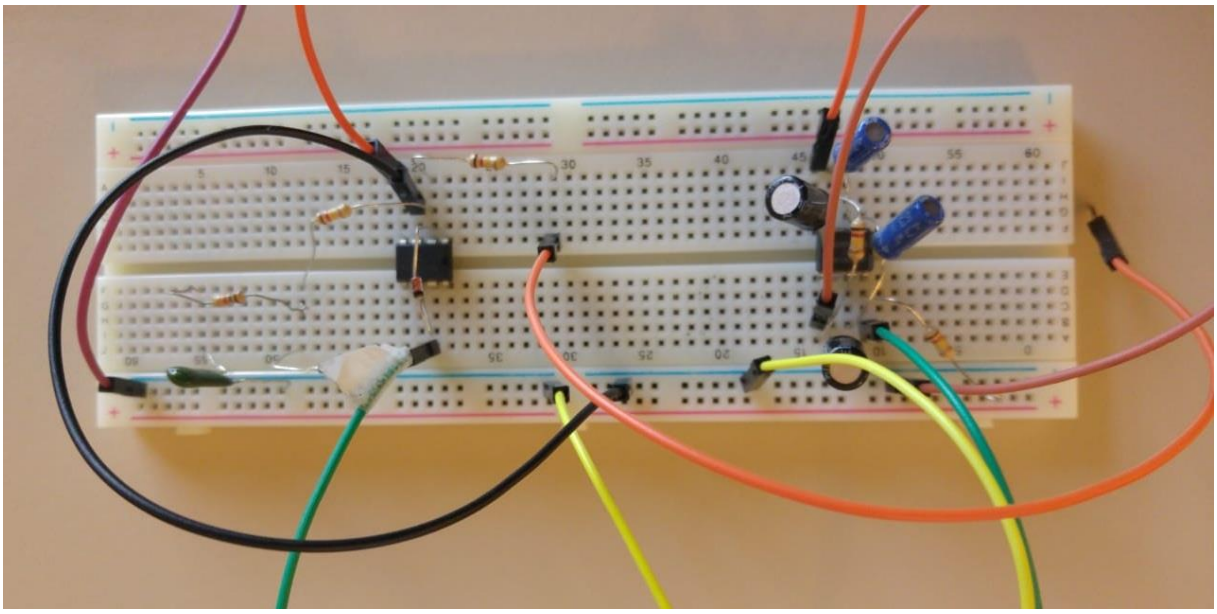


Figure 4. Circuit built on breadboard.

2.2 Software

2.2.1 Data Acquisition and Processing

The software part of the project involves in programming of the micro-controller and user interface application. The Arduino IDE software is used to program the micro-controller for purposes which are Wi-Fi communication between ESP-32 and wireless remoter, reading of analog data and received data's processing. The micro-controller receives the analog signal from its analog pin, when analogRead() function is called, 12 bit ADC having 3.3 V reference voltage on the ESP-32 turns the signal into a value between 0 and 4095.[2] By considering the reference voltage and bit number of ADC, corresponding voltage value is calculated. In the project, analogRead() is called 2000 times for each iteration. As it was observed experimentally, number of calls ensures to sample at least a period of a 50 Hz signal.[3] In the sampling process, maximum value is taken, and then it is stored in an array. There are 20 maximum values in the array. The program works in the same way for the next maximum values.

Thus, their average can be calculated in order to eliminate sudden changes that arise from the environmental effects and device's itself.

[2] Espressif Systems, "ESP32 Series Datasheet," 2019, accessed August 6, 2019, https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf

[3] Christopher McNally and Bruce Land, Arduino Based Wireless Power Meter, report, Master of Engineering (Electrical), Cornell University (2010).

After the average is calculated, amount of current flowing through the main cable is computed through following equation.

$$Current = \frac{V_{max} * 3100}{\sqrt{2} * R}$$

R is the burden resistor and 3100 is coefficient of the transducer. In Turkey, mains electricity's rms voltage value is 220 V. Thus, power consumption is found by multiplication of the current with the rms voltage value. For the cost estimation, firstly, a command keeps adding power consumption to calculate cumulative power consumption. Secondly, then since the micro-controller is programmed in a way that it sends data for only once per second and electricity prices are given in kW, the cumulative power consumption is divided by (3600 * 1000) and multiplied with electricity price. The unit price is cost per kW consumption and all taxes are included on it.

$$Cost = \frac{\text{Total Power Consumption} * \text{Unit Price}}{3600 * 1000} \text{ TL}$$

2.2.2 Wi-Fi Connection and IoT Cloud Server

The micro-controller is connected to internet through a wireless remoter. It has its own Wi-Fi module. Thus, Wi-Fi connection between ESP-32 and remoter is set up by defining SSID and password of the Wi-Fi network and using WiFi.begin() function that comes from Arduino IDE's Wi-Fi library. After the internet connection is completed, the data is sent to IoT cloud server. As the cloud server, ThingsBoard platform was used in the project.

2.2.3 Mobile Application (GUI)

Mobile application was developed on MIT App Inventor utility for mobile phones whose operating system is Android. It comprises all the user interface. Application is connected to cloud server and displays the data with features which are immediate power consumption, hourly, daily, weekly and monthly total cost estimations. Thus, customer can see the immediate energy consumption in the household and know the approximate cost that he will receive at the end of the month and it allows user to optimize and minimize the consumption.

Figure 5. Opening screen of the app.

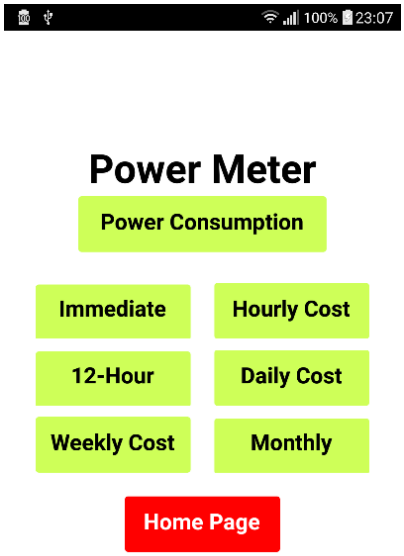


Figure 6. Main menu of the app.

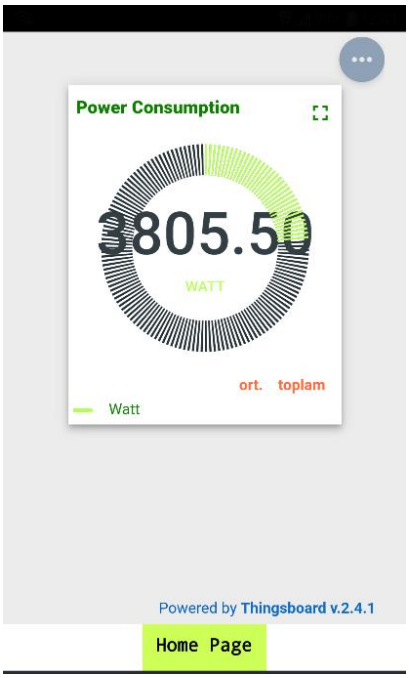


Figure 7. Immediate power consumption tab of the app.

3 Discussion

Initially, except for minor errors the device worked successfully. One of them arose from the micro-controller, its ADC converted analog voltage data value to digital voltage data value of around 0.1 V less depending on the signal's magnitude. Therefore, the problem was solved by adding 0.1 V through the programming. The solution didn't work values less than 6 mV. For this reason, the problem limited device's ability to measure power consumptions less than 222,57 W. However, if usual total power consumption rates in the household is considered, issue can be negligible. The device can measure immediate power consumption values which are between 222,57 W - 12.241 kW.

Furthermore, as it was stated in the previous sections, firstly, the design was tested on a breadboard. Connections on the breadboard was made through many jumpers. Since there were too many jumper connections, voltage signal reached ESP-32 was noisy despite a low pass filter was connected to the circuit. As it was aimed in printed circuit board design process, PCB has visibly reduced noise on the signal.

Tests were conducted on hot air gun and kettle. According to data sheet of the hot air gun, its power consumption is around 1530 W for 220 V mains rms voltage. Device's test result for the hot air gun was around 1450 W. In addition to hot air gun, kettle's consumption is around 2520 W and the device measured it as around 2440 W. Furthermore, when both devices were on, power-meter displayed it around 3810 W. By comparing results with values given in the devices' datasheets, Error percentages are found as 5.6 %, 3,17 %, 5.92 % for hot air gun, kettle and the case when both devices are on are respectively. Margin of error might result from either circuit components' tolerance percentages or + / - 5 % tolerance of the mains voltage.

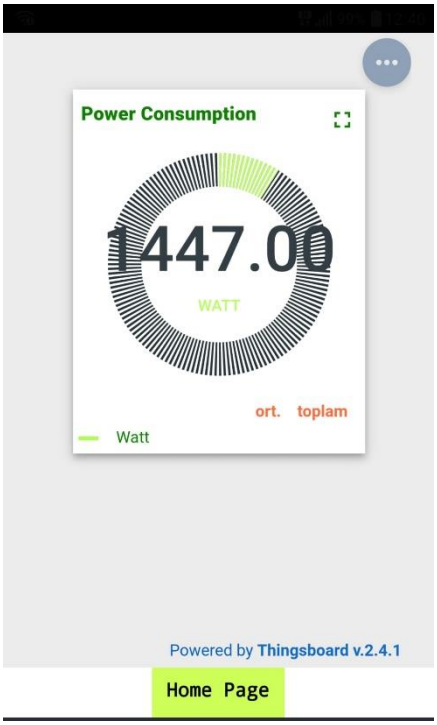


Figure 8. Hot air gun's immediate power consumption: 1.447 kW

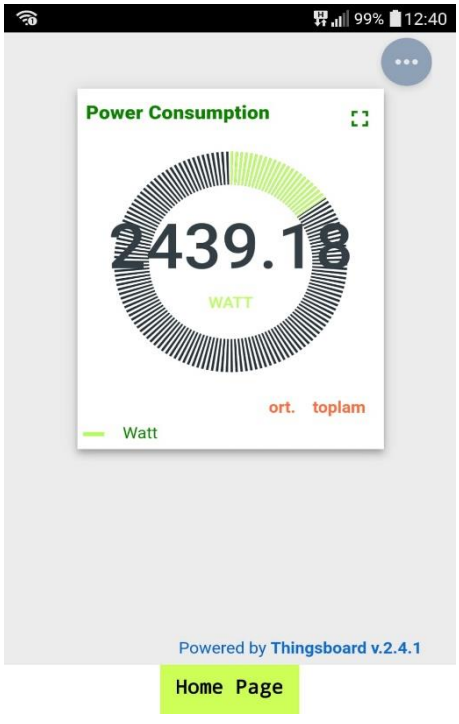


Figure 9. Kettle's immediate power consumption: 2.439 kW

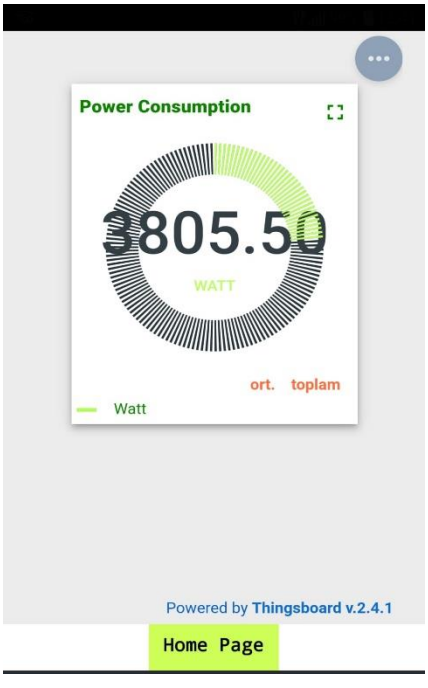


Figure 10. Immediate power consumption of both devices: 3.805 kW

4 Conclusion

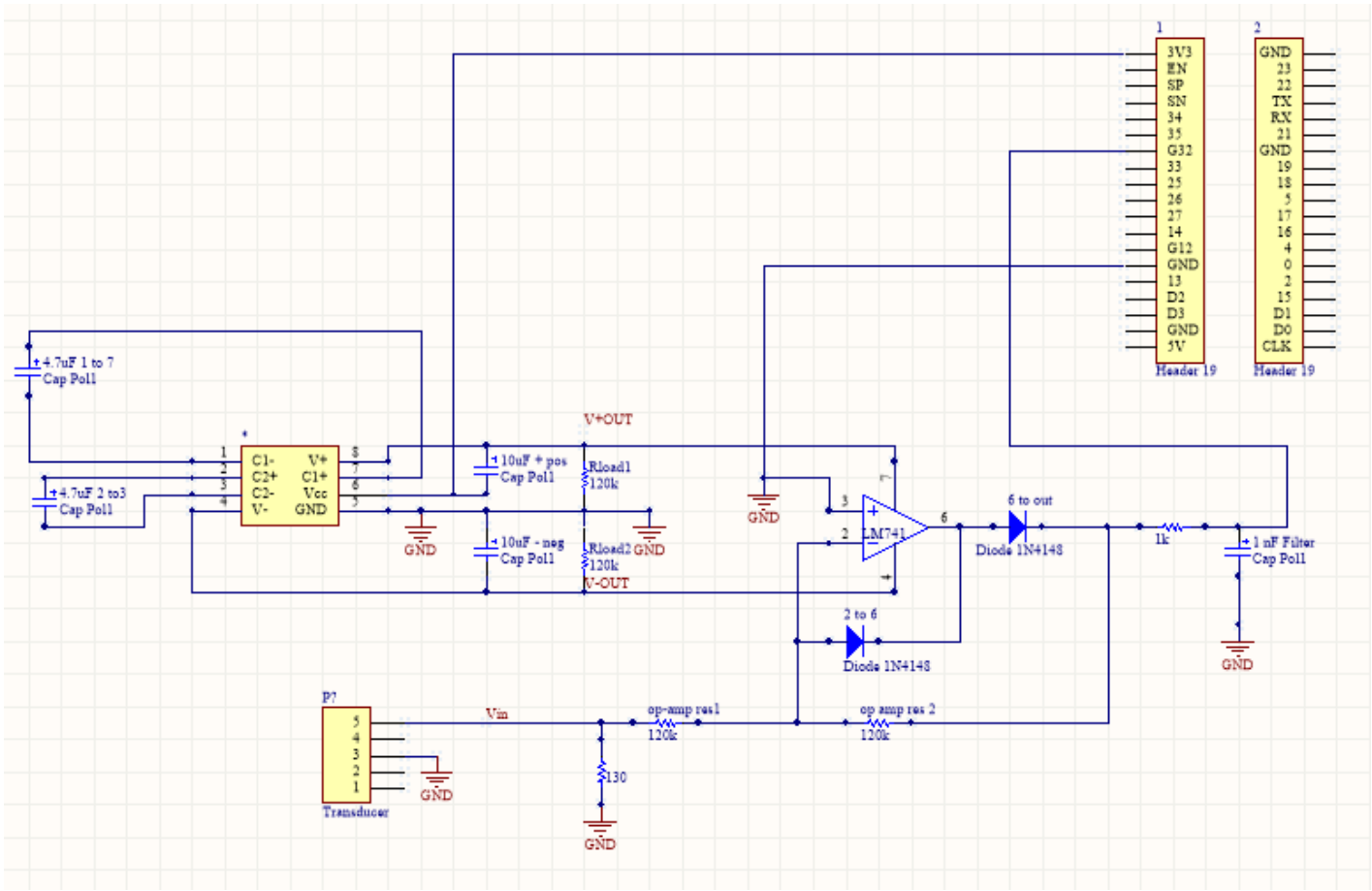
In conclusion, the device worked successfully in capturing and processing the voltage data within an acceptable margin of error. Power consumption was calculated by conversion of voltage data into power data. This power data is used for calculating daily, weekly and monthly power consumption and power cost, which is stored in IoT cloud server. The developed mobile application takes this information from IoT cloud server and shows graphs of power consumption and cost data to the user.

In future studies, researchers might improve and develop the Arduino code and the interface of mobile application. A solid box might also be built to protect PCB from external effects such as atmospheric noise and physical damage. The results of this project might be improved if future researchers can solve the minor problems mentioned earlier.

The learning outcomes for project members are laboratory working conditions and ethics, practicing on advanced programming techniques, getting familiar with microcontrollers and how they work, using different software tools such as Altium Designer to design printed circuit boards, and also soldering circuit components into PCB.

5 Appendix

Circuit Design



Arduino Code

```
void loop()
{
    if(!tb.connected())//connecting to thingsboard
    {
      reconnect();
    }
    float totalVoltage = 0.0;
    float measurement = 0.0;
    float maxReading = 0.0;
    for (int i = 0; i < 20; i++){ //take 20 samples
      for(int counter = 0; counter < 2000; counter++)
      {
         measurement = analogRead(32) * 3.3 / 4095.0;
         if(measurement > maxReading)
         {
            maxReading = measurement;
            //Serial.println(maxReading);
            if(maxReading < 0.06){ //make max 0 in case of noise
              maxReading = 0;
              }
         }
      }
      if(maxReading != 0){//esp32 voltage measurement error
        maxReading += 0.1;
      }
     maxvalues[i] = maxReading;


     maxReading = 0.0; //reset maximum value
    }
    for (int i = 0; i < 20; i++){//total voltage measured in 20 samples
      totalVoltage += maxvalues[i]; //its average will be taken
    }
    float current = (totalVoltage / 20.0) / sqrt(2) * 3100 / 130;
    float power = current * 220;
    Serial.print(power);
    Serial.println (" W");
    Serial.println("Sending data to Thingsboard:");
    float unit_price = 0.0; //set unit price value here
    cumulative += power;
    cost = cumulative / 3600.0 * unit_price / 1000.0; //total cost
    tb.sendTelemetryFloat("power", power);
    tb.sendTelemetryFloat("cost", cost);
    delay(550);
    tb.loop();
}
```

```
void reconnect()
{
  while(!tb.connected())
  {
    Serial.println("Connecting to ThingsBoard server ...");
    if(tb.connect(thingsboardServer, TOKEN))
    {
      Serial.println("Connected to the server !");
    }
    else
    {
      Serial.print("Connection failed");
      Serial.println(" : retrying in 1 seconds");
      delay(1000);
    }
  }
}
```

6 References

- Espressif Systems. "ESP32 Series Datasheet." 2019. Accessed August 6, 2019. https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf.
- McNally, Christopher, and Bruce Land. Arduino Based Wireless Power Meter. Report. Master of Engineering (Electrical), Cornell University. 2010.