

A Low Cost Live Electricity Consumption Monitor using IoT and The Analyses of Electricity Consumption of Appliances in a Turkish Household

Bariş Sanlı , www.barissanli.com , barissanli2@gmail.com

WARNING : DO NOT ATTEMPT TO DO THIS EXPERIMENT AT HOME, DO NOT TOUCH YOUR BREAKER PANEL OR METER. THERE IS A RISK OF DEATH AND HEALTH HAZARD. NO RESPONSIBILITIES IF ANY ATTEMPT TO REPEAT THE PROCEDURES IN THIS ARTICLE. NEVER EVER INTERFERE WITH ELECTRIC CABLES AT HOME

Contents

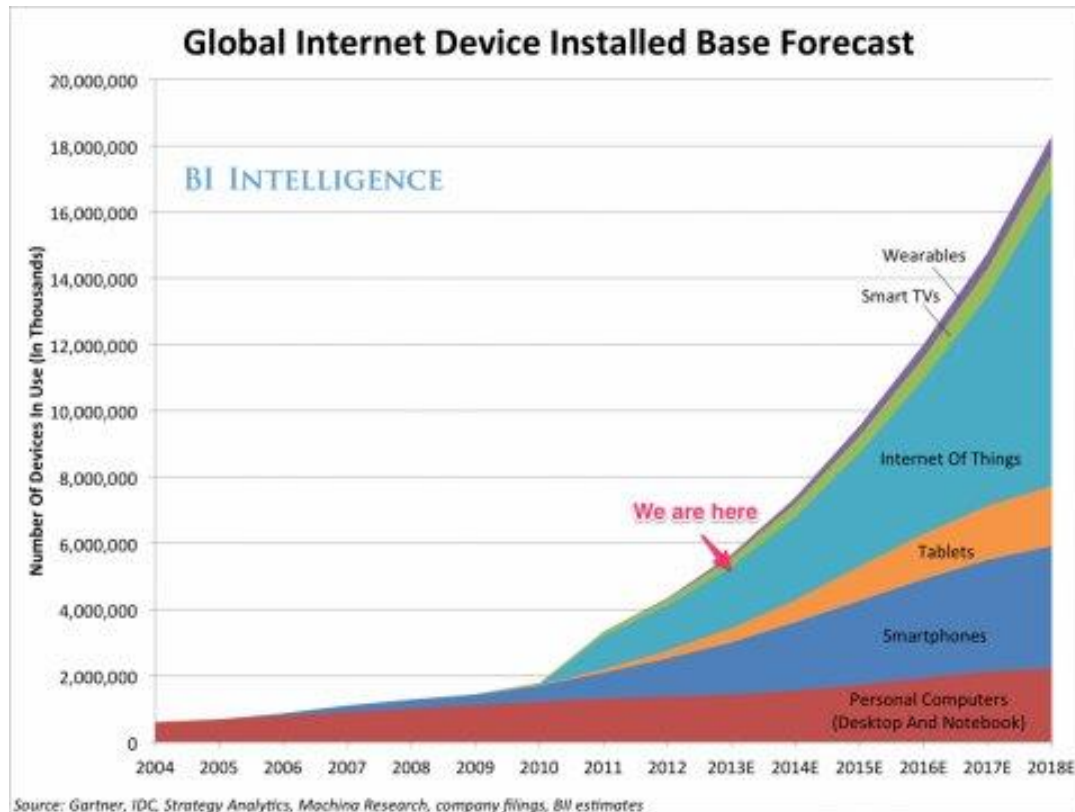
Summary	1
Introduction	2
Turkish Electricity Distribution Perspective and Rules	2
In Practical Terms	4
SCT Circuit	5
Spark Core – “Core”	7
Building the Program	9
The Code	10
Google Spreadsheet Connection	12
Data and Results	14
Conclusion	17

Summary

In this project, a non invasive current transformer and Spark Core (renamed Particle recently) is used to monitor a Turkish household consumption for several days and broadcasted live data over Google Spreadsheets. The aim of the project is to built a low cost, easy to program and monitor energy monitor to record household consumption hence improve energy usage of a consumer. At the end of monitoring period, a series of precautions to lower electricity bills are investigated.

Introduction

“Internet of Things (IoT) will be huge “ Business Insider claimsⁱ. Fifteen years ago it was hardly imaginable that all the objects surrounding us will be internet connected, by then Internet was the playground of desktop computers. However, as the sensors became cheaper and internet connections more widespread the IoT has taken off. The trend seems to accelerate in the following years.



The IoT term has been coined by Kevin Ashton, British technology pioneer in 1999. His definition of the system is “system that connects the physical world to the Internet via ubiquitous sensors that gather and report data”. In 2020, the population of IoT will reach 50 billion by some other analystsⁱⁱ.

It is not IoT history or discussion this article will explain, but rather how easy it is for someone to setup a certain IoT system.

Turkish Electricity Distribution Perspective and Rules

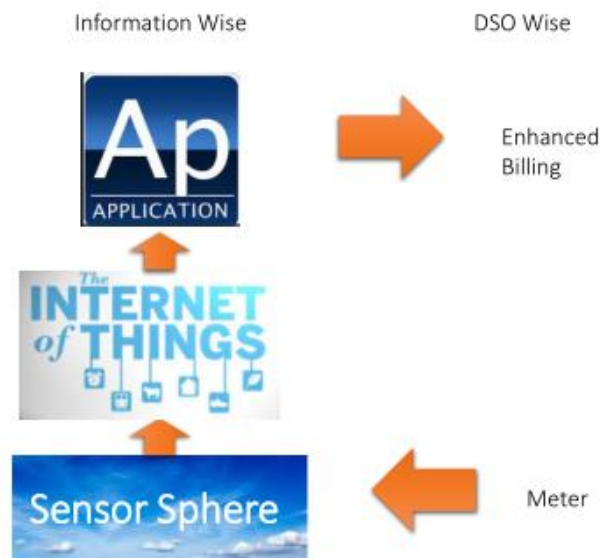
Turkish distribution companies are recently privatized and currently their main concern is to build high level “smart” systems to monitor their system from a macro view point. The smart meter deployment for households has not been on the agenda due to regulation. Turkish regulator EMRA has limited the regulated costs of these meters such that no smart grid with remote control can be deployed. EMRA insists that utilities should first prove the cost effectiveness of the system. So there are no widespread smart meter deployment, neither the Government has published a road map.

However, during my presentation in European Utility Week in Amsterdam(2014)ⁱⁱⁱ, I emphasized that in the bigger picture, IoT and Sensor Sphere makes the smart meter less important for household consumer. My main question was “When everything in your home is a sensor and you can connect them through your Wifi, why do you need utilities to gather your data or smart meter as a door

keeper?”. Of course Time of Use tariffs or user specific rates will be important, but from a household’s perspective smart meter’s abilities and benefits are yet to be tested.

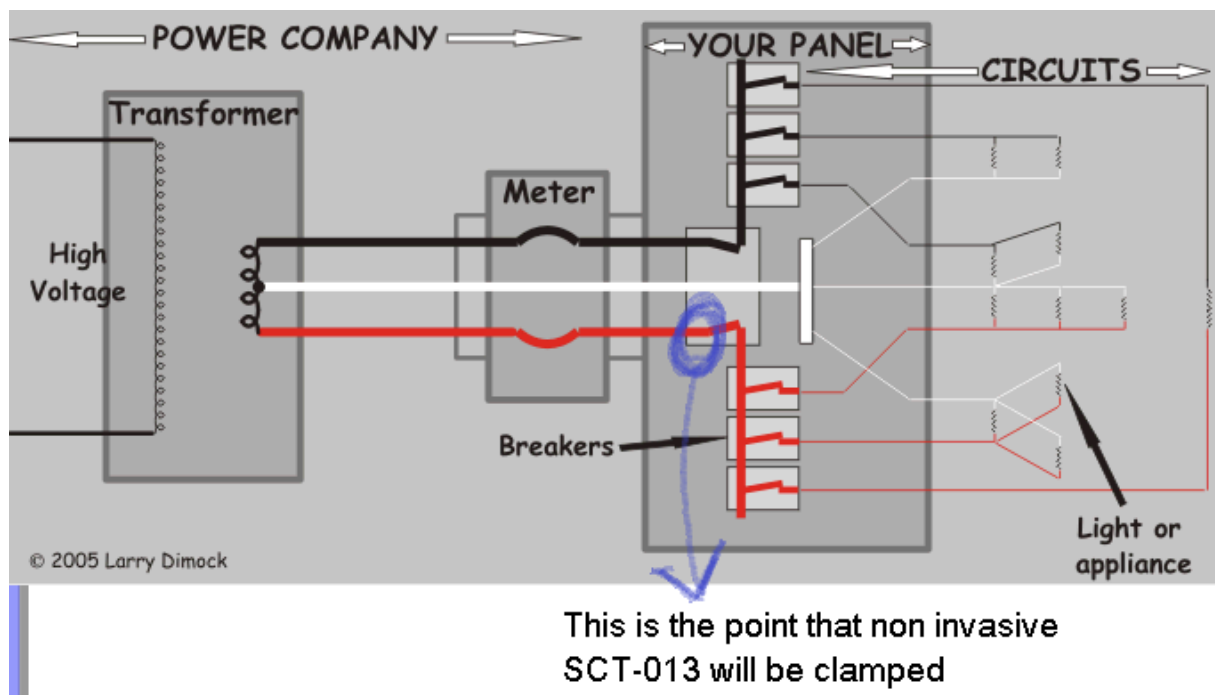
Big Picture

- Future is empowering consumer
- Consumer appliances generate more data/interactive
- No need to meter?
- IP is enough
- House server can do everthing DSO can do



Therefore, before my utility starts to install smart meters, I would test the my own smart monitoring system. This article is also a testimony to my main question of “Why smart meters when you have IoT?”. The answer is not binary.

After the new Turkish electricity law in 2013 the ownership of meters have changed. All meters have been transferred to the distribution company overnight. Since meter is distribution companies’, I can’t put my sensors around the meter. So I have to find another way.

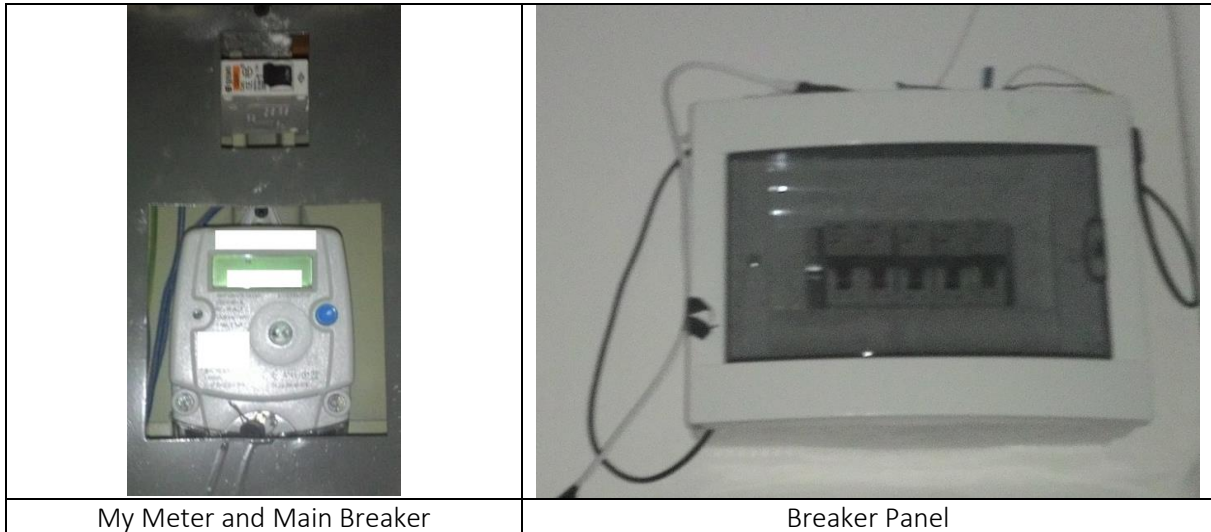


Home electricity system and SCT-013’s location^{iv}

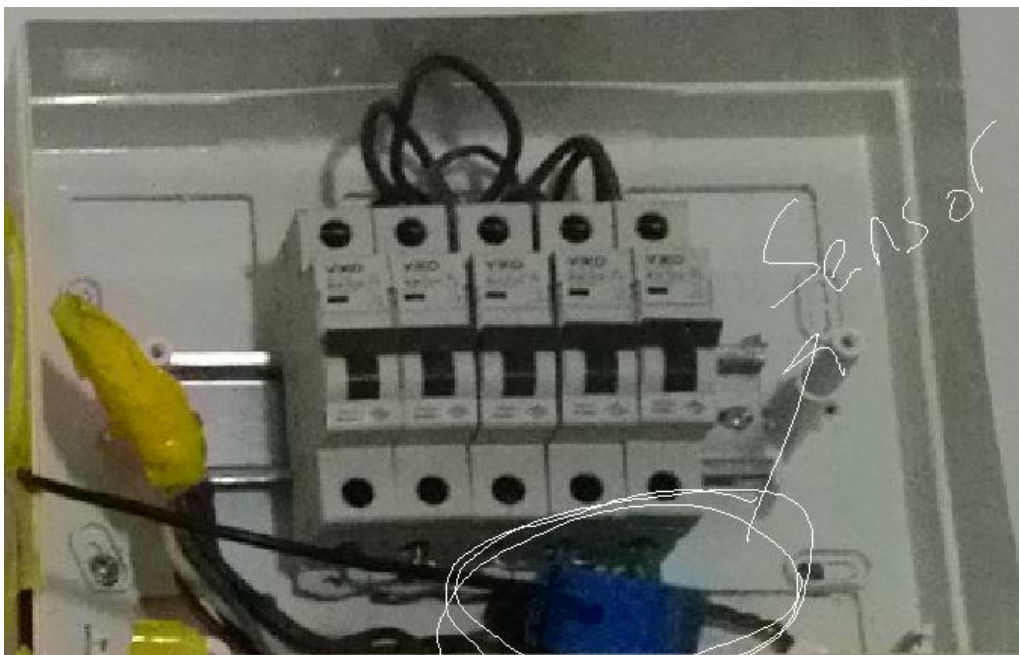
In the home electricity system meter can't be touched, so the breaker panel is used. From the breaker panel, if you can locate the active line coming from the meter or going to the household system you can put the non-invasive sensor at that point.

In Practical Terms

The meter can not be touched. The breaker panel is inside the house, therefore a non invasive sensor can be installed. The main problem is the death and health risk hazard and a fire hazard, so from this point **on do not attempt to do what I have done** at your home. Remember the setting is for Turkish electricity system.



I first shutdown the main breaker at the top of meter(Main breaker). So no electricity can flow to my breaker panel and household appliances. I couldn't pull the cables since they seem to stuck, so I didn't interfere with the system. I put the non invasive SCT-013 to a point where the "active" line leaves the breaker as in the picture below:



The sensor I used is a SCT-013-000. It is a non-invasive current sensor, that you can open and clamp around the cables the measure the current flowing through that cable.



A SCT sensor^v

This sensor does not measure voltage, so I assumed voltage on the line as constant (this is an assumption, generally voltage fluctuates around 220 V for Europe). I tried to test the current sensor with electricity heaters and hardly achieved any significant results. My main strategy was to measure an electricity load first with a PowerMeter and then with SCT, then find a coefficient to match these two numbers. However SCT was either giving very noisy signal or no signal at all.

SCT Circuit

First thing you have to be careful about SCT is you need to know the details. From a Turkish provider, I checked the exact name of my sensor and find out the datasheet. Datasheet is important because you need to calculate burden resistor.

Burden resistor is the resistor that you will read the voltage difference across to calculate current. Generally 33 ohms or 17 ohms can be used. But the resistor value depends on your circuits voltage(5V or 3.3V), your voltage level(110V or 230 V), number of turns of SCT, maximum Amperes etc.

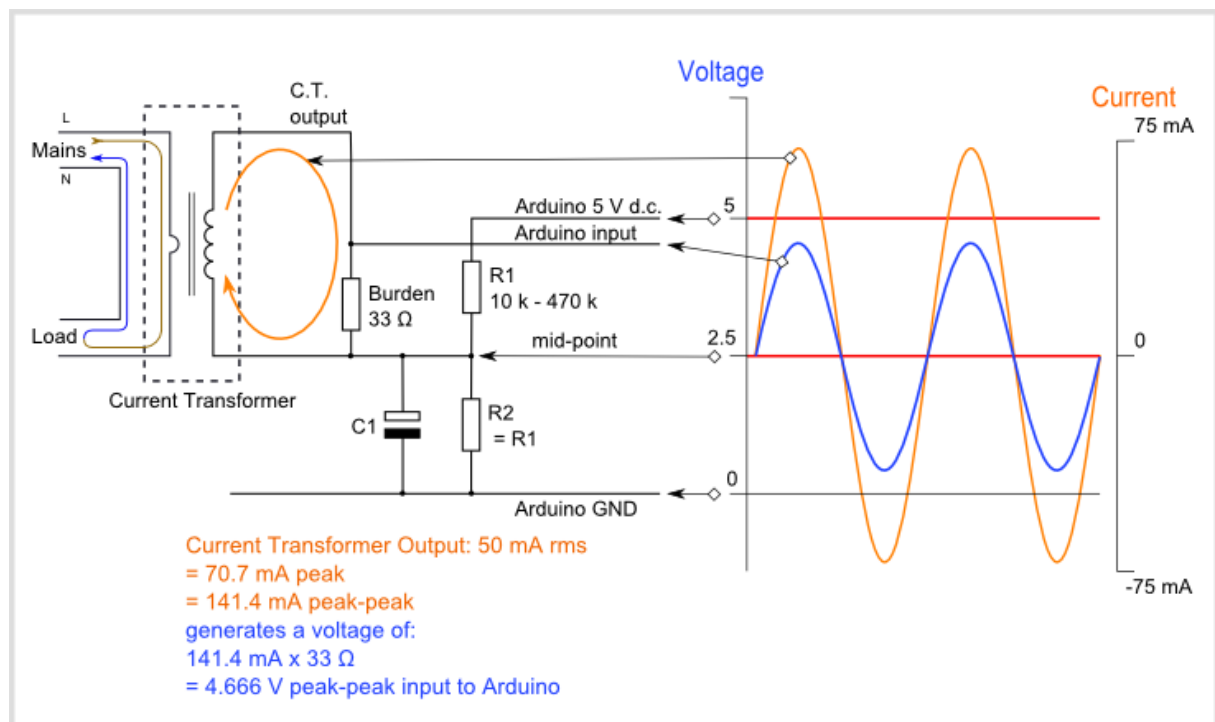
One issue was the Spark Core's 3.3 Vs. Arduino examples have comfortable 5Vs but for this example Spark core's voltage level has changed the calculations. One other problem was the voltage difference between my laptop's USB outage and a Samsung mobile phone charger's output. The calculations, I made with my laptop seemed not to fit when I replaced power supply with a Samsung charger.

Luckily, Open Energy Monitor has a BurdenResistorCalculator.xls file. From the datasheet I found the turns and voltage levels, I know my voltage levels. In Turkey the households are around 6-8 kW (my house is small). So approximately 68 ohms will do the job.

C	D	E
Burden Resistor Calculator		
Editable Fields		
240	V Supply	
current transformer		
30	A max	
50	mA output	
1800	turns	
30	A Desired max current	
70.71068	mA peak output	
42.42641	A max current (p2p)	
0.02357	A max peak current	
3.3	V max micro input voltage	
70.00357	Ω Burden Resistor	
68	closest lower standard R Value	
66	closest resistance if using 2 in series	
68	USE A SINGLE RESISTOR	
0.036765	A actual max peak current	
66.17647	A actual max current (p2p)	
	A max current (rms) before exceeding max Input Voltage	
46.79383		
Resistor Calculator		
	Std Resistor Val	(+)

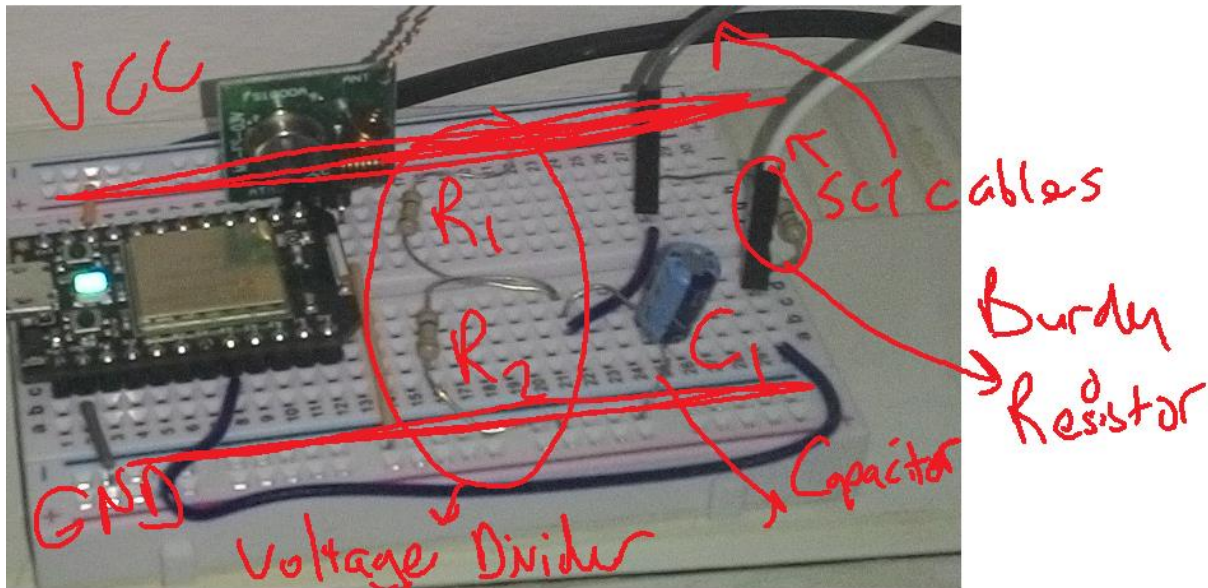
Burden Resistor Calculation^{vi}

After calculating the burden resistor, the connections to the Arduino or Spark Core can be done through this schematics:



Connecting SCT through a resistor and capacitor circuit^{vii}

Remember SCT has two cables. One cable is set to the reference voltage through voltage divider ($R_1=R_2$ in the above schematics), the other cable is connected to this point through burden resistor. A 10 uF capacitor is also used(C_1).



My connections for the schematics above

I used 330kohms for voltage divider and you should calculate your burden resistor yourself. Otherwise, the output may clip and deceive you. Remember you are already making some error by assuming the voltage is constant, since you didn't put a voltage sensor to your system.

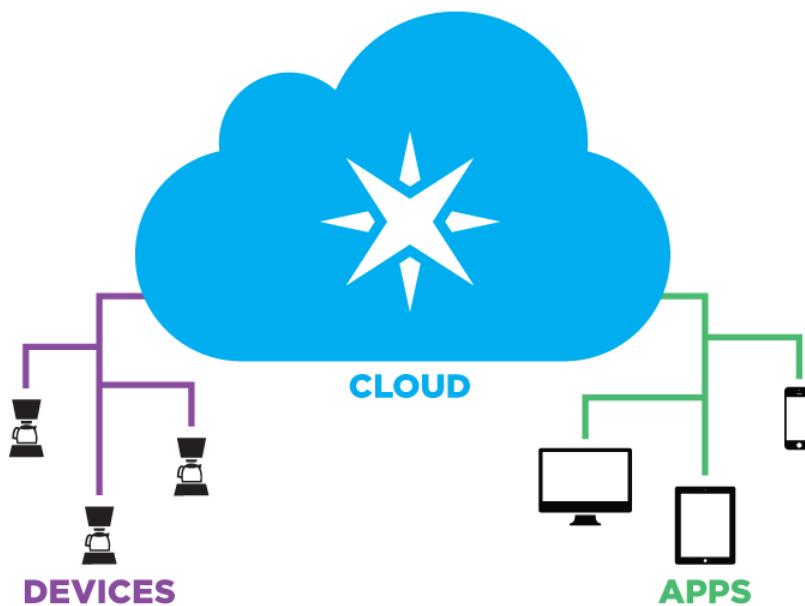
This will make your sensor ready for connecting with spark core and to the Internet

Spark Core – “Core”

“Spark Core is a \$39 development kit for creating Wi-Fi connected products”^{viii}. It is based on STM32 microcontroller with a Texas Instrument's CC3300 Wi-Fi Module. Spark was renamed “Particle” recently. The picture below shows my Spark Core from a Turkish supplier.

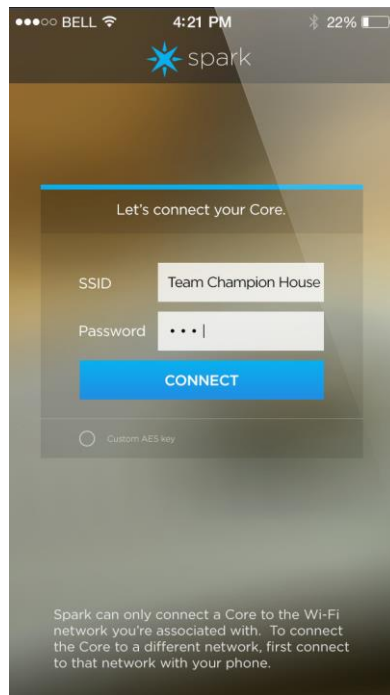


Although I didn't intend to buy the breadboard and the USB cable, it came with a hefty price of \$66. (180 TL). My main motivation was to understand the workings of a Arduino+WiFi+Cloud system. There are much cheaper solutions around. But Spark Core is compatible with Arduino programs.



Spark Core connects through Cloud

After the arrival of my Core, I installed the “Spark Core” program from the Android store. Following the procedures from <http://docs.particle.io> , I finally connected my spark core to my wireless network. Your phone and spark core should be connected to the same WiFi network to identify Spark Core.



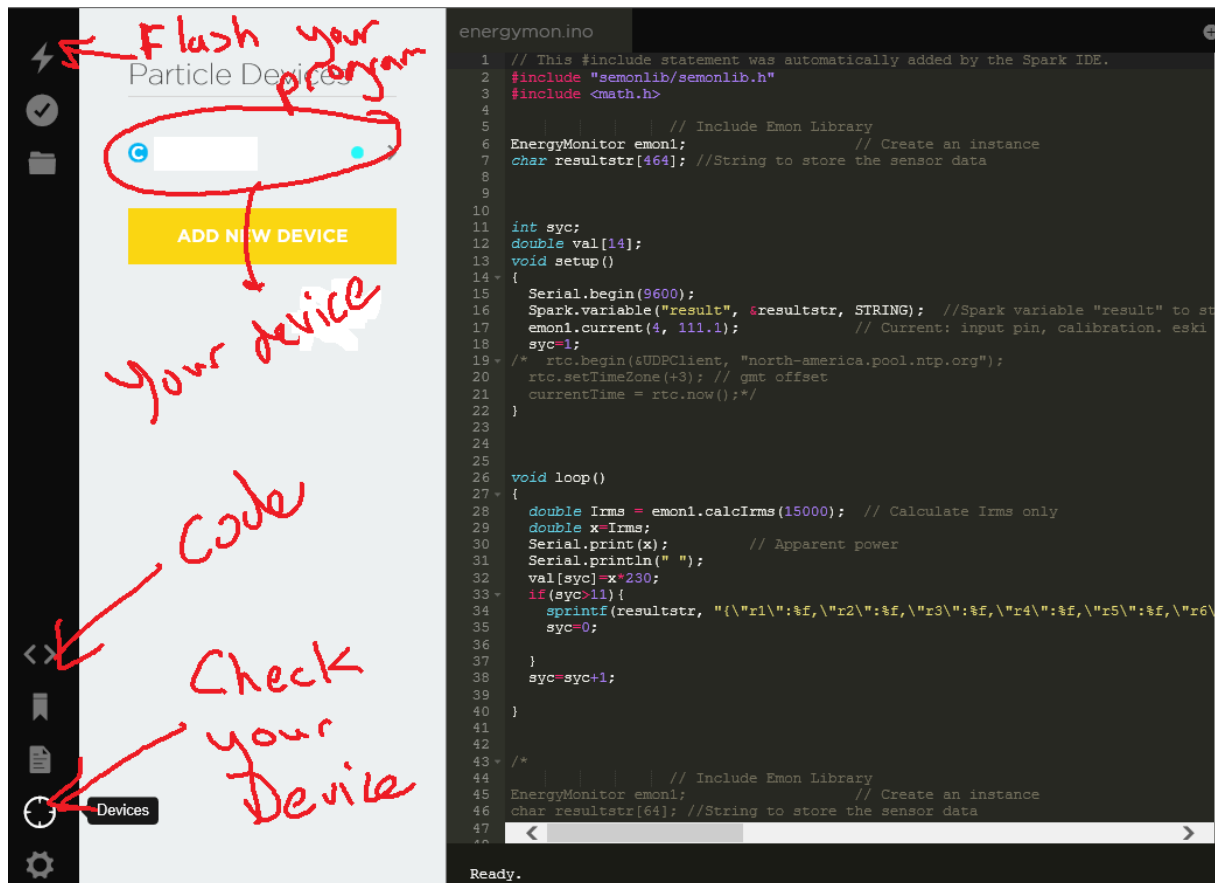
Mobile phone app to identify and connect Core to Cloud

With the user name and password you identified your Core, you can now log on to Web IDE to program.

Building the Program

When you login to the build.particle.io with the username and password you choose from the mobile phone app, you can code everything from the Cloud and flash it to your connected Core. So first login to the

[Http://build.particle.io](http://build.particle.io)



My building environment looks like the following screen.

This is where you insert your code and associate it with libraries. In my program I used two libraries

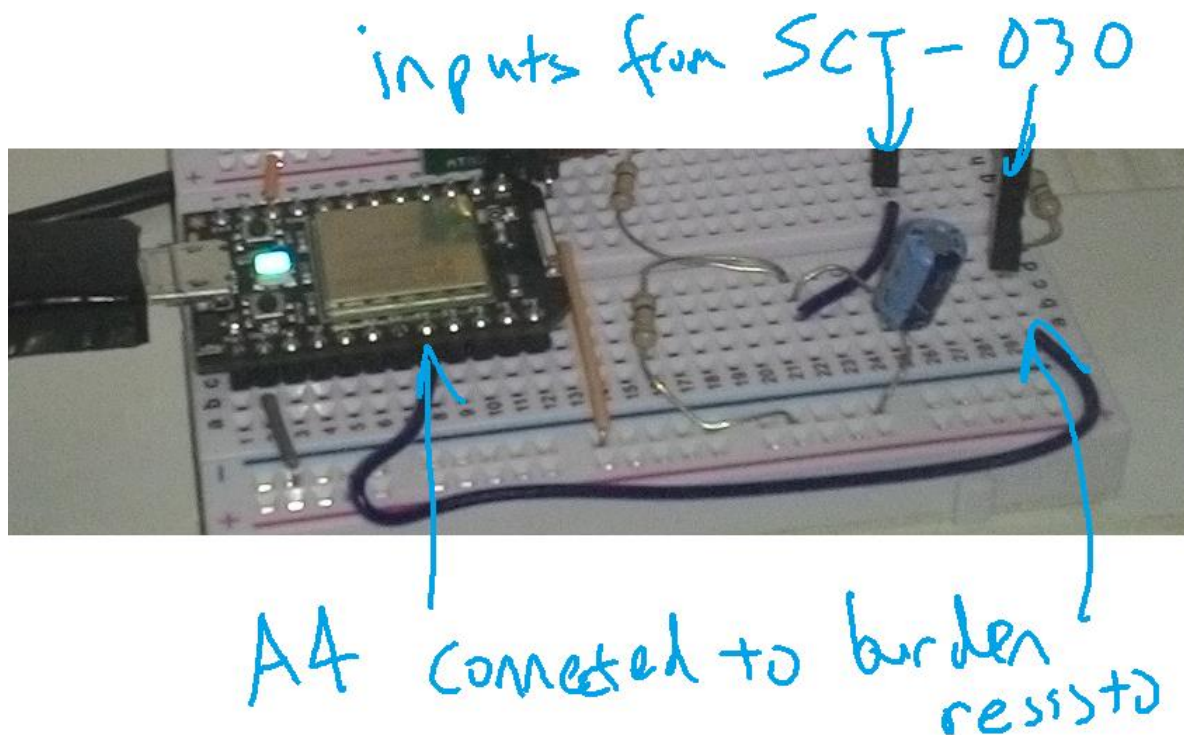
- Semonlib.h
- Math.h (I need this one for some other calculations like log values)

You can follow the tutorials on the web page to learn more about programming the Core. When Core program is verified and flashed(uploaded) from the Cloud to your device, the color of the led changes to magenta.

If not, you have to reset your device, which was not that easy.

The Code

From the Web IDE, select the library Semonlib(Spark Emon Library – Spark version of Arduino Emon lib). I used the Analog 4 (A4) as the input from the SCT-030 circuit.



My code is different from the ones on the net, because I gather 11 samples in a minute and push them to the cloud as a series of variables namely(r1,r2,r3...etc)

The detailed explanation for my code is as follows:

- a. Insert the libraries you will use:

```
#include "semonlib/semonlib.h"
#include <math.h>
```

- b. Define an instance of energy monitor

```
EnergyMonitor emon1;
```

- c. Define a char array for submitting data

```
char resultstr[464]; //String to store the sensor data
```

- d. A variable to count till 11-> a counter

```
int syc;
```

- e. A double(floating number) array to store 11 samples

```
double val[14];
```

- f. Setup part

```
void setup()
{
```

- a. Serial connection speed 9600 for testing purposes

```
Serial.begin(9600);
```

- b. A cloud variable "result" that will link us to the data from resultstr array

```
Serial.begin(9600);
Spark.variable("result", &resultstr, STRING); //Spark variable "result" to store sensor
```

- c. Energy monitor initialization

```
emon1.current(4, 111.1);
```

- d. Initialize syc-> counter index to 1

```
syc=1;
```

g. Loop Part

```
void loop()
{
```

- a. Calculate Irms (every 6 seconds)

```
double Irms = emon1.calcIrms(15000); // Calculate Irms only
```

- b. Put the Irms into x (This is an unnecessary step for this example, you can calculate from Irms)

```
double x=Irms;
```

- c. Print x from the serial ->for debug purposes and add a line end to the serial

```
Serial.print(x); // Apparent power
Serial.println(" ");
```

- d. Copy Irms*230 to the current val indexed by syc variable

```
val[syc]=x*230;
```

- e. When syc arrives 12 (that means 11 variables are stored in val array)

```
if(syc>11){
```

- i. Put all the values in val to resultstr as r1 , r2 , r3 ,r4

```
sprintf(resultstr, "\r1\":%f,\r2\":%f,\r3\":%f,\r4\":%f,\r5\":%f,\r6\":%f,\r7\":%f,\r8\":%f,\r9\":%f,\r10\":%f,\r11\":%f", val[1],val[2],val[3],val[4],val[5],val[6],val[7],val[8],val[9],val[10],val[11]); //Write sensor data to string
```

- ii. Reset the syc to zero

```
syc=0;
```

- f. Increment the syc

```
syc=syc+1;
```

The heart of the code is the g.e.i part which stores all the data into resultstr variable. Although there are other ways to do this more efficiently, I think the following is much more understandable:

```
sprintf(resultstr,
"\r1\":%f,\r2\":%f,\r3\":%f,\r4\":%f,\r5\":%f,\r6\":%f,\r7\":%f,\r8\":%f,\r9\":%f,\r10\":%f,\r11\":%f", val[1],val[2],val[3],val[4],val[5],val[6],val[7],val[8],val[9],val[10],val[11]); //Write sensor data to string
```

Code calculates some Irms from Enmon library, then stores it in val array at the index of syc, then increases syc until syc hits 12. When there are 11 variables in the val array, it posts all the data to the cloud.

Why I did this is because google spreadsheets time triggers' maximum resolution is 1 minute. Because, I can not push data to Google instantaneously but every minute, I tried to send 11 data every minute.

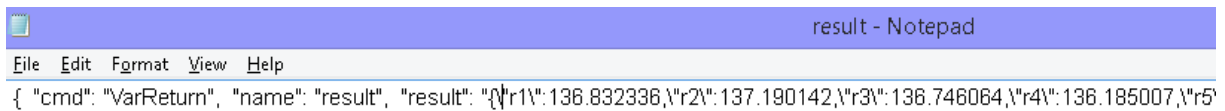
Google Spreadsheet Connection

After flashing the program, you can connect

https://api.particle.io/v1/devices/device_id/result?access_token=access_token web site to download your data.

device_id is the unique device id(from devices) your spark core is assigned, and access_token (from settings) is the token string given from build.spark.io

Web response will be a text file with data such as follows.

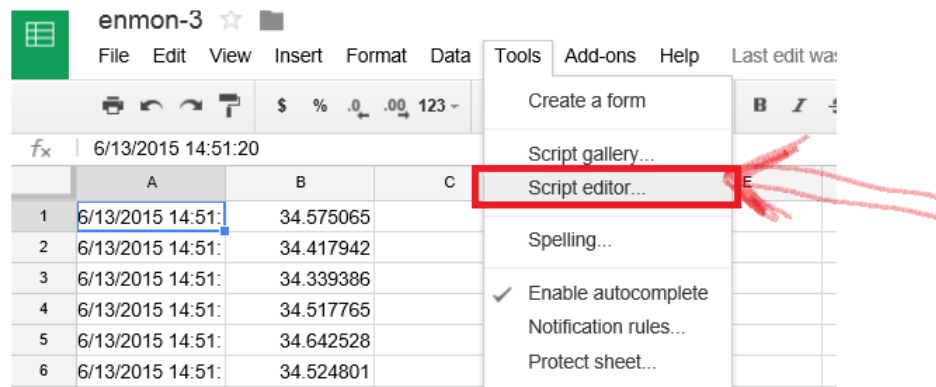


Now the important thing is to get this data to be stored in Google Sheets. This webpage demonstrates a very good example with Google Sheets:

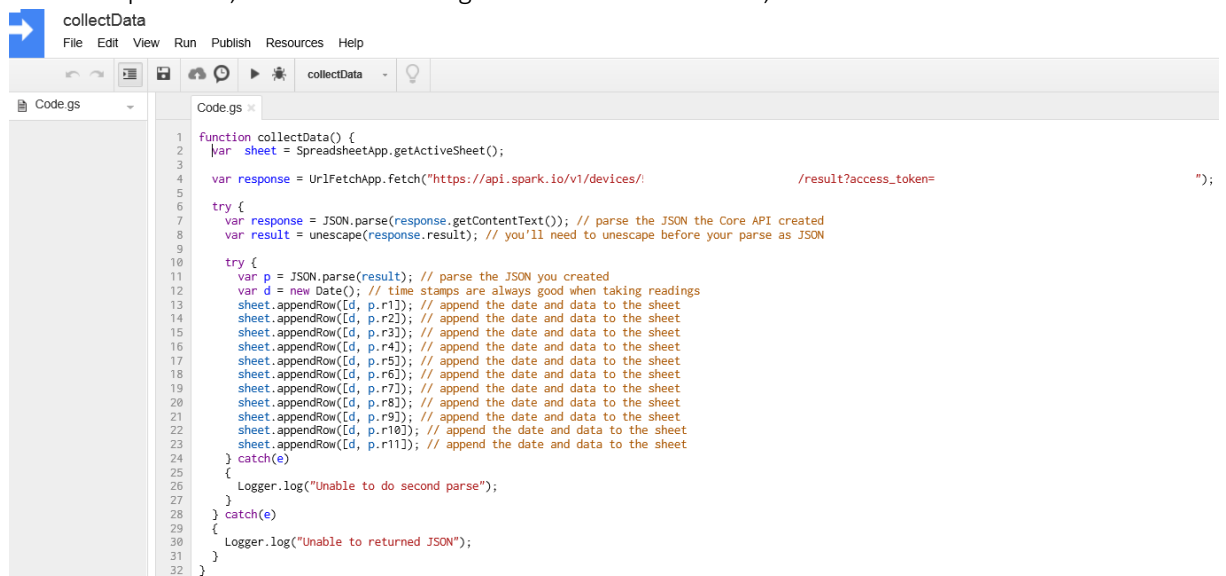
<http://www.instructables.com/id/Datalogging-with-Spark-Core-Google-Drive/?ALLSTEPS>

My procedure is as follows:

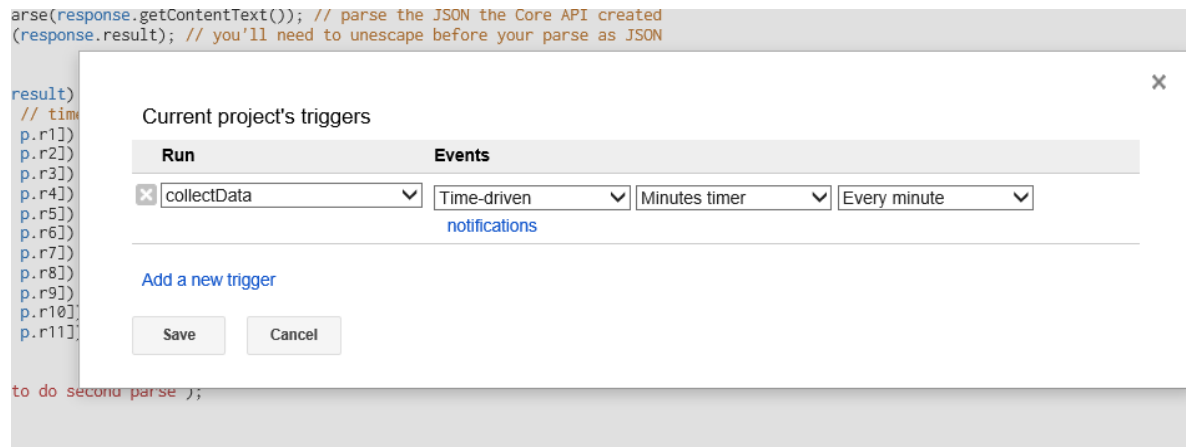
1. Open a new Google Spreadsheet from Google Drive
2. From Tools-> open Script Editor



3. In the Script Editor, insert the following code “collectData” code, and save it.



4. Add a time trigger from menu Resources > Current project triggers as follows



5. Save and Run, if no errors then close

My collectData code is the same as the example from the link above, but with p.r1, p.r2, p.r3 lines

```
function collectData() {
    var sheet = SpreadsheetApp.getActiveSheet();

    var response =
    UrlFetchApp.fetch("https://api.spark.io/v1/devices/device_id/result?access_token=token_id");

    try {
        var response = JSON.parse(response.getContentText()); // parse the JSON the Core API
        created
        var result = unescape(response.result); // you'll need to unescape before your parse as
        JSON

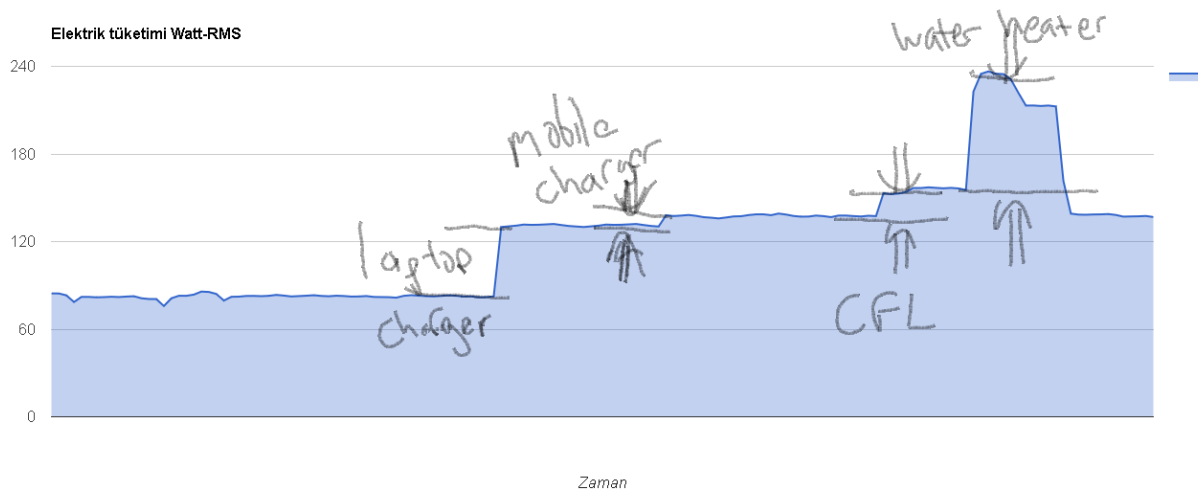
        try {
            var p = JSON.parse(result); // parse the JSON you created
            var d = new Date(); // time stamps are always good when taking readings
            sheet.appendRow([d, p.r1]); // append the date and data to the sheet
            sheet.appendRow([d, p.r2]); // append the date and data to the sheet
            sheet.appendRow([d, p.r3]); // append the date and data to the sheet
            sheet.appendRow([d, p.r4]); // append the date and data to the sheet
            sheet.appendRow([d, p.r5]); // append the date and data to the sheet
            sheet.appendRow([d, p.r6]); // append the date and data to the sheet
            sheet.appendRow([d, p.r7]); // append the date and data to the sheet
            sheet.appendRow([d, p.r8]); // append the date and data to the sheet
            sheet.appendRow([d, p.r9]); // append the date and data to the sheet
            sheet.appendRow([d, p.r10]); // append the date and data to the sheet
            sheet.appendRow([d, p.r11]); // append the date and data to the sheet
        } catch (e)
        {
            Logger.log("Unable to do second parse");
        }
    } catch (e)
    {
        Logger.log("Unable to returned JSON");
    }
}
```

Insert your own device_id and token_id from procedures explained above.

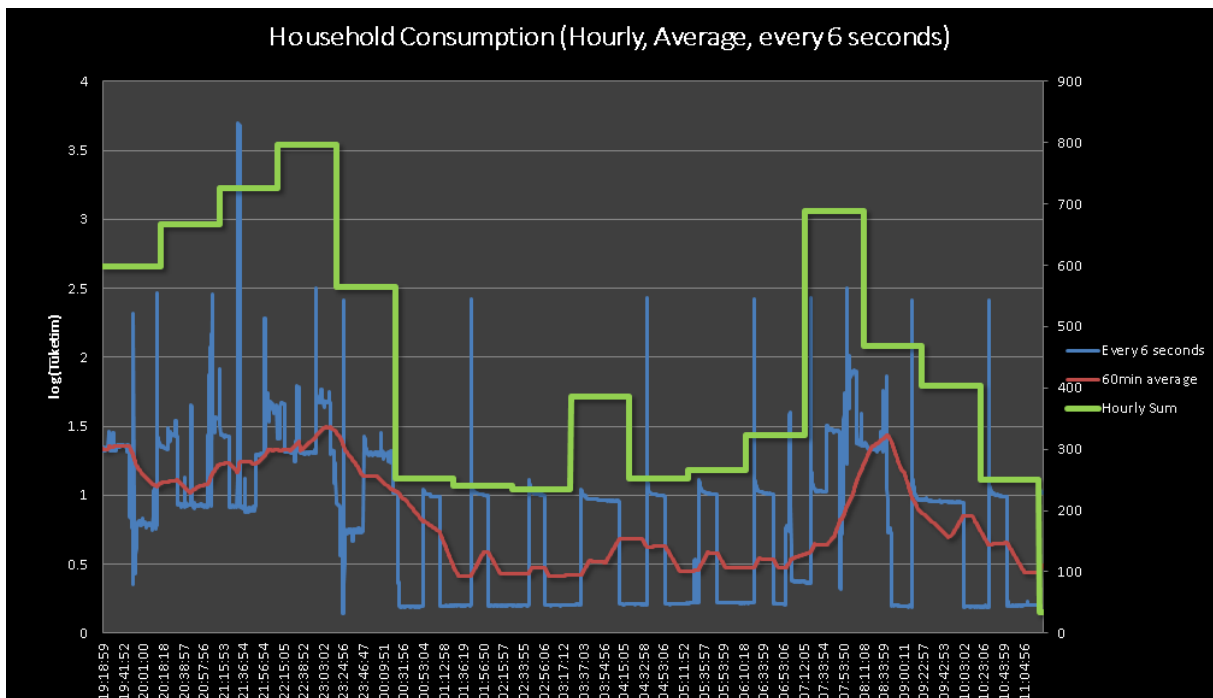
Data and Results

After running the code for several days, some patterns emerged from my daily consumption. During the experiment period, I was the only occupant of the house.

An initial view of the data was as follows in a quite short span. This shows me plugging the chargers and going to kitchen:



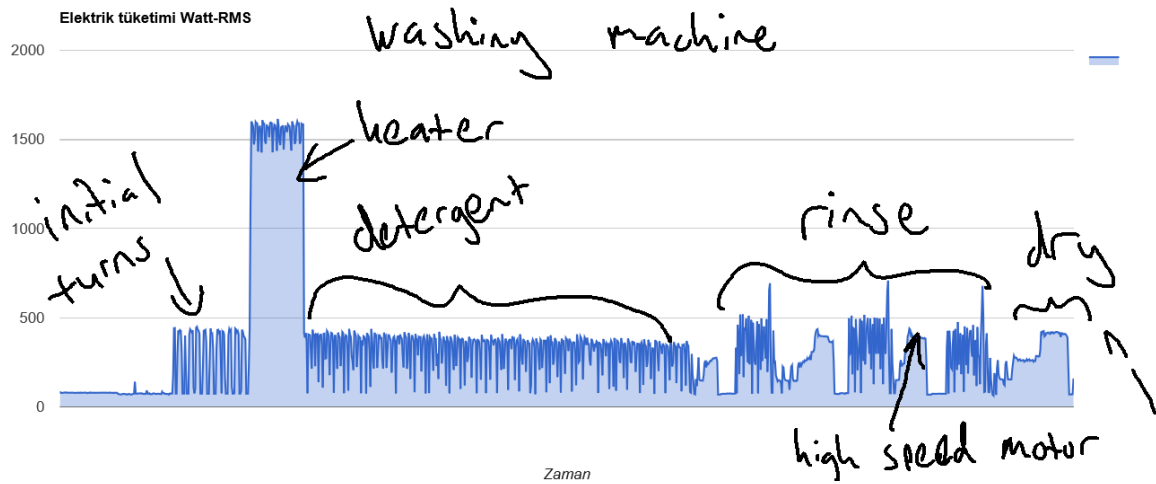
As data accumulated, the difference between hourly data, average data and instant becomes more visible. The graph below illustrates that point.



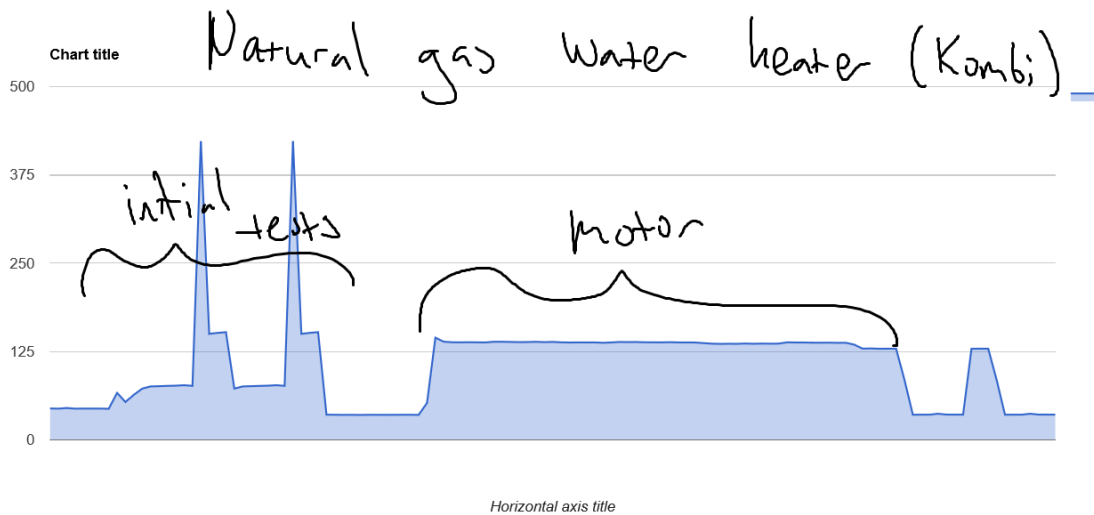
- Green line is the sum of all values in one hour and it is quite discrete
- Red line is the moving average over 60 minutes, it reflects the peaks in consumption with a time delay
- Blue is the instant or every 6th second data, which is very peaky.

As more data is collected following data of household appliances appeared.

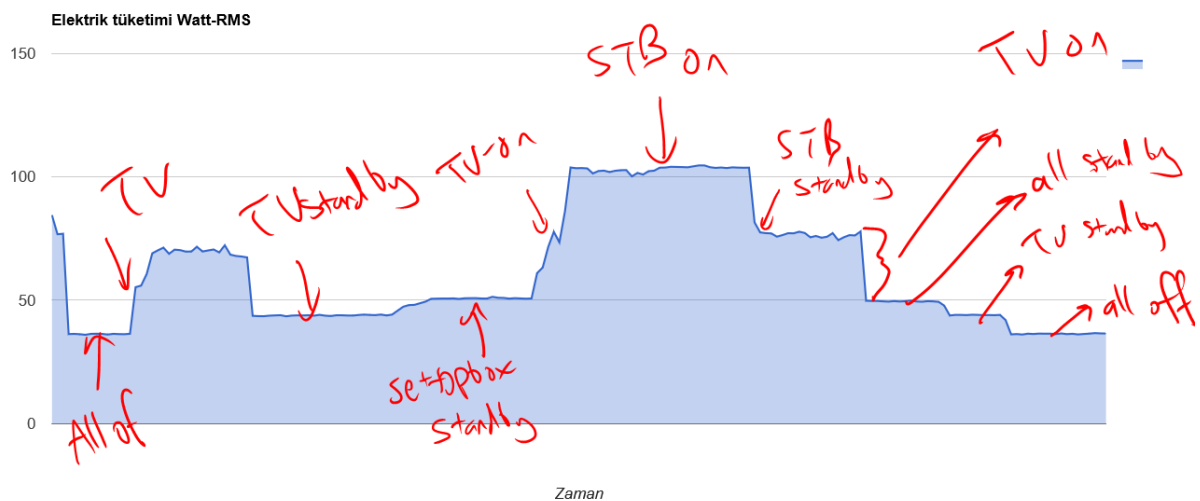
A washing machine's electricity consumption looks like this:



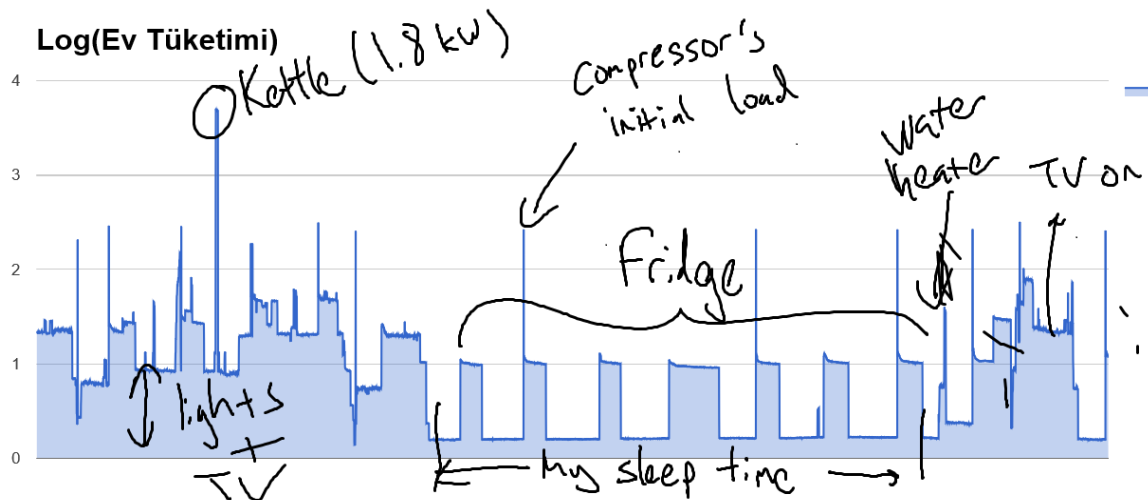
A Natural Gas water heater still needs electricity to pump hot water:



Another one is how much electricity my TV and cable TV box consumes, during On/Off and stand by periods.



Fridge(A+++) is also visible with a very distinct pattern



Conclusion

Internet of Things (IoT) is changing the world as the costs go down and the devices become more accessible. Now people can have more data about their activities as I demonstrated above. Back to my original point: as Turkish utilities can not deploy smart meters for households, IoT devices can give consumers more insight about their daily activities and consumption.

What I learned from this experiment is

- Do not let the things at stand by, turn off or unplug them
- My daily consumption varies a lot in detail
- There are room for improving my carbon footprint

As I will gather more data, I will share it with everyone interested. I hope this project will make me more energy efficient.

Bariş Sanlı, 14 June 2015 , barissanli.com , barissanli2@gmail.com

Code for Spark Core

```
// This #include statement was automatically added by the Spark IDE.
#include "semonlib/semonlib.h"
#include <math.h>

// Include Emon Library
EnergyMonitor emon1; // Create an instance
char resultstr[464]; //String to store the sensor data

int syc;
double val[14];
void setup()
{
  Serial.begin(9600);
  Spark.variable("result", &resultstr, STRING); //Spark variable "result" to store sensor data string
```

```

    emon1.current(4, 111.1);          // Current: input pin, calibration. eski hali 10000
    syc=1;
    /* rtc.begin(&UDPCClient, "north-america.pool.ntp.org");
    rtc.setTimeZone(+3); // gmt offset
    currentTime = rtc.now();*/
}

void loop()
{
    double Irms = emon1.calcIrms(15000); // Calculate Irms only
    double x=Irms;
    Serial.print(x);          // Apparent power
    Serial.println(" ");
    val[syc]=x*230;
    if(syc>11){
        sprintf(resultstr,
"%r1\":"%f,%r2\":"%f,%r3\":"%f,%r4\":"%f,%r5\":"%f,%r6\":"%f,%r7\":"%f,%r8\":"%f,%r9\":"%f,%r10\":"%f,%r11\":"%f",val[1],va
l[2],val[3],val[4],val[5],val[6],val[7],val[8],val[9],val[10],val[11]); //Write sensor data to string
        syc=0;

    }
    syc=syc+1;
}

```

ⁱ Here's Why 'The Internet Of Things' Will Be Huge, And Drive Tremendous Value For People And Businesses

Read more: <http://www.businessinsider.com/growth-in-the-internet-of-things-market-2014-1#ixzz3d3CZALZH>

ⁱⁱ <http://blogs.wsj.com/accelerators/2014/09/22/ted-leonsis-an-exciting-time-for-the-internet-of-things/>

ⁱⁱⁱ <http://www.barissanli.com/calismalar/2014/20141105-bsanli-Towards%20a%20Smarter.pdf>

^{iv} http://www.thecircuitdetective.com/home_electrical_system_diagram.gif

^v <http://openenergymonitor.org/emon/buildingblocks/report-yhdc-sct-013-000-current-transformer>

^{vi} http://openenergymonitor.org/emon/sites/default/files/BurdenResistorCalculator_0.xls

^{vii} <http://openenergymonitor.org/emon/buildingblocks/ct-sensors-interface>

^{viii} <https://www.particle.io/prototype>