

Análisis de una carga doméstica.

Alumno: Victor Khomyakov

1. Definición de la carga

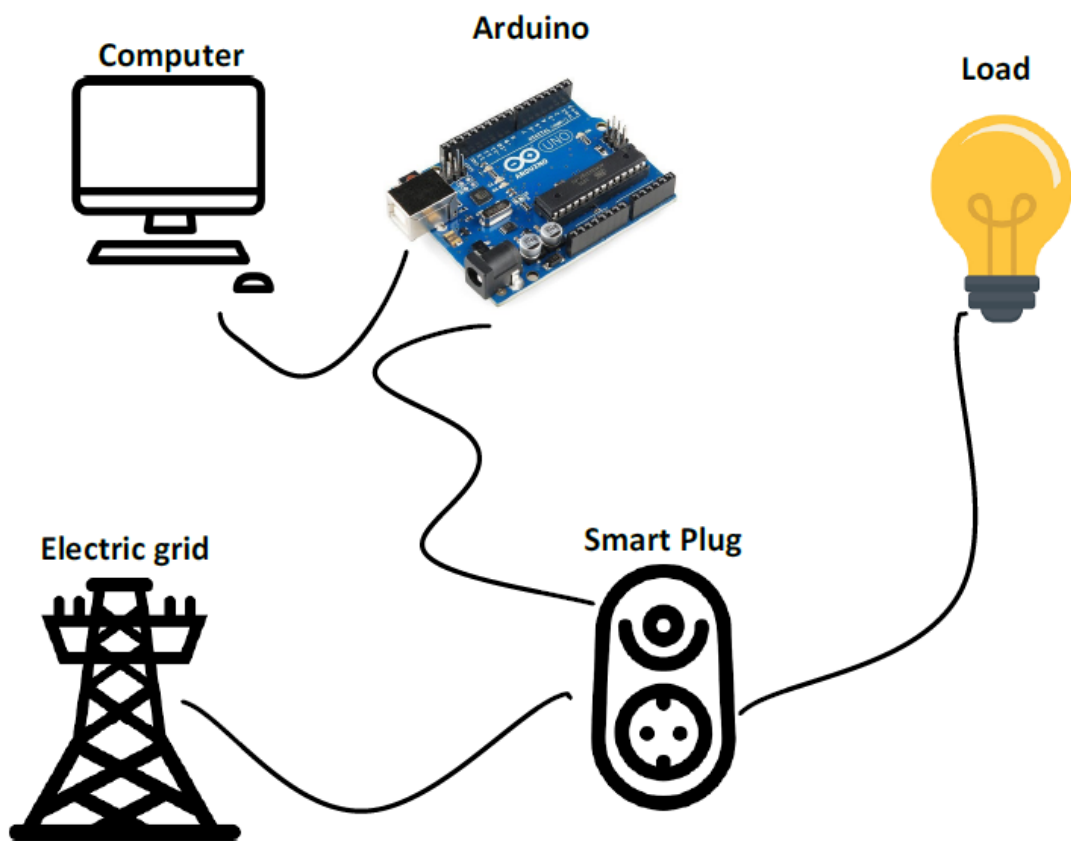
Lavadora: modelo FAR F-LAV605.



Los parámetros de la lavadora están presentados en la figura a continuación.



2. Montaje de la instalación



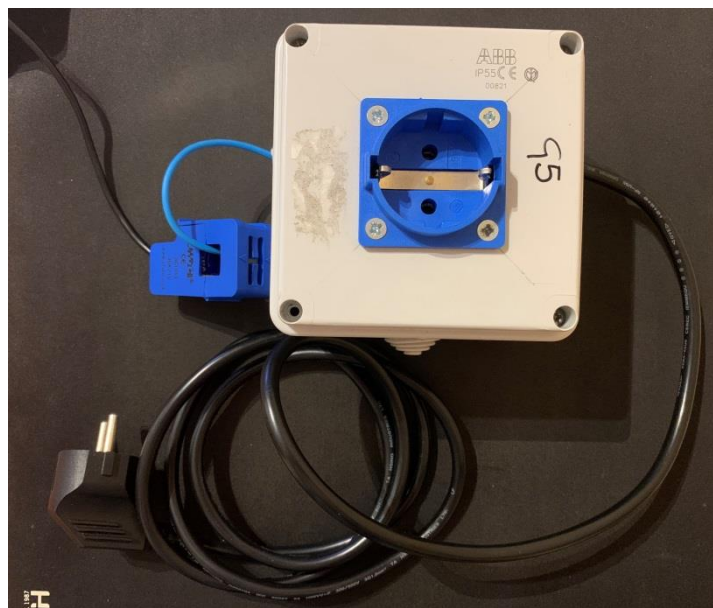
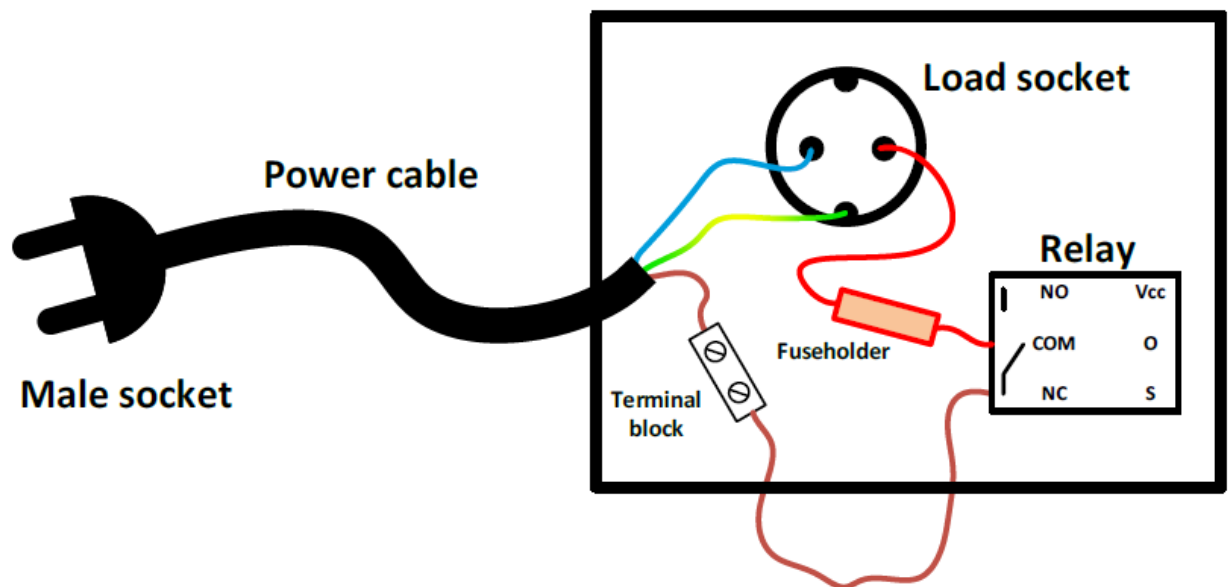
i. Montaje de hardware

El primer paso antes de medir la corriente es ensamblar los elementos del enchufe inteligente para tener una toma de corriente disponible lista para ser medida.

Los elementos para este primer ensamblaje serán los siguientes:

- Caja de conexión con enchufe hembra.
- Cable de alimentación de red con una toma de corriente macho.
- Fusible de 10 A.
- Portafusibles en línea.
- Cable eléctrico y un bloque de terminales para hacer las conexiones.

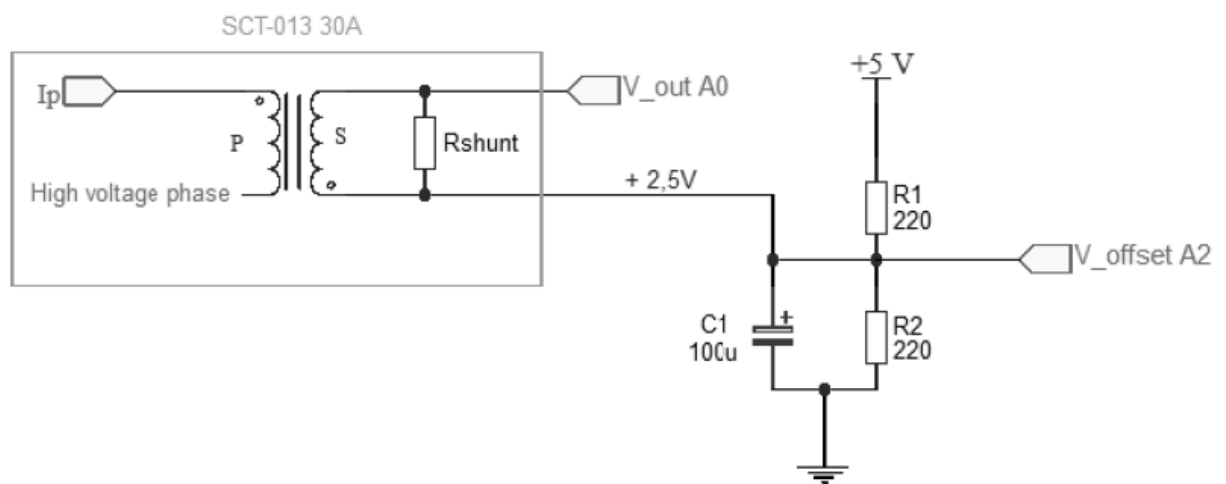
El ensamblaje básico se muestra en las figuras a continuación.



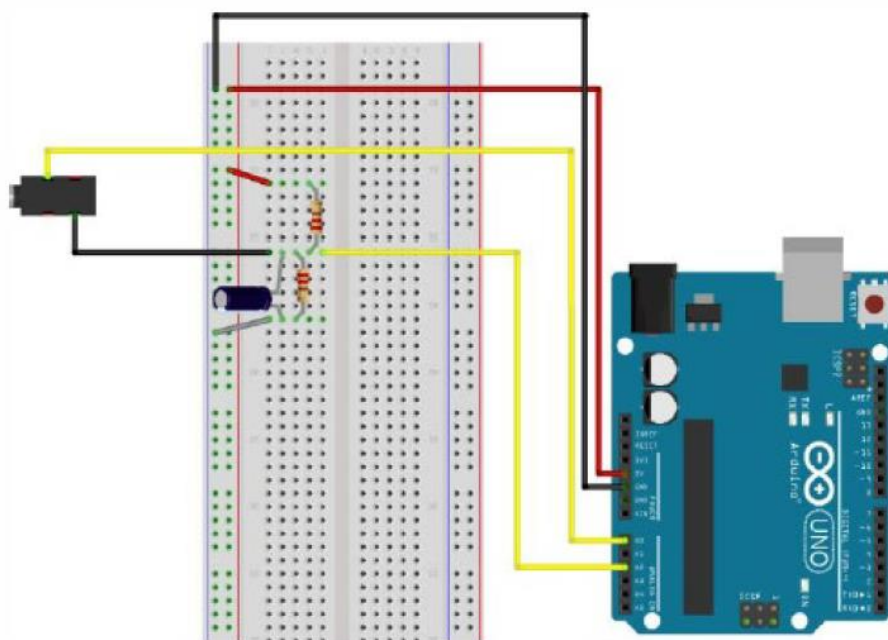
ii. Montaje de medición de corriente

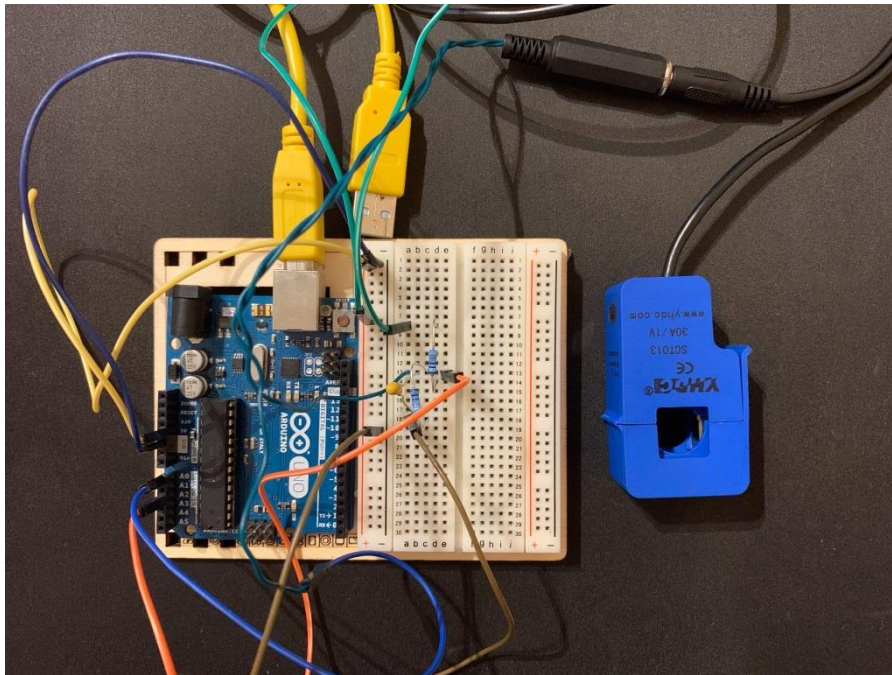
El SCT013 es un dispositivo transformador de corriente (CT) que proporciona una medida proporcional a la corriente que fluye a través del circuito principal. La medición se realiza por inducción electromagnética.

El circuito eléctrico se representa en la figura a continuación. Las resistencias pueden ser de cualquier valor, pero deben ser iguales (en este caso, se han utilizado dos resistencias de $220\ \Omega$). El condensador se puede agregar para actuar como un filtro del ruido en caso de picos de voltaje.



Las conexiones físicas con la placa de pruebas se realizaron de la siguiente manera:





iii. Instalación de software y recopilación de datos

Primero, se establecen los parámetros iniciales (definición de pin, variables auxiliares, parámetros de red), que usaremos en el código.

```
arduino_SerialPort_to_pc


unsigned long time_now = 0;
unsigned long time1_ant = 0, time2_ant = 0;
unsigned long count = 0;
float sum1 = 0, sum2 = 0;
char comando;

const int SensorPin = A0 , RefPin = A2 ;
const int Rshunt = 33;

unsigned long time_ant = 0, difTime = 0, act_time = 0;
const int sampleDuration = 20;
int count_integral = 0;
double rawSquaredSum = 0;
double Iant = 0;
double ADC_sensor;
double ADC_ref;
double V_sens;
double V_ref;

double freq = 50;
double n_trafo = 1000;
double Irms = 0;
```


Después, se inicializa las comunicaciones del puerto serie. Además se lee data de sensores, como ADC y la tensión referente. Se traduce la entrada de ADC medida a los valores de voltaje y se calcula el valor instantáneo de corriente.



```
// Configure the relay digital pin, and the serial port interface
void setup()
{
  Serial.begin (115200);
}

// Loop function
void loop()
{
  act_time = micros();
  difTime = act_time - time_ant;

  int RawValue = 0;

  if (difTime >= 1000) {
    time_ant = act_time + (difTime - 1000);

    /***** Read the ADC input from the sensor and the voltage reference point *****/
    /***** WRITE HERE THE CODE *****/
    ADC_sensor = analogRead(SensorPin);
    ADC_ref = analogRead(RefPin);

    /***** Translate the ADC input measured to voltage values *****/
    /***** WRITE HERE THE CODE *****/
    V_sens = ADC_sensor*5.0/1023;
    V_ref = ADC_ref*5.0/1023;

    // Calculate the instantaneous current using the voltage difference and the burden resistor value
    double Iinst = n_trafo * (V_sens - V_ref) / Rshunt;

    // Print the instantaneous current
    //Serial.println(Iinst);

    // Calculate the integral
    rawSquaredSum += Iinst * Iinst * 0.001;
```

Para obtener el valor más preciso, se divide tiempo en periodos de 20 ms y para cada periodo calculamos valor Irms a través de integral. A continuación, se hace un ciclo para calcular valor promedio de Irms cada 5 segundos, también obteniendo valor de la potencia. Los datos se imprimen separando con punta y coma.

```
arduino_SerialPort_to_pc

// Each 20 ms, calculate the RMS
if (count_integral >= sampleDuration)
{
    // Calculate the RMS
    Irms = sqrt(freq * rawSquaredSum)-0.110476804;
    // Counter and integral reset
    count_integral = 0;
    rawSquaredSum = 0;

    // Low-pass filter
    double Ifilt = 0.95 * Iant + 0.05 * Irms;
    Iant = Ifilt;

    // Print the RMS
    // Serial.print("Ifilt = ");
    // Serial.print(Irms);
    // Serial.print(';');
}
```

```
arduino_SerialPort_to_pc $

// Check the current time in milliseconds
time_now = millis();

// Each 1 second, measure the A0 and A1 ports
if (time_now - time1_ant > 1000)
{
    // Increment the time counter
    count++;
    // Accumulate the ADC measurements each second, to calculate latter and average value each 5 seconds
    sum1 += Irms;
    sum2 += Irms*230;
    // Update the "1 second" time flag
    time1_ant = time_now;
}

// Each 5 seconds, calculate the average value of the A0 and A1 measurements, and the state of the relay output pin
// and write the values with the serial port using semicolons to separate them
if (time_now - time2_ant > 5000)
{
    Serial.print(sum1/count);
    Serial.print(';');
    Serial.print(sum2/count);
    Serial.println(" ");

    // Reset the variables to calculate the average results
    sum1 = 0;
    sum2 = 0;
    // Reset the time counter and update the "5 second" time flag
    count = 0;
    time2_ant = time_now;
}

}
```

Al final, se utiliza el código de Python, que en este trabajo sirve para crear y guardar los datos en fichero tipo csv.

```
# importing libraries
import serial
import datetime as dt
import re
import csv

# creating communications object using Serial
arduino = serial.Serial('COM3', 115200)
print("Starting!")

# try-except-finally loop for data acquisition
try:
    while True:
        # Check if there is new info from the Arduino and read it
        data_bytes = arduino.readline()
        # Decoding the message into UTF-8
        data = data_bytes.decode("utf-8")

        # if data has been read, print and save it
        if data:
            print(data)
            # strip data string into 3 different values for each reading
            [l1,l2] = re.findall(pattern=r"[-+]?[d*]\.[d+]/\d+", string=data)
            # store data and readings into a list
            row = [dt.datetime.now(), float(l1), float(l2)]
            # save reading row into the csv file
            with open('data.csv', 'a', newline='') as csvFile:
                writer = csv.writer(csvFile)
                writer.writerow(row)
                csvFile.close()
# handling KeyboardInterrupt by the end-user (CTRL+C)
except KeyboardInterrupt:
    # closing communications port
    arduino.close()
    print('Communications closed')
```

3. Representación de datos

Se realizaron tres experimentos con diferentes tipos de lavado. Los parámetros de cada uno de ellos se presentan en la tabla a continuación.

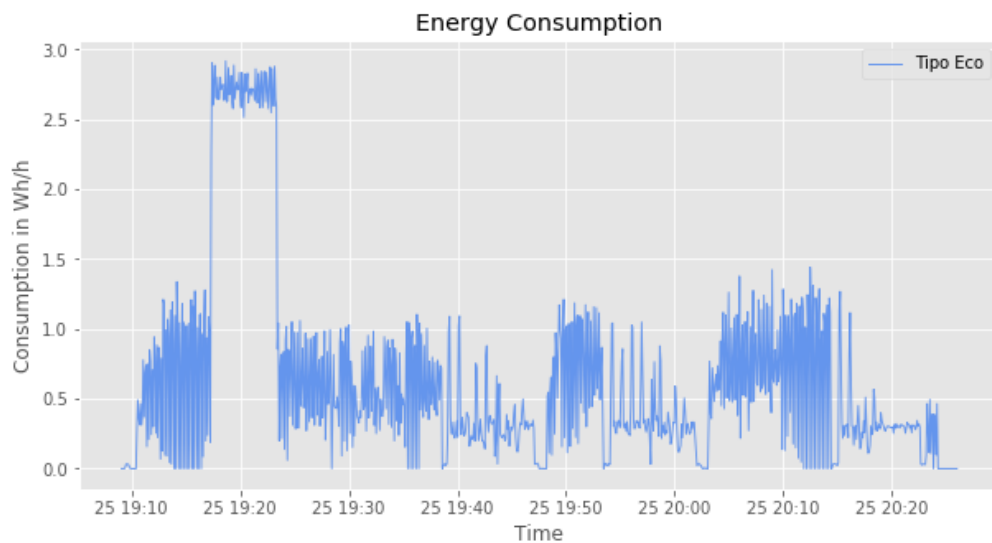
TIPO DE LAVADO	TIPO DE ROPA	TEMPERATURA	MAX CAPACIDAD ROPA SECA(kg)	CAPACIDAD ROPA REAL(kg)
ECO	ALGODON	30°	5	≈3
DELICADO	ALGODON	30°	2,5	≈2
RAPIDO	ALGODON	30°	2	≈2

Tipo ECO

En el siguiente gráfico se representa la energía consumida mientras lavado tipo Eco. Se puede determinar tres periodos interesantes:

- 1) Primeros 5 minutos se ve proceso de llenar la lavadora con el agua
- 2) Los siguientes 7-8 minutos la lavadora calienta el agua y por lo tanto se ve el máximo consumo de energía, que está a punto de llegar hasta 3 Wh/h
- 3) Al final, el último periodo se puede definir como el centrifugado que dura casi 10 minutos y el consumo llega hasta 1,3 Wh/h

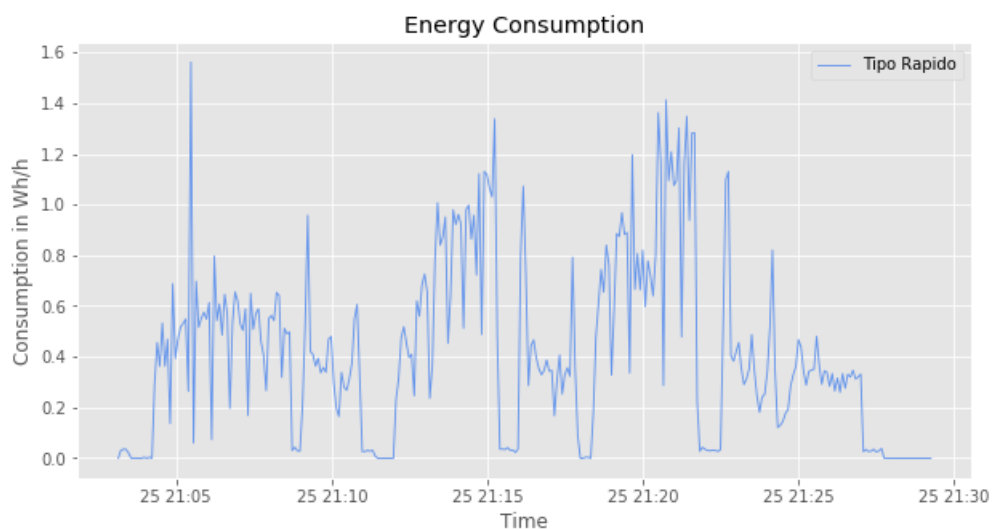
El consumo final de energía es 0,607 Wh.



Tipo RAPIDO

El siguiente tipo de lavado es un lavado rápido, que es muy similar al tipo anterior en sus periodos, solo en una escala de tiempo diferente, que es tres veces menos. El máximo consumo de energía llega hasta casi 1,6 Wh/h.

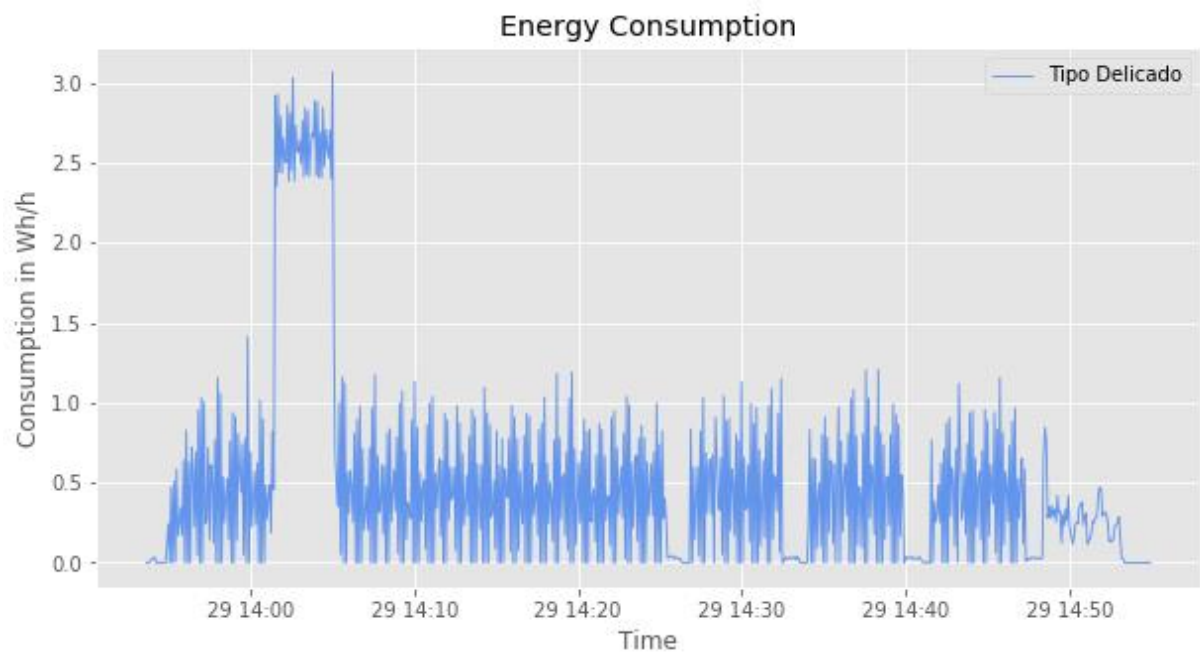
El consumo final de energía es 0,130 Wh.



Tipo DELICADO

Este tipo de lavado justifica su nombre, ya que no hay grandes espacios en el gráfico como en los dos tipos anteriores. Aunque todavía hay un período de calentamiento de agua cuando el consumo alcanza su pico, 3 Wh/h. Sin embargo, el proceso de lavado es más suave para la ropa(menos rotaciones por minuto) y se puede ver que la misma velocidad de rotación se mantiene constantemente. Además el proceso de centrifugado es más sencillo comparando con lavados anteriores.

El consumo final de energía es 0,365 Wh.

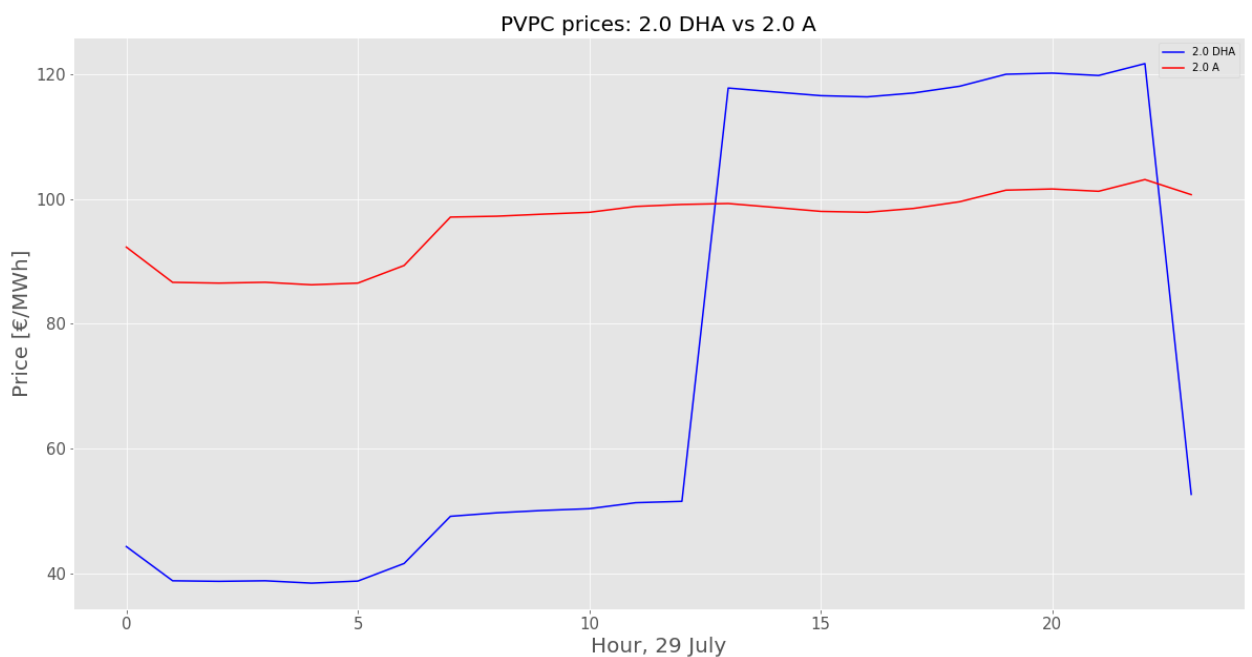
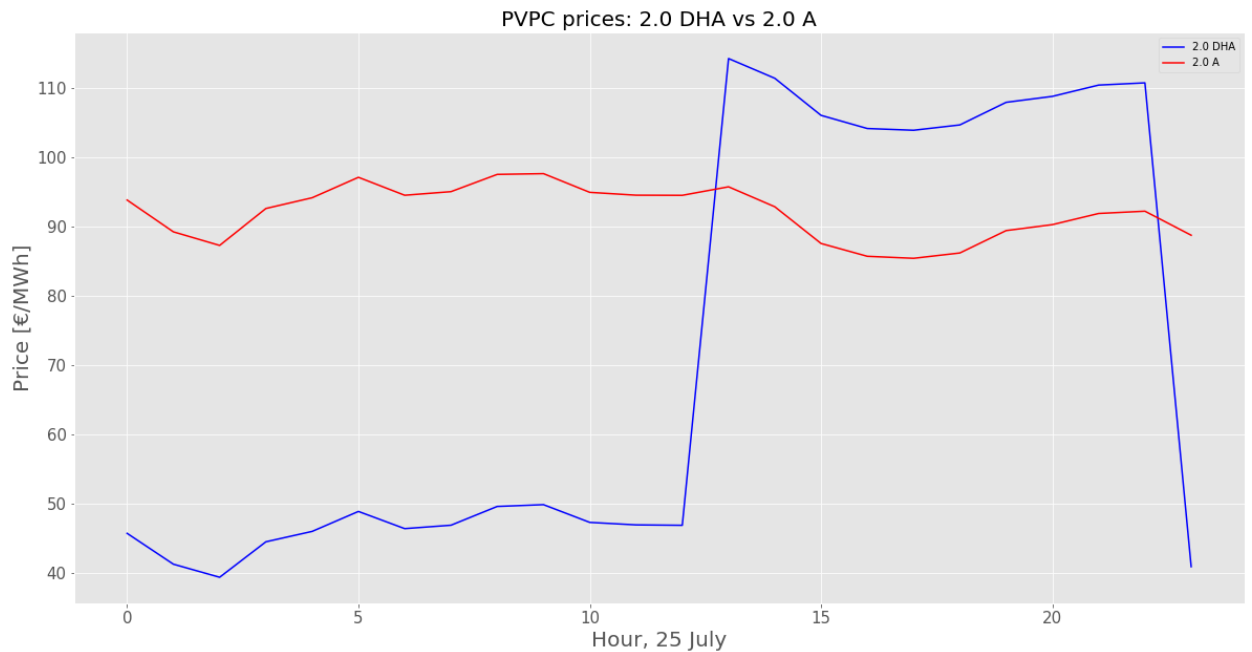


Comparación de los lavados presentados

TIPO DE LAVADO	DURACIÓN	EL CONSUMO DE ENERGÍA
ECO	01:16:55	0,607 Wh
RAPIDO	00:26:06	0,130 Wh
DELICADO	01:01:14	0,365 Wh

4. Análisis económico

Como resultado de esta parte del trabajo, se obtuvieron precios por hora de la energía para los días 25 y 29 de julio en el mercado eléctrico, mediante la consulta de los datos de la API REE a través del script de Python. En los siguientes gráficos se representan los precios horarios para dos tarifas PVPC 2.0 A y 2.0 DHA para dos días distintos.



Primero, debe tenerse en cuenta que las mediciones se llevaron a cabo en la casa, donde el contrato fue de tarifa PVPC 2.0 A. Usando los datos obtenidos de la API REE, se calculó el costo de cada lavado por la energía consumida. A continuación, se seleccionó el período de tiempo óptimo para el lavado, por ejemplo, si fuera una persona nocturna, que soliera poner la lavadora a las horas nocturnas(desde la 1 hasta las 6 de la madrugada) podría pagar el coste lo mínimo posible, como son las horas más baratas de todo el día. Pero es muy poco probable, por lo tanto se tiene en cuenta las horas más adecuadas, aunque no son más baratas. Fueron elegidos dos periodos de tiempo: 1) desde las 7 hasta las 11 para el día 29 de julio, el coste promedio es 49,87 Euros; 2) desde las 9 hasta las 13 para el día 25 de julio, el coste promedio es 47,72 Euros. Después, se calcularon los costos óptimos. Todos los resultados se representan en la tabla a continuación.

TIPO DE LAVADO	TIEMPO DE LAVADO REAL	TIEMPO DE LAVADO ÓPTIMO	EL COSTO REAL	EL COSTO ÓPTIMO
ECO	2020-07-25 19:09:04 - 20:26:00	2020-07-25 09:00:00 - 13:00:00	0.054 Euros	0.029 Euros
RAPIDO	2020-07-25 21:03:08 - 21:29:15	2020-07-25 09:00:00 - 13:00:00	0.014 Euros	0.006 Euros
DELICADO	2020-07-29 13:53:41 - 14:54:55	2020-07-29 07:00:00 - 11:00:00	0.036 Euros	0.018 Euros

5. Conclusión

Durante este trabajo, se obtuvieron conocimientos importantes y necesarios para obtener los datos de consumo eléctrico de un electrodoméstico utilizando Arduino, su posterior análisis en Python y también para conseguir los datos del mercado eléctrico utilizando REE API. Como resultado, se realizó un análisis de la lavadora y su consumo de energía, se realizó una comparación con el uso óptimo. Para terminar, se puede concluir que tener un contrato de PVPC 2.0 DHA y utilizar la lavadora por la mañana, puede reducir el costo al menos dos veces, sin embargo, falta un análisis completo de comportamiento de consumidor y su consumo diario, para que se pueda decir precisamente cuál tarifa es más adecuada para el.