Divulgaciones Matemáticas Vol. 19, No. 2 (2018), pp. 1–15

Medidor y graficador de voltaje, corriente y potencia aparente con monitor en tiempo real por Internet y local usando Arduino y Raspberry PI

Measurer and plotter of voltage, current and apparent power with real-time monitor by Internet and local using Arduino and Raspberry PI

Yemala Castillo Brito (ycastillo@unibe.edu.ec)
Facultad de Tecnologías de la Información y Comunicación.
Universidad Iberoamericana del Ecuador
Quito, Ecuador

Gerardo Herrera Roldan (gherrera2k1@gmail.com)
Coordinación de Carrera Tecnología Superior en Automatización e Instrumentación
Tecnológico Superior "El Pacífico"
Quito, Ecuador

Jesús Gómez (jgomez@unibe.edu.ec)
Facultad de Hotelería y Turismo
Universidad Iberoamericana del Ecuador
Quito, Ecuador

Resumen

Actualmente hay muchos requisitos de soluciones de telemetría configurables. El objetivo general de este trabajo es construir un multímetro de variables eléctricas con monitor gráfico en tiempo real y en red (usando una página web) usando Arduino y Raspberry PI. El método y el procedimiento: Linux, Apache, programado con C^{++}, Python, aproximaciones con series de Taylor, anotaciones con residuos de LaGrange Cauchy. Se trazaron 7 ciclos de onda instantáneos de voltaje, corriente, potencia aparente y voltaje teórico sin problemas. Se tomaron medidas de voltaje, corriente y potencia aparente, que coincidieron con los valores de las cargas, en casos de carga resistiva pura y el otro tipo de carga fue una lámpara fluorescente. La comparación de la tensión medida con la teórica, que coincidió en el período, más la fase, no se sincronizó. La función resultante puede tener una función más precisa si tiene una gran cantidad de términos de Taylor. La clase de medidas indirectas y directas fue 1.5.

Palabras y frases clave: Multímetro, graficador, voltaje, corriente, potencia, Arduino; Raspberry.

Abstract

Recibido 30/07/2018. Revisado 04/08/2018. Aceptado 04/09/2018.
MSC (2010): Primary 68U20; Secondary 68N99.
Autor de correspondencia: Jesús Gómez

Currently there are many requirements for configurable telemetry solutions. The general objective of this work is to build a multimeter of electrical variables with graphic monitor in real time and in network (using a web page) using Arduino and Raspberry PI. The method and procedure: Linux, Apache, programmed with C^{++}, Python, approximations with Taylor series, annotations with Lagrange Cauchy residues. 7 instantaneous wave cycles of voltage, current, apparent power and theoretical voltage were drawn without problems. Measures of voltage, current and apparent power were taken, which coincided with the values of the loads, in cases of pure resistive load and the other type of load was a fluorescent lamp. The comparison of the voltage measured with the teorical, which coincided in the period, plus the phase, was not synchronized. The resulting function can have a more precise function if it has a large number of Taylor terms. The class of indirect and direct measures was 1.5.

Key words and phrases: Multimeter, plotter, voltage, current, power, Arduino, Raspberry.

1 Introducción

En la actualidad, existen nuevos desafíos gerenciales y requerimientos de data en tiempo real (Telemetría o lo más cercano a ella), debido al crecimiento del internet de las cosas (IoT). Además, se incorpora el uso de tecnologías libres para tener una opción libre de los costos de patentes.

Equipos como osciloscopios, vatímetros, de buena precisión y clase baja son sumamente costosos, y sus servicios de mantenimientos también. Por tales motivos, se busca una nueva opción que resuelva estas problemáticas.

Este trabajo tuvo como objetivo general construir un medidor de variables eléctricas (voltaje y potencia aparente) con monitor graficador en tiempo real y por red (usando una página web) empleando Arduino y Raspberry PI.

2 Marco Teórico

Para el desarrollo de la investigación se utilizaron las fórmulas matemáticas descritas a continuación:

- Voltaje alterno, según Alexander en [1].

$$V(t) = vo \cdot \sin(wt + \phi)$$

- Voltaje Eficaz (RMS), ver [1].

$$Vrms(t) = \sqrt{\frac{\int_{t_0}^{t_0} V^2(t)dt}{T}}$$

- Aproximación de Taylor del Seno, según Rivera en [10]

$$\sin(x) = \sum_{i=0}^{\infty} (-1)^i \frac{x^{2i+1}}{(2i+1)!} \qquad \forall\, x \in \mathbb{R}$$

- Potencia aparente, ver [1]

$$S(t) = V(t)I(t)$$

- Velocidad angular y frecuencia, según Valkerbug en [12]

$$w = \frac{2\pi}{T} = 2\pi f$$

3 Método y procedimiento

El procedimiento para calcular los datos se realizó a través de la elaboración de un programa de Python con las características necesarias para ser una herramienta aceptable y funcional. Este se crea resolviendo las ecuaciones que se requieren para llegar al resultado. Para ensamblar el circuito eléctrico se realizaron los siguientes pasos:

1. Armar en el Arduino los medidores de voltaje (para voltaje hay que conectar un transformador para poder aumentar el rango de voltaje) y corriente.

2. Conectar ya ensamblado por USB al Raspberry pi, como se indica en Ray en [9].

3. En el Raspberry se realiza un programa ó "script" en Python (es lo más rápido), éste Importa la librería grafica "Gnuplot" y se adjuntó el código Python (Ver Programa 2).

4. Analizar el código, realizando:

   - Leer del puerto USB las mediciones de voltaje y corriente del Arduino.
   - Llenar una lista con estos datos en Python.
   - Tomar los datos y calcular la potencia aparente y el error (opcional).
   - Tomar como argumento estos datos para la herramienta graficadora "Gnuplot".
   - Configurar cada gráfica, ejes, títulos y escalas.
   - Exportar en fichero jpg o png cada grafica en cada 12 segundos.

5. Al mismo tiempo el servidor web (servicio apache de linux) tiene en la página principal un código que actualiza la página donde se muestra gráficas cada 15 segundos.

4 Resultados

4.1 Adquisición de datos

Se inició trabajando con el muestreo de 300 datos por segundo de voltaje, corriente y potencia con respecto al tiempo, pero para no sobrecargar el Arduino se bajó el muestreo a 150 muestras por segundo, graficando Voltaje Vs. Tiempo (ver Figura 1), Corriente (ver Figura 2), Potencia Aparente Vs. Tiempo (ver Figura 3).

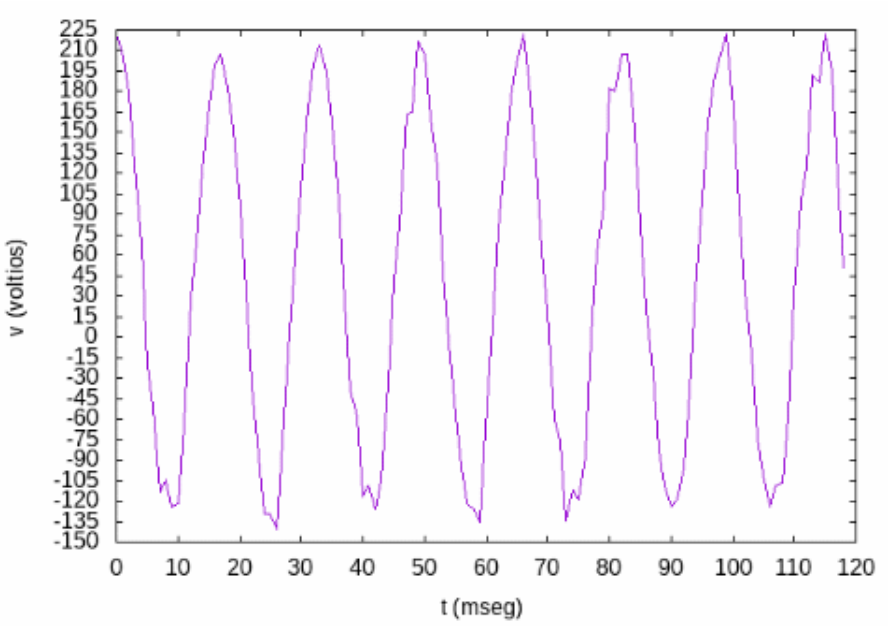

Figura 1: *Voltaje Vs. Tiempo*

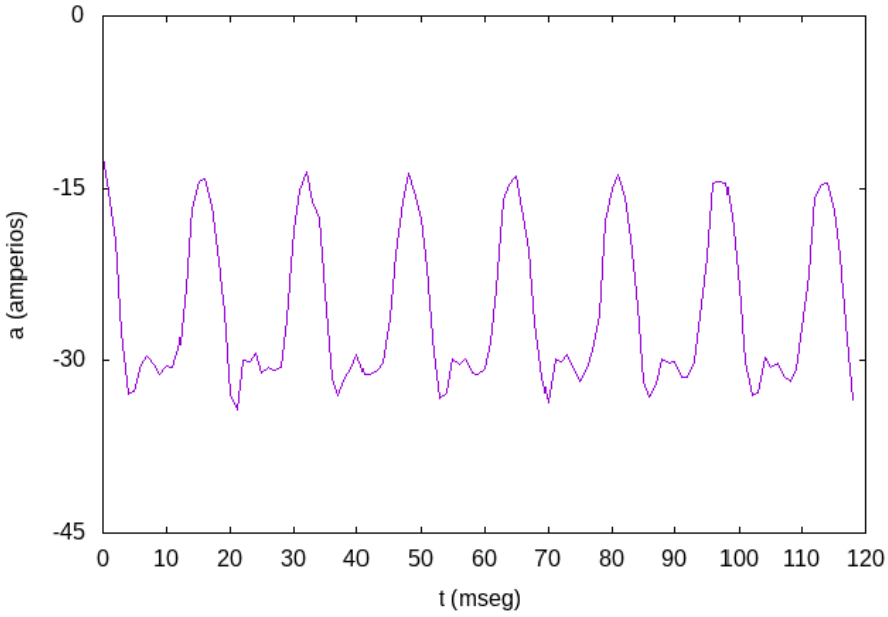

Figura 2: *Corriente Vs. Tiempo*

En la Figura 1 se presenta la gráfica que identifica la resolución del sensor de voltaje= 0,00488 V, según especificaciones de su Datasheet en Dealextreme (ver [14]), pero como se le agregó un transformador de relación 19,17; queda, la nueva resolución= 0,0935 V, es decir, cualquier valor de la gráfica tiene un error (Error= $\pm 0,0935\,V$).

En la gráfica presente en la Figura 2 se encontró el error total del sensor de corriente es= $1,5\,\%A$, según especificaciones de su Datasheet en Allegro (ver [13]), entonces la gráfica tiene un error (Error= $\pm 1,5\,\%$). Es decir, si toma un valor de la gráfica de $20\,A$. El valor real= $20 \pm 20 \cdot 0,015 = (20 \pm 0,3)\,A$.

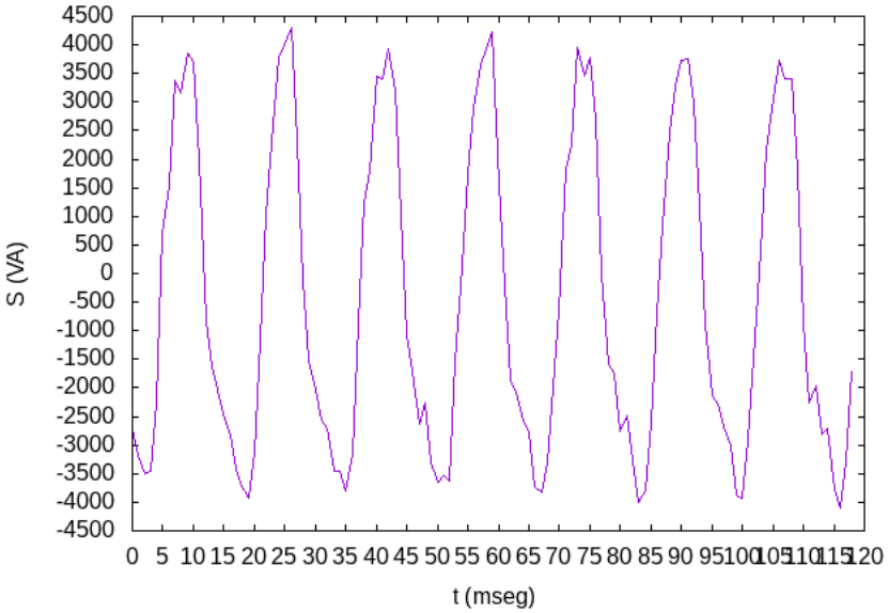

Figura 3: *Potenia aparente Vs. Tiempo*

La carga usada en la gráfica de la Figura 3 fue un bombillo fluorescente. La medición de esta magnitud es indirecta, $S = V \cdot A$, el error de S, según Helfrick en [5].

$$\frac{\Delta S}{S} = \sqrt{\left(\frac{\Delta V}{V}\right)^2 + \left(\frac{\Delta I}{I}\right)^2}$$

Con los datos de $V = 105\,v$; $I = -35\,A$; $\Delta V = 0,0935$; $\Delta I = 1,5\,\%\,I$; $S = 3515,5\,W$. Sustituyendo $\frac{\Delta S}{S} = 0,0015$; $\Delta s = 0,0015 \cdot S = 52,86\,w$; El valor real de $S = (3515,5 \pm 52,86)\,w$, porcentualmente este error= $1,504\,\%$.

A continuación se presenta la Figura 4, una captura de pantalla del monitor web del graficador. Como se puede observar, una vez graficado se obtuvo el resultado esperado, aunque se encontró inconveniente con la gráfica teórica de la aproximación de Taylor. (ver Figura 5).

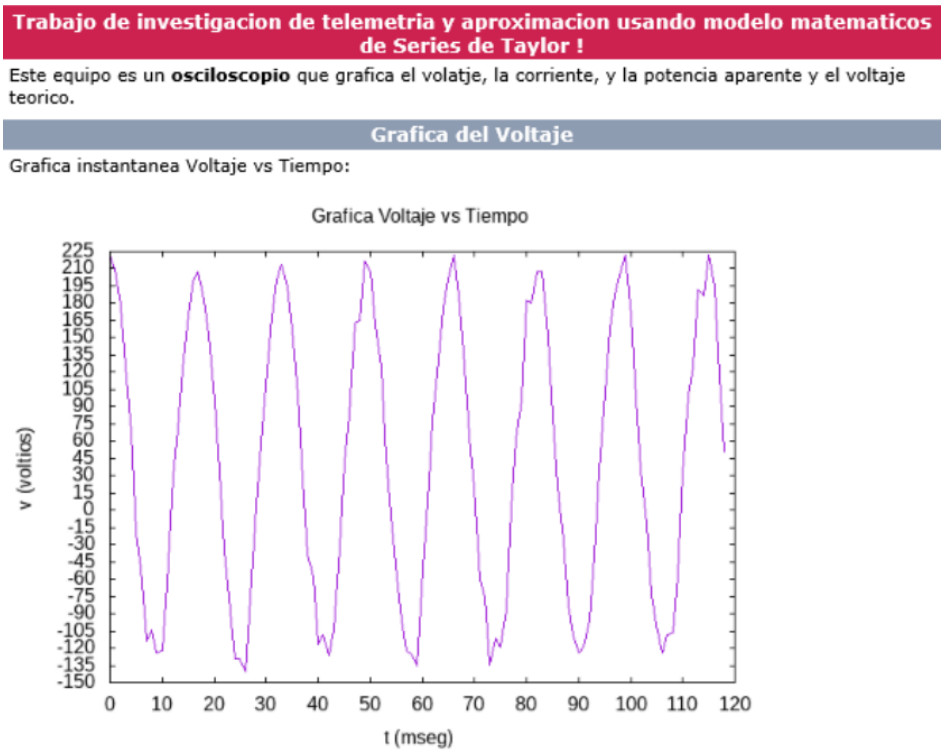

Figura 4: *Pantalla de la página web del monitor de red*

5 Diseño de un medidor y graficador de voltaje, corriente y potencia aparente.

Para la programación se usó lineamientos de lectura de puerto serie de ejemplos de Banzi (ver [3]), cofundador del proyecto Arduino. A continuación se presenta el código de medidor en Arduino mega. (Programa 1).

5.1 Programa 1.

El lenguaje de este programa en C^{++} para Arduino

```
#define␣muestras␣300␣//␣200␣muestras␣equivalen␣a
Int␣voltaje[muestras];␣//␣10␣ciclos␣completos␣de␣la␣se\~nal.
Int␣intensidad[muestras];
Unsigned␣long␣tiempo[muestras];␣//␣se␣utiliza␣el␣mismo␣formato␣que␣millis().
Unsigned␣int␣voltaje_max;Int␣voltios[muestras];␣Int␣inten[muestras];
Float␣sumaaparente=0;␣Float␣voltajee=0;␣Float␣rms_corriente=0;
Int␣aparente[muestras];Float␣amperios[muestras];
```

```
Float␣p=0;␣Long␣suma=0;␣Float␣sumaa=0;␣Float␣s=0;Float␣fp=0;Float␣phi=0;
Float␣reactiva=0;
Void␣setup()␣{
//␣inicialización␣del␣puerto␣serie.
Serial.begin(230400);
}
Void␣loop()␣{
Suma=0;Sumaa=0;Sumaaparente=0;
Muestreo();Calculos();Envio();Delay(16000);
Reset␣();
}
Void␣reset(){
For(int␣i=0;␣i<muestras;␣i++){
Voltios[i]=0;Amperios[i]=0;Aparente[i]=0;Sumaaparente=0;P=0;Suma=0;Sumaa=0;
}
}
Void␣calculos(){
For(int␣i=0;␣i<muestras;␣i++){
Voltio[i]=(voltios[i]-107)*2.44;␣Amperios[i]=(inten[i]-510)*0.07;
Aparente[i]=voltios[i]*amperios[i];␣Sumaaparente=sumaaparente+aparente[i];
P=sumaparente/muestras;
Suma=suma+pow(voltios[i],2);Sumaa=sumaa+pow(amperios[i],2);
}
Voltajee=sqrt((suma/muestras));Rms_corriente=sqrt((sumaa/muestras));
S=voltajee*rms_corriente;Fp=p/s;Phi=acos(fp);Reactiva=s*sin(phi);
}Void␣muestreo(){
Unsigned␣long␣tiempo_objetivo;Unsigned␣long␣cuenta;
For(int␣i=0;␣i<muestra;␣i++){
Tiempo_objetivo␣=␣millis()+1;␣//sumando␣un␣milisegundo
Voltios[i]␣=␣analogread(a0)*0.85;␣//se␣muestrea␣a␣1␣khz.
Inten[i]␣=␣analogreads(a_1);Tiempo[i]␣=␣millis();Cuenta␣=␣millis();
While␣(cuenta␣<␣tiempo_objetivo){
Cuenta␣=␣millis();
}}}
Void␣envio(){
Serial.print(rms_corriente);␣Serial.print(",");Serial.print(voltaje);
Serial.print(",");Serial.print(p);Serial.print(",");Serial.print(fp);
Serial.print(",");Serial.print(reactiva);Serial.print(",");Serial.println(s);
For(int␣i=0;␣i<muestras;␣i++){
Serial.print␣(voltios[i]);Serial.print(",");}
For(int␣i=0;␣i<muestras;␣i++){
Serial.print␣(amperio[i]);Serial.print(",");}
For(int␣i=0;␣i<muestras;␣i++){
Serial.print␣(aparente[i]);Serial.print(",");}
For(int␣i=0;␣i<muestras;␣i++){
Serial.print␣(i);Serial.print(",");}
}
```

Así mismo, se realizó el Código graficador (Programa 2) en Raspberry PI 3 plus, en lenguaje Python. Utilizando rutinas de ejemplos y referencias de Pilgrim presentes en [7] y para ajustar los comandos de Gnuplot se usaron ejemplos y referencias de Jarnet expuestas en [6].

5.2 Programa 2.

El lenguaje de este programa en Python 3

```
#␣-*-␣coding:␣utf-8␣-*-
Import␣httplib2␣#␣librería␣para␣clientes␣http␣(si␣no␣la␣tienes␣tienes␣que
bajartela␣e␣instalarla␣antes)
Import␣serial␣#␣librería␣del␣puerto␣serie␣para␣Arduino␣(si␣no␣la␣tienes,
tienes␣que␣bajartela␣e␣instalarla␣antes)
Import␣timeImport␣gnuplot
Gplot␣=␣gnuplot.gnuplot(debug␣=␣1)
G2␣=␣gnuplot.gnuplot(debug␣=␣1)
G3␣=␣gnuplot.gnuplot(debug␣=␣1)
Gplot.title("grafica␣voltaje␣vs␣tiempo")
G2.title("grafica␣corriente␣vs␣tiempo")
G3.title("grafica␣potencia␣aparente␣(s)␣vs␣tiempo")
Gplot.xlabel("t␣(mseg)")
G2.xlabel("t␣(mseg)")
G3.xlabel("t␣(mseg)")
Gplot.ylabel("v␣(voltios)")
G2.ylabel("a␣(amperio)")
G3.ylabel("s␣(va)")
#␣set␣style␣of␣plot.␣We␣want␣lines
GplotG1("set␣style␣data␣lines")␣#pone␣el␣grafico␣de␣tipo␣lineas␣;gh2018,
gherrera2k1@gmail.com
G2("set␣style␣data␣lines")␣#pone␣el␣grafico␣de␣tipo␣lineas␣;gh2018,
gherrera2k1@gmail.com
G3("set␣style␣data␣lines")␣#pone␣el␣grafico␣de␣tipo␣lineas␣;gh2018,
gherrera2k1@gmail.com
#␣gplot("set␣grid")␣#␣pone␣la␣cuadricula␣;gh2018,␣gherrera2k1@gmail.com
Gplot("set␣xtic␣10")
G2("set␣xtic␣10")
G3("set␣xtic␣5")
Gplot("set␣ytic␣15")
G2("set␣ytic␣15")
G3("set␣ytic␣500")
Ser␣=␣serial.serial('/dev/ttyacm0',␣230400)#abrimos␣puerto␣serie␣para␣ardu␣mega
Ser.readline()␣#␣esperamos␣a␣que␣el␣Arduino␣envíe␣una␣línea␣para␣evitar␣leer
luego␣una␣corrupta.
Conn␣=␣httplib2.http()␣#␣creamos␣la␣conexión␣http.
While␣true:
c␣=␣0
data␣=␣[]
```

```
data2 = []
data3 = []
while (c < 1):datostring = ser.readline() # leemos una nueva línea enviada por
el Arduino
datos = str(datostring).split(",") # separamos los datos recibidos
rms_corriente=(datos[0]) # guardamos el valor de voltaje de la luz
voltajee= (datos[1])
p= (datos[2])
fp= (datos[3])
reactiva= (datos[4])
s= (datos[5])
c= c + 1
# data.append(s) #
for gh in range(1,120):
data.append(datos[gh])
for gh in range(301,420):
data2.append(datos[gh])
for gh in range(601,720):
data3.append(datos[gh])
#time.sleep(31)
#data = [datos[1], datos[0], 1.0,0.73, datos[1], -0.7] # the data to plot
gplot("set terminal svg")
g2("set terminal svg")
g3("set terminal svg")
gplot.hardcopy (filename='/var/www/html/imagenes/grafica1.png', terminal='png')
# write
g2a.hardcopy (filename='/var/www/html/imagenes/grafica2.png', terminal='png')
# write
g3.hardcopy (filename='/var/www/html/imagenes/grafica3.png', terminal='png')
# write
gplot.plot(data) # plot the data
g2.plot(data2) # plot the data
g3.plot(data3) # plot the data
time.sleep(15)
```

6 Diseño de un graficador de voltaje teórico usando aproximación de Taylor.

6.1 Programa 3.

El lenguaje de este programa son comandos internos de Gnuplot

```
f(x)=x-(x**3)/gamma(4)+(x**5)/gamma(6)-(x**7)/gamma(8)+(x**9)/gamma(10)-
(x**11)/gamma(12)+(x**13)/gamma(14)-(x**15)/gamma(16)
plot[-5:5][-2:2]f(x)
```

Como se puede apreciar, para el polinomio de Taylor, se usa la función Gamma como susti-

tución del factorial, porque es la opción disponible para "Gnuplot", la relación entre el factorial y Gamma, está descrita por Casteleiro en [4], dada por:

$$\Gamma(n) = (n-1)!$$

Este código presentado genera la Figura 5, encontrada a continuación:

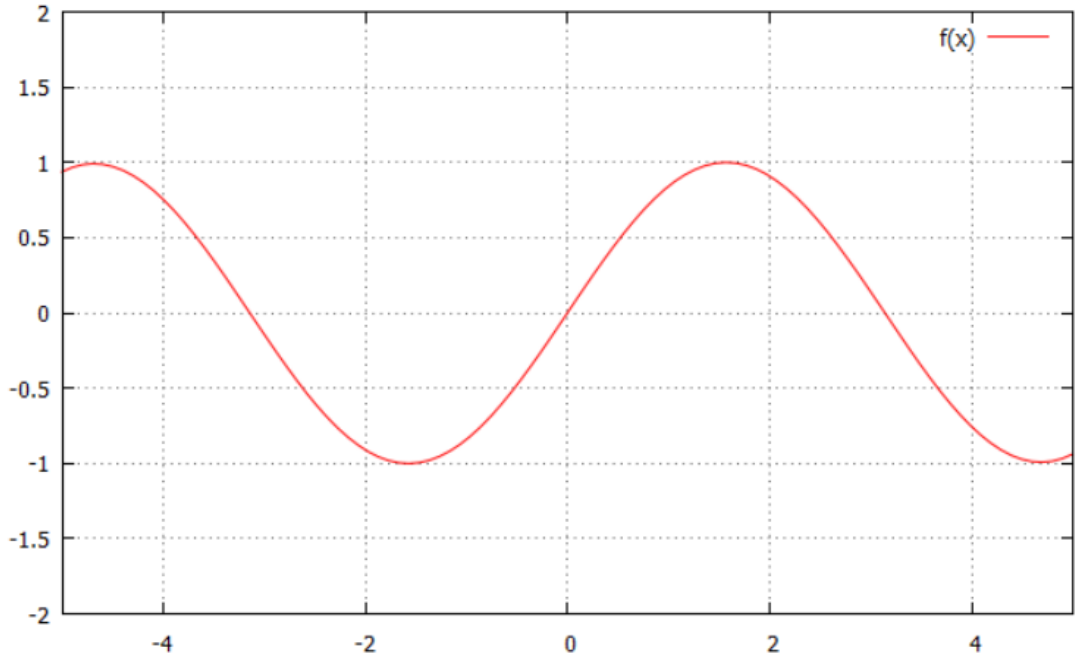

Figura 5: *Gráficas de aproximación de Taylor del Voltaje*

Con este análisis se obtuvo la gráfica voltaje teórico, $V_{teórico}(x)$, la cual se identifica simplemente $V(t)$ de ahora en adelante y, para el tratamiento práctico del error simplemente, se restó el valor de la función $sin(x)$, menos el valor del polinomio de Taylor, $P_{Taylor}(sinx)$. Así, según Apostol en [2]

$$Error(x) = \sin(x) - P_{Taylor}(\sin(x))$$

Pero, mejor aún es usar, el término complementario de una serie de Taylor, según la ecuación Lagrange y Cauchy del residuo presentada en [2]. Siendo operativamente imposible trabajar con infinitos términos, se hace

$$V(t) = V_o \sin(wt+\phi) = V_o \sin(x) = V_o \left(x - \sum_{i=1}^{\infty} (-1)^i \frac{x^{2i+1}}{(2i+1)!} \right)$$

Luego, para un Taylor de grado 15, queda $V(t) = V_o \sin(wt+\phi)$, es decir,

$$V_o \sin(x) = V_o \left(x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \frac{x^{11}}{11!} + \frac{x^{13}}{13!} - \frac{x^{15}}{15!} + \sum_{i=8}^{\infty} (-1)^i \frac{x^{2i+1}}{(2i+1)!} \right),$$

es decir,

$$V(t) = V_o \sin(wt + \phi) = V_o \sin(x) = V_o \left(x - \sum_{i=1}^{7} (-1)^i \frac{x^{2i+1}}{(2i+1)!} + \sum_{i=8}^{\infty} (-1)^i \frac{x^{2i+1}}{(2i+1)!} \right)$$

Ahora, se considera el primer término de la segunda serie y despreciamos los siguientes infinitésimos de esa segunda serie, usando la cota M, se resuelve

$$V(t) = V_o \sin(wt + \phi) = V_o \sin(x) = V_o \left(x - \sum_{i=1}^{7} (-1)^i \frac{x^{2i+1}}{(2i+1)!} + M \sum_{i=8} (-1)^i \frac{x^{2i+1}}{(2i+1)!} \right)$$

El término acotado se denomina $Error(x)$ o término complementario de un polinomio de Taylor, así

$$Error(x) = M \sum_{i=8} (-1)^i \frac{x^{2i+1}}{(2i+1)!}$$

En este caso el término complementario de polinomio de grado 15 (T_{15}), con $x = -5$ y M la cota máxima del $\sin(x) = 1$

$$E(-5) = \left| \frac{x^{17}}{(17)!} M \right|$$

Evaluando $Error(x)$ en $x = -5$ se tiene queda

$$Error(x) = -0,002144972,$$

si el error porcentual o clase es igual a $\frac{0,002144972}{\sin(-5)}$. Clase= $0,22$ (Con un Taylor de un grado menor la clase es mayor a 2, y el patrón de medición no puede ser menos exacto).

6.2 Ensamblaje del medidor y graficador de voltaje, corriente y potencia aparente.

Adicionalmente se realizó el diagrama de conexión eléctrica del prototipo del medidor de variables presentado en Figura 6.

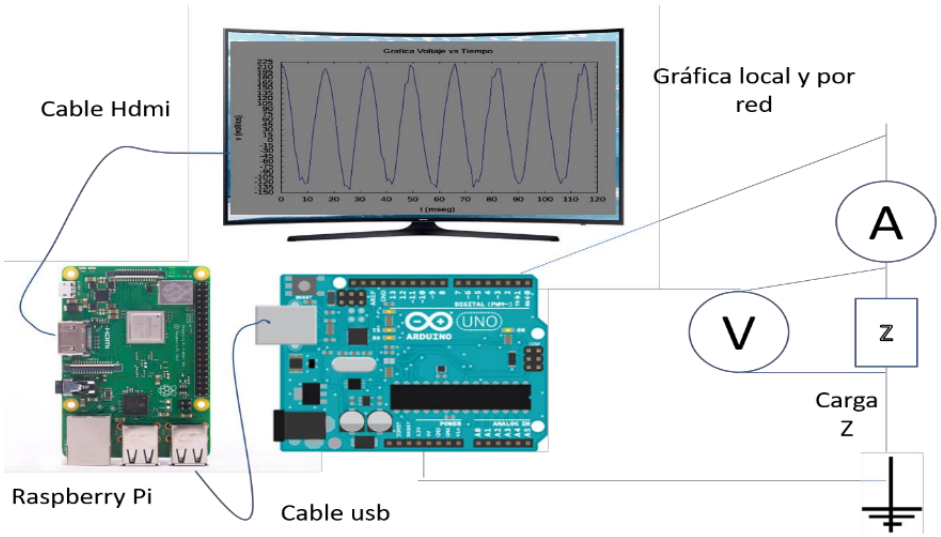

Figura 6: *Circuito eléctrico prototipo*

Así mismo, se presenta la conexión usada para aumentar el alcance del sensor de $25V$ a $357V$, según Trujillo en [11], la cual traslada la señal $9V$, por tal motivo se agrega una batería de $9v$ para cancelar esa traslación, adicional se usa un transformador de medidas para alimentar la bobina voltimétrica, indica Ras en [8]. La cual se observa en la Figura 7.

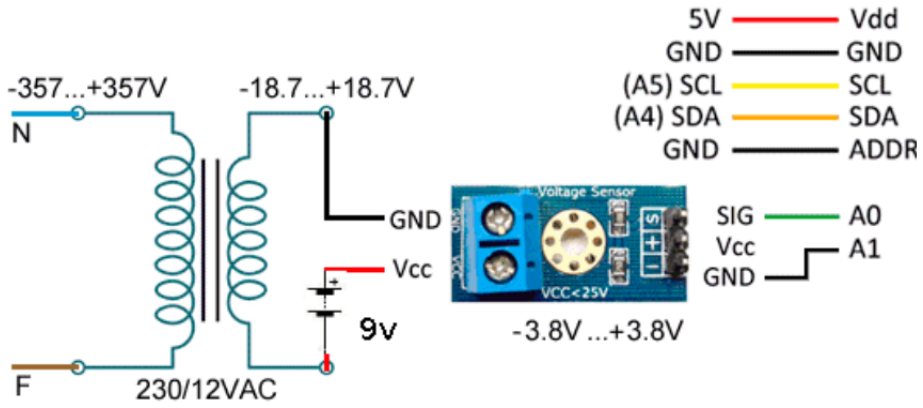

Figura 7: *Circuito y conexión eléctrica del sensor de voltaje*

Figura 8: *Circuito eléctrico ensamblado*

7 Conclusiones

Una vez realizada y culminado el proceso de graficación, se puede decir que se concluye que:

- Se pudo graficar 7 ciclos o períodos de onda del voltaje instantáneo, corriente instantánea, potencia aparente instantánea y voltaje teórico instantáneo sin inconvenientes.
- Se graficó la potencia aparente, la cual coincidió con los valores de las cargas, en los casos de carga resistiva pura y el otro tipo de carga fue un bombillo o foco fluorescente.
- Se observó remotamente desde el computador conectado a la red LAN las gráficas de los distintos parámetros eléctricos característicos.
- Comparando el voltaje medido con el teórico, se observa que coinciden en período, mas no en fase. La función resultante de voltaje teórico será una función más exacta en la medida que tenga un gran número de términos.
- Comparando el voltaje medido con el teórico, se observa que coinciden en periodo, mas no en fase. La función resultante de voltaje teórico será una función más exacta en la medida que tenga un gran número de términos.
- Referente a la potencia aparente S, el error $= 1,504\,\%$, prácticamente, la clase$= 1,5$; lo cual es muy bueno para ser una medición indirecta.
- Se necesitó un Polinomio de Taylor de grado 15, para que la clase del voltaje teórico fuera menor que la clase de las demás mediciones.

8 Recomendaciones

Para ver por internet las gráficas hay que hacer un reenvió (forwarding) del tráfico de red del puerto 80 del Raspberry en el router conectado al ISP (Internet Server Provider) de la red.

Agregar botones en el monitor de página web, que permitan cambiar el rango del eje X, del eje Y, y poder moverse por la gráfica.

Aumentar los términos de Taylor, para tener mayor precisión, como tal falta sincronizar la gráfica de voltaje medido con el voltaje teórico, por algún método.

Referencias

[1] Alexander, C. y Sadiku M. *Fundamentos de circuitos eléctricos.* 3^a Edición . México: McGraw-Hill, 370-468. 2006.

[2] Apostol, T. Calculus. *Cálculo con funciones de una variable, con una introducción al álgebra lineal.* 2^a Edición. Barcelona. España: Editorial Reverte, 341-368. 2001.

[3] Banzi, M. y Shiloh, M. *Introducción a Arduino.* California, USA. Anaya Editores, 68-69. 2016.

[4] Casteleiro, J. y Paniagua R. *Cálculo integral.* Madrid, España. ESIC Editorial, 473-478. 2002.

[5] Helfrick, A. y Cooper W. *Instrumentación electrónica moderna y técnicas de medición.* Barcelona, México. Prentice-Hall Hispoamericana, 1-8. 1991

[6] Jarnet , P. *GnuPlot in Action.* New York, USA. Manning Publications, 7-299. 2010.

[7] Pilgrim, M. *Dive into Python 3.* New York, USA. Apress Press, 1-344. 2009.

[8] Ras, E. *Transformadores de potencia, de medida y de protección.* Barcelona, España. Marcombo Boixareu Editores, 7^a Edición, 179-180. 1998.

[9] Ray, R. *Raspberry Pi: Guía paso a paso para dominar el Hardware y Software de Raspberry PI 3.* New York, USA. CreateSpace Independent Publishing Platform, 2018s, 7-299. 2010.

[10] Rivera, A. *Cálculo integral. Sucesiones y series de funciones.* México, México, Editorial Patria, 1^a Edición, 228-232. 1998.

[11] Trujillo, E. *Analizador de consumo de potencia eléctrica con arduino.* Madrid, España. Universidad Carlos III de Madrid, 1^a Edición, 24-25. 2015. Documento on line disponible en: `https://e-archivo.uc3m.es/handle/10016/23628`.

[12] Valkerbug, V. *Análisis de redes.* Editorial Limusa. ISBN: 9789681801786, 3^a Edición, 35-36. 1999.

[13] ACS712 Datasheet. *AllDatasheet.es.* Allegro Microsystems, n.d. Documento on line disponible en: `http://www.alldatasheet.es/datasheet-pdf/pdf/168326/ALLE-GRO/ACS712.html`.

[14] B25 Modulo de tablero voltaje del Sensor para Arduino – azul. *DealExtreme.com*. N.p., n.d. Documento on line disponible en: http://www.dx.com/es/p/b25-voltage-sensor-board-module-for-arduino-blue-379810?tc=EUR&gclid=CjwKEAjw3uWuBRD_s-3a8-_h6j0SJA-CqgtH3p0Yo6N5hhX5qfkR8gDnND2X14jsGRk6rmQvFHBBuRoCiTbw_wcB#.VdodYZR_v1s