UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

PROYECTO

RASPBERRY PI

Ingeniería Civil Telemática

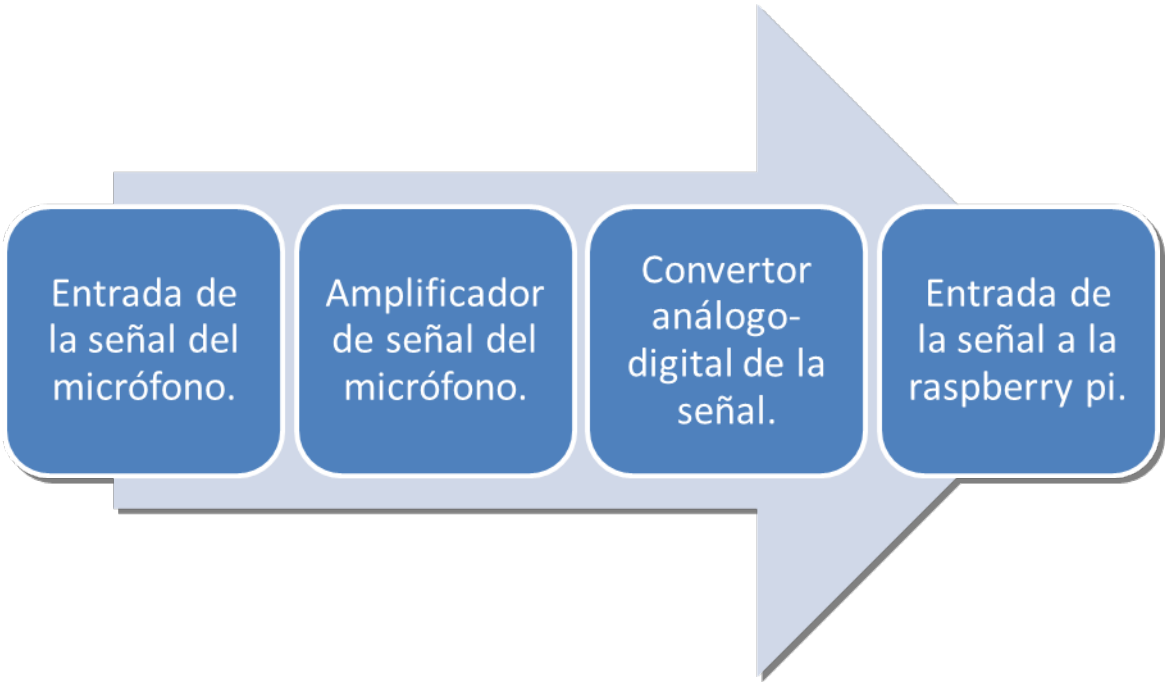
Felipe Escobar 201330048-0

Pablo Oñate 201330050-2

Valentina Yévenes 201330057-k

PROYECTO RASPBERRY PI

El proyecto consta de un registro de sonido y posterior medición de voltaje. Para esto, el proceso a seguir será el siguiente:



ENTRADA DE LA SEÑAL DEL MICRÓFONO.

En este caso utilizaremos un micrófono cuyo radio de funcionamiento va de 1.5 a 15 volts al cual, para que funcione, se le entregarán 12 volts usando una fuente de poder. Entregará una señal senoidal cuyos peaks serán muy bajos (200mV) para que la Raspberry Pi la lea correctamente.

AMPLIFICADOR DE SEÑAL

Se utilizará un amplificador UA741CN con el que ampliaremos la señal que entrega el micrófono para que sea leída correctamente por la Raspberry Pi y, por lo tanto, se obtenga un espacio muestral suficiente para lograr que se grafique correctamente.

CONVERSOR ANÁLOGO-DIGITAL.

La Raspberry Pi solo recibe señal digital, por lo tanto, hay que transformarla. Esto se realiza con un MCP3201, éste se encargará de convertir la señal análoga de voltaje que emite el micrófono en una digital con un valor binario.

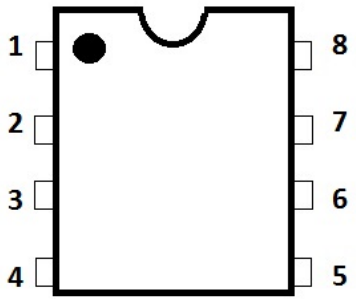
Con esto la Raspberry Pi recibirá una señal que podrá ser leída y posteriormente graficada.

LISTADO DE ELEMENTOS A UTILIZAR

1 Protoboard
Cables Para Conexión
1 Conversor Análogo Digital MCP3201
1 Amplificador UA741CN
Raspberry Pi con Librerías RPi.GPIO y WiringPi instaladas sobre Raspbian.
1 Micrófono Electret
Resistencias de diversos valores
2 Condensadores, uno de 10 [µf] y otro de 1 [µf]
1 Fuente de Poder Variable (o fija de 12.0[V])

Para la conexión se usa una Protoboard la que nos ayudará para conectar todo y enviar la información a la RPi.

Tanto el conversor como el amplificador son circuitos integrados y cuentan con un círculo en una esquina que es el indicador del pin número 1. Se cuentan hacia abajo y luego hacia arriba según el dibujo.



El micrófono será el encargado de registrar el sonido, las resistencias se utilizan para regular y disminuir la corriente que circula por el circuito y en ocasiones para evitar fugas indeseadas de energía.

Un condensador es un dispositivo capaz de almacenar energía en forma de campo eléctrico.

COMPONENTES UTILIZADOS

MCP3201

Fabricante: Microchip
Función: ADC (Conversor Análogo Digital) con Interfaz SPI y tecnología CMOS de bajo consumo
Resolución: 12 bits
Voltaje de Funcionamiento: 2.7 – 5.5 [V]

Es utilizado para transformar la Señal Analógica del Micrófono Electret, el cual entrega un Voltaje Peak de 300[mV] tratada posteriormente por un Amplificador Operacional. Entregar directamente una Señal Analógica al Raspberry Pi puede reiniciar o averiar el dispositivo por exceso de corriente. Se utiliza debido que su Voltaje de Funcionamiento del Raspberry Pi es de 3.3 o 5[V], lo cual simplifica su operación a tan solo conectarlo a la fuente provista de la Raspberry Pi a través de los puertos GPIO y sus 8 pines de conexión son una interfaz amigable de conectarse en comparación a otros ADC como por ejemplo el ADC0808 que tiene la UTFSM en su Pañol y es más adecuado para circuitos más complejos.

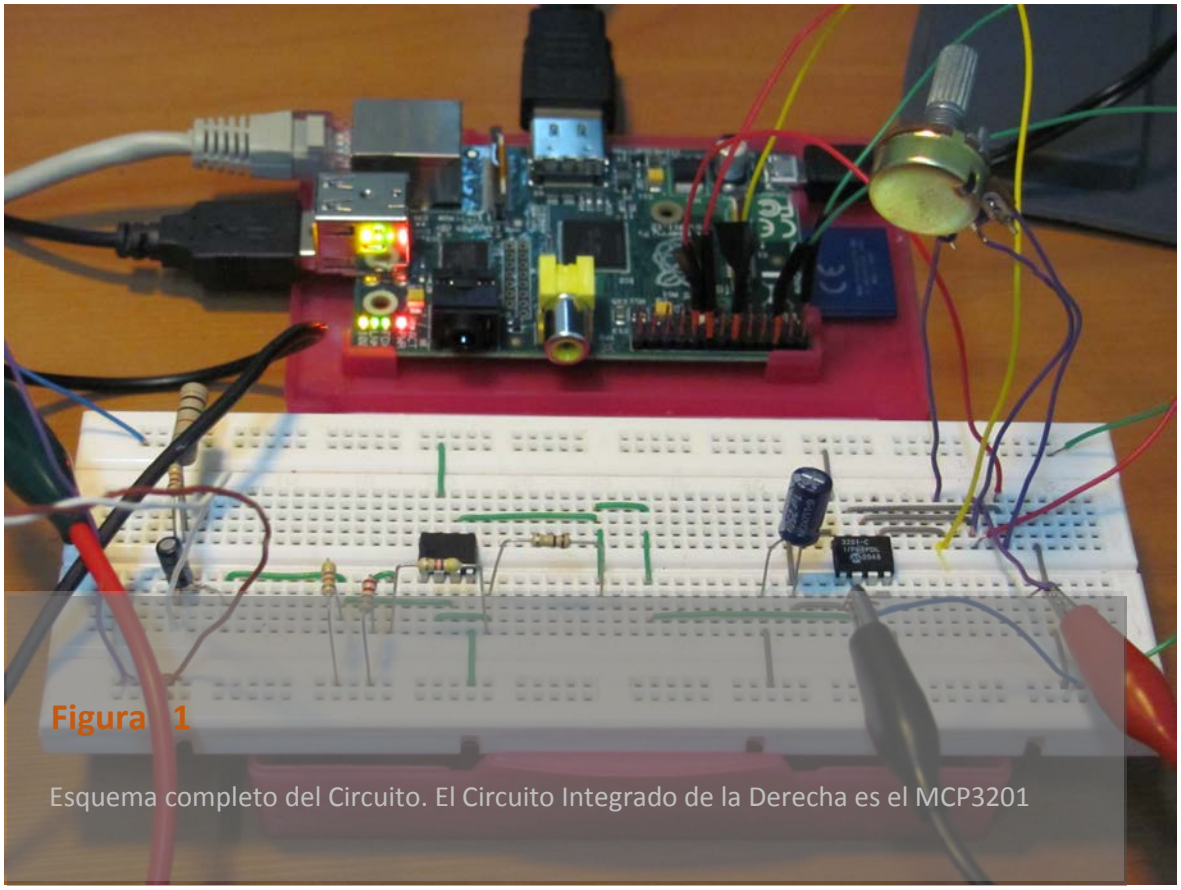


Figura 1

Esquema completo del Circuito. El Circuito Integrado de la Derecha es el MCP3201

UA741CN

Fabricante: SGS - Thompson Electronics
Función: Amplificador Operacional sin Compensación de Frecuencia
Voltaje de Funcionamiento utilizado: 12[V]

El micrófono Electret correctamente polarizado y conectado a una fuente de 15[V] genera un voltaje Peak-To-Peak de 250[mV]. Si la fuente es de 12[V] como la utilizada por nosotros genera un Voltaje Peak-to-Peak de 200[mV]. Si necesitamos una señal de salida de 3.29 [V] analógica lista para ser convertida por el MCP3201 a digital, se requiere que el Amplificador Operacional tenga una ganancia de 16.45

Para ello el Amplificador Operacional en modo No Inverso debe ser complementado con una resistencia de 5[kΩ] y 100[kΩ], tal como muestra la simulación realizada

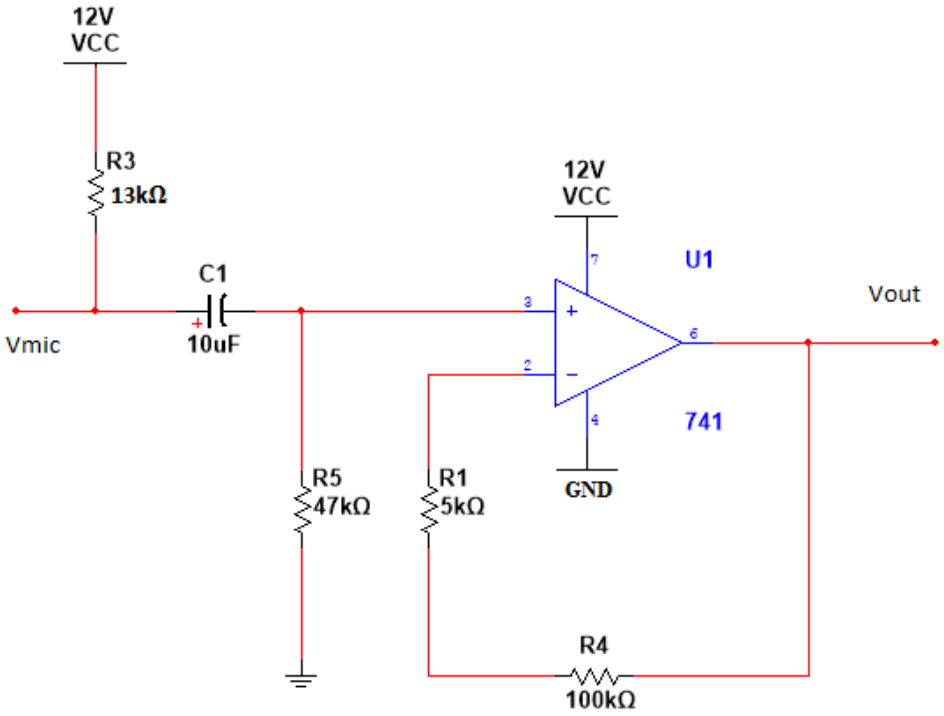


Figura 2: Simulación en Multisim de Conexión entre el Micrófono y el Amplificador UA741

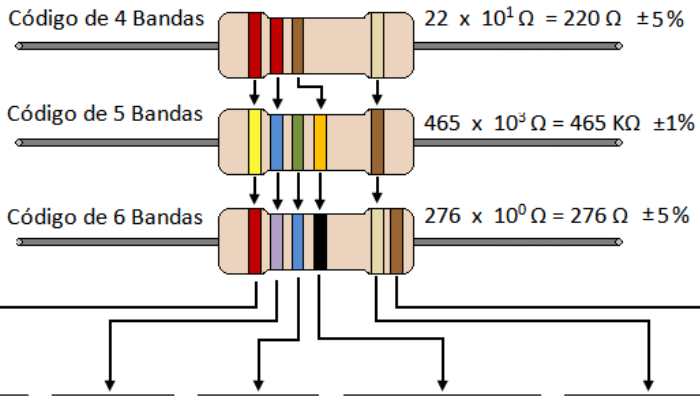
FUENTE DE PODER

Fue necesaria la instalación de una fuente de poder externa, ya que para el correcto funcionamiento de este tipo de micrófonos se requiere al menos 9[V] (precisamente por el Voltaje Peak que el micrófono entrega). Utilizamos una fuente de poder variable para poder ajustar sin mayores inconvenientes el circuito ante cualquier situación o mejora. Cabe destacar que la Raspberry Pi es limitada en este aspecto y por ello utilizamos la fuente externa, para así no sobreexigirle entrega de voltaje y eliminando las limitaciones prácticas al respecto de los resultados deseados.

Figura 3

Osciloscopio de 2 Canales junto a una Fuente de Poder Variable

CÓDIGO DE COLORES DE LAS RESISTENCIAS



	Banda 1	Banda 2	Banda 3	Multiplicador	Tolerancia	Coeficiente Temperatura
Negro	0	0	0	10^0		
Marrón	1	1	1	10^1	±1.00%	100 ppm/°C
Rojo	2	2	2	10^2	±2.00%	50 ppm/°C
Naranja	3	3	3	10^3		15 ppm/°C
Amarillo	4	4	4	10^4		
Verde	5	5	5	10^5	±0.50%	
Azul	6	6	6	10^6	±0.25%	10 ppm/°C
Violeta	7	7	7	10^7	±0.10%	5 ppm/°C
Gris	8	8	8	10^8	±0.05%	
Blanco	9	9	9	10^9		
Dorado				10^{-1}	±5.00%	
Plateado				10^{-2}	±10.00%	

CONEXIÓN DEL CIRCUITO AL RASPBERRY PI

La forma para conectar los circuitos integrados será según la siguiente tabla:

PIN en MCP3201	PIN en Raspberry Pi	Nombre	PIN en UA741CN	PIN en Raspberry Pi	Nombre
1	+3.3V	VREF	1	NO*	OFFSET NULL 1
2	NO*	IN+	2	NO*	Entrada Inversora
3	GND	IN-	3	NO*	Entrada No inversora
4	GND	Vss	4	NO*	Alimentación Negativa
5	GPIO18	CS	5	NO*	OFFSET NULL 2
6	GPIO23	DOUT	6	NO*	Salida
7	GPIO24	CLK	7	NO*	Alimentación Positiva
8	+3.3V	Vdd	8	NO*	NC

- Estos pines no van conectados directamente a la interfaz GPIO de la Raspberry Pi, van conectados a distintas partes de la circuitería utilizada, ver esquema.

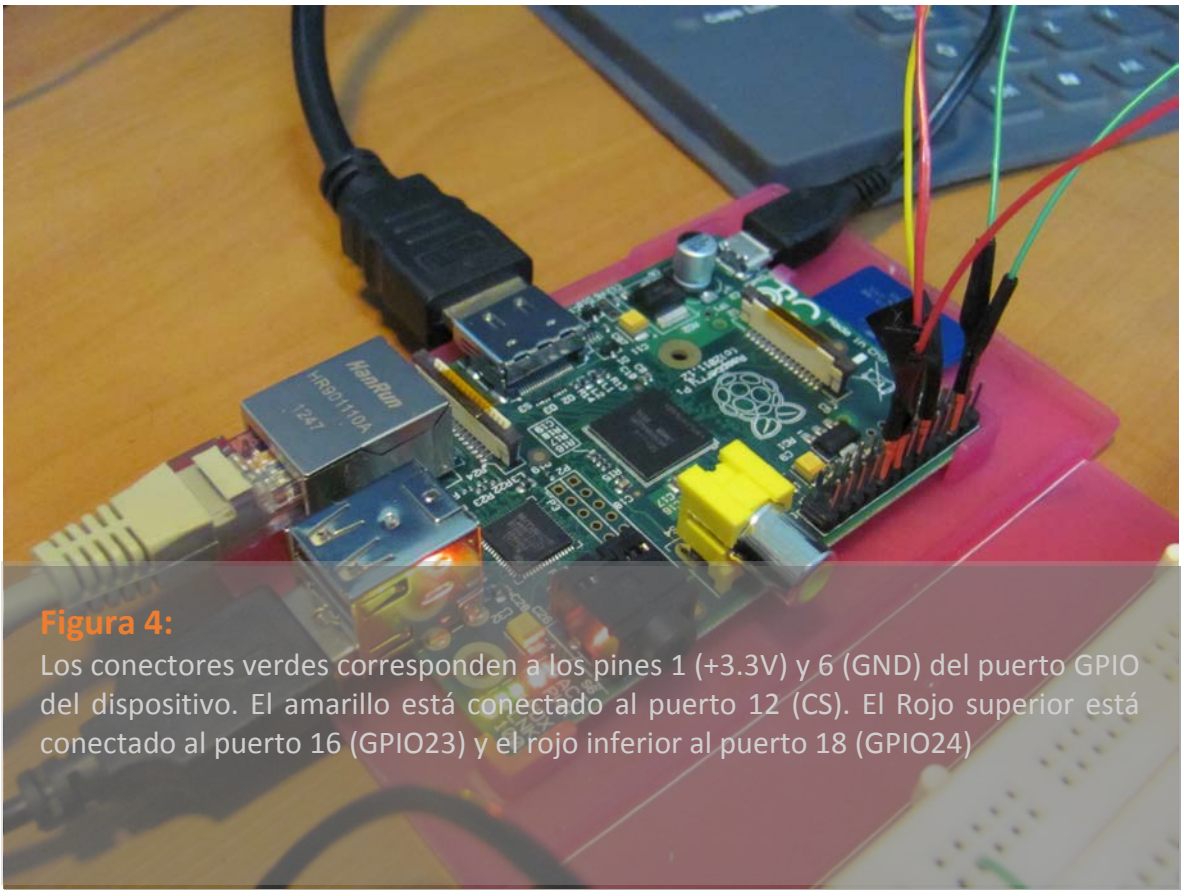


Figura 4:
Los conectores verdes corresponden a los pines 1 (+3.3V) y 6 (GND) del puerto GPIO del dispositivo. El amarillo está conectado al puerto 12 (CS). El Rojo superior está conectado al puerto 16 (GPIO23) y el rojo inferior al puerto 18 (GPIO24)

Los pines físicos son distintos a los puertos nombrados según la interfaz GPIO, y la distribución depende de la librería que se utilizará para hacer el programa. Por ejemplo, en el cuadro anterior dice que el pin número 5 corresponde al pin número 18 de la GPIO, que en realidad es el pin físico número 24.

RPi.GPIO	Wiring Pi Pins	GPIO Pins		Wiring Pi Pins	RPi.GPIO
—	—	1	2	—	—
0	8	3	4	—	—
1	9	5	6	—	GND
4	7	7	8	15	14
—	—	9	10	16	15
17	0	11	12	1	18
21	2	13	14	—	—
22	3	15	16	4	23
—	—	17	18	5	24
10	12	19	20	—	—
9	13	21	22	6	25
2	14	23	24	10	8
—	—	25	26	11	7

RPi.GPIO es la librería de Python para utilizar los GPIO de la RPi.
Para obtenerla visite: https://pypi.python.org/pypi/RPi.GPIO

Wiring Pi es la librería de C para utilizar los GPIO de la RPi.
Para obtenerla visite: http://wiringpi.com/download-and-install/

PROGRAMACIÓN DEL PROYECTO

En este Proyecto utilizamos la Programación de Python por la facilidad de ser implementada y la sencillez de entender las funciones. La Librería RPi.GPIO es desarrollada bajo Licencia MIT y es de Código Libre bajo los estándares CC3.

Para obtenerla puede visitar el sitio https://pypi.python.org/pypi/RPi.GPIO, o bien desde la terminal ejecutar el siguiente código en la Raspberry Pi:

```
sudo wget https://pypi.python.org/packages/source/R/RPi.GPIO/RPi.GPIO-
0.5.3a.tar.gz
tar zxf RPi.GPIO-0.5.3a.tar.gz
cd RPi.GPIO-0.5.3a
sudo python setup.py install
sudo reboot
```

Para verificar la correcta instalación se puede seguir el siguiente script y ejecutarlo

```
sudo python
>>> import RPi.GPIO as GPIO
>>> GPIO.RPI_REVISION
2
>>> GPIO.VERSION
'0.5.3a'
>>>
```

De igual manera instalamos las librerías scipy, numpy y matplotlib para trabajar con archivos, vectorización y gráficas. Para instalarlas solo basta ejecutar:

```
sudo apt-get install python-scipy python-numpy python-matplotlib
```

SCRIPT 1: GENERACIÓN DE DATOS Y ALMACENADO

```
from time import sleep
import RPi.GPIO as GPIO

# Abre el archivo y si no existe lo crea
archive=open('resultado.txt', 'w')

# A continuacion definimos tiempo
# Asignamos los nombres CS al GPIO18, DOUT al GPIO 23 y CLK al GPIO2
# Fijamos el Voltaje de Referencia de 3.29[V], entregado por la
raspberry
tiempo = 0
CS = 18
DOUT = 23
CLK = 24

Vref = 3.29
# Desactiva las Advertencias que la Librería RPi.GPIO entrega en
pantalla por ciertos eventos
GPIO.setwarnings(False)
# Fijamos por defeto el modo de reconocer los puertos GPIO al
utilizado por el Convenio Broadcom, es decir, a lo que fisicamente se
indica
GPIO.setmode(GPIO.BCM)
# Fijamos al GPIO18 como puerto de Salida (entrega datos del Raspberry
al Circuito, precisamente al ADC MCP3201
GPIO.setup(CS, GPIO.OUT)
# Fijamos al GPIO23 como puerto de entrada (Recibe y entrega al
Raspberry bits para que nosotros los interpretemos)
GPIO.setup(DOUT, GPIO.IN)
# Fijamos al GPIO24 como puerto de Salida, el Raspberry a través del
GPIO tiene reloj interno, necesario para hacer cambios de estado o
recibir nuevo bit
GPIO.setup(CLK, GPIO.OUT)

# Este loop infinito hace poner en modo BAJO(False) o ALTO(True) los
diferentes puertos para entregar y/o recibir bits
while (True):
        GPIO.output(CS, True)
        GPIO.output(CLK, True)
        GPIO.output(CS, False)
        binData = 0
        i1 = 14
# El ADC tiene resolución de 12 bits, por ende, debe leer al menos 12
bits (0 ó 1) para conformar una medida o lectura correcta aprovechando
su capacidad


# Cuando este ciclo finaliza se obtiene una lectura y se entrega a la
pantalla, pasando a hacer nuevas lecturas indefinidamente
        while (i1 >= 0):
                # Cierra el reloj del Raspberry para leer un pulso
```

```
                GPIO.output(CLK, False)
            # Se obtiene una lectura de 1 bit
                bitDOUT = GPIO.input(DOUT)
            # Se Abre el puerto CLK
                GPIO.output(CLK, True)
            # Si el bit obtenido es mucho menor a i1 entrega un
False y lo entrega al ciclo anterior para capturarlo como bit de la
cadena a entregar
                bitDOUT = bitDOUT << i1
                binData |= bitDOUT
            # descenso luego de terminar el proceso
                i1 -= 1
        # Se abre el puerto CS
        GPIO.output(CS, True)
        # Produce que los bits más significativos menores a 12 se
aproximen a cero
        binData &= 0xFFF
        # Realiza la conversión del binario a un valor voltaje válido
para mostrar y almacenar
        res = Vref * binData/4096.0
        # Se almacena la coordenada en el archivo especificado
previamente por el usuario, archivo del tipo CSV
        archive.write(str(Counter))
    archive.write(';')
    archive.write(str(res))
    archive.write('\n')
    # Imprime en pantalla el voltaje medido, suma 0.2 al contador de
tiempo y en 200 milisegundos realiza un nuevo loop
    print('Voltaje Medido = ' + str(res) + '[V]')
        Counter  += 0.2
    sleep(0.2)
```

SCRIPT 2: VECTORIZACIÓN Y GRAFICACIÓN DE DATOS

Utilizando las librerías matplotlib y numpy para graficar y obtener los datos desde el archivo CSV generado por nuestra aplicación, este script permite la apertura de una ventana gráfica donde se ve la gráfica obtenida a partir de nuestro sondeo.

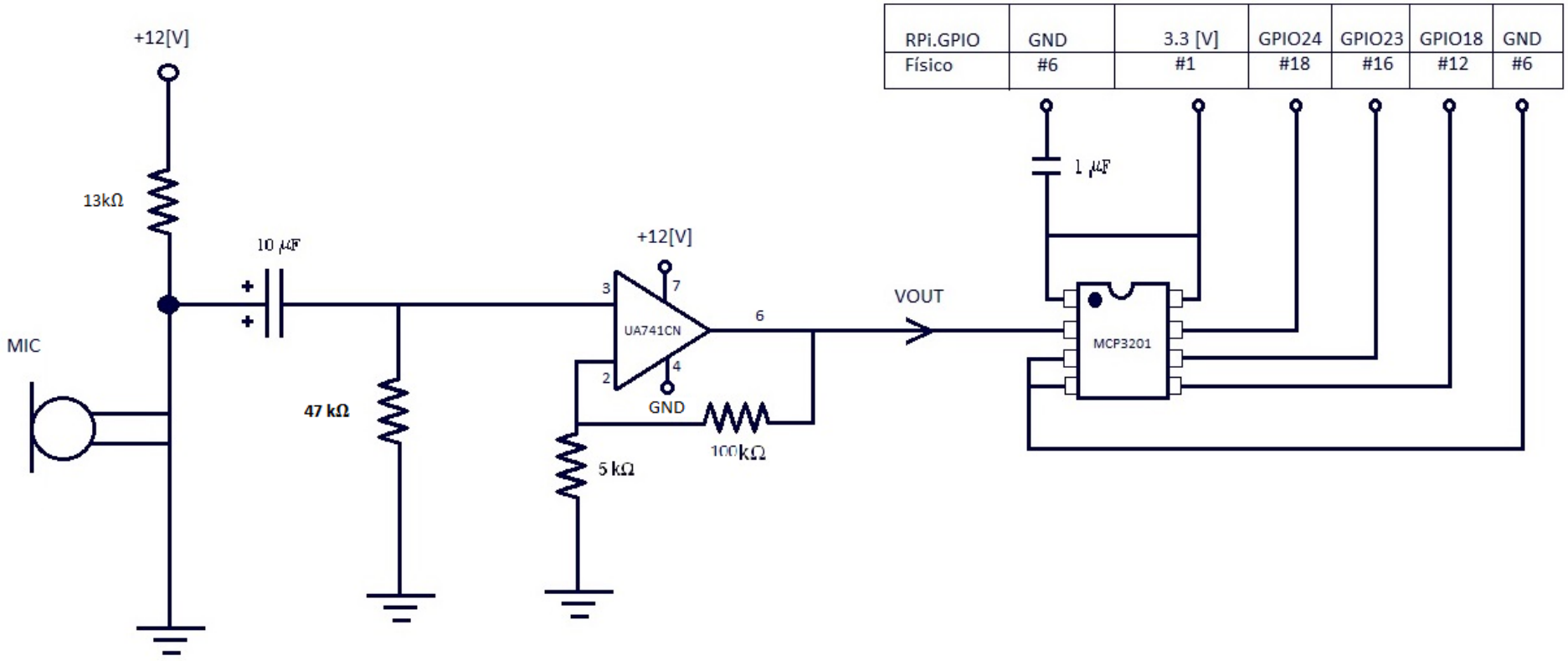
```
import numpy as np
import pylab as pl

arreglo = np.loadtxt('resultado.txt',delimiter=';')
tiempo = []
voltaje =[]
for i in range(len(arreglo)):
      tiempo.append(arreglo[i][0])
      voltaje.append(arreglo[i][1])
pl.plot(tiempo,voltaje)
pl.title("Voltaje a traves del tiempo")
pl.xlabel("Tiempo [s]")
pl.ylabel("Voltaje [V]")
pl.show()
```

BASH PARA EJECUTAR Y PRESENTAR EL PROYECTO

Usando los conocimientos sobre Linux vistos en clase, decidimos automatizar el proceso de nuestro Proyecto usando un fichero basado en bash, muy sencillo en donde el archivo CSV tiene por defecto como nombre resultado.txt:

```
#!/bin/bash
echo Ejecutando el Script en Python. Presionar Ctrl+C para saltar paso hasta finalizar.
cd /home/pi/prueba
sudo python script2.py
echo Abriendo los datos obtenidos...
nano resultado
echo Graficando...
sudo python listar.py
clear
echo Cierre la ventana gráfica para finalizar la presentación. Gracias
```

ESQUEMA Y CIRCUITO DEL SISTEMA UTILIZADO EN EL PROYECTO



IMÁGENES

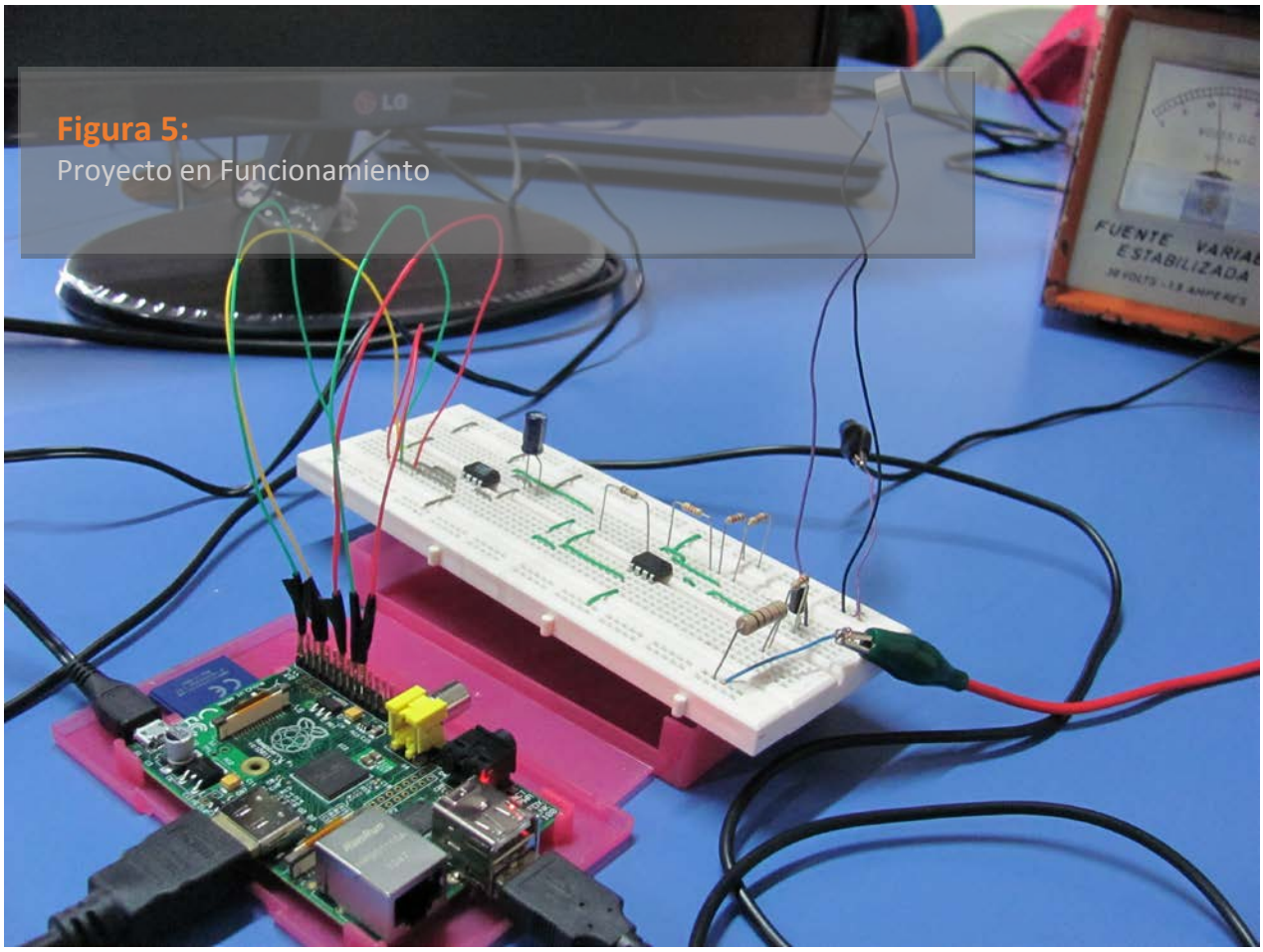
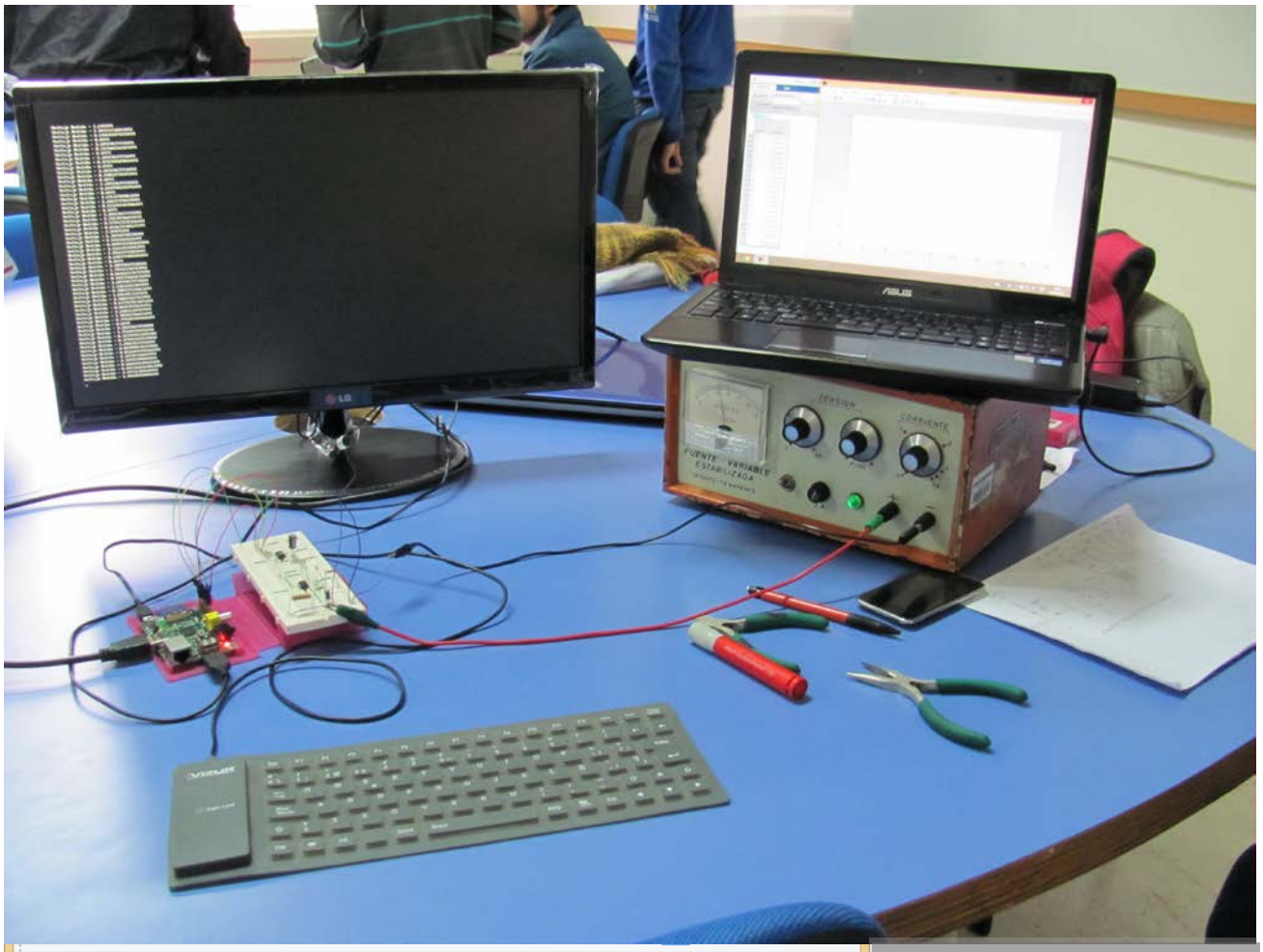


Figura 5:
Proyecto en Funcionamiento



problema3.py | script22.py | muestragrande.txt

```
0;0.190363769531
0.2;0.117270507813
0.4;0.0
0.6;1.24820800781
0.8;2.39360351563
1.0;3.19040039063
1.2;3.28919677734
1.4;3.28919677734
1.6;2.59762207031
1.8;1.78877685547
2.0;1.36869140625
2.2;0.730129394531
2.4;0.48193359375
2.6;0.696394042969
2.8;0.608842773438
3.0;0.118876953125
3.2;0.0819287109375
3.4;0.0
3.6;1.00081542969
3.8;2.15745605469
4.0;3.15907470703
4.2;3.28919677734
```

length : 74660 lines : 380 Ln : 1 Col : 1 Sel : 0 | 0 UNIX ANSI as UTF-8 INS

Figura 6:
Midiendo Voltajes y mostrando en pantalla los valores obtenidos, Muestra del Archivo generado por el proyecto (Archivo tipo CSV)

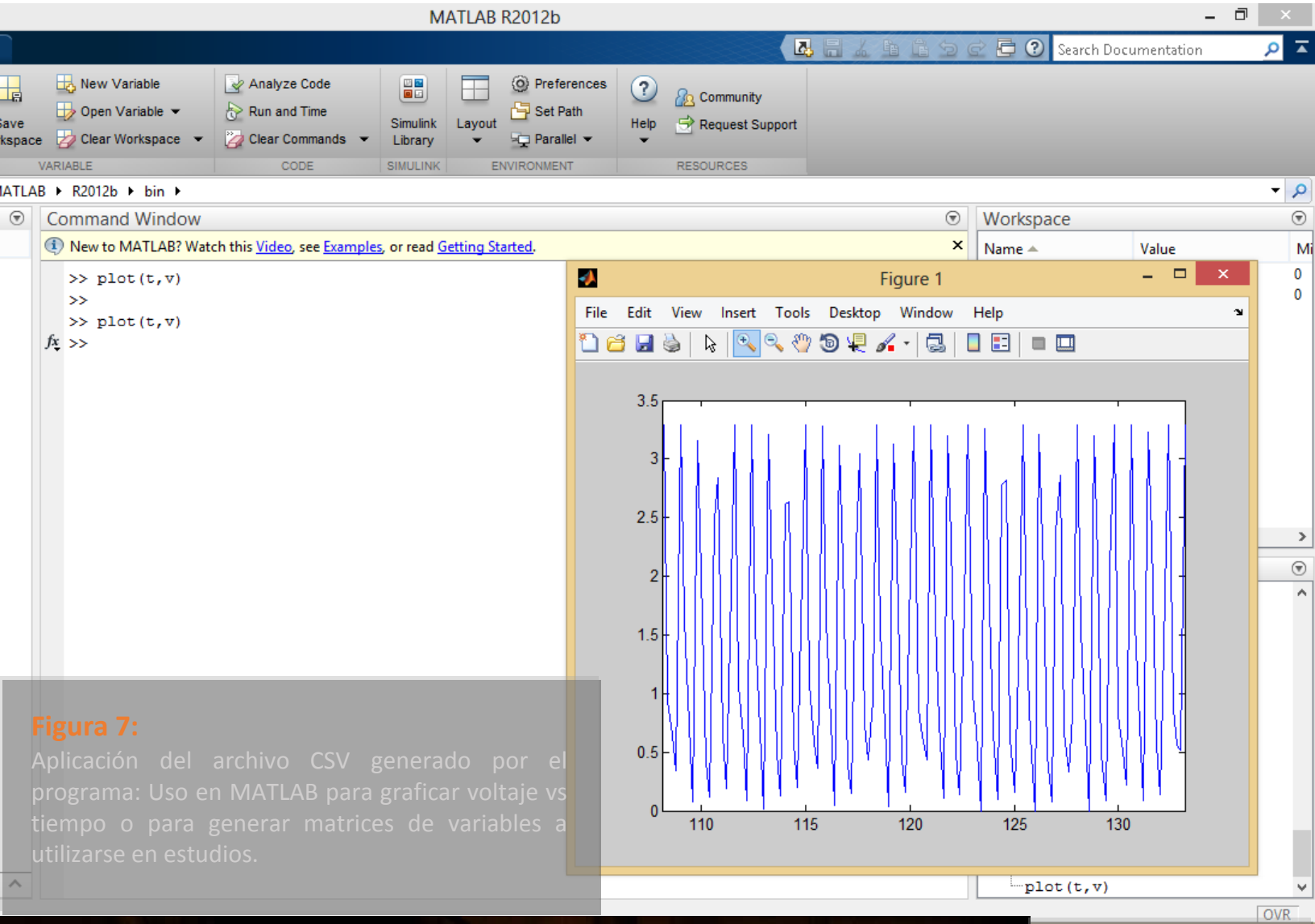


Figura 7:
Aplicación del archivo CSV generado por el programa: Uso en MATLAB para graficar voltaje vs tiempo o para generar matrices de variables a utilizarse en estudios.

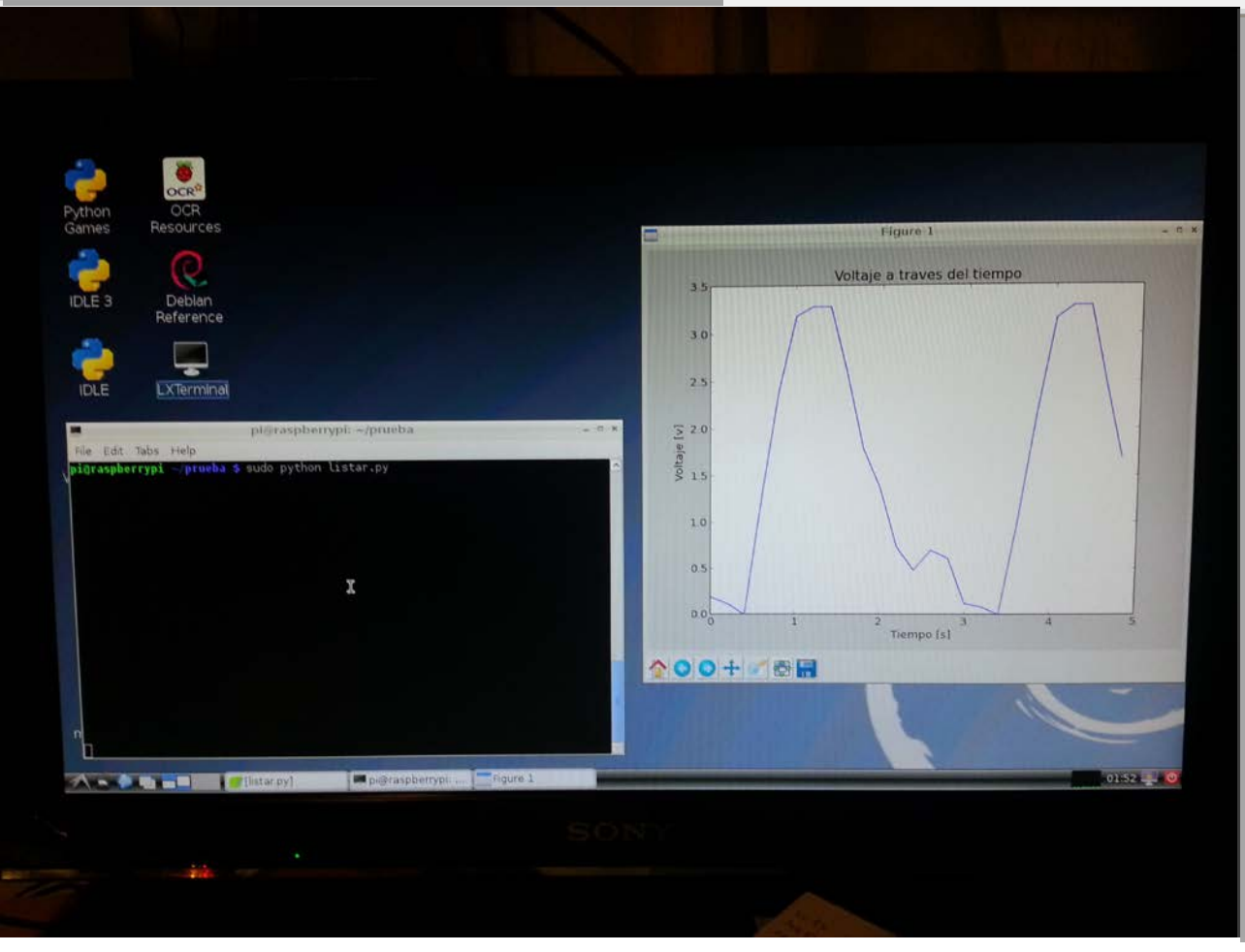


Figura 8:
Parte del Script que Grafica la función en la Raspberry Pi obtenida ya almacenados los datos

REFERENCIAS

- Para conectar el circuito integrado MCP3201 se tomó como referencia la siguiente página: http://www.internetdelascosas.cl/2013/03/10/usando-adc-en-raspberry-pi-mcp3201/

- Para introducirnos en el desafío de conectar un Amplificador Operacional para un Sensor de Sonido tomamos como referencia la siguiente página: http://www2.elo.utfsm.cl/~mineducagv/docs/ListaDetalladadeModulos/Sensor%20de%20sonido.pdf

- MCP3201 Specifications Datasheet: http://ww1.microchip.com/downloads/en/devicedoc/21290d.pdf

- Aplicación Electrodroid, para referencia en Resistencias y Cálculo de Ganancias en Amplificadores Operacionales: https://play.google.com/store/apps/details?id=it.android.demi.elettronica&hl=es_419

Figura 9: Interfaz Gráfica de ElectroDroid para Android

RECOPILADO DE INFORMACIÓN EN LA NUBE

Carpeta en Nube con Fotos, Scripts, Apuntes y Datasheets utilizados de toda la documentación recopilada, luego consultada y/o utilizada en este Proyecto de Raspberry Pi

https://app.box.com/iwg101

NUESTRAS CONCLUSIONES:

Como equipo logramos aprender sobre la sorprendente Interfaz General Multipropósito de Entrada y Salida (GPIO) que dispone la Raspberry Pi. Eso nos abre un camino muy grande para aplicaciones y experimientaciones caseras de tal manera que fomentemos y aprendamos más sobre Electrónica e incluso sobre Tecnologías de Información. Eso es algo que ningún computador tiene tan a la vista o mejor dicho tiene tan accesible al precio y la facilidad que la Raspberry Pi si lo ofrece.

Ver tanto potencial en un dispositivo dotado de un procesador ARM v6, el cual tiene una arquitectura tecnológica basada en el ahorro de energía y la integración de los diversos microsistemas embebidos que nos permiten transformar un pequeño dispositivo en un gran computador.

Agradecemos a la Universidad por la experiencia y el honor de tener este dispositivo del porte de una tarjeta bancaria en nuestro poder, lo cual nos incentiva de sobremanera en poder experimentar e ir más allá de lo que los docentes nos transmiten día a día en las aulas y laboratorios y que en el fondo es el espíritu de la universidad en liderar el aprendizaje en Ingeniería.

Como grupo, usted podrá apreciar que accedimos a implementar una mejora al código y la funcionalidad de nuestro proyecto en función a las sugerencias realizadas por Docentes y Visitantes de la Muestra realizada el Lunes 9 de Septiembre en la Universidad, destacándose así la anhelada función de graficar nuestros resultados una vez terminada la recopilación de datos ya descrita en este informe.

Queda un desafío de poder implementar gráficos en tiempo real, algo que hacen los osciloscopios. Ya el camino está a la vista…

Atentamente,

Felipe Escobar
Pablo Oñate
Valentina Yévenes