Fenómenos Transitorios Eléctricos

ESTACIONARIO → **TRANSITORIO** → **ESTACIONARIO**

Proceso muy general → se lo encuentra en muchas fenómenos / áreas de la física:
→ determina la existencia de un tiempo característico y de una amplitud de la señal "perturbada".
→ frecuencias características (ondas estacionarias)

SONIDO

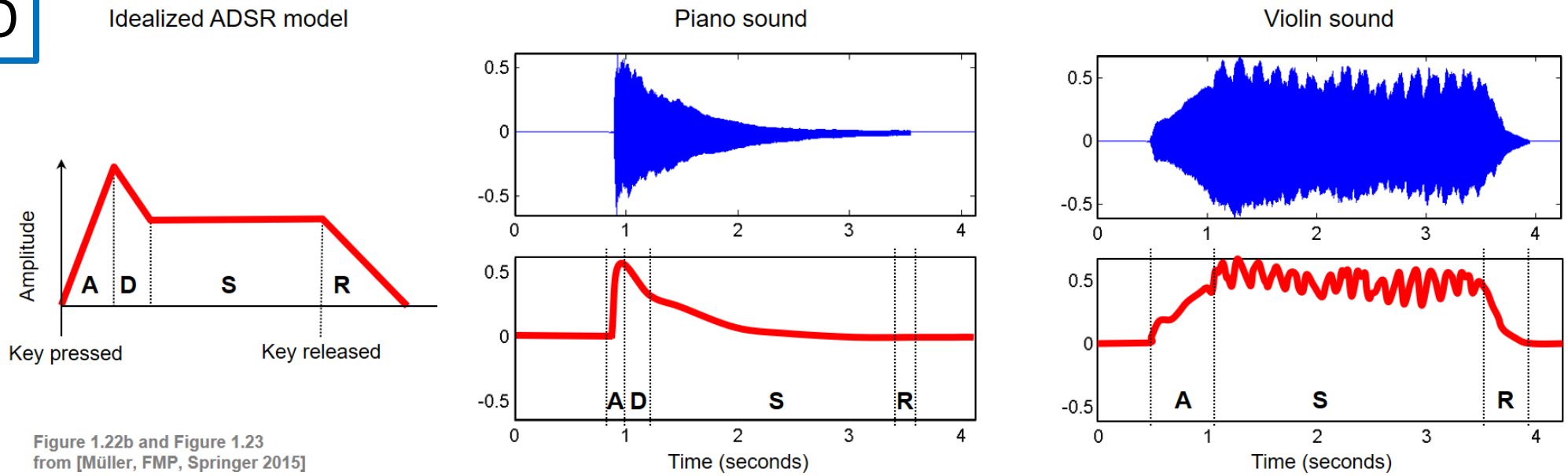


Figure 1.22b and Figure 1.23
from [Müller, FMP, Springer 2015]

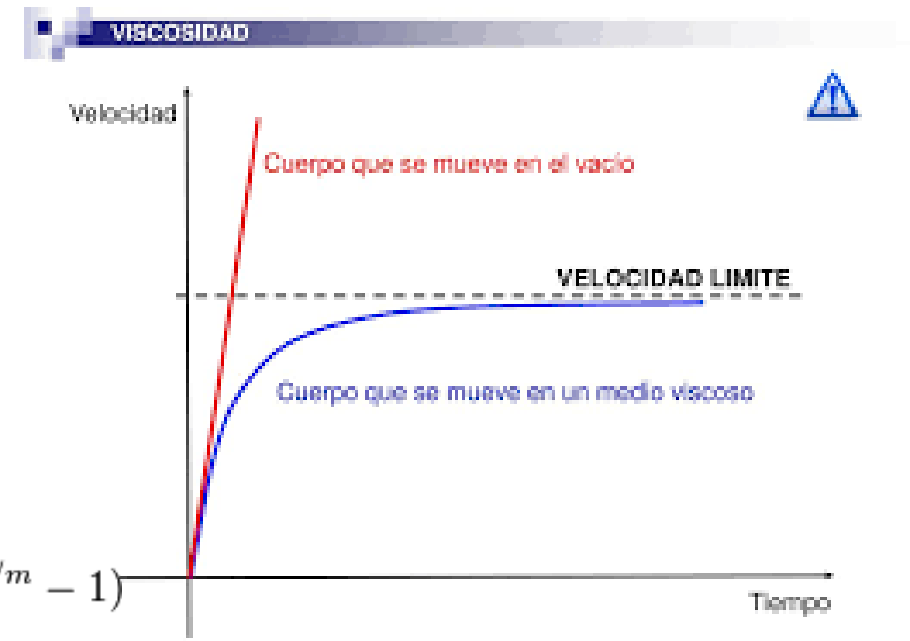
Fenómenos Transitorios

Caída libre (en medio viscoso)

$$\ddot{y} - b = 0 \quad \text{(sin rozamiento)}$$

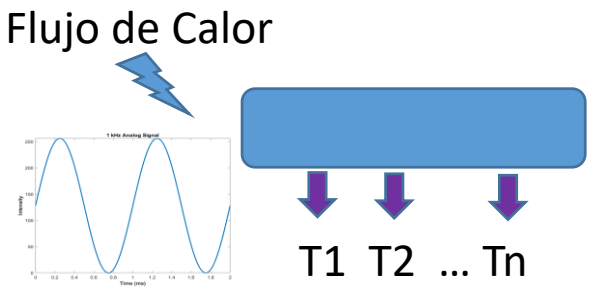
$$\ddot{y} + a\dot{y} - b = 0$$

$$\begin{cases} v_y = v_0 e^{-k_w t/m} + \dfrac{mg}{k_w}(e^{-k_w t/m} - 1) \\ y = h_0 - \dfrac{mgt}{k_w} + m\left(\dfrac{mg + k_w v_0}{k_w^2}\right)(e^{-k_w t/m} - 1) \end{cases}$$

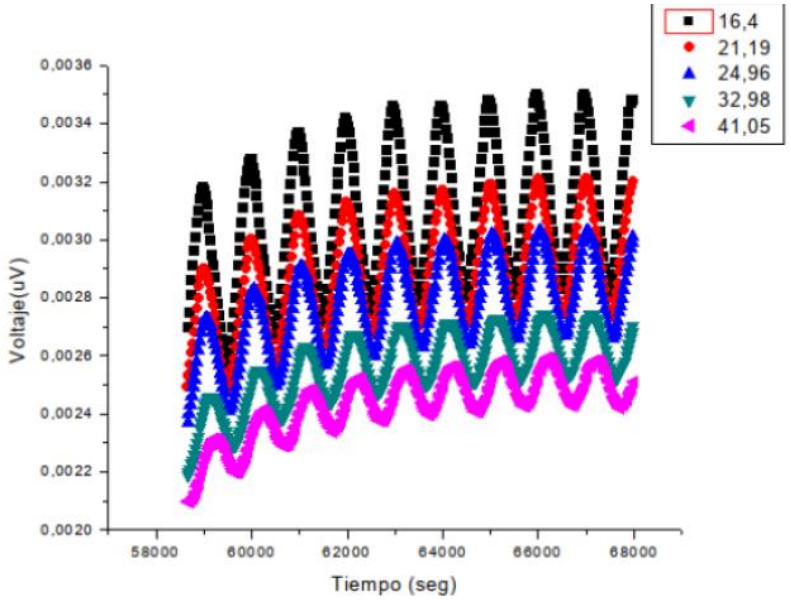


Difusión del calor

$$\frac{\partial^2 T}{\partial x^2} = \frac{1}{\kappa}\frac{\partial T}{\partial t}$$

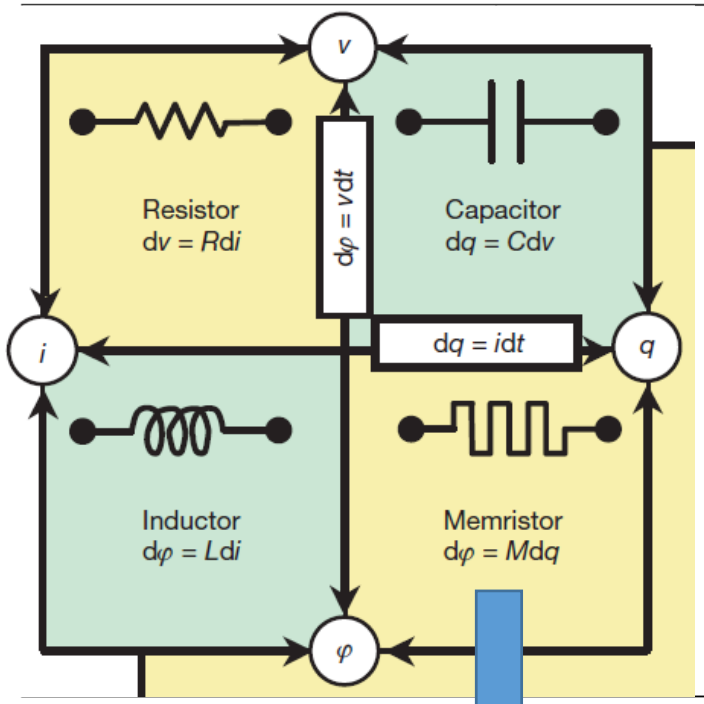


$$T(x,t) \simeq A_0 e^{-\epsilon x} \cos \omega (t - \frac{x}{v})$$



Transitorios Eléctricos: el circuito RC serie

Resistencia (R)- Inductancia (L) – Capacitancia (C)



$$V_B - V_A = -i\,R$$

$$V_L = L\left(\frac{di}{dt}\right)$$

$$i = C\left(\frac{dV_c}{dt}\right)$$

Nature 453 (2008)

The missing memristor found

$$M(q) = \mathcal{R}_{OFF}\left(1 - \frac{\mu_V \mathcal{R}_{ON}}{D^2} q(t)\right)$$

Dmitri B. Strukov[1], Gregory S. Snider[1], Duncan R. Stewart[1] & R. Stanley Williams[1]

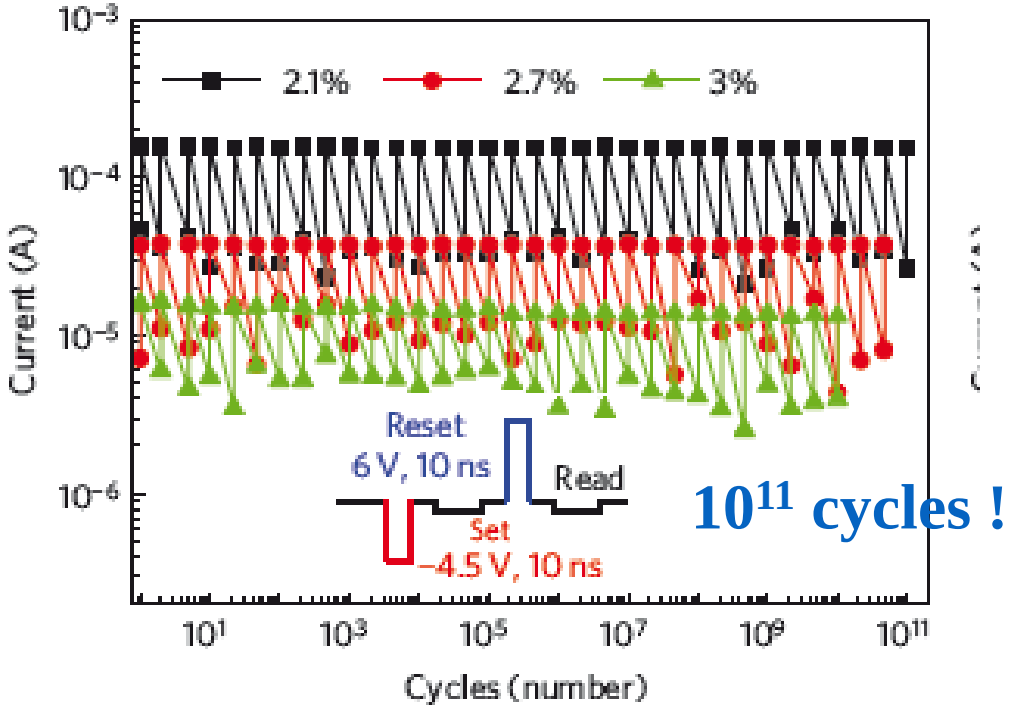
Memristor: memoria no-volátil

•tiempo de conmutación: 10 ns - durabilidad: 10^{11} ciclos

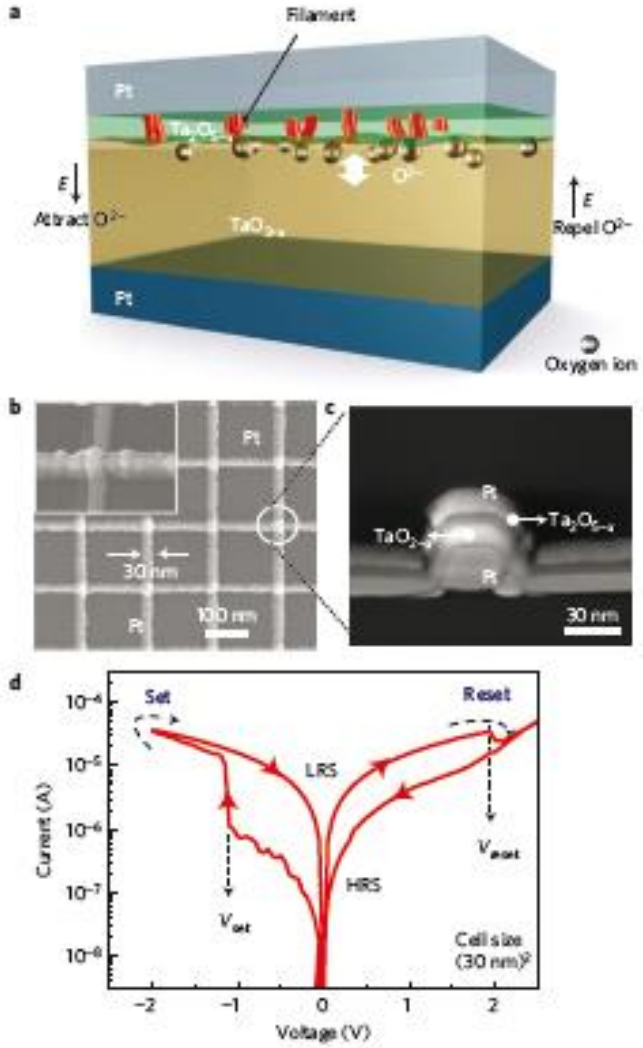
nature materials
ARTICLES
PUBLISHED ONLINE: 10 JULY 2011 | DOI: 10.1038/NMAT3070

A fast, high-endurance and scalable non-volatile memory device made from asymmetric Ta_2O_{5-x}/TaO_{2-x} bilayer structures

Myoung-Jae Lee[1]*, Chang Bum Lee[1], Dongsoo Lee[1], Seung Ryul Lee[1], Man Chang[1], Ji Hyun Hur[1], Young-Bae Kim[1], Chang-Jung Kim[1]*, David H. Seo[1], Sunae Seo[2], U-In Chung[1], In-Kyeong Yoo[1] and Kinam Kim[3]

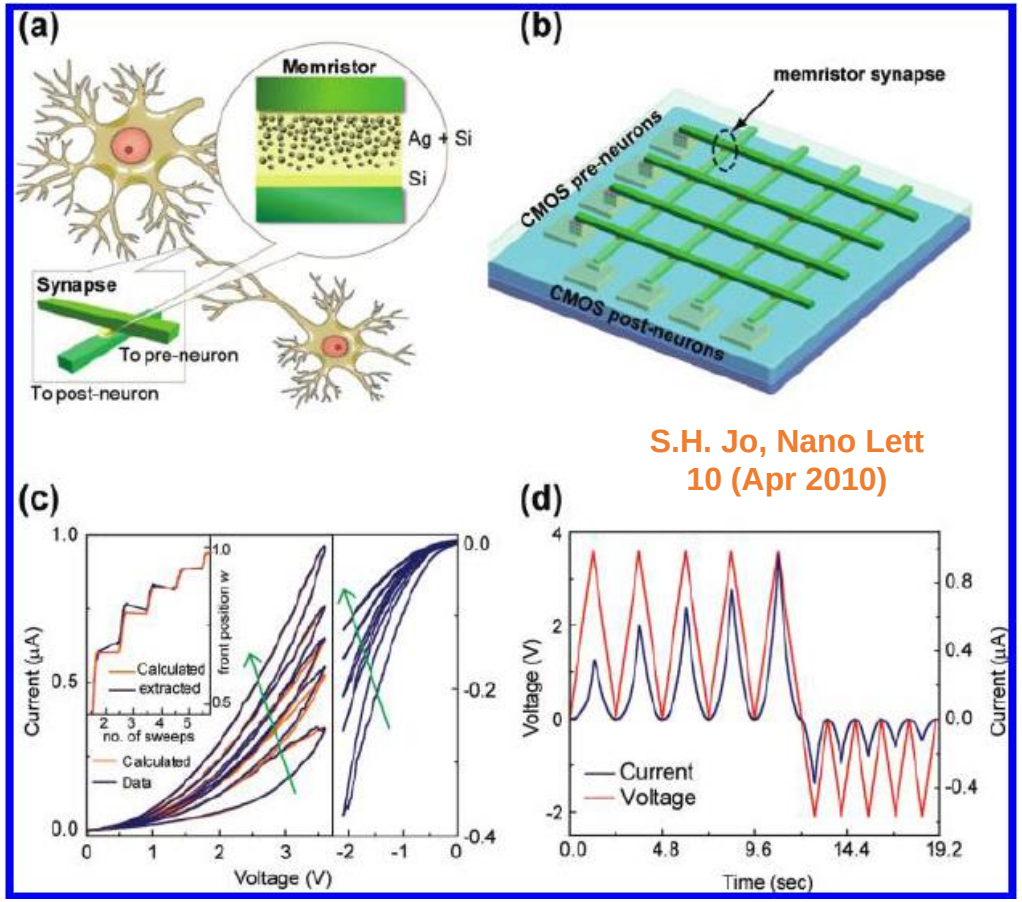


10^{11} cycles !

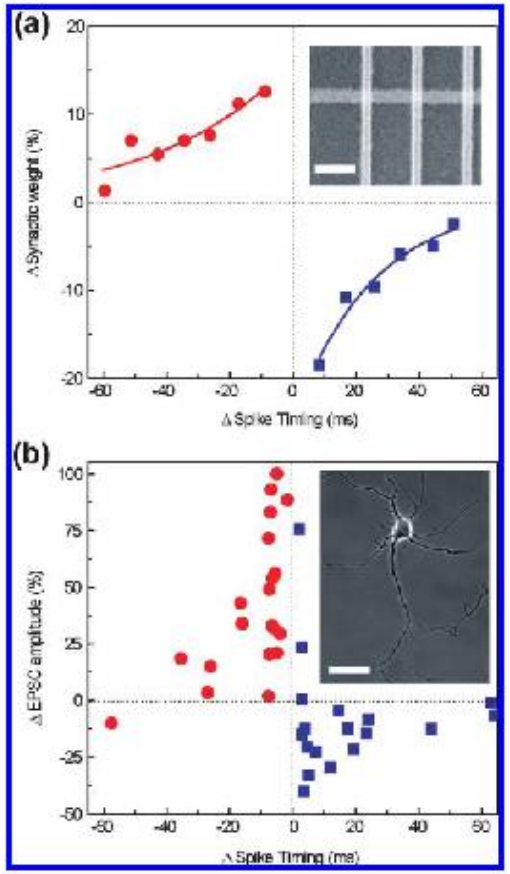


Memristor → Sinapsis: Aprendizaje = Plasticidad

•Emulación de sinapsis – Spike Timing Dependent Plasticity (STDP)

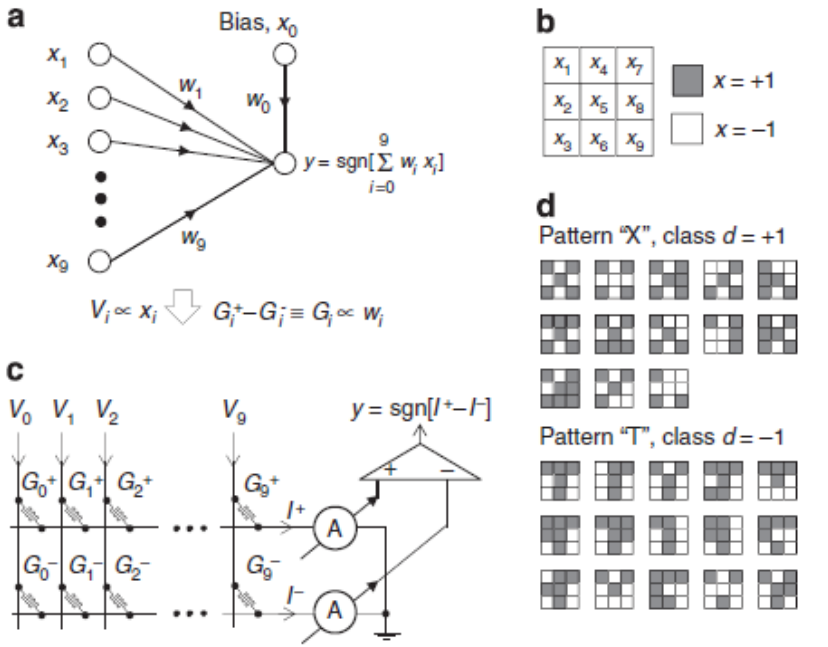


S.H. Jo, Nano Lett
10 (Apr 2010)



Memristor: desarrollo de electrónica inteligente

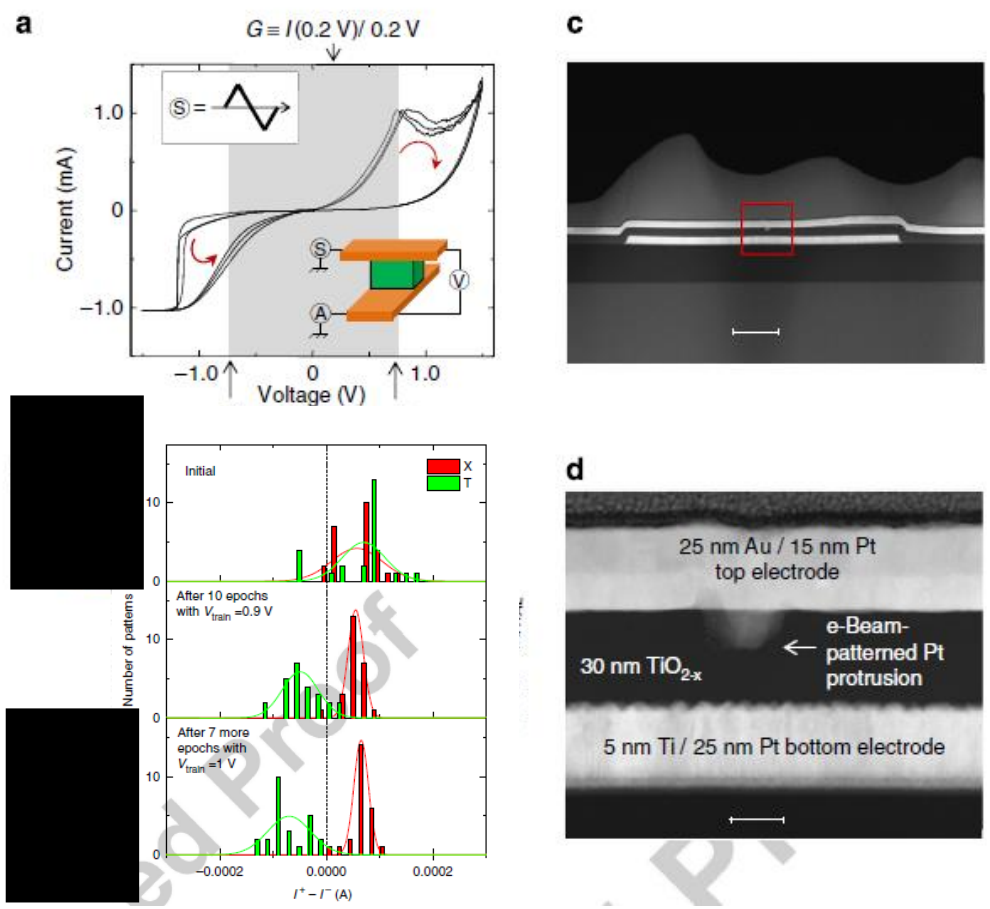
- **El perceptrón electrónico → red neuronal de estado sólido**



ARTICLE
Received 25 Jan 2013 | Accepted 29 May 2013 | Published xx xxx 2013
DOI: 10.1038/ncomms3072

Pattern classification by memristive crossbar circuits using *ex situ* and *in situ* training

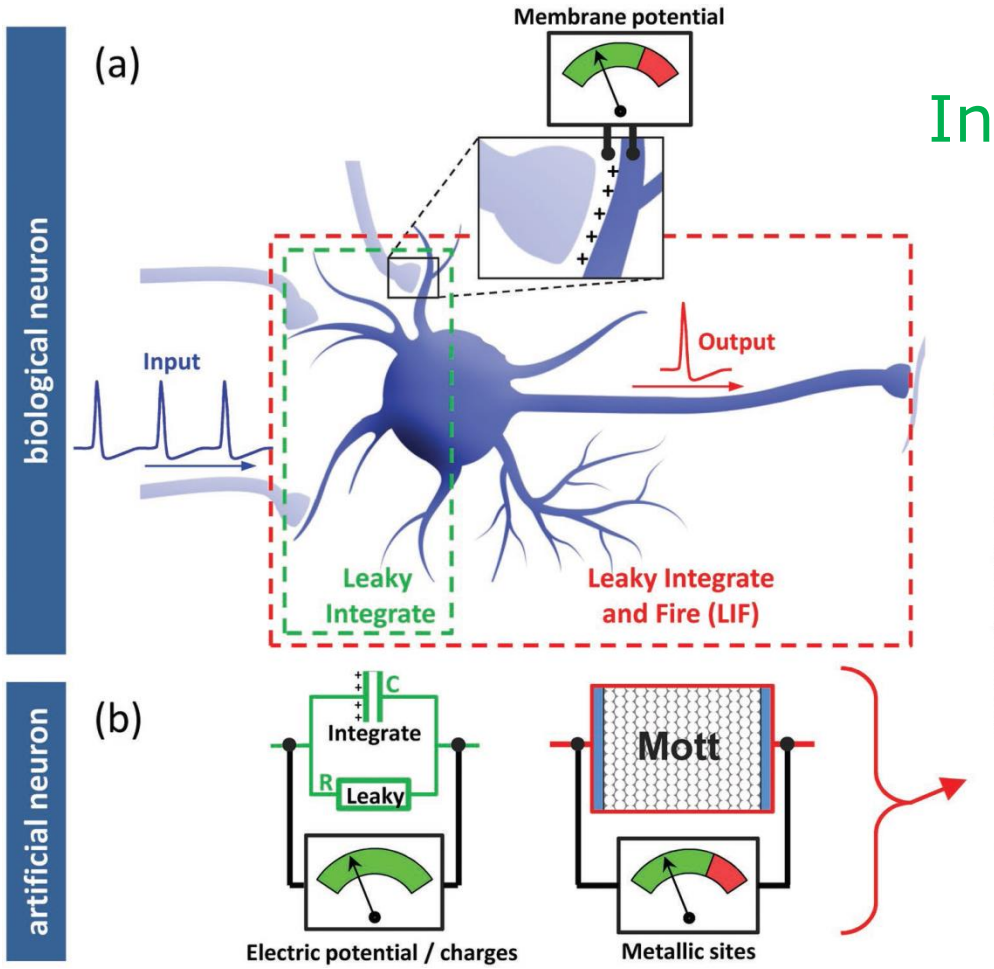
Fabien Alibart[1,†], Elham Zamanidoost[1] & Dmitri B. Strukov[1]



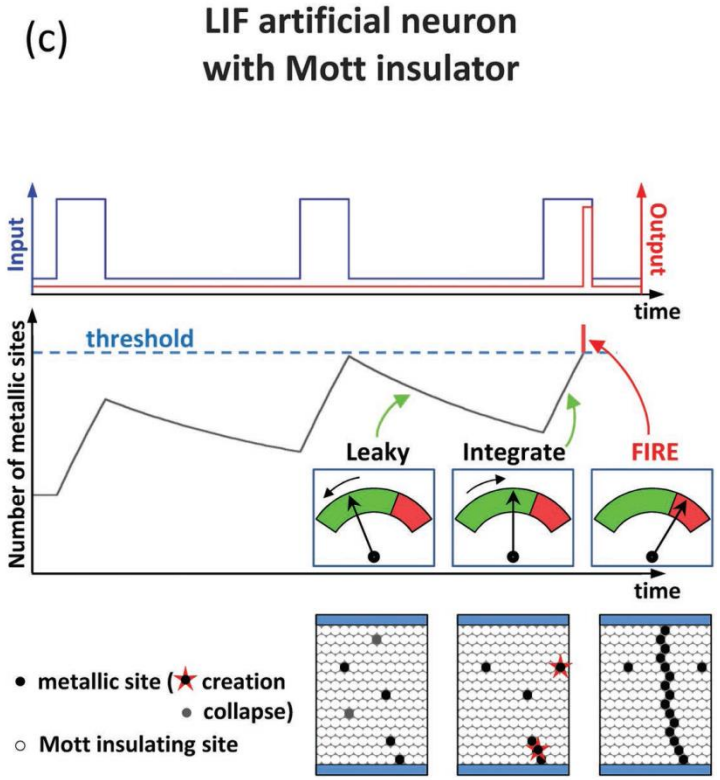
F. Alibart et al., NatComms 3072 (2013)

Memristor: neurona de estado sólido

Memristores basados en aislantes de Mott como posibles neuronas?



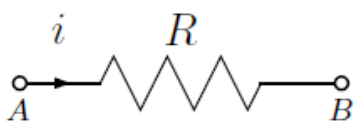
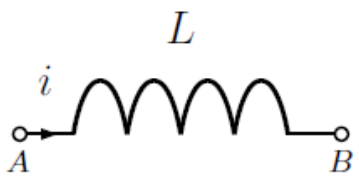
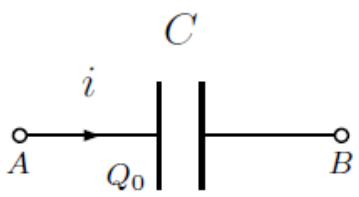
El modelo de Leaky Integrate and Fire (LIF)



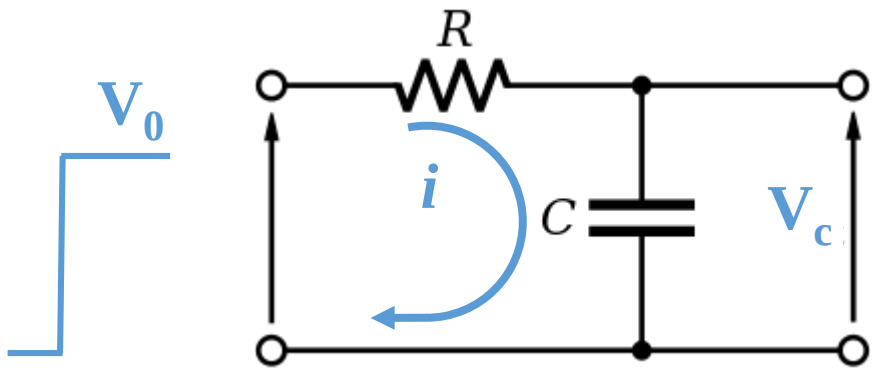
P. Stoliar,..,F. Tesler, M. Rozenberg et al., Adv. Funct. Mater (2017)

Transitorios Eléctricos: el circuito RC serie

Resistencia (R)- Inductancia (L) – Capacitancia (C)

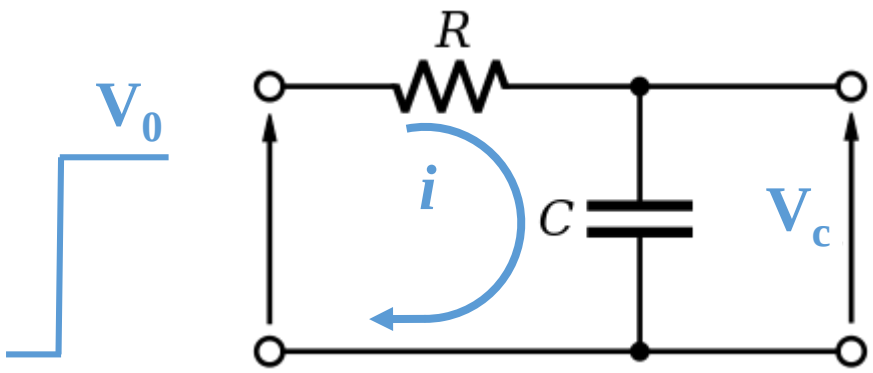
	$\Delta V_R \equiv V_B - V_A = -i\,R$
	$\Delta V_L \equiv V_B - V_A = -L\,\mathrm{d}i/\mathrm{d}t$
	$\Delta V_C \equiv V_B - V_A = -\frac{Q_0}{C} - \frac{1}{C}\int i\,\mathrm{d}t$

V(t≤0) = 0. Qué pasa al aplicar V_0 (t>0)?



El circuito RC serie

El enfoque experimental



PREGUNTAR
OBSERVAR

-Habrá un tiempo característico?

-La i, V_R, V_C alcanzarán un estado estacionario?

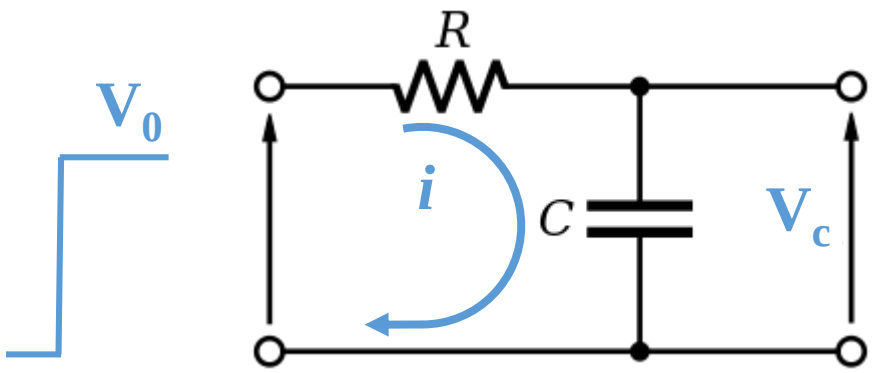
-Cómo evolucionarán i(t) y Vc(t)?

-Diseño del experimento

El circuito RC serie

El enfoque teórico

MODELIZAR
ANALIZAR



$$V_0 = i\,R + \frac{1}{C}\int_0^t i\,\mathrm{d}t' + V_C(0) \qquad t > 0$$

$$0 = R\,\frac{\mathrm{d}i}{\mathrm{d}t} + \frac{1}{C}\,i \qquad t > 0$$

Ecuación diferencial ordinaria, lineal, homogénea de orden 1

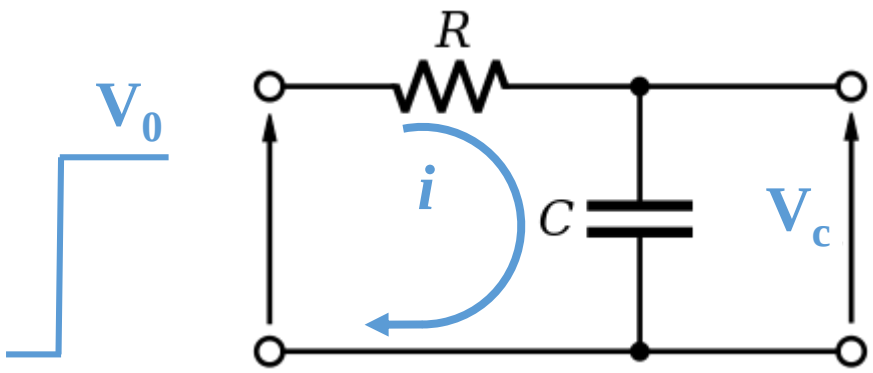
$$i(t) = \mathrm{A}\,e^{Bt} \qquad \frac{di}{dt}(t) = \mathrm{AB}\,e^{Bt} = B\,i(t)$$

$$(RCB + 1)e^{Bt} = 0 \;\rightarrow B = -\frac{1}{RC}$$

El circuito RC serie

El enfoque teórico

MODELIZAR
ANALIZAR



Ecuación diferencial ordinaria, lineal, homogénea de orden 1

$$i(t) = \mathrm{A}\, e^{Bt} \qquad \frac{di}{dt}(t) = \mathrm{AB}\, e^{Bt} = B\, i(t)$$

$$(RCB + 1)e^{Bt} = 0 \;\rightarrow B = -\frac{1}{RC}$$

$$V_0 = i\,R + \frac{1}{C}\int_0^t i\,\mathrm{d}t' + V_C(0) \qquad t > 0$$

$$0 = R\,\frac{\mathrm{d}i}{\mathrm{d}t} + \frac{1}{C}\,i \qquad t > 0$$

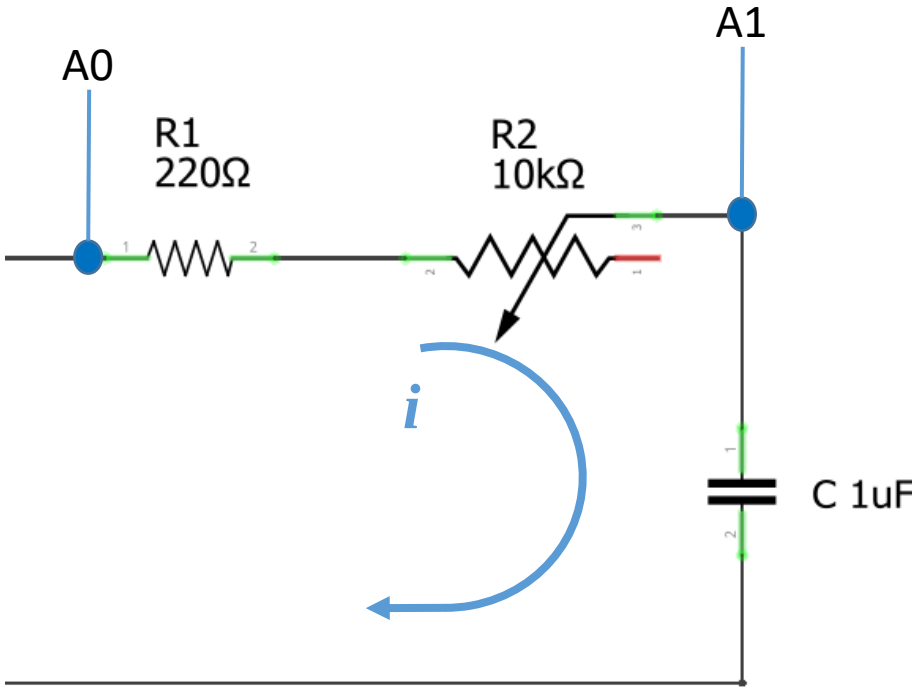
$$i(0) = \frac{V_0 - V_C(0)}{R}$$

$$\boxed{i(t) = \frac{V_0 - V_C(0)}{R}\, e^{-t/RC}}$$

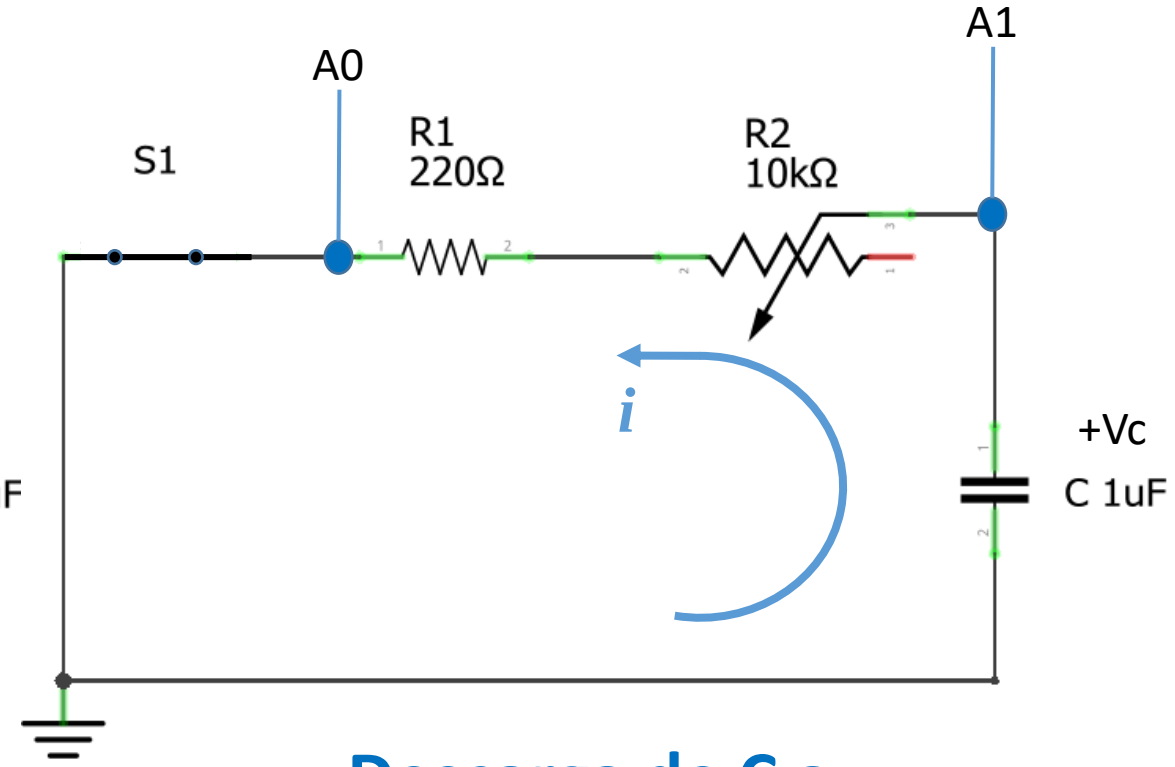
$$\boxed{V_C(t) = V_C(0) + [V_0 - V_C(0)]\,\left(1 - e^{-t/RC}\right)}$$

El circuito RC serie: estudio experimental

Diseño del experimento: **Medir t, Medir i(t), medir V(t) / variar R1+R2**



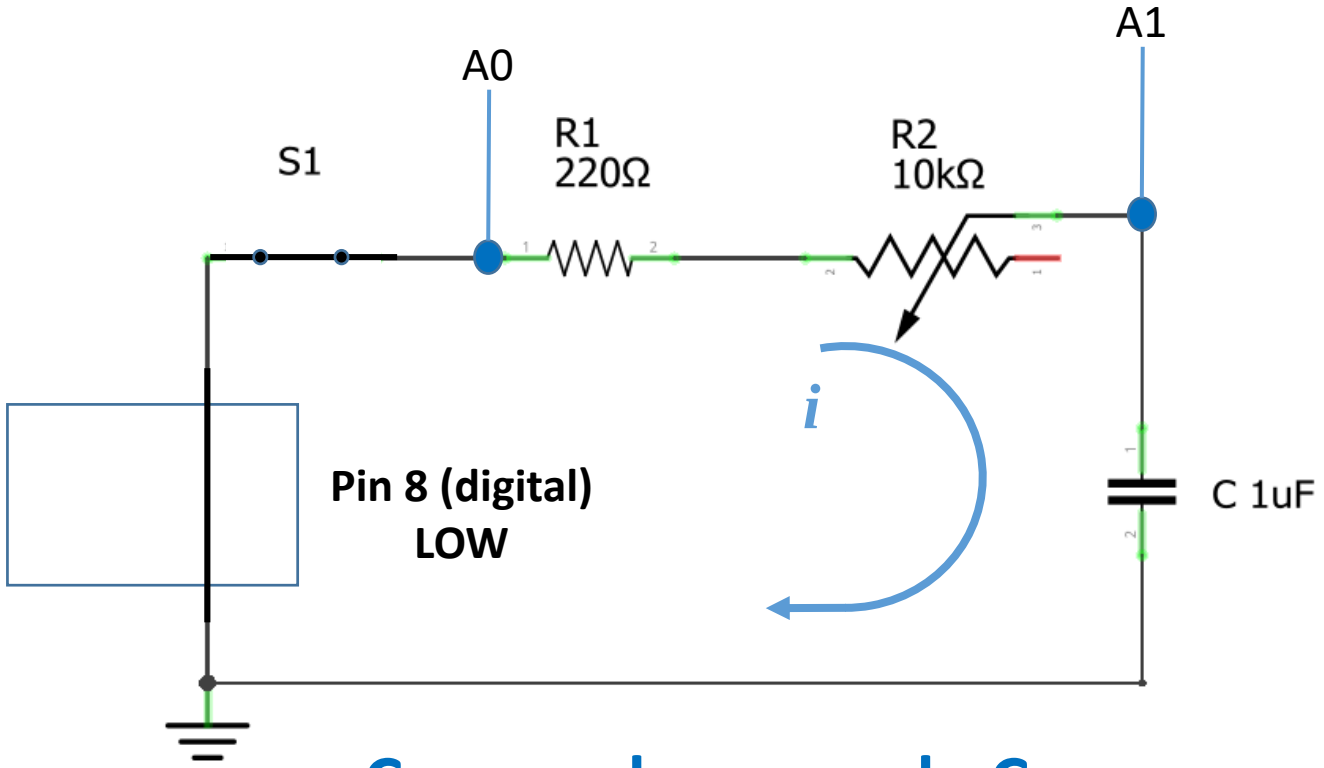
Carga de C a través de R1+R2



Descarga de C a través de R1+R2

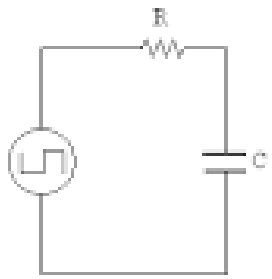
El circuito RC serie: estudio experimental

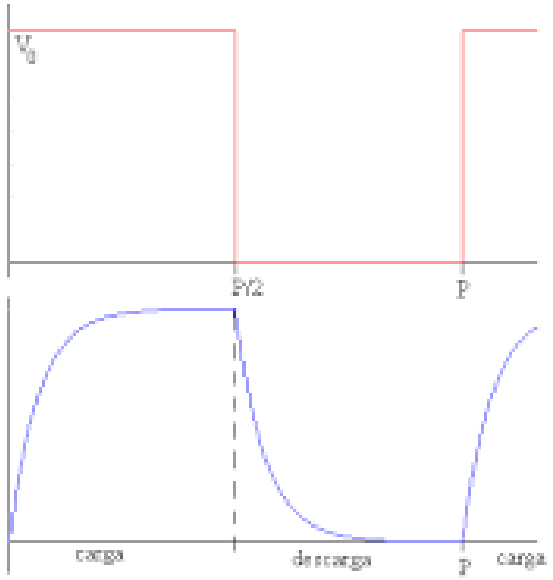
Diseño del experimento: Medir t, Medir i(t), medir V(t) / variar R1+R2



$i = (V_{A0} - V_{A1}) / (R1 + R2)$; $[\frac{\Delta i}{i} = \frac{\Delta V}{V} + \frac{\Delta R}{R}]$

$Vc = V_{A1}$

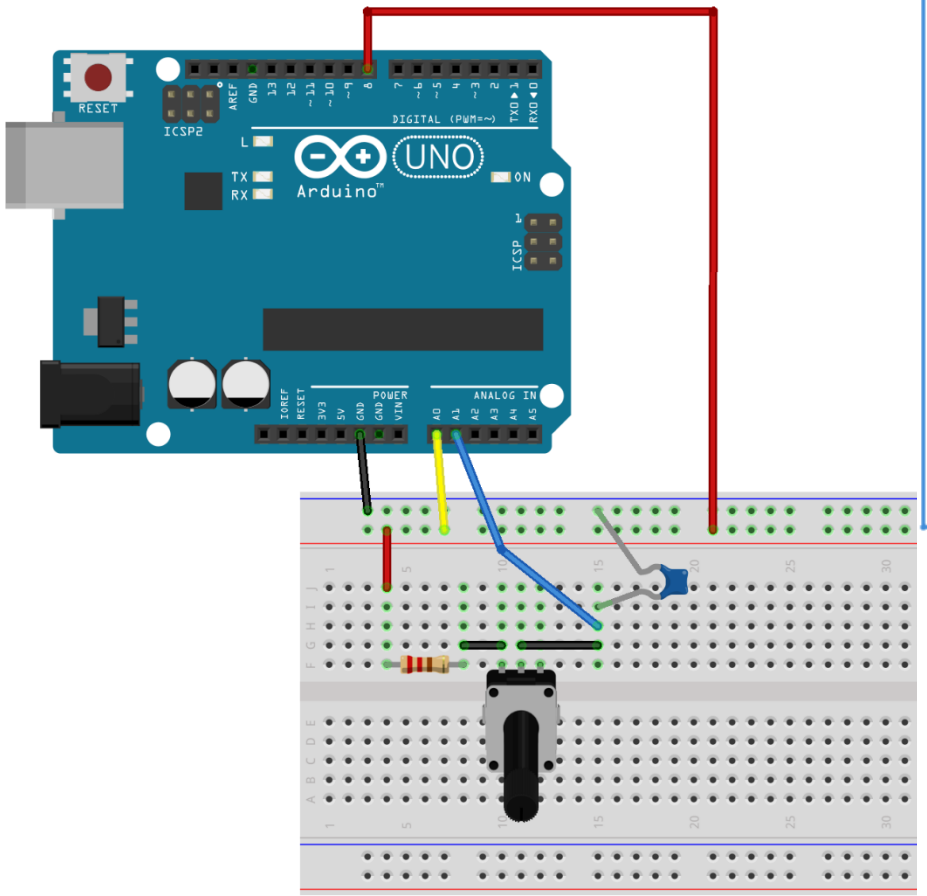


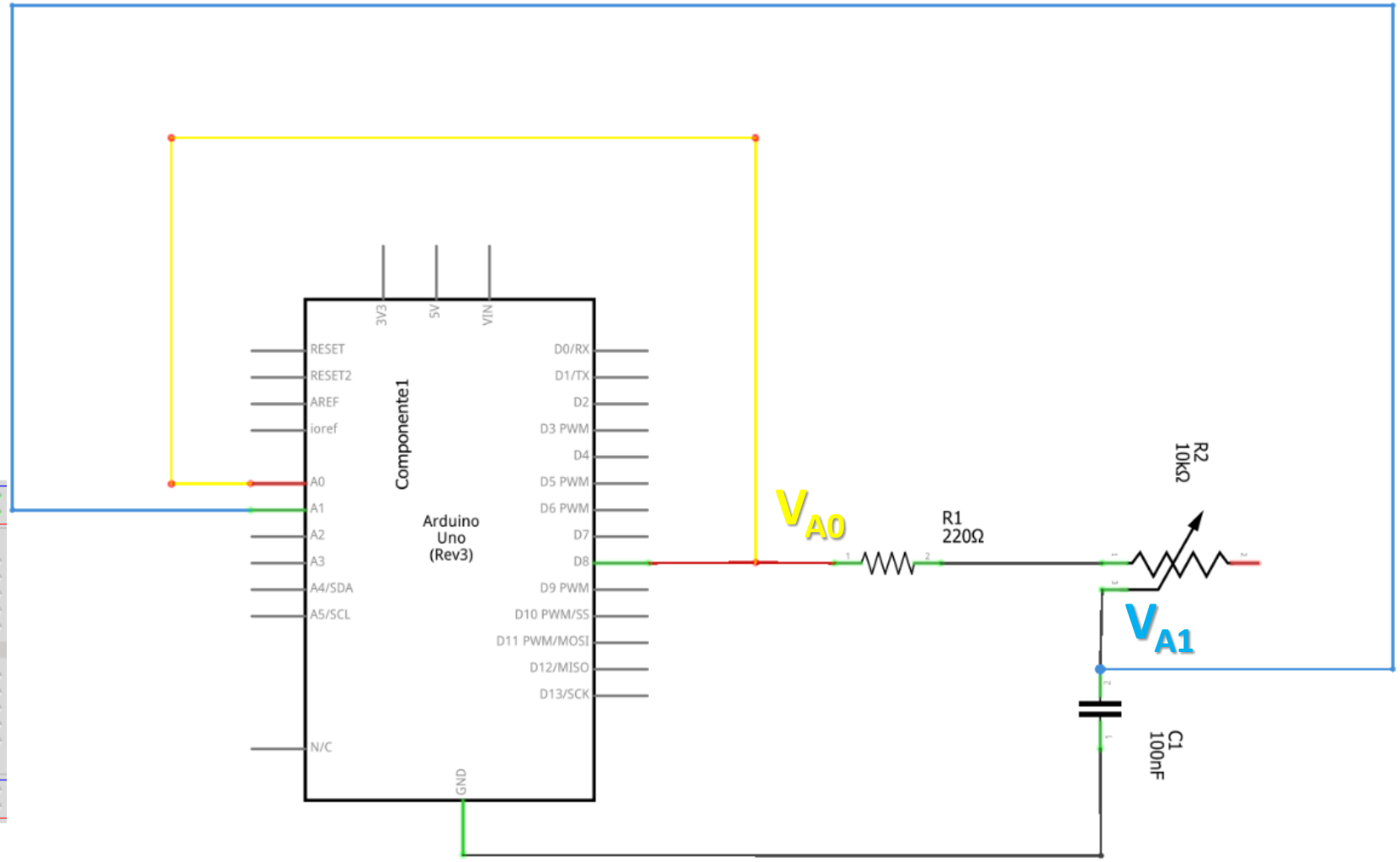


Carga y descarga de C a través de R1+R2

Tiempo de medición > Tiempo característico (τ)

El circuito RC serie: estudio experimental





fritzing

El circuito RC serie: estudio experimental

Recordar la nomenclatura para la selección de los condensadores

$$\underbrace{10}\underbrace{4} = 10 \times 10^{4}$$

10 0000 pF

Respetar la polaridad de los “electrolíticos”

El circuito RC serie: estudio experimental

Transitorio_RC_Lab3 Arduino 1.8.13

Archivo Editar Programa Herramientas Ayuda

Transitorio_RC_Lab3

```
/*  LABORATORIO 3 - DF - FCEyN - UBA
 *  Este programa permite medir los voltajes de las entradas
 *  analógicas A0 y A1 en función del tiempo
 *  Enciende a alto (5 V) una salida digital (pin 8) para medir el transitorio de un cir
 *  RC y LC y mide 60 veces la evolución de estos 2 voltajes. Luego aplica un valor bajo
 *  (0 V) y repite las 60 mediciones. Finalmente envía por el puerto serie las
 *  120 mediciones del tiempo y de ambos voltajes
 */

int V0[160], V1[160];
long tiempo[160];
int espera=0; // Cambiar para ajustar la ventana de datos a la frecuencia medida // 200
/* CAMBIAR "espera" según la frecuencia de la señal a medir: espera=0--> 1 medición cada
 *  para frec>4[illegible] 1000 Hz // 100 161-450 Hz // 300 74-160 Hz // 1000 26-73 Hz // 3000 9
 * Para frec> 1000 Hz fijar espera=0 y cambiar el prescaler de 04 a 03 como se indica má
 */

void setup() {

Serial.begin(57600); // velocidad de transferencia de la interfaz serie
pinMode(8, OUTPUT); // fija al pin digital 8 como Salida

/* CAMBIAR "0x04" o "0x03" según la frecuencia de la señal a medir
 *  USAR "0x04" para frecuencias < 1000 Hz y regular con "espera" para frecuencias bajas
 *  USAr "0x03" para frecuencias >= 1000 Hz con espera=0
 */
ADCSRA = (ADCSRA & 0xf8) | 0x03; // clear prescaler | set prescaler to /16
ADCSRA |= 1<<ADEN; // make sure ADC is enabled
```

Se miden 80 tríos de datos (t, VA0, VA1) para el semi-ciclo de carga (Pin 8 “HIGH”) y otros 80 tríos de datos para el semi-ciclo de descarga (Pin 8 “LOW”)

Recomendación para ajustar el valor de “espera”: espera=0 → mide 80 datos (t, VA0, VA1) en 4 ms

Espera=n μs de espera entre cada terna de medición → 80 x n μs. Si n=200 → (80X0.2 + 4)ms = 20 ms para los 80 datos (semi-ciclo)

Guardado.

```
El Sketch usa 4258 bytes (13%) del espacio de almacenamiento de programa. El máximo es 32256 bytes.
Las variables Globales usan 1512 bytes (73%) de la memoria dinámica, dejando 536 bytes para las variables locales. El máximo es 2048 bytes.
```

79 Arduino Uno en COM4

Escribe aquí para buscar Escritorio ESP 20:04 24/9/2020

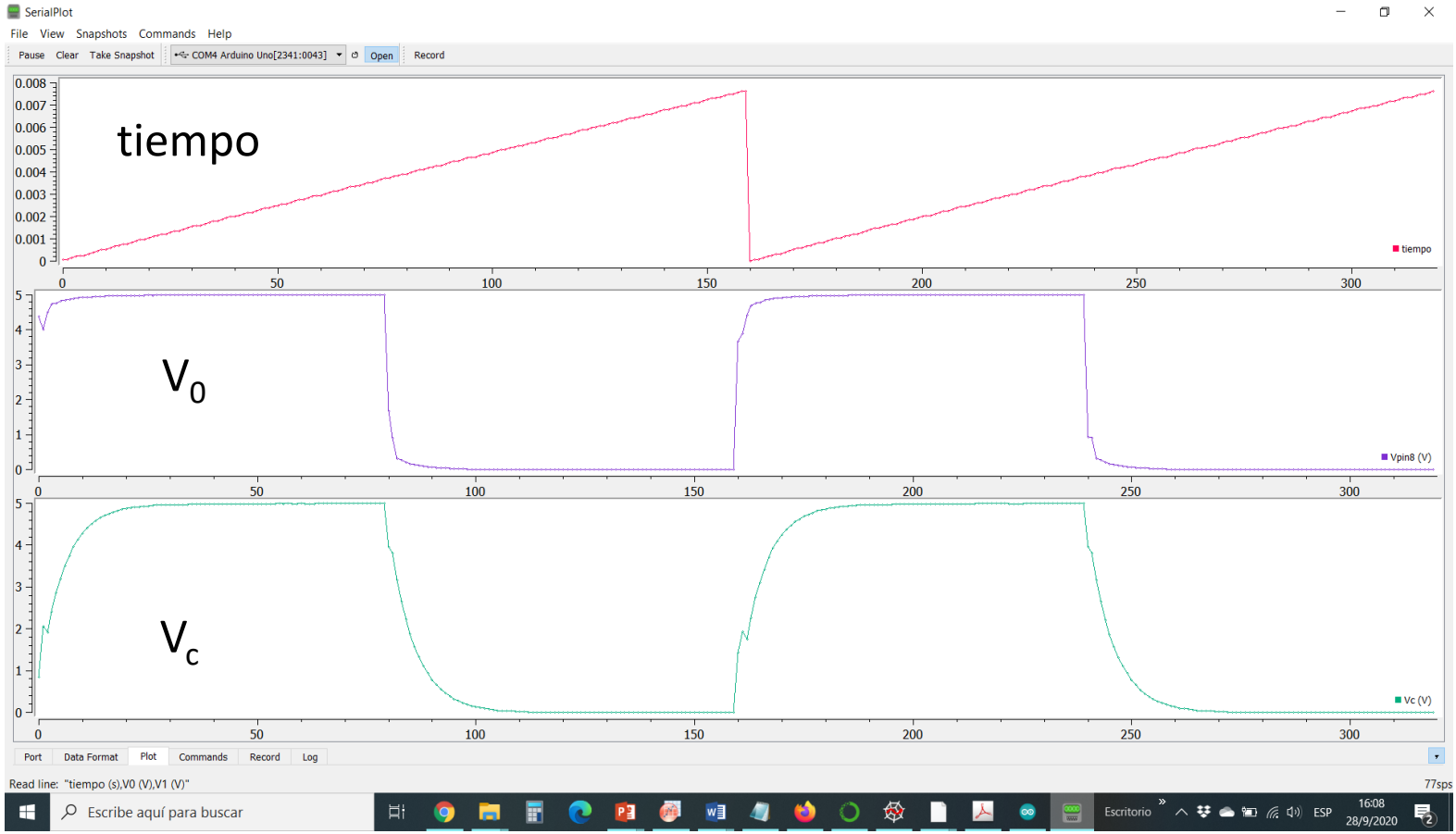
El circuito RC serie: estudio experimental

Tiempo de medición OK respecto del tiempo característico

Se observa la saturación de $V_{A1}(t)$

Es decir $T_{med} > 4\,\tau$

→ No cambiar "espera"



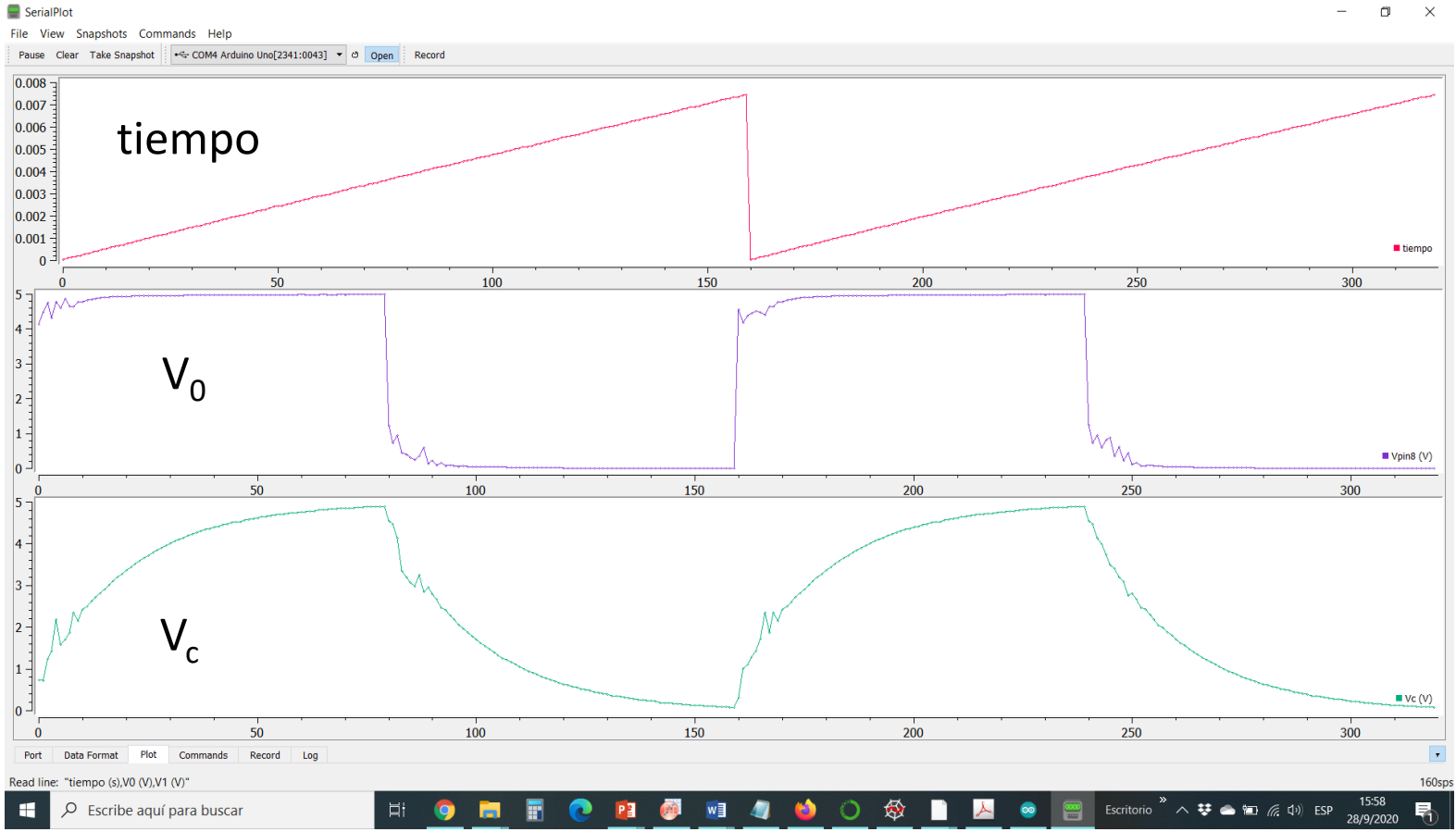
El circuito RC serie: estudio experimental

Tiempo de medición corto respecto del tiempo característico

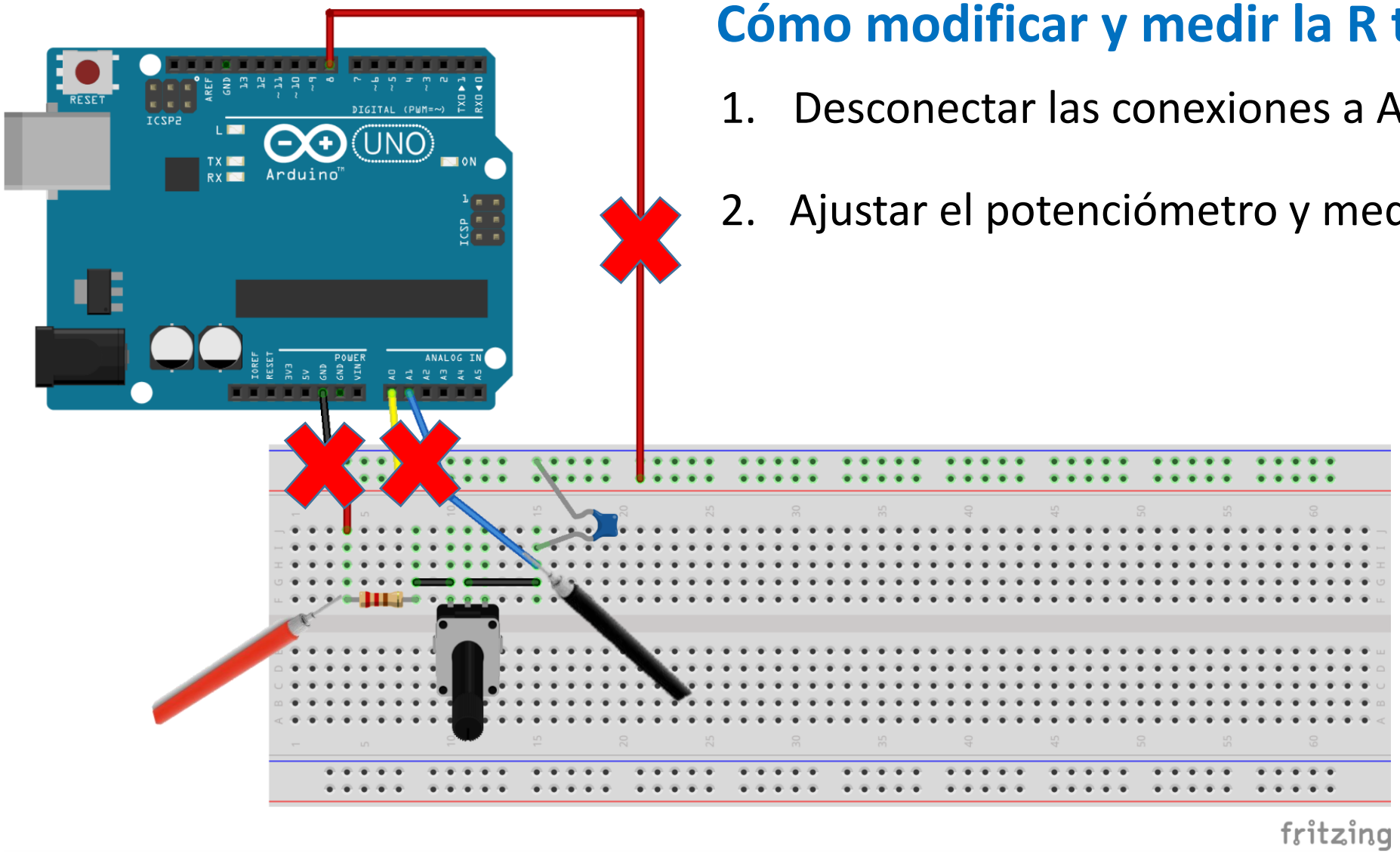
No se observa la saturación de $V_{A1}(t)$

Es decir $T_{med} < 4\,\tau$

→ Aumentar "espera"



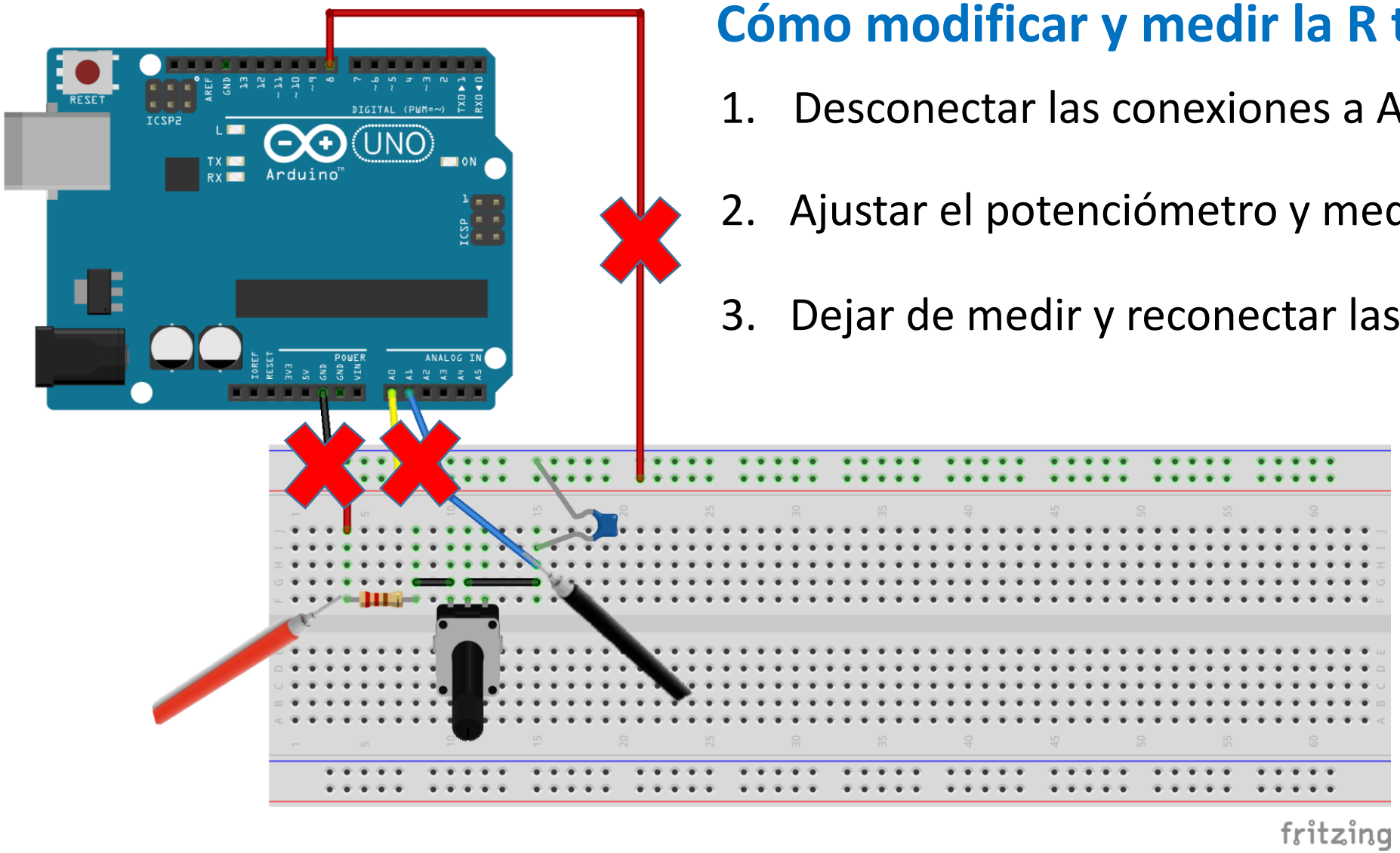
El circuito RC serie: estudio experimental



Cómo modificar y medir la R total?

1. Desconectar las conexiones a Arduino
2. Ajustar el potenciómetro y medir la R total

El circuito RC serie: estudio experimental



Cómo modificar y medir la R total?

1. Desconectar las conexiones a Arduino
2. Ajustar el potenciómetro y medir la R total
3. Dejar de medir y reconectar las conexiones a Arduino

El circuito RC serie con Arduino: un desafío a la sistematicidad!

Pasos:

1. Abrir el IDE de ARDUINO (Transitorio_RC_Lab3.ino) y compilarlo (cargarlo en Arduino)
2. Abrir el SerialPlot: setear plot y fijar nombre de los archivos de salida
3. Fijar R2 y medir R1+R2 con el multímetro
4. Conectar los pines de Arduino
5. Monitorear la señal con el SerialPlot
6. Si se observa la saturación de Vc seguir con **7** / caso contrario→ ir a **9**
7. Grabar (prop. de indexación automática) archivo con 3-4 ciclos de carga y descarga
8. Volver a **3**

Pasos':

9. Cerrar el SerialPlot
10. Cambiar el valor de la variable "espera" en el IDE de ARDUINO (Transitorio_RC_Lab3.ino)
11. Compilar este IDE (cargarlo en Arduino)
12. Abrir el SerialPlot
13. Monitorear la señal con el SerialPlot
14. Si se observa la saturación seguir con **7** / caso contrario volver a **9**

Scripts de Python para graficar y analizar los datos

C:\Users\Charly\NubeDF\PythonProgs\Cursos\Lab3\RCtransitorio_Ch.py

Código fuente Ejecutar Depurar Terminales Proyectos Herramientas Ver Ayuda

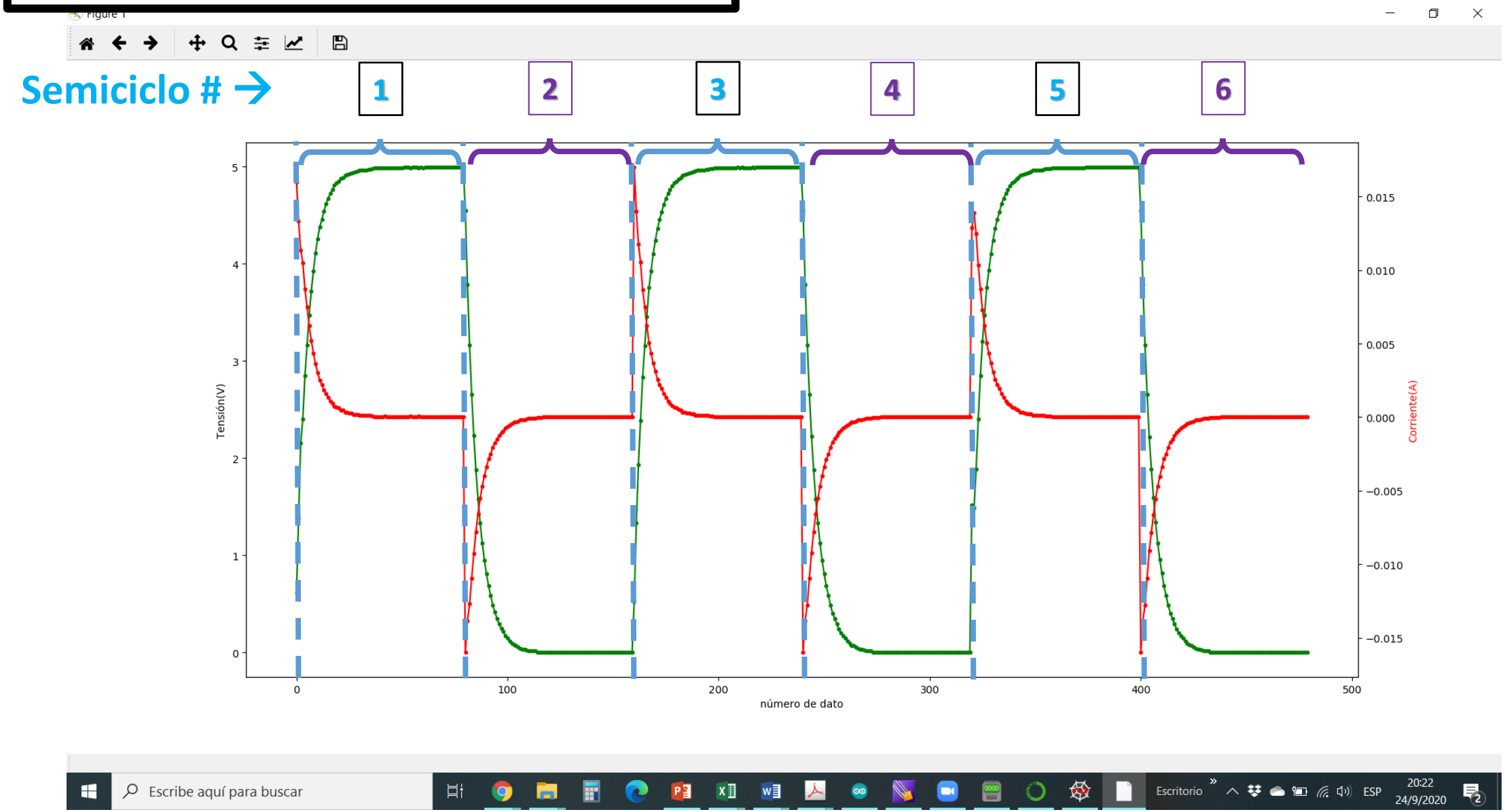
1) RCTransitorio.py → ajusta la Vc(t) y la i(t) con las expresiones que figuran en "enfoque teórico". Se extrae el valor del tiempo tau para cada R elegida

```
22
23  #Elijo el directorio donde se encuentran los archivos que voy a analiza
24  os.chdir (r'C:\Users\Charly\NubeDF\Arduino\Data\Transitorios')
25
26  print("nombre del archivo completo con terminación .txt incluida")
27  file = input()
28  print("Valor de la R (ohm)")
29  Resist = float(input())
30
31  #Importo los datos del archivo
32  data = np.loadtxt(file,dtype=float,delimiter = ',',skiprows= 1)
33
34
35
36  x1i = data[:,0] #columna de tiempo en segundos
37  y1i = data[:,2] #caida de tensión sobre el capacitor
38  y2i= (data[:,1]-data[:,2])/Resist   #corriente del circuito
39
40
41
42
43  #vdif=
44  #const
45  numdat
46
```

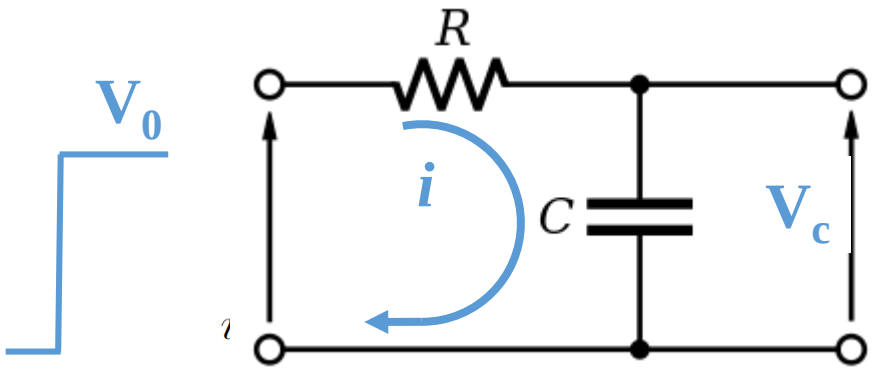
```
58
59  #%%
60  print("Seleccionar el semi-ciclo de carga o descarga
61  print("Por ej: 1-3-5 para carga o 2-4-6 para descarg
62  print("(si es que tienen 3 ciclos completos medidos)
63  ciclo = int(input())
64
65  #limitar los datos al rango de interés
66
67  datoinicial1=(ciclo-1)*80
68  datofinal1=datoinicial1+79
69
70
71  x1=x1i[datoinicial1:datofinal1]
72  y1=y1i[datoinicial1:datofinal1]
73  y2=y2i[datoinicial1:datofinal1]
74  x1corr=x1i[datoinicial1+5:datofinal1]
75  y2corr=y2i[datoinicial1+5:datofinal1]
76
77  #Graficamos los datos en la zona elegida en función
82  y2x=y1x.twinx()
```

Se graban los valores de tau vs R en el archivo: "SalidaRCTransitorio.txt"

Cómo elegir los “semi-ciclos”



Análisis del “semi-ciclo” elegido



$$i(t) = \frac{V_0 - V_C(0)}{R}\, e^{-t/RC}$$

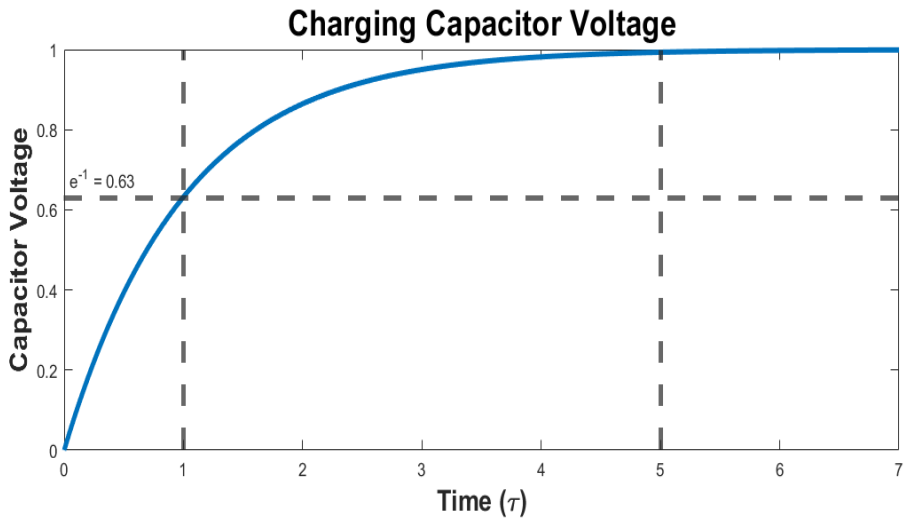
$$V_C(t) = V_C(0) + [V_0 - V_C(0)]\,\left(1 - e^{-t/RC}\right)$$

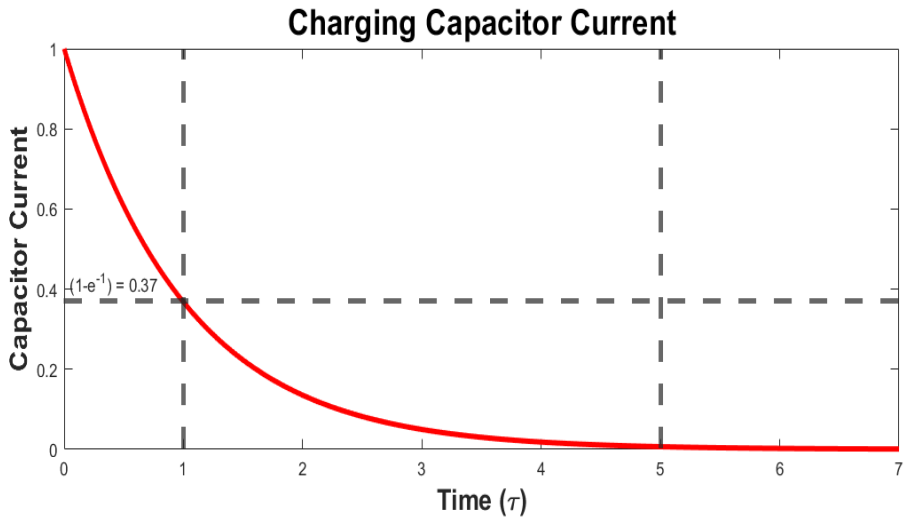
Carga: Vc(0)=0

$$V_c = V_0(1 - e^{-\frac{t}{\tau}}) \;, \qquad i(t) = \frac{V_0}{R}(1 - e^{-\frac{t}{\tau}})$$

Descarga: V_0=0
$V_c(0)=V_0$

$$V_c = V_0\, e^{-\frac{t}{\tau}} \;, \qquad i(t) = \frac{-V_0}{R}\, e^{-\frac{t}{\tau}}$$





Scripts de Python para graficar y analizar los datos

2) Ajuste RC.py: ajusta los datos obtenidos de tau vs R → Extrae el valor de C

Spyder (Python 3.7)

Archivo Editar Buscar Código fuente

C:\Users\Charly\NubeDF\PythonProgs\Cursos\Lab

Derivadas y rho de T.py | dataSaver

```python
import statsmodel
import os
from IPython import get_ipython

#%%
#selecciono si las figuras aparecen en la terminal (inline) o en ventana emergente (qt5)
#get_ipython().run_line_magic('matplotlib', 'inline')
get_ipython().run_line_magic('matplotlib', 'qt5')

#Elijo el directorio donde se encuentran los archivos que voy a analizar
os.chdir r'C:\Users\Charly\NubeDF\Arduino\Data\Transitorios')

#Leo el archivo txt tiene que estar en la misma carpeta que el programa

file = 'SalidaRCtransitorio.txt'
#en el archivo que leo las columnas están separadas por coma, la primera fila tiene el título enta
Misdatos1=[]
Misdatos = np.loadtxt(file1, delimiter=",",skiprows=1)
#print(Misdatos1)
#Ordeno los datos con la tensión de entrada
Misdatos1=Misdatos[np.argsort(Misdatos[:, 2])]

"""
#leer los datos desde una archivo csv delimitado por tabs
print("nombre del archivo completo con terminación .csv incluida")
file = input()
Misdatos1=np.genfromtxt(file,delimiter='\t',skip_header=1)
"""

#Datos ordenados col0 : Vfuente, col1: Vresistencia, Col2. Corriente en mA
y0= Misdatos1[:,0] #valores de tau
errory0 = Misdatos1[:,1] #incertidumbre de tau
x0= Misdatos1[:,2] #valores de r
errorx0=Misdatos1[:,3]/10 #valores de incertidumbre de r

plt.ion()
```

LSP Python: listo

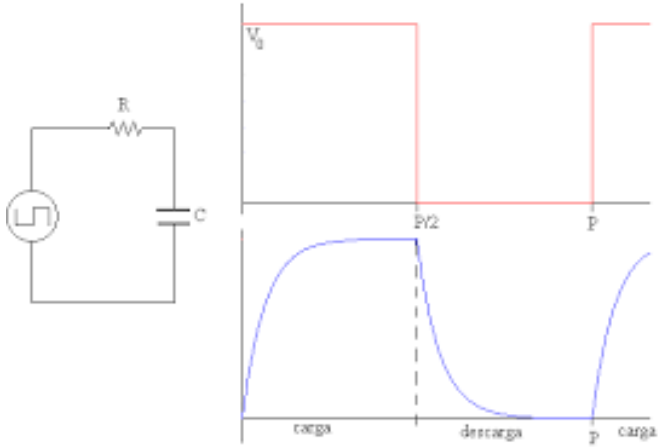
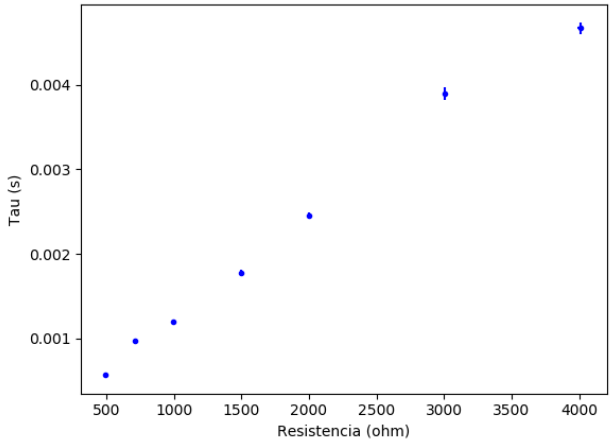
Escribe aquí para buscar

\NubeDF\Arduino\Data\Transitorios

C:\Users\Charly\NubeDF\PythonProgs\Cursos\Lab3\Ajuste RC.py

Derivadas y rho de T.py | dataSaver.py | dataPlotter.py | Ploteando_varias_figs.py | FeTeSe_filtrados.py | RCtransitorio_Ch.py | Ajuste RC.py

```python
plt.xlabel('Resistencia (ohm)');
plt.ylabel('Tau (s)');
plt.show()


print('Chequeamos que el comportamiento de Tau vs R sea lineal antes de


#Ajuste por cuadrados minimos ponderado con incertidumbres en y
w=1/errory0
X = sm.add_constant(x0)
wls_model = sm.WLS(y0,X, weights=w)
results = wls_model.fit()
o,C=results.params
#intervalo de confianza para ordenada al origen y pendiente
oint,Cint=results.conf_int(alpha=0.05)
deltaC=abs((Cint[1]-Cint[0])/2)
deltao= abs((oint[1]-oint[0])/2)

print("C=(", C," +/- ",deltaC,") F")
print("ordenada al origen=(", o," +/- ",deltao,") V")

#calculo las bandas de confianza y predicción
from statsmodels.stats.outliers_influence import summary_table
from statsmodels.sandbox.regression.predstd import wls_prediction_std

st, data, ss2 = summary_table(results, alpha=0.05)
```

Transitorios Eléctricos: puntos de control

1. Trabajar sistemáticamente evitando cometer errores en las conexiones
2. Monitorear que la señal digital de 5 V dé una señal cuadrada “limpia” (sin saltos ligados al ruido y abrupta en sus rampas ascendente y descendente)
3. Obtener ciclos de carga y descarga COMPLETOS → ajustar “espera” adecuadamente
4. Lograr ajustes adecuados de Vc(t) e i(t) de los cuales extraigan el tiempo característico para cada R (y C=1 μF)
5. Comprobar la linealidad entre τ y R → verificar la dependencia de τ con R y C → determinar C
6. Verificar las incertezas consideradas en todos los casos