dx.doi.org/10.24254/CNIB.18.42

Instbio: librería de Python para simular circuitos electrónicos de instrumentos biomédicos

R. M. Higuera Gonzalez[1,*], A. Martínez García[1], V. López Castillo[1]
[1] Departamento de Ingeniería Biomédica y Electrónica, Tecnológico de Estudios Superiores de Ixtapaluca, Edo. de México
* m.sc.hgrm@ieee.org

Resumen— En el siguiente artículo se presenta el desarrollo e implementación de la librería *Instbio* para observar la respuesta de los circuitos electrónicos que componen instrumentos biomédicos. Los resultados obtenidos se compararon con resultados experimentales, los cuales comprueban que el uso de la librería es eficiente para simular los circuitos electrónicos y se encuentran dentro del rango esperado debido a la tolerancia de los resistores utilizados.

Palabras clave—**Phyton, ECG, EMG, Simulador**

I. Introducción

Para realizar el diseño de los circuitos electrónicos de los instrumentos biomédicos, se requiere simular cada uno de los circuitos que integra el instrumento. Una de las desventajas que esto implica, es que las licencias de los diferentes simuladores son costosas y las versiones de prueba se ofrecen por un período de tiempo limitado o bajo algunas restricciones. De estos simuladores, los más comunes son *Proteus* y *Orcad*. En el caso de la licencia de prueba de *Proteus*, ésta permite simular los circuitos pero no permite guardar los archivos. Otra desventaja es que no cuenta con el amplificador de instrumentación INA128/129, que es generalmente utilizado para la etapa de preamplificación de los aparatos biomédicos. La desventaja del software libre de *Orcad* (*Orcad Lite*) es que solo permite simular un cierto número de nodos y si el circuito se extiende, ya no es posible continuar con la simulación. Además, la licencia de *Proteus* ronda los $6,59200 [1] y la de *Orcad* $6,800.00 [2]. Esto es una desventaja para los estudiantes de ingeniería, ya que difícilmente cuentan con los recursos para adquirir la licencia de alguno de estos dos softwares.

En este artículo se describe como se realizó la librería *Instbio* en *Python*, la cual ayuda a observar la respuesta de cada una de las etapas de instrumentos biomédicos tales como: electrocardiógrafo, electromiógrafo y electroencefalógrafo. Esta librería les ayuda a los alumnos a observar la respuesta de los circuitos antes de probarlos en protoboard. También se puede observar qué tanto varía la señal de salida tomando en cuenta la tolerancia de los resistores.

II. Metodología

El desarrollo de la librería *Instbio* se realizó utilizando la distribuidora *Anaconda* y el editor *Spyder,* el cual está pensado para facilitar el uso de *Python* a científicos e ingenieros [3]. *Instbio* puede mostrar la respuesta de las diferentes etapas de los circuitos que componen los aparatos biomédicos tales como los electrocardiógrafos, electromiógrafos y electroencefalógrafos. Las etapas principales de estos aparatos son: preamplificación, filtrado y amplificación final. El primer paso consiste en obtener cada uno de los modelos matemáticos de las diferentes etapas.

Preamplificación: Debido a que las señales biomédicas tienen amplitudes muy pequeñas en el orden de mV o hasta los µV, estas se deben de amplificar. En esta etapa se utiliza un amplificador de instrumentación debido a su alta impedancia de entrada y a que reduce el ruido en modo común [4]. En diferentes trabajos [5-7] el amplificador de instrumentación utilizado fue el INA128 (Fig. 1). La ganancia está definida por:

$$G = 1 + \frac{50k\Omega}{RG} \tag{1}$$

Y el voltaje de salida está definido por:

$$Vout = 1 + \frac{50k\Omega}{RG}\,(Vin^{-} - Vin^{+}) \tag{2}$$

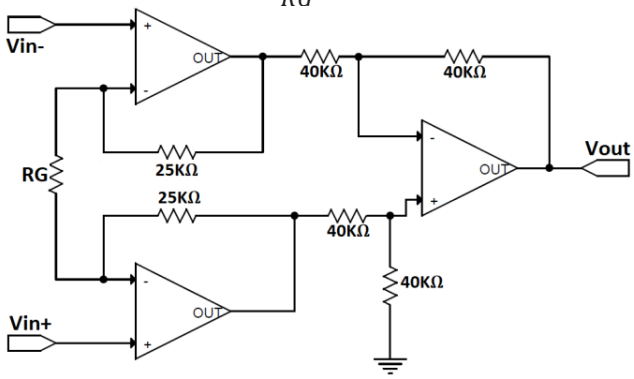

Fig. 1. Diagrama interno del amplificador de instrumentación INA128.

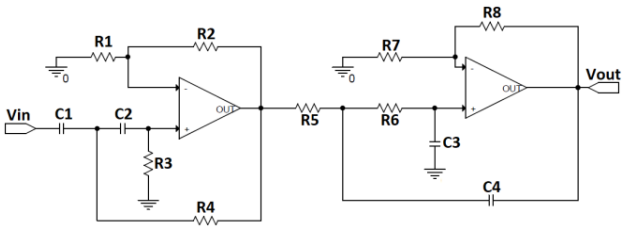

Fig. 2. Diagrama del filtro pasa-banda de cuarto orden.

Filtrado: debido a que las señales tienen una amplitud pequeña, son susceptibles al ruido, por lo que deben de pasar por una etapa de filtrado. Debido a que las señales biomédicas tienen un ancho de banda ya predefinido, se debe utilizar un filtro pasa-banda (Fig. 2) para eliminar las señales que se encuentren fuera del ancho de banda. La función de transferencia de un filtro pasa-banda de cuarto orden se define multiplicando la función de transferencia del filtro pasa-alta de segundo orden y el filtro pasa-baja de segundo orden [8].

$$H_{PB} = \left(\frac{K_1 s^2}{s^2 + \left(\frac{\omega_{01}^2}{Q}\right)s + \omega_0^2}\right)\left(\frac{K_2 \omega_{02}^2}{s^2 + \left(\frac{\omega_{02}^2}{Q}\right)s + \omega_{02}^2}\right) \quad (3)$$

.

Las señales biomédicas también se ven afectadas por el ruido producido por la toma de corriente (60Hz) y, debido a que esta frecuencia se encuentra dentro del ancho de banda de las señales biomédicas, se debe implementar un filtro Notch (Fig. 3).

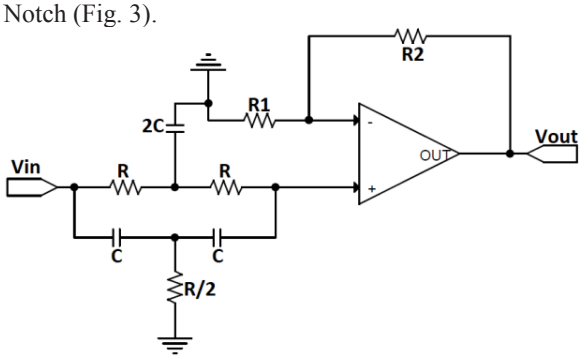

Fig. 3. Filtro Notch de segundo orden.

La función de transferencia del Filtro Notch está definida por [8]:

$$H_{RB} = \frac{K(\omega_0^2 + s^2)}{s^2 + \left(\frac{\omega_0^2}{Q}\right)s + \omega_0^2} \quad (4)$$

Finalmente, se amplifica la señal filtrada. Para esto se utiliza un amplificador no inversor mostrado por Fig. 4.

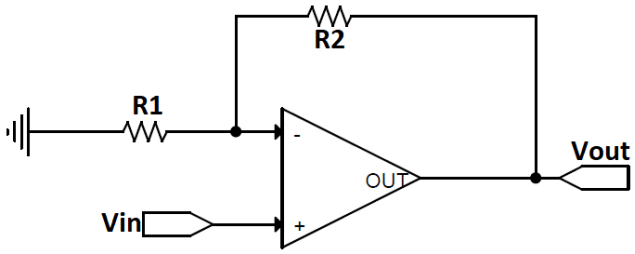

Fig. 4. Amplificador no inversor.

El voltaje de salida está dado por:

$$V_{out} = \left(1 + \frac{R2}{R1}\right)V_{in} \quad (5)$$

Después de obtener cada uno de los modelos matemáticos de los circuitos, se empezó a realizar cada una de las funciones de los circuitos en *Python*. Además se realizaron funciones para filtros pasa-baja de primer y segundo orden, pasa-altas de primero y segundo orden, pasa-banda de segundo orden y amplificador inversor. Por último se probó cada una de las funciones de la librería *Instbio*.

III. RESULTADOS Y DISCUSIÓN

Primero se debe de incluir la librería *Instbio* en la dirección C:\Program Files\Anaconda3\Lib, después se debe importar en el archivo donde se va a trabajar. En la Fig. 5 se muestra como se importa la librería al archivo, para utilizarla se le da un sobrenombre pequeño en este caso bio.

```
7 import numpy as np
8 import matplotlib.pyplot as plt
9 import Instbio as bio
```

Fig. 5. Importación de la librería *Instbio* al archivo de *Python*.

Para probar la parte de la etapa de preamplificación se utilizaron dos señales de ECG las cuales fueron obtenidas del banco de señales physiobank. Ambas señales tienen una amplitud de 1mV y la segunda señal esta invertida en comparación con la primera señal. Los demás parámetros son: RG=5.6kΩ, Vdd=10V y Vee=-10V. Para obtener el voltaje de salida se debe de declarar la función correspondiente a la etapa de preamplificación en este caso bio.INA128a (RG, Vin1, Vin2, Vdd, Vee). En la Fig. 5 se puede observar como varía la señal de salida cuando se tiene en cuenta la tolerancia de 5% del resistor RG.

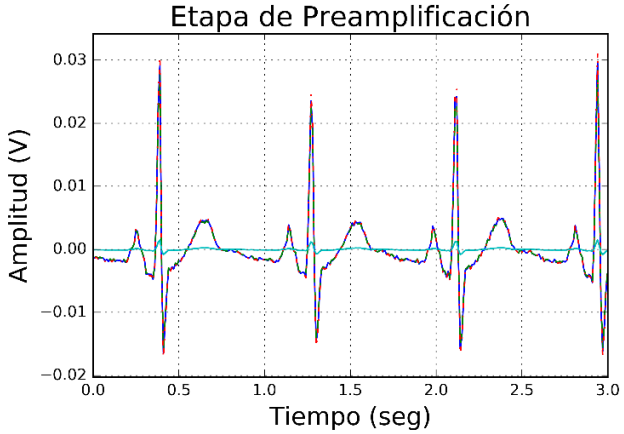

Fig.5. Respuesta de la etapa de preamplificación.

Ya que se comprobó que funcionaba la librería intsbio, se obtuvieron datos experimentales del voltaje de salida (Vout) de la etapa de preamplificación, para variar este voltaje se utilizó un potenciómetro (RG) de 5kΩ. Como voltaje de entrada se utilizaron ondas senoidales con amplitud de 78 mV y una frecuencia de 60Hz. Los valores obtenidos de la amplitud de Vout se compararon con los obtenidos del software Orcad lite y la librería Instbio (Fig.6).

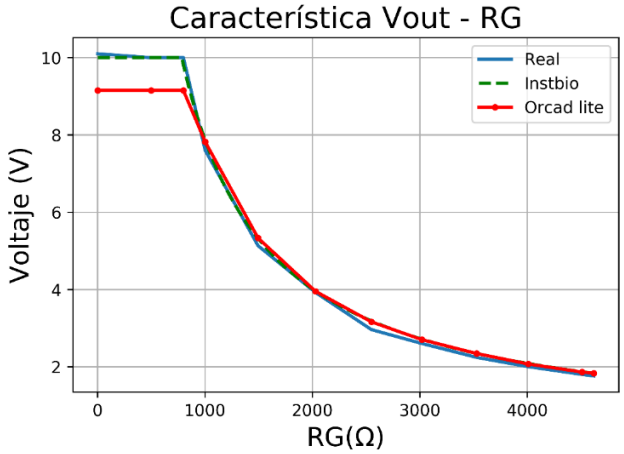

Fig. 6. Característica Vout-RG.

En la gráfica de la Fig. 6 se muestra que para valores pequeños de RG el amplificador de instrumentación se encuentra saturado por lo cual la amplitud del voltaje de salida es igual al voltaje de alimentación (10V), en la gráfica se puede observar que la librería creada se aproxima más a los datos experimentales que la simulación con Orcad lite. A medida que incrementa el valor de RG la ganancia disminuye por lo cual la amplitud del voltaje de salida disminuye, en esta parte la diferencia máxima entre los datos experimentales y la librería instbio es del 6.32%.

Después de verificar que la función de preamplificación funcionaba correctamente se realizó una prueba a la etapa de filtrado. En este caso se diseñó un filtro pasa-banda para un electrocardiógrafo cuyo ancho de banda es de 0.1Hz -100Hz, los valores de los componentes del filtro pasa-banda se muestran en la tabla 1.

Tabla 1. Parámetros introducidos al filtro pasa-banda de cuarto orden.

Componente	Valor
R1	1kΩ
R2	560Ω
R3	160kΩ
R4	160kΩ
C1	10µF
C2	10µF
R7	1kΩ
R8	560Ω
R5	16kΩ
R6	16kΩ
C3	0.1µF
C4	0.1µF

Para simular el filtro pasa-banda se utiliza la función: bio.fpba4(R1,R2,R7,R8,R5,R6,R3,R4,C3,C4,C1,C2,Vin,fs). La Fig.7 muestra la respuesta en frecuencia del filtro pasa-banda. Se utilizó una escala semilog para observar mejor la respuesta del filtro.

En la Fig. 8 se muestra la respuesta del filtro pasa-baja de segunda, cuya frecuencia de corte es de 100Hz. Para simular el filtro pasa-baja se necesita utilizar la función: bio.fpb2(R7,R8,R5,R6,C3,C4,Vin,fs).

En la Fig. 9 se muestra la respuesta del filtro pasa-alta de segunda orden, cuya frecuencia de corte es de 0.1Hz. Para simular el filtro pasa-alta se necesita utilizar la función: bio.fpa2(R1,R2,R3,R4,C1,C2,Vin,fs).

En la Fig. 10 se muestra la respuesta del filtro notch de segundo orden, cuya frecuencia de corte es de 60Hz. Para simular el filtro notch se necesita utilizar la función: bio.notch(R1,R2,R,C,Vin,fs). Los valores de los componentes del filtro notch se muestran en la tabla 2.

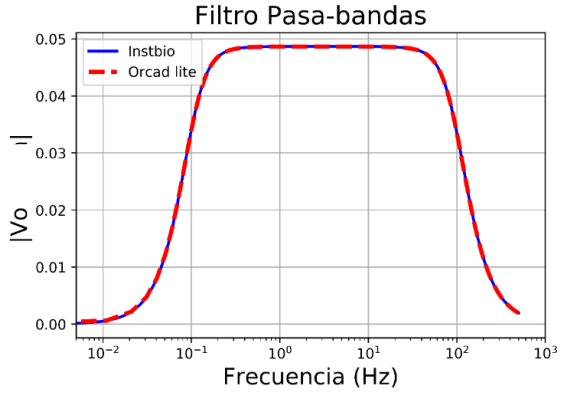

Fig.7 Respuesta en frecuencia del filtro pasa-banda de cuarto orden.

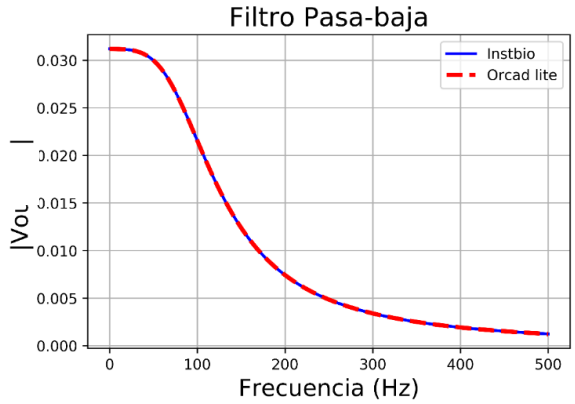

Fig.8. Respuesta del filtro pasa-baja de segundo orden.

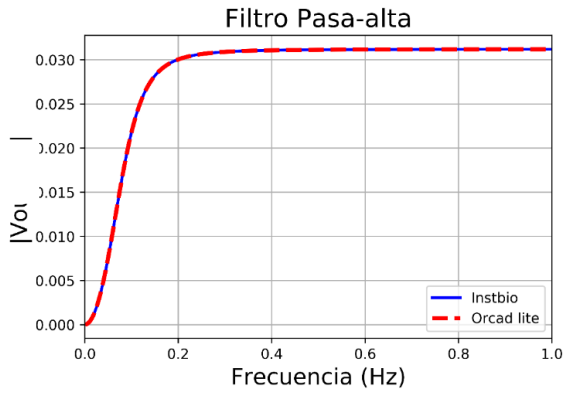

Fig.9. Respuesta del filtro pasa-alta de segundo orden.

Tabla 2. Parámetros introducidos al filtro notch de segundo orden.

Componente	Valor
R1	1kΩ
R2	290Ω
R	2.65kΩ
C	1 µF

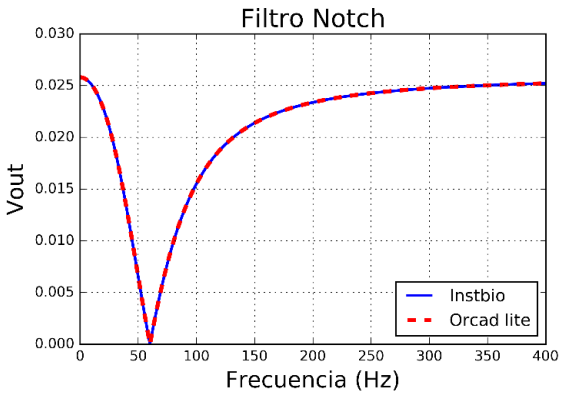

Fig.10. Respuesta del filtro notch de segundo orden.

V. Conclusión

Observando los resultados reportados en este artículo se demuestra que la librería de uso libre *Instbio* nos sirve para simular cualquiera de los circuitos electrónicos tales como amplificadores, filtros pasa-banda, filtros pasa-baja, filtros pasa-altas y filtros notch. Las gráficas obtenidas demuestran que los resultados son congruentes con lo obtenido del simulador Orcad lite.

Como trabajo futuro se tiene pensado realizar una GUI de distribución libre para facilitar el uso a los usuarios. Otra mejora que se pretende incluir es que mediante redes neuronales la misma librería te recomiende los valores comerciales de resistores y capacitores para una frecuencia de corte o una ganancia dada.

BIBLIOGRAFÍA

[1] V. Rossano. "Simulación de circuitos electrónicos". 1ª ed. RedUsers: Buenos Aires, 2013.
[2] I. Pérez. "Introducción a OrCAD 10.0". España, Universidad Carlos III de Madrid.
[3] M. Summerfield. "Rapid GUI Programming with Python and Qt: the definitive guide to PyQt programming", Prentice Hall, 2008.
[4] Kitchin, C., & Counts, L. "A designer's guide to instrumentation amplifiers". Analog Devices, 2004.
[5] F. J. R. Miranda, "Diseño y construcción de un electrocardiógrafo de doce derivaciones y detector de pulsos cardiacos con visualización de trazos en pc y dispositivo móvil vía bluetooth", 2016.
[6] G. E. V. Picón, "Diseño y construcción de un electrocardiógrafo de 12 derivaciones para el análisis de señales cardiacas", 2012.
[7] M. Dai, X. Xiao, X. Chen, H. Lin, W. Wu, and S. Chen, "A lowpower and miniaturized electrocardiograph data collection system with smart textile electrodes for monitoring of cardiac function", Australasian physical & engineering sciences in medicine, vol. 39, no. 4, pp. 1029–1040, 2016.
[8] Sedra, A. S., & Smith, K. C. "Circuitos microelectrónicos", 2006.
[9] Rashid, M. H. "Circuitos microelectrónicos: análisis y diseño". International Thomson, 2000.