EMULADOR HÍBRIDO DE MEMRESISTOR CON APLICACIÓN EN CIRCUITOS CAÓTICOS

HYBRID MEMRISTOR EMULATOR WITH APPLICATION IN CHAOTIC CIRCUITS

Eduardo Elpidio Rodríguez Martínez
Instituto Potosino de Investigación Científica y Tecnológica, México
eduardo.rodriguez@ipicyt.edu.mx

Juan Gonzalo Barajas Ramírez
Instituto Potosino de Investigación Científica y Tecnológica, México
jgbarajas@ipicyt.edu.mx

Recepción: 24/noviembre/2023 **Aceptación:** 26/diciembre/2023

Resumen

En los últimos años, el memresistor ha generado un creciente interés en diversas áreas de investigación. Sin embargo, hasta ahora, los memresistores no están fácilmente accesibles. En este artículo, se propone el diseño de un emulador de memresistor que emplea una interfaz analógica-digital, un microcontrolador y una fuente de poder controlada digitalmente. Se implementaron memresistores de tipo genérico en dos circuitos caóticos, y se muestran resultados numéricos que reflejan su comportamiento caótico. La metodología incluye el desarrollo de la interfaz analógica-digital, el uso del microcontrolador y el control digital de la fuente de poder. Estos resultados preliminares proporcionan una alternativa accesible y versátil para la emulación de memresistores y su implementación en circuitos caóticos, ya que ofrece la ventaja de emular diversos modelos memresistivos y ajustar sus características con relativa facilidad en comparación con los emuladores analógicos y digitales convencionales.

Palabras Clave: Circuitos caóticos, Emulación de memresistores, Memresistor, Sistemas memresistivos.

Abstract

In recent years, the memresistor has generated increasing interest in various research areas. However, until now, memresistors are not easily accessible. In this

article, the design of a memresistor emulator that uses an analog-digital interface, a microcontroller and a digitally controlled power supply is proposed. Generic type memresistors were implemented in two chaotic circuits, and numerical results reflecting their chaotic behavior are shown. The methodology includes the development of the analog-digital interface, the use of the microcontroller and the digital control of the power source. These preliminary results provide an accessible and versatile solution for the emulation of memresistors and their implementation in chaotic circuits, since it offers the advantage of emulating various memresistive models and tuning their characteristics with relative ease compared to conventional analog and digital emulators.

Keywords**:** *Chaotic circuits, Memristor, Memristor emulation.*

1. Introducción

El memresistor es un dispositivo electrónico de dos terminales descrito por una relación constitutiva no lineal entre la carga y el flujo magnético. El memresistor puede considerarse como una resistencia variable en el tiempo cuyo valor de resistencia depende de la historia de la corriente que ha pasado a través de él, es decir, su resistencia cambia como resultado de la integral de corriente que ha pasado a través del dispositivo. Adicionalmente cuando la corriente que fluye a través de él es cero, la suma de la corriente se vuelve constante y, por lo tanto, la resistencia permanece sin cambios [Chua, 1971]. Desafortunadamente, una realización experimental del memresistor no se pudo obtener hasta el año 2008, en que investigadores de los Laboratorios HP fabricaron el primer memresistor basado en nano estructuras metal-aislante-metal de $\mathrm{Ti}O_2$ dopado [Strukov, 2008]. En general, los memresistores se describen como elementos no lineales lo que los vuelve una buena opción para diseñar circuitos capaces de mostrar dinámicas caóticas. Uno de los primeros trabajos de circuitos caóticos que incluyen memresistores fue propuesto por Chua e Itho en 2008 [Itho, 2008], donde mostraron mediante simulación numérica una galería de osciladores caóticos obtenidos a partir del circuito de Chua, substituyendo el elemento no lineal por un memresistor con relación constitutiva lineal por partes (PWL). Por otro lado, en [Muthuswamy, 2010]

se presenta la implementación de un circuito caótico con un memresistor, basado en el circuito de Chua. Así mismo, Muthuswamy también mostró que un circuito caótico puede ser obtenido con solo tres elementos: un capacitor, un inductor y sistema memresistivo activo (una generalización del memresistor) [Muthuswamy, 2010]. Sin embargo, hasta el momento, los memresistores no están disponibles fácilmente, por lo tanto, la mayoría de las implementaciones experimentales con memresistores en circuitos caóticos se han apoyado de circuitos emuladores, los cuales imitan el comportamiento y las propiedades del memresistor [Khalil, 2019], [Zhang, 2013]. La implementación de estos emuladores generalmente se compone de dispositivos analógicos, empleando circuitos integradores y multiplicadores. Sin embargo, estos circuitos emuladores presentan algunas limitaciones. Entre ellas se encuentra el rango de frecuencias de operación, que puede estar limitado. Además, su implementación requiere un número significativo de componentes y en caso de implementar un modelo de memresistor diferente, será necesario diseñar un nuevo circuito de emulación, lo que conlleva una falta de flexibilidad. Otra forma de implementar los memresistores es mediante emuladores digitales, que, a diferencia de los analógicos, el modelo del memresistor se programa en un microcontrolador el cual calcula el valor de resistencia adecuado dada una entrada de corriente o voltaje. Este tipo de emulador ofrece la capacidad de emular modelos de memresistor más complejos y diversos, por lo que se han utilizado en la implementación de circuitos neuromórficos [Pershin, 2010], [Zhao, 2018]. Sin embargo, algunas de las limitaciones que presentan es que la discretización, conversión, la perdida de resolución y la velocidad de muestreo afectan el desempeño del emulador.

Por otro lado, los emuladores híbridos de memresistor son aquellos que se componen de circuitos analógicos y dispositivos digitales unificados, es decir, tienen una interfaz analógica-digital, un microcontrolador para programar el modelo que determina el comportamiento del memresistor y fuente de voltaje o corriente que controla su salida mediante el microcontrolador. En comparación con los emuladores digitales y analógicos, el uso de una fuente controlada permite una mejor frecuencia de operación máxima ya que la fuente controlada puede ser mucho

más flexible y del mismo modo se conservan las ventajas de usar un microcontrolador [Biolek, 2014].

En este trabajo se presenta el diseño de un emulador de memresistor híbrido basado en el método presentado por [Kolka, 2011]. Este enfoque implica el control digital de una fuente de corriente o voltaje mediante el uso de un microcontrolador. Para la realización de este diseño, se empleó un Arduino Mega256 para programar el modelo del memresistor y las ecuaciones correspondientes al circuito caótico propuesto. La elección de este tipo de microcontrolador se basa en su bajo costo y facilidad de uso, lo que lo convierte en una opción accesible y adecuada para su implementación en circuitos caóticos. Los emuladores desarrollados se implementaron en dos circuitos caóticos, y se presentan los resultados de los atractores caóticos obtenidos a través de simulaciones numéricas y experimentales utilizando la plataforma de simulación Proteus.

2. Métodos

El enfoque de diseño de emuladores híbridos parte de la definición de sistemas memresistivos (SMR) controlados por corriente (CC) y controlados por voltaje (CV). Los sistemas memresistivos controlados por corriente están descritos por las ecuaciones 1 y 2.

$$v = R_M(x, i, t)i \tag{1}$$

$$\dot{x} = f_i(x, i, t) \tag{2}$$

Por otro lado, los sistemas memresistivos controlados por voltaje se definen mediante las ecuaciones 3 y 4.

$$i = G_M(x, v, t)v \tag{3}$$

$$\dot{x} = f_v(x, v, t) \tag{4}$$

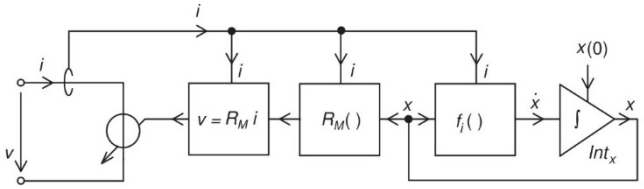
Las ecuaciones 1 y 3 son ecuaciones algebraicas que describen el voltaje o corriente del memresistor, mientras que 2 y 4 son ecuaciones de estado que contienen un vector de estados $x = [x_1, x_2, \dots, x_n]$ y su primera derivada con respecto al tiempo. R_M y G_M representan la memeresistencia y memductancia respectivamente. Para modelar estos sistemas a través de fuentes controladas por

voltaje o corriente, las ecuaciones 1 y 3 deben transformarse en otra ecuación, que calcula el voltaje o corriente del puerto analógico, es decir, el voltaje o corriente en las terminales del emulador, de modo que se denotarán como “VPE” para un sistema controlado por voltaje, ecuación 5 e “IPE” para un sistema controlado por corriente, ecuación 6.

$$v(t) = R_M(t)i(t) \tag{5}$$

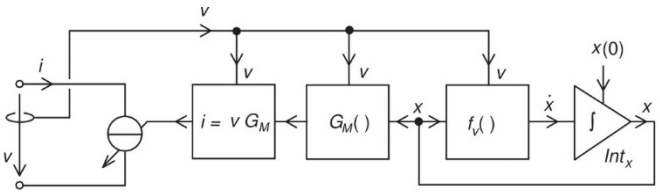
$$i(t) = G_M(t)v(t) \tag{6}$$

Las figuras 1 y 2 muestran los diagramas para emular sistemas memresistivos controlados por corriente y voltaje, respectivamente.



Fuente: elaboración propia.

Figura 1 Diagrama del método emulación.



Fuente: elaboración propia.

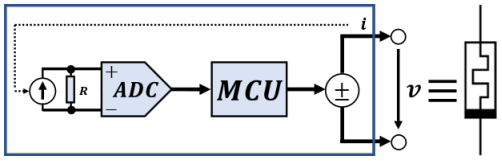
Figura 2 Diagrama del método de emulación.

Estos esquemas se basan en la idea de que una resistencia, simbolizando al memresistor, puede ser implementada mediante una fuente de voltaje o corriente, controlada en función del voltaje o corriente del puerto analógico (VPE o IPE). Por ejemplo, en el caso de un sistema memresistivo controlado por corriente (CCSMr), se implementa mediante una fuente de voltaje y su valor de voltaje es calculado a través de VPE. Este cálculo requiere la detección de la corriente i que fluye a través del puerto memresistivo y el conocimiento del valor real de la memresistencia R_M. La memresistencia se calcula en el bloque $R_M(\cdot)$ mediante la ecuación algebraica que describe la corriente i y el vector de estado x. El estado se determina mediante

la ecuación de estado, específicamente a través de la integración de la salida $\dot{x}$ del bloque etiquetado como $f_i(\cdot)$.

En el caso de un sistema memresistivo controlado por voltaje (VCSMr), la memductancia G_M (valor recíproco de la memresistencia) se regula mediante las variables de estado y el voltaje v en el puerto analógico. De esta manera, la memductancia se determina a partir del voltaje detectado en las terminales del emulador. El procedimiento para calcular el estado del sistema es idéntico al utilizado en los CCSMr.

Siguiendo el principio de emulación mencionado anteriormente, se propone un emulador de memresistor híbrido para un sistema memresistivo controlado por corriente y otro por voltaje. Este emulador consta de un microcontrolador, un convertidor analógico-digital, una fuente de voltaje o corriente y una interfaz digital-analógica, para implementarlo en circuitos caóticos. El esquema del emulador propuesto se ilustra en la figura 3. La implementación del emulador se llevará a cabo mediante una fuente de voltaje controlada digitalmente, encargada de establecer el voltaje y corriente necesario en las terminales del puerto analógico.



Fuente: elaboración propia.

Figura 3 Esquema del emulador propuesto.

De acuerdo con el esquema, i representa la corriente que circula en el circuito y R_M corresponde al valor de memresistencia. Para determinar el valor de memresistencia R_M o memductancia G_M, es necesario medir la caída de voltaje mediante el resistor R. Este valor de voltaje se procesa mediante un convertidor analógico-digital, de tal manera que el valor muestreado es procesado por un microcontrolador para obtener el estado interno del memresistor resolviendo las ecuaciones 2 y 4. Utilizando la variable de estado, se calcula el valor de memresistencia o memductancia y se obtiene el valor de voltaje de acuerdo con las ecuaciones 5 y 6.

3. Resultados

Siguiendo la metodología de diseño de emuladores de memresistores descrita en la sección anterior, se ha implementado un emulador híbrido para desarrollar dos circuitos caóticos. El primero de ellos se basa en el circuito caótico más simple propuesto por Muthuswamy, pero con la novedad de emplear un enfoque digital en el desarrollo del emulador del memresistor en lugar de uno analógico, lo cual conlleva ventajas significativas en términos de flexibilidad. La figura 4 muestra el esquema del circuito, el cual se compone de tres elementos conectados en serie: un inductor lineal pasivo, un capacitor lineal pasivo y un memresistor localmente activo. El modelo de memresistor empleado es de tipo genérico y se define por las ecuaciones 7 y 8.

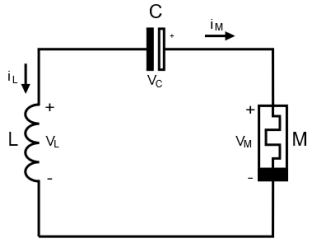
$$v = R(z)i \tag{7}$$

$$\dot{z} = f(z, i) \tag{8}$$

Donde se tienen a las ecuaciones 9 y 10.

$$R(z) = \beta(z^2 - 1) \tag{9}$$

$$f(z, i) = i - \alpha z - iz \tag{10}$$

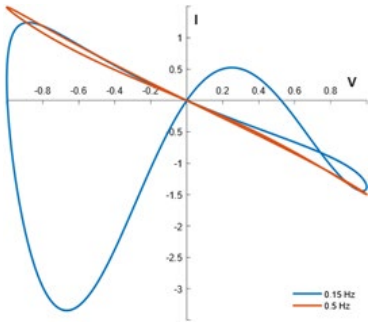


Fuente: elaboración propia.

Figura 4 Circuito caótico de tres elementos.

La ecuación 7 representa el voltaje en las terminales del memresistor y depende de la corriente y la memresistencia, ecuación 9. La ecuación 8 describe la evolución de la variable de estado del memresistor en función de la corriente i y la variable de estado, ecuación 10. Debido a que la memresistencia $R(z)$ es negativa en el intervalo $-1 < \mathrm{z} < 1$, el memresistor es localmente activo, es decir, su curva v-i se puede encontrar en el primer y tercer cuadrante como se observa en la figura 5. En la figura se muestra el lazo de histéresis pinchado del memresistor con cruce por

cero en el origen y se observa la reducción del área de los lóbulos cuando se aumenta la frecuencia de la señal de entrada $\mathrm{i(t)} = \mathrm{sin(\omega t)}$ con $\omega = 0.5$ Hz y 0.15 Hz, respectivamente. Esta característica es única de los memresistores.



Fuente: elaboración propia.

Figura 5 Lazo de histéresis pinchado del memresistor.

La ecuación 11 define la dinámica del circuito y se obtiene al establecer como variables de estado el voltaje en el capacitor, la corriente en el inductor y el estado interno del sistema memresistivo.

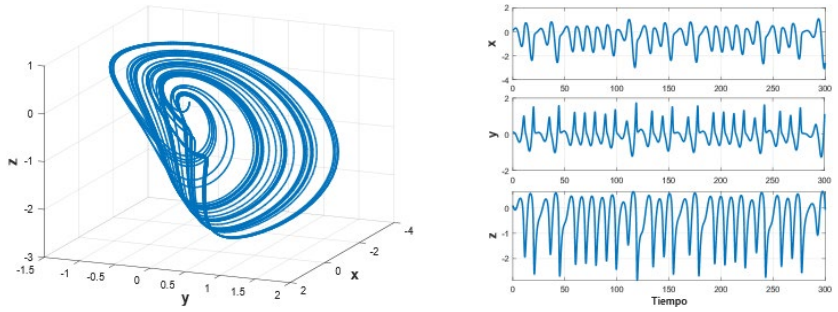
$$\begin{aligned} x(t) &\triangleq v_c(t) \\ y(t) &\triangleq i_L(t) \\ z(t) &= estado\ interno\ del\ sistema \end{aligned} \tag{11}$$

Aplicando las leyes de Kirchhoff y utilizando las relaciones constitutivas del capacitor, inductor y memresistor se obtiene la ecuación 12, que representa las ecuaciones del circuito en términos de sus variables de estado.

$$\begin{aligned} \dot{x} &= \frac{y}{C} \\ \dot{y} &= -\frac{1}{L}(x + \beta(z^2 - 1)y) \\ \dot{z} &= -y - \alpha z + yz \end{aligned} \tag{12}$$

Dados los siguientes valores de los parámetros, $C = 1$, $L = 3$, $\beta = 1.5$, y $\alpha = 0.6$. Se obtiene el atractor caótico de la figura 6.

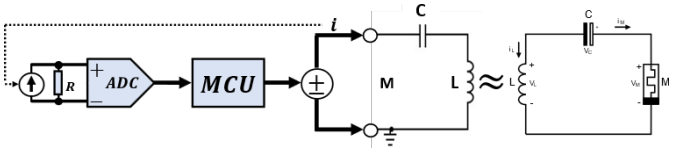
Por otro lado, en la figura 7 se presenta un diagrama del circuito caótico con el emulador propuesto, el cual ha sido implementado mediante una fuente de voltaje controlada digitalmente. Esta fuente establece el voltaje $v_M(t)$ necesario en las terminales del puerto analógico utilizando la ecuación 5.



a) Atractor. *b) Estados.*

Fuente: elaboración propia.

Figura 6 Atractor caótico y trayectorias de sus estados.



Fuente: elaboración propia.

Figura 7 Circuito caótico con un emulador de memresistor híbrido.

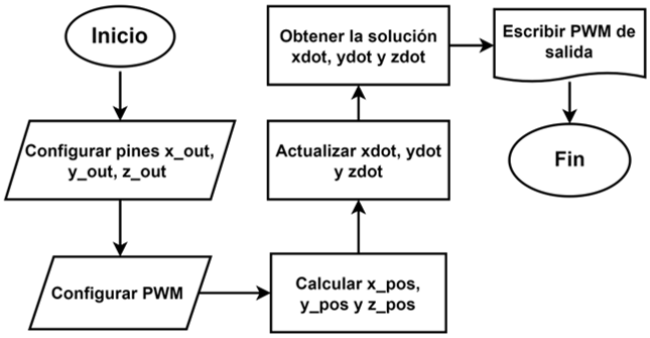
A continuación, se describe el algoritmo que debe realizar el microcontrolador para obtener el estado interno del sistema y el valor de voltaje V_M. Este proceso se lleva a cabo en tres etapas clave:

- **Adquisición de Datos:** el microcontrolador adquiere el valor de corriente i del sistema utilizando el convertidor analógico-digital (AD).
- **Resolución de la Ecuación de Estado:** se aplica el algoritmo de Euler para resolver la ecuación de estado del memresistor. El algoritmo utiliza un intervalo de muestreo (dT) para realizar los cálculos, ecuación 13.

$$z = z + ((i - \alpha\, z - iz)dT) \tag{13}$$

- **Generación de Señal de Salida:** Una vez que se ha calculado el nuevo valor de z, se utiliza para calcular el voltaje V_M mediante ecuaciones 7 y 9.

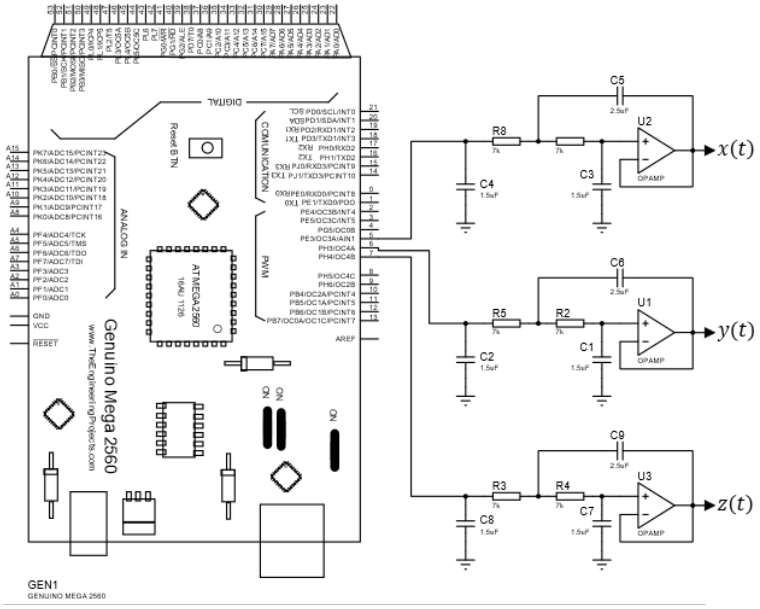
Finalmente, el microcontrolador establece el valor calculado de V_M en el convertidor digital-analógico (DA). La implementación de este sistema se realizó en el simulador de Proteus que contiene la librería para simular el Arduino Mega 256. A través de los puertos PWM del Arduino se transmitieron las variables de estado $\mathrm{x(t)}$, $\mathrm{y(t)}$ y $z(t)$. En la figura 8, se muestra un diagrama de flujo del código de Arduino.



Fuente: elaboración propia.

Figura 8 Diagrama de flujo del programa del Arduino.

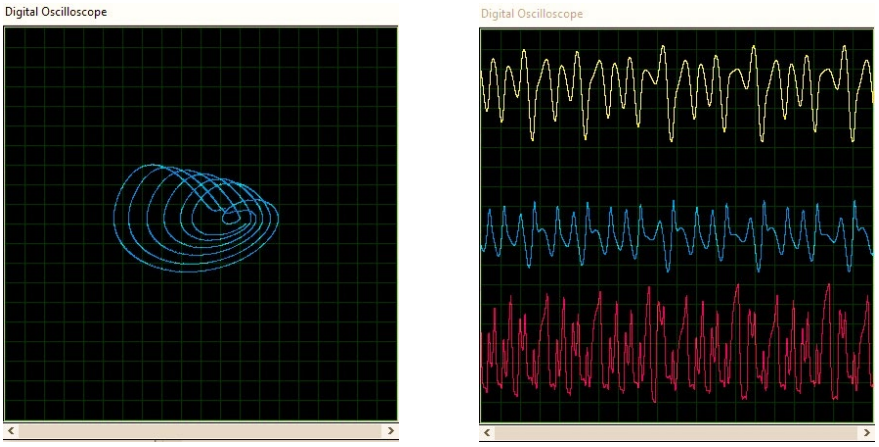
El diagrama comienza con la configuración de las salidas PWM (x_out, y_out y z_out) y entra en un bucle principal. Dentro de este bucle, el código calcula las derivadas de las variables $x(t)$, $y(t)$, así como la variable de estado $z(t)$ del memresistor, las actualiza con un paso de tiempo (dT), y calcula las posiciones x_pos, y_pos y z_pos. Estos valores de posición se escriben en las salidas PWM correspondientes utilizando la función analogWrite(). Para esta implementación, la fuente de voltaje controlada proviene del mismo Arduino y suministra el voltaje necesario para mantener el circuito oscilando. Además, se aplicó un filtro de paso bajo para eliminar el ruido de las señales de salida, figura 9.



Fuente: elaboración propia.

Figura 9 Esquema del circuito en Proteus.

En la figura 10, se muestra el atractor del sistema, así como las trayectorias de las variables de estado. Se observa que algunas de las trayectorias se ven afectadas por el rendimiento del microcontrolador.

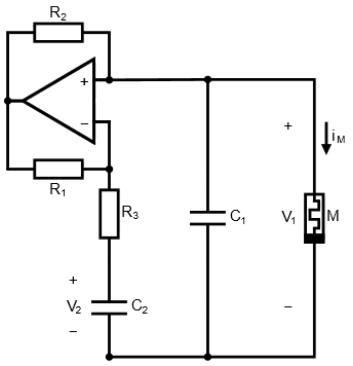


a) Atractor. *b) Trayectoria de los estados.*

Fuente: elaboración propia.

Figura 10 Atractor caótico y sus trayectorias obtenidas en Proteus.

Por otro lado, la segunda implementación se basa en el circuito memresistivo de Wien Bridge [Xu, 2019]. En la figura 11, se presenta el esquema del circuito, el cual consta de tres resistores, dos capacitores y un memresistor pasivo. En esta implementación, se empleó un memresistor genérico controlado por voltaje, cuyo modelo se describe mediante las ecuaciones 14, 15, 16 y 17.



Fuente: elaboración propia.

Figura 11 Circuito memresistivo de Wien-bridge.

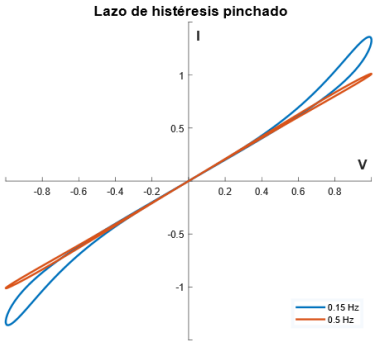
$$i = G(z)v \tag{14}$$

$$\dot{z} = f(z, v) \tag{15}$$

$$G(z) = \gamma(z^2 + \beta) \tag{16}$$

$$f(z, v) = -v - hz + v^2 z \tag{17}$$

En este modelo de memresistor el signo antes del parámetro β en la ecuación 16 denota su polaridad. Si el signo es positivo, se trata de un elemento pasivo; de lo contrario, será activo. La ecuación 17 describe el comportamiento de la variable de estado en función del voltaje. En la figura 12, se presenta el lazo de histéresis pinchado, que en este caso se encuentra en el segundo y cuarto cuadrante, lo que indica que se trata de un memresistor pasivo.



Fuente: elaboración propia.

Figura 12 Lazo de histéresis pinchado del memresistor.

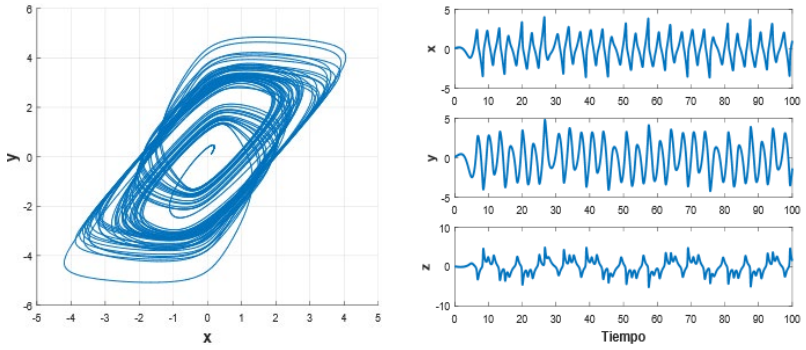
Aplicando las leyes de Kirchhoff y la relación entre el voltaje y la corriente de los elementos, se obtiene la ecuación 18, que define la dinámica del circuito.

$$\begin{gathered} C_1 \frac{dv_1}{dt} = \frac{(R_1/R_2)v_1 - v_2}{R_3} - (1 + z^2)v_1 \\ C_2 \frac{dv_2}{dt} = \frac{(R_1/R_2)v_1 - v_2}{R_3} \\ \frac{dz}{dt} = -v_1 - hz + {v_1}^2 z \end{gathered} \tag{18}$$

Realizando los siguientes cambios de variables: $v_1 = x$, $v_2 = y$, $R_1/R_2R_3C_1 = R_1/R_2R_3C_2 = a$, $1/R_3C_1 = 1/R_3C_1 = b$ y $1/C_1 = 1$, se obtiene la ecuación 19.

$$\begin{gathered} \dot{x} = (ax - by) - (1 - cz^2)x \\ \dot{y} = ax - by \\ \dot{z} = -x - hz + x^2 z \end{gathered} \tag{19}$$

Al emplear los valores $a = 3.05$, $b = 1$, $c = 0.5$ y $h = 2$, se genera el atractor caótico que se muestra en la figura 13, obtenido a partir de la ecuación 19.



a) Atractor. *b) Estados.*

Fuente: elaboración propia.

Figura 13 Atractor caótico y trayectorias de sus estados.

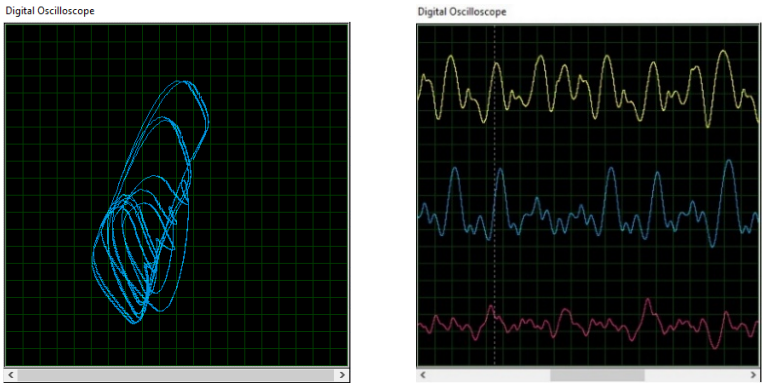
Utilizando la metodología empleada en el circuito anterior, para este caso se estableció un valor de corriente controlado digitalmente $i_m(t)$ en las terminales del puerto analógico. Para calcular el valor de memductancia será necesario medir el voltaje v del puerto analógico a través de un convertidor analógico digital. Con el valor muestreado, el microcontrolador calculará la corriente i_m utilizando la ecuación 6 y la variable de estado del memresistor. A continuación, se describe el algoritmo que debe realizar el microcontrolador para obtener el estado interno del sistema y el valor de corriente i_M. Este proceso se lleva a cabo en tres etapas clave:

- **Adquisición de datos:** En el primer paso, el microcontrolador adquiere el valor de voltaje v del sistema utilizando el convertidor analógico-digital (AD).
- **Resolución de la ecuación de estado:** A continuación, se aplica el algoritmo de Euler para resolver la ecuación de estado del memresistor. El algoritmo utiliza un intervalo de muestreo (dT) para realizar los cálculos, como se observa en la ecuación 20.

$$z = z + \left((-v + hz + v^2 + z)dT\right) \tag{20}$$

- **Generación de la señal de salida:** Una vez que se ha calculado el nuevo valor de z, se utiliza para calcular el valor de corriente i_M mediante las ecuaciones 14 y 16. Finalmente, se obtiene el valor de corriente i_m y el

microcontrolador lo establece en la fuente con apoyo de un convertidor digital-analógico. A través de los puertos PWM del Arduino, se transmitieron las variables de estado $\mathrm{x(t)}$, $\mathrm{y(t)}$ y $z(t)$. El código de Arduino se implementó de forma muy simular al diagrama de flujo presentado en la figura 8. La figura 14 se muestra el atractor caótico del circuito, así como las trayectorias de las variables de estado obtenidas en Proteus.



a) Atractor. *b) Trayectoria de los estados.*

Fuente: elaboración propia.

Figura 14 Atractor caótico y sus trayectorias obtenidas en Proteus.

4. Discusión

Los emuladores son circuitos electrónicos que permiten replicar el comportamiento de los memresistores. Estos emuladores se pueden clasificar en tres grupos principales: analógicos, digitales e híbridos. Los emuladores analógicos se dividen en dos tipos: directos e indirectos. El emulador de tipo indirecto utiliza un resistor no lineal y un mutador, que transforma la característica $v - i$ no lineal del resistor al dominio $q - \varphi$, mientras que el de tipo directo implementa directamente las ecuaciones que caracterizan al memresistor mediante una configuración de circuito adecuada.

A diferencia de los emuladores analógicos, los emuladores digitales utilizan un microcontrolador para implementar el modelo del memresistor. Esto ofrece la posibilidad de modificar el modelo a emular a través del microcontrolador y adaptarlo a diferentes configuraciones y características específicas. Por lo tanto, el uso de un microcontrolador facilita la implementación del modelo en comparación con un

emulador completamente analógico, que puede volverse más complicado de construir dependiendo del grado de complejidad del modelo a emular. En [Pershin, 2010] se presenta un emulador digital compuesto por un potenciómetro digital, un convertidor analógico a digital ADC y un microcontrolador.

Sin embargo, una de las desventajas de utilizar componentes digitales es que la discretización, conversión, la perdida de resolución y la velocidad de muestreo afectan considerablemente la frecuencia de trabajo del emulador. En contraste, el emulador de memresistor propuesto en este trabajo, no sufre de estas limitaciones ya que su estructura híbrida combina circuitos analógicos y dispositivos digitales en una configuración unificada. Además, este enfoque de emulación no requiere de un potenciómetro digital, ya que el puerto analógico del emulador se implementa mediante una fuente de alimentación.

Es importante destacar que en este trabajo se utilizó la fuente de voltaje proporcionada por defecto en la tarjeta Arduino como la fuente de alimentación principal. Se determinó que este voltaje predeterminado cumple con los requisitos específicos de la aplicación, proporcionando la energía necesaria para el funcionamiento del emulador.

Por otro lado, en 1985 Leon O. Chua estableció las condiciones para que un circuito presente comportamiento caótico [Chua, 1985], entre las cuales se encuentran:

- Contener al menos un elemento no lineal.
- Contener al menos un elemento activo.

El elemento activo proporciona la energía necesaria para que el circuito oscile de manera autónoma. En este sentido, el uso de una fuente de voltaje proporciona la energía necesaria para mantener el comportamiento caótico, sin la necesidad de utilizar circuitos adicionales para generar energía.

5. Conclusiones

El memresistor fue definido por Leon O. Chua en 1971 como el cuarto elemento fundamental de los circuitos, relacionando el flujo magnético y la carga eléctrica. A medida que se ha estudiado su comportamiento se ha descubierto que está

presente en sistemas naturales y artificiales, lo que indica que existía mucho antes de su definición. Debido a sus propiedades no lineales, se ha propuesto como un elemento que proporciona la no linealidad necesaria para generar caos. Se ha utilizado, por ejemplo, para reemplazar un elemento en el circuito de Chua, un circuito autónomo con comportamiento caótico.

En este trabajó se presentó el diseño de un emulador de memresistor híbrido basado en un método de emulación que consiste en controlar digitalmente una fuente de voltaje mediante un controlador, con su implementación en circuitos caóticos. El emulador propuesto se compone de un microcontrolador que estable el voltaje o corriente deseados en las terminales del dispositivo, un convertidor analógico-digital y una fuente de voltaje con una interfaz digital-analógica.

Se emularon dos tipos de memresistor en este trabajo. El primer modelo de memresistor emulado es de tipo genérico y localmente activo. El segundo modelo también es de tipo genérico y puede cambiar su polaridad, de modo que se puede comportar como un elemento pasivo o activo. Ambos tipos de memresistor tienen comportamiento no lineal que proporciona las características necesarias para generar caos. Además, se mostró la implementación en Proteus de dos circuitos caóticos. En ambos casos, se utilizó el algoritmo de Euler para obtener la versión discretizada de los sistemas caóticos. Este algoritmo utiliza menos operaciones matemáticas y es de fácil implementación, lo que lo hace apto para plataformas de acceso libre como Arduino. El uso del microcontrolador permite modificar el modelo y características del memresistor sin necesidad de realizar cambios significativos en la estructura del emulador.

6. Bibliografía y Referencias

[1] Biolek, D. Memristor Emulators. In: Adamatzky, A., Chua, L. (eds). Memristor Networks. Springer, Cham, 2014.

[2] Chua, L. O. Memristor-the missing circuit element. IEEE Trans. Circuit Theory, No. 5, Vol. 18, 507-519, 1971.

[3] Chua, L.O. The Genesis of Chua's circuit. Archiv fur Elektronik und Ubertragungstechnik, No.4, Vol. 46. 250-257, 1992.

[4] Itho, M., & Chua, L. O. Memristor oscillators. International Journal of Bifurcation and Chaos, No. 11, Vol. 18, 3183-3206, 2008.

[5] Khalil, N. A., & Said, L. A., & Radwan, A. G. et al. A Simple BJT Inverse Memristor Emulator and Its Application in Chaotic Oscillators. 2019 Fourth International Conference on Advances in Computational Tools for Engineering Applications (ACTEA), Beirut, Lebanon, 1-4, 2019.

[6] Kolka, Z., & Biolek, D., & Biolková, V. Hybrid modelling and emulation of mem-systems. International Journal of Numerical Modelling: Electronic Networks, Devices and Fields, Vol. 25, 216-225, 2012.

[7] Muthuswamy, B. Implementing memresistor based chaotic circuits. International Journal Bifurcation and Chaos, No. 5, Vol. 20, 1335-1350, 2010.

[8] Muthuswamy, B., & Chua, L. O. Simplest chaotic circuit. International Journal of Bifurcation and Chaos, No. 5, Vol. 20, 1567-1580, 2010.

[9] Pershin, Y. V., & Di Ventra, M. Experimental demonstration of associative memory with memristive neural networks. Neural Networks, No. 7, Vol. 23, 881-886, 2010.

[10] Strukov, D., & Snider, G., & Stewart, D. et al. The missing memresistor found. Nature, Vol. 453, 80-83, 2008.

[11] Xu, B., & Wang, G., & Wang, X. et al. A third-order memristive Wien-bridge circuit and its integrable deformation. Pramana - J Phys 93, 42, 2019.

[12] Zhang, X., & Yu, Y., & Zhang, W. et al. A simple memristor emulator. Proceedings of the 32nd Chinese Control Conference, Xi'an, China, 8718-8721, 2013.

[13] Zhao, L., & Hong, Q., & Wang, X. Novel designs of spiking neuron circuit and STDP learning circuit based on memristor. Neurocomputing, Vol. 314, 207-214, 2018.