DISEÑO COMPORTAMENTAL DE NEURONAS DE IMPULSO EN HARDWARE RECONFIGURABLE BASADAS EN EL MODELO DE IZHIKEVICH

BEHAVIORAL DESIGN OF SPIKING NEURONS IN RECONFIGURABLE HARDWARE BASED ON IZHIKEVICH MODEL

Isaac Macias Mendoza
Universidad de Guadalajara, México
Isaacm.macias@alumnos.udg.mx

Juan José Raygoza Panduro
Universidad de Guadalajara, México
juan.rpanduro@academicos.udg.mx

Edwin Christian Becerra Alvares
Universidad de Guadalajara, México
Edwin.becerra@academicos.udg.mx

Mario Jiménez Rodríguez
Universidad de Guadalajara, México
mario.jimenez@academicos.udg.mx

José Luis González Vidal
Universidad Autónoma del Estado de Hidalgo, México
jlvidal@uaeh.edu.mx

José Roberto Reyes Barón
Centro de Enseñanza Técnica Industrial CETIs, México
jrreyes@ceti.mx

Recepción: 7/noviembre/2021 **Aceptación:** 7/marzo/2022

Resumen

Los modelos matemáticos de neuronas han logrado ser computacionalmente eficientes y biológicamente plausibles, como es el modelo de Izhikevich. La implementación de neuronas artificiales en hardware se ha extendido en áreas como: control, robótica, entre otros. Sin embargo, migrar un modelo a hardware en un sistema reconfigurable se complica debido a la alta ocupación de recursos lógicos. En este trabajo se presenta el diseño de dos neuronas de impulsos que replican el comportamiento del modelo de Izhikevich y la propuesta de modificación

de estas para una funcionalidad más amplia. Se realizan simulando el modelo como sistema dinámico en software con un tiempo de 100 ms, para obtener los valores de iteraciones en los que se generan impulsos. Son implementadas en lenguaje de descripción de hardware utilizando máquinas de estado. Como resultados se obtienen neuronas que ocupan 10 y 13 Flip-Flops de 1536 disponibles en una FPGA Spartan 3.

Palabras Clave: FPGA's, modelo de Izhikevich, neurona de impulso.

Abstract

Mathematical models of neurons have managed to be computationally efficient and biologically plausible like Izhikevich's model. The implementation of artificial neurons in hardware has spread in areas such as: control, robotics, among others. However, migrating a model to hardware in a reconfigurable system is complicated due to the high occupation of logical resources. This work presents the design of two spiking neurons that replicate the behavior of the Izhikevich model and the proposal to modify these for a broader functionality. They are carried out by simulating the model as a dynamic system in software with a time of 100 ms, to obtain the values of iterations in which spikes are generated. They are implemented in hardware description language using state machines. The results are neurons that occupy 10 and 13 Flip Flops out of 1536 available in a Spartan 3 FPGA.

Keywords: *FPGA's, Izhikevich model, Spiking neurons.*

1. Introducción

El uso de neuronas artificiales en hardware inició en 1951 por Marvin Minsky quien construyó la primera neuro-computadora denominada SNARC (Stochastic Neural Analog Reinforcement Calculator) [Kelemen, 2007]. Esto permitió avances en la implementación de redes neuronales artificiales (RNAs) mediante hardware, los cuales están ligados al desarrollo de la tecnología microelectrónica de alta escala de integración (VLSI) y de los arreglos de compuertas lógicas programables en campo (FPGAs). Los constantes desarrollos facilitaron la importación de algunos modelos de RNAs en hardware, extendiendo su aplicación en áreas como:

computación neuro-mórfica [Kousanakis, 2017], control [Pearson, 2007], robótica [Rocke, 2007], reconocimiento de patrones [Rice, 2009], identificación de caracteres codificados [Lammie, 2018], decodificación en radio [Sandoval, 2017], hardware neuro-mórfico de baja potencia, así como algoritmos híbridos de redes neuronales para mejora de rendimiento [Diehl, 2016], [Han, 2020]. Eugene M. Izhikevich propuso un modelo matemático de neuronas de impulso, el cual es biológicamente plausible y computacionalmente eficiente, sin embargo, al igual que otros modelos matemáticos de neuronas que comparten características, importarlo en hardware se complica debido a la alta ocupación de recursos lógicos que requieren.

El propósito del trabajo es el diseño comportamental de dos neuronas de impulsos con variantes en la salida, en función del tipo de entrada, basadas en el modelo simple de neuronas de Izhikevich, teniendo como objetivo facilitar la implementación de modelos matemáticos de neuronas en hardware.

2. Métodos

Modelo simple de neuronas de impulso de Izhikevich

El modelo simple de Izhikevich es biológicamente plausible, como el modelo Hodgkin-Huxley y computacionalmente eficiente como el modelo Leaky Integrate and Fire [Izhikevich, 2003]. Quiere decir que es capaz de emular el funcionamiento de una neurona biológica real y a la vez reduce la complejidad computacional. En la figura 1 se muestran las propiedades neuro-computacionales de neuronas de impulso biológicas que este modelo puede exhibir.

Al poseer estas características y ventajas como la capacidad de aproximarse a los comportamientos medidos en neuronas biológicas se convierte en un modelo de interés científico. Está representado por dos ecuaciones diferenciales ordinarias que corresponden al potencial de membrana ecuación 1 y a la variable de recuperación ecuación 2, además de una condición de reinicio ecuación 3 que ocurre cuando el impulso es generado respectivamente.

$$v' = 0.04v^2 + 5v + 140 \pm u + I \tag{1}$$

$$u' = a(bv - u) \tag{2}$$

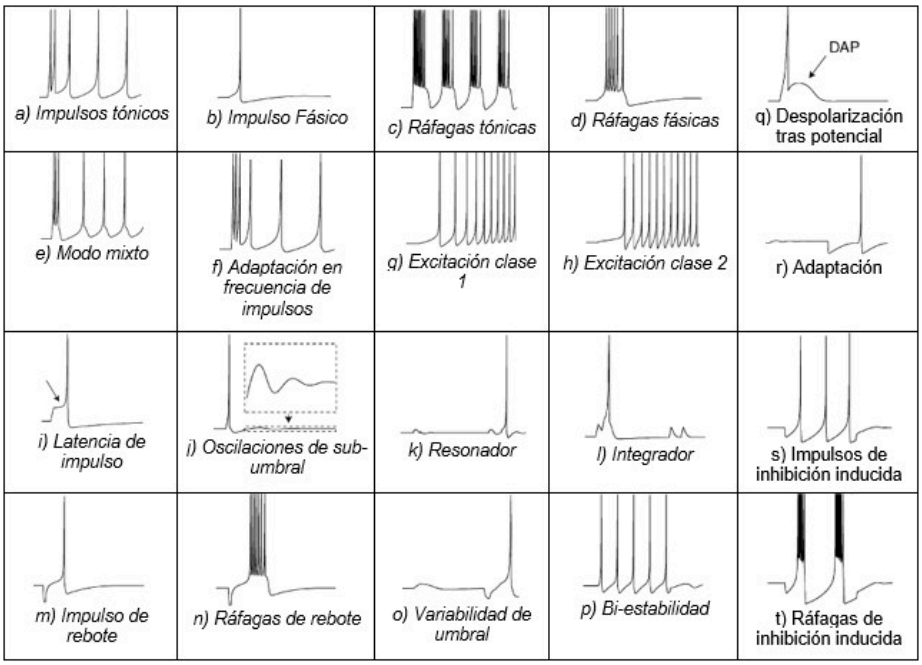

Figura 1 Propiedades neuro-computacionales de neuronas [Izhikevich, 2004].

El reinicio después de que el impulso se genera es representado por la ecuación 3.

$$si\ v \geq 30mV,\ entonces \begin{cases} v \leftarrow c \\ u \leftarrow u + d \end{cases} \quad (3)$$

Donde, v y u son variables adimensionales, mientras que a, b , c y d son parámetros adimensionales, $' = d/dt$ corresponde a la derivada con respecto al tiempo. A continuación, se describe cada elemento del modelo:

- v : Representa el potencial de membrana de la neurona.
- u : Representa una variable de recuperación de la membrana.
- a : Describe la escala de tiempo de la variable de recuperación.
- b : Describe la sensibilidad de la variable de recuperación a las fluctuaciones subumbrales del potencial de membrana.
- c : Describe el valor de reinicio posterior al impulso del potencial de membrana.
- d : Describe el reinicio posterior al pico de la variable de recuperación u.

Los coeficientes de la ecuación 1, fueron elegidos para que el potencial de membrana v tuviera escala de mV y escala de tiempo en ms. Se observa que la condición en la ecuación 3, donde el potencial de membrana se compara con 30

mV, hace referencia al pico del impulso y no al umbral ya que el valor del umbral de la neurona se encuentra entre −70 mV y −50 mV, este es dinámico como en las neuronas biológicas. El modelo es capaz de exhibir todos los patrones de disparo conocidos de las neuronas corticales, sólo con la elección de los parámetros a, b, c, d. En la figura 2 se muestra de forma visual a v y u, así como las variables a, b , c y d. [Izhikevich, 2003].

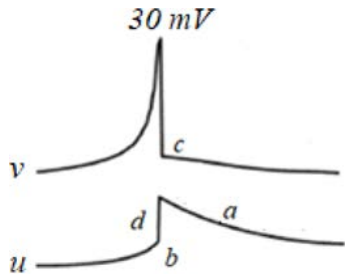

Figura 2 Potencial de membrana y variable de recuperación [Izhikevich, 2004].

Comportamientos del modelo

Para el diseño de las neuronas se seleccionaron dos propiedades del modelo, las cuales corresponden a impulsos tónicos e impulso fásico. En el comportamiento de impulso fásico, la neurona dispara un sólo impulso cuando es estimulada y posteriormente permanece en reposo, aunque el estímulo de entrada aun esté activo. Esto se puede apreciar con mayor claridad en la figura 3.

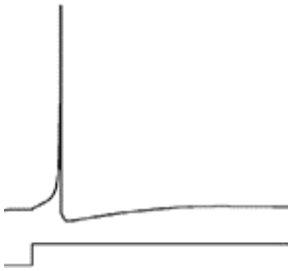

Figura 3 Impulso fásico [Izhikevich, 2004].

El comportamiento de impulsos tónicos tiene como característica permanecer disparando picos, mientras el estímulo de entrada se encuentre presente y se mantiene en reposo cuando no lo está. Como se muestra en la figura 4.

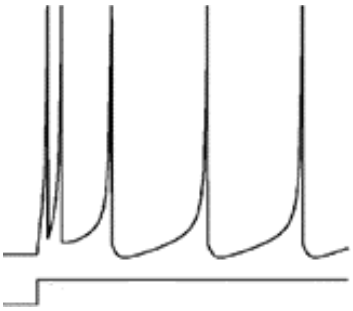

Figura 4 Impulsos tónicos [Izhikevich, 2004].

Para obtener estos comportamientos con el modelo de Izhikevich, las variables a, b, c y d deben de tomar los valores de la tabla 1, así como los valores iniciales de voltaje y corriente de estímulo. Los datos de la tabla 1 son proporcionados por el autor del modelo en su página oficial [Izhikevich, 2021].

Tabla 1 Parámetros de comportamientos de neurona.

Variables	Impulsos tónicos	Impulso fásico
a	0.02	0.02
b	0.2	0.25
c	-65	-65
d	6	6
I	14	0.5
v	-70	-64

Modelo como sistema dinámico

Este modelo puede ser adaptado en forma de un sistema dinámico utilizando las ecuaciones 4 y 5.

$$x' = f(x) \tag{4}$$

$$x(t+\tau) = x(t) + \tau f\big(x(t)\big) \tag{5}$$

Donde

τ : Representa el paso en cada iteración.

$x(t+\tau)$: Representa el valor que adquiere la variable en la siguiente iteración.

$x(t)$: Representa el valor de la variable en la iteración actual.

$\tau f(x(t))$: Evalúa la función para después multiplicarla por el paso.

Después de sustituir en las ecuaciones 1 y 2 se obtienen las ecuaciones 6 y 7 [Salamanca, 2017].

$$v(t+\tau) = v(t) + \tau(0.04v^2 + 5v + 140 - u + I) \quad (6)$$

$$u(t+\tau) = u(t) + \tau * a\big(bv(t) - u(t)\big) \quad (7)$$

3. Resultados

Simulación en software

La simulación del modelo se realizó en Matlab. Se utilizaron los parámetros de la tabla 1 y la ecuación 6. Para cada comportamiento la simulación cuenta con un paso de 0.25 y un tiempo 100 ms, teniendo un total de 400 iteraciones. Las figuras 5 y 6 muestran el resultado de la simulación, en las cuales se observan los impulsos que se asemejan a los mostrados en las figuras 3 y 4.

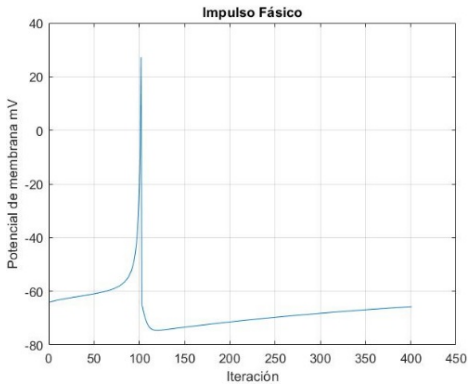

Figura 5 Resultados de simulación de impulso fásico.

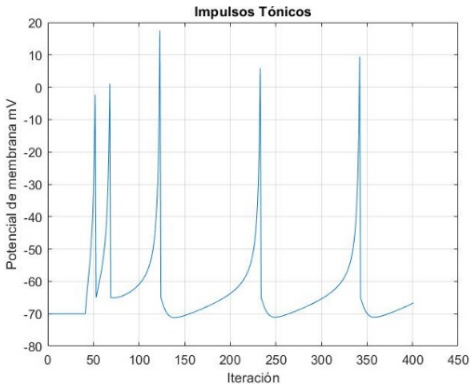

Figura 6 Resultados de la simulación de impulsos tónicos.

Las tablas 2 y 3 muestran los valores del potencial de membrana y la iteración. Se observa que el valor en el que se genera un impulso es -65 mV, debido que el reinicio de la variable es instantáneo.

Tabla 2 Iteraciones y potencial de impulsos tónicos.

Iteración	53	54	69	70	124
v	-65	-63.5672132	-65	-65.0347536	-65
Iteración	125	234	235	343	344
v	-65.9727947	-65	-65.9497911	-65	-65.9510899

Tabla 3 Iteraciones y potencial de impulso fásico.

Iteración	1	..	..	..	103	104
v	-64	…	…	…	-65	-66.56291

Diseño de neuronas en lenguaje de descripción de hardware VHDL

El diseño de las neuronas en lenguaje de descripción de hardware se realiza mediante las tablas de datos 2 y 3. Esa información es utilizada para crear un algoritmo de máquinas de estado finito, que cumple con la función de replicar de forma comportamental el modelo matemático de Izhikevich simulado en Matlab. Las figuras 7 y 8 muestran los diagramas de estados de cada comportamiento.

Cada diagrama tiene 2 entradas y 1 salida, que corresponden a x (entrada), *contador* (entrada) y y (salida), donde x representa el estímulo de la neurona, y es encargada de generar el impulso de salida y el *contador* corresponde a la iteración en que se dispara la neurona.

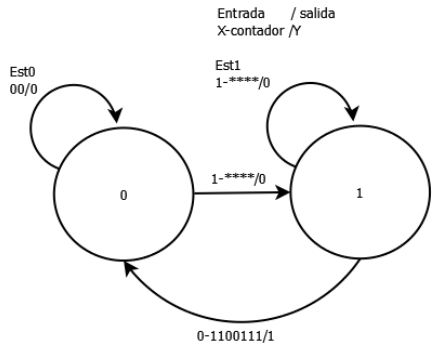

Figura 7 Diagrama de estados del comportamiento impulso fásico.

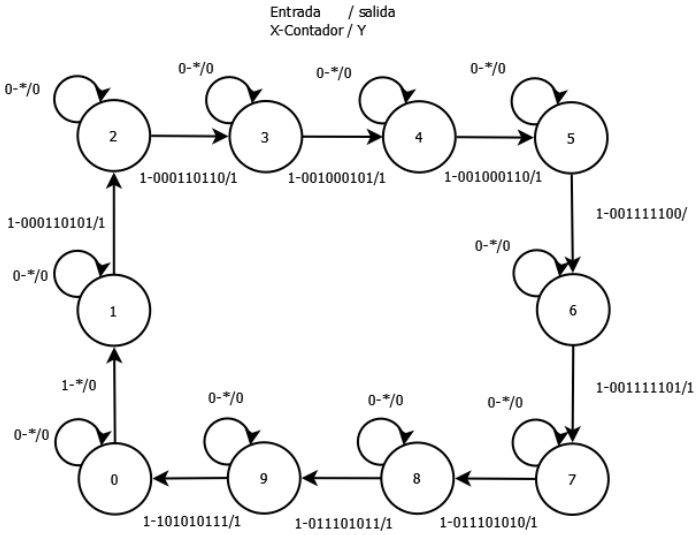

Figura 8 Diagrama de estados del comportamiento impulso tónico.

El diagrama de estados de impulso fásico está compuesto de dos estados. Para que exista un cambio de estado es necesario que se aplique un estímulo a la entrada x y permanezca activo, el símbolo "*" indica que no interesa el valor que tenga el *contador* en ese momento, es decir, no importará hasta llegar al estado 1, cuando esto ocurre, si tiene el valor 103 (en binario) el siguiente ciclo de reloj regresará al estado 0 con la salida *y* activa replicando el impulso de la tabla 3 y permanecerá en ese estado.

En este trabajo se implementó una variante en el comportamiento de la neurona fásica. Tiene como característica que con un solo estímulo de entrada produce un solo impulso de salida. Esto se puede observar en la figura 9 de resultados de comportamiento de las neuronas, en donde el estímulo de entrada de la neurona deberá ser de mínimo un ciclo reloj, dando como resultado la misma salida.

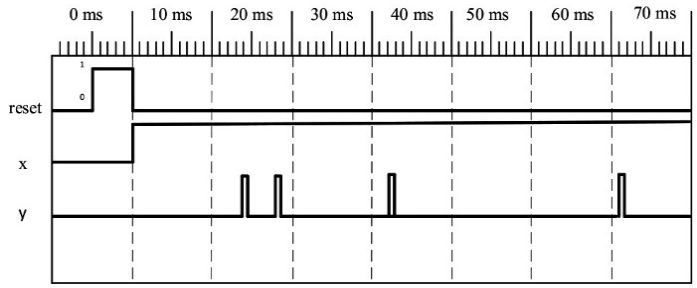

Figura 9 Resultados del test bench impulsos tónicos.

El diagrama de estados del comportamiento de impulsos tónicos está formado por 10 estados. La condición de transición de estado para que se genere un impulso de salida deberá ser que el estímulo de entrada x este activo y el *contador* genere los valores 53, 79, 124, 234 y 343 como se observa en la tabla 2. Esto se repite mientras el estímulo de entrada este activo. La variante propuesta de la neurona tónica tiene como característica generar solo una vez los impulsos de salida mencionados en el comportamiento original, pero con un estímulo de entrada. Esto se puede observar en la figura 10 de resultados de comportamiento de las neuronas, en donde el estímulo de entrada de la neurona deberá ser de mínimo un ciclo reloj.

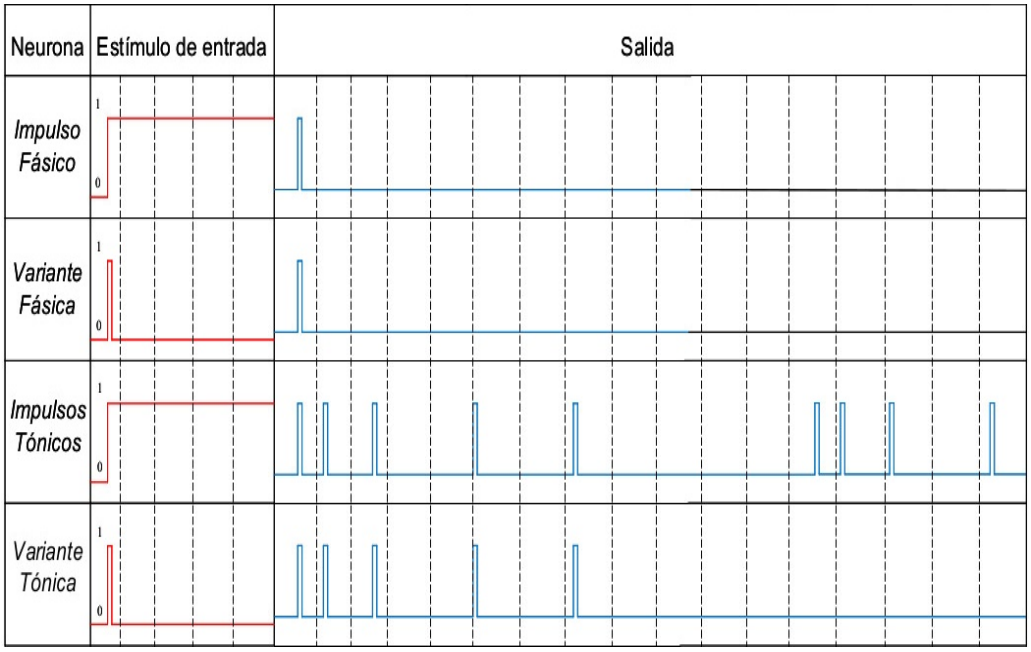

Figura 10 Resultado de las señales de entrada y respuesta de cada neurona.

En este trabajo se presenta el diseño de cuatro neuronas comportamentales en lenguaje de descripción de hardware. Por medio del software Xilinx ISE Design Suite, en una FPGA Spartan 3.

La estructura general de las neuronas diseñadas está compuesta por tres entradas (clk, $reset$ y x) y una salida (y), sin embargo, el reinicio no se considera dentro del diagrama de estados debido que la señal de reinicio es evaluada en cada ciclo de reloj. En la figura 11 se puede observar la entidad de una neurona.

```
entity neurona is
            Port ( clk : in  STD_LOGIC;
            reset : in  STD_LOGIC;
            x : in  STD_LOGIC;
            y : out  STD_LOGIC);
end neurona;
```

Figura 11 Entidad general para una neurona en lenguaje VHDL.

La arquitectura está integrada por un contador de ciclos de reloj de 9 bits en conjunto con un detector de flancos de subida que incrementa su valor en uno cada ciclo. A partir de estos elementos y utilizando los diagramas de estados, se hizo una máquina de estados donde el contador se considera como una entrada interna, la señal $reset$, clk y el estímulo x externa. El circuito de la neurona está compuesto por tres procesos: contador, lógica de salida y lógica de cambio de estado. En la figura 12 se muestra uno de los tres procesos que integran la arquitectura de la neurona, el cual corresponde al proceso de contador.

```
architecture comportamiento of neurona is
type estados is (estados_de_neuronas);
signal estado_presente, estado_futuro : estados;
signal contador : std_logic_vector (8 downto 0);
begin
process (clk,reset)
   begin
      if (reset ='1') then
            contador <= "000000000";
      elsif(clk'event and clk='1' and x = '1') then
         contador <= contador+1;
      else
      contador <= contador;
end if;
end process;
```

Figura 12 Proceso del contador descrito en lenguaje VHDL.

La transición de estado complementada por el contador está dada por las tablas 2 y 3, es decir, el contador debe generar el valor correspondiente de una iteración donde se produce un impulso. Para el caso de la neurona tónica y su variante, cuando un impulso es generado, el siguiente ciclo de reloj se desactiva la salida y espera que el contador genere el siguiente valor de un impulso. Esto es, cuando el

contador genera el valor 53 activa la salida y en el valor 54 se desactiva, repitiendo en cada impulso. En la figura 13 se muestra el bloque esquemático genérico de las neuronas implementadas.

Figura 13 Bloque esquemático genérico.

Las neuronas corresponden a impulsos tónicos y fásico con sus respectivas variantes. El Test Bench es configurado con un tiempo de 100 ms, un periodo de reloj de 0.25 ms con un ciclo de trabajo del 50% dando un tiempo en alto y bajo de 0.125 ms. La señal $reset$ y el estímulo x, se activan en estado alto (1) y se desactivan en estado bajo (0). En el caso x se configura para cada comportamiento. En la figura 14 se observa el resultado del comportamiento impulso fásico, en la parte izquierda se observan las señales de entrada y salida.

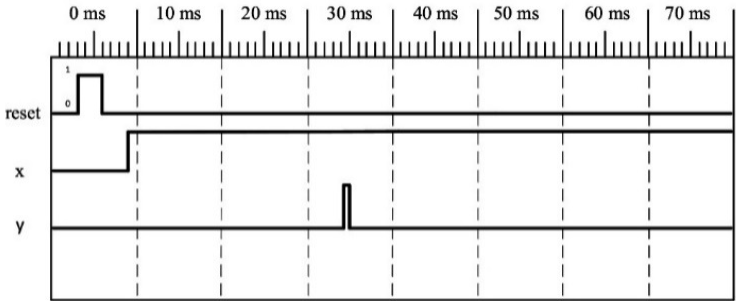

Figura 14 Resultado del test bench de impulso fásico.

La parte superior muestra la escala de tiempo en milisegundos, esto corresponde al instante de tiempo. En ella se puede ver que los primeros instantes de tiempo se reinicia la neurona, después el estímulo de entrada cambia a estado alto y se

mantiene. La señal de salida replica un impulso alrededor del instante 30 ms, esto indica que en ese momento el contador genero el valor de 103 en binario que corresponde al impulso de la tabla 3. En la figura 15 se muestra el resultado de la variante de impulso fásico. Al inicio se reinicia la neurona, después se activa el estímulo de entrada por un tiempo de 3 ms para después permanecer inactivo. En el instante de tiempo aproximado de 35 ms se genera el impulso.

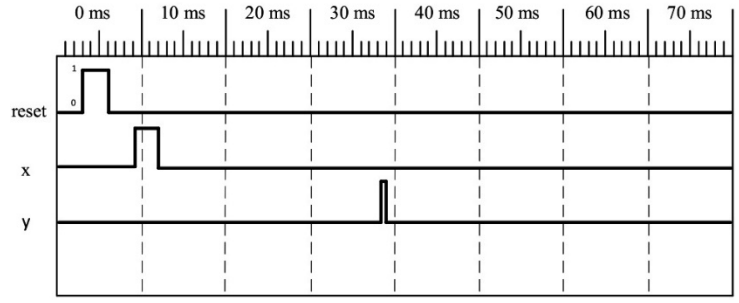

Figura 15 Resultado del test bench de la variante de impulso fásico.

En la figura 9 se observa el resultado del comportamiento impulsos tónicos. Al principio se reinicia la neurona, después se activa el estímulo de entrada y se mantiene. La salida de la neurona genera impulsos en los instantes de tiempo aproximados de 19, 24, 37 y 65 ms, esto corresponde a la iteración 53, 69, 124 y 234 de la tabla 2, el último impulso no se puede apreciar debido a el tamaño total de tiempo requerido para observar el comportamiento.

En la figura 16 se muestra la variante. De la misma manera al principio se reinicia la neurona, seguido se activa el estímulo de entrada, pero solo por unos instantes de tiempo y se desactiva. Se aprecia que de forma similar se generan los impulsos en los instantes de tiempo del comportamiento original.

En la figura 10 se observa el estímulo de entrada de cada neurona, así como la respuesta que produce cada una de ellas.

En las tablas 4, 5, 6 y 7 se muestra el resumen de ocupación de recursos lógicos que utiliza cada neurona. Para la neurona de impulso fásico y su variante, se muestra que la ocupación de Flip Flops, LUTs y Slices es el mínimo, equivalente al 1% del total que ofrece el dispositivo Spartan 3.

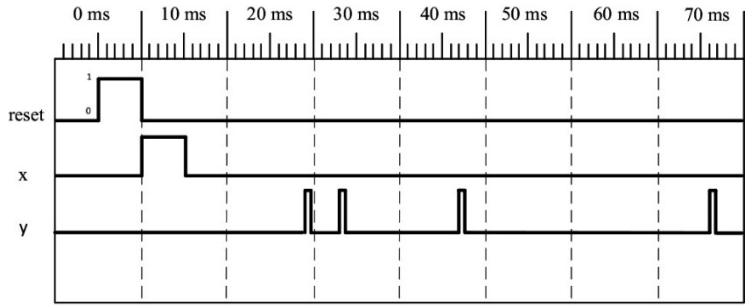

Figura 16 Resultado del test bench de la variante de impulsos tónicos.

Tabla 4 Recursos utilizados en impulso fásico.

Resumen de utilización de impulso fásico			
Utilización lógica	**Usado**	**Disponible**	**Utilización**
Número de Slice Flip Flops	10	1,536	1%
Número de LUTs de 4 entradas	6	1,536	1%
Número de Slices ocupadas	8	768	1%
Número de IOBs	4	63	6%

Tabla 5 Recursos utilizados en impulso fásico variante.

Resumen de utilización de impulso fásico variante			
Utilización lógica	**Usado**	**Disponible**	**Utilización**
Número de Slice Flip Flops	10	1,536	1%
Número de LUTs de 4 entradas	6	1,536	1%
Número de Slices ocupadas	8	768	1%
Número de IOBs	4	63	6%

Tabla 6 Recursos utilizados en impulsos tónicos.

Resumen de utilización de impulsos tónicos			
Utilización lógica	**Usado**	**Disponible**	**Utilización**
Número de Slice Flip Flops	13	1,536	1%
Número de LUTs de 4 entradas	70	1,536	4%
Número de Slices ocupadas	40	768	5%
Número de IOBs	4	63	6%

Tabla 7 Recursos utilizados en impulsos tónicos variante.

Resumen de utilización de impulsos tónicos variante			
Utilización lógica	**Usado**	**Disponible**	**Utilización**
Número de Slice Flip Flops	13	1,536	1%
Número de LUTs de 4 entradas	64	1,536	4%
Número de Slices ocupadas	37	768	4%
Número de IOBs	4		6%

Para la neurona de impulsos tónicos y su variante se muestra que la ocupación de Flip Flops, LUTs y Slices equivalen al 1%, 4% y 5%, donde su variante reduce 1% en Slices. De esta forma si se replicaran cuatro neuronas de un tipo en el mismo dispositivo tendrían una ocupación aproximada de Flip Flops del 4%.

4. Discusión

Los recientes trabajos de implementación de neuronas en hardware se pueden mencionar a [Salamanca, 2017], en la cual se propone una arquitectura global de neurona de impulso que puede seleccionar 13 comportamientos diferentes, dando como resultado un diseño con mucha funcionalidad, pero con un coste mayor en ocupación de hardware reconfigurable. En este trabajo se propuso desarrollar dos tipos de neuronas independientes reduciendo la ocupación lógica, para la implementación de redes de mayor dimensión. Además, dos modificaciones en el comportamiento de cada neurona para mejorar su funcionalidad y buscar nuevas aplicaciones.

5. Conclusiones

En este trabajo se presenta una propuesta de diseño para neuronas comportamentales que replican el funcionamiento del modelo simple de Izhikevich en lenguaje de descripción de hardware. La arquitectura de la neurona está diseñada utilizando únicamente elementos combinacionales y secuenciales, como contadores, comparadores, detector de flancos y compuertas lógicas. Se observa en las tablas de ocupación de recursos que el porcentaje se mantiene en 1% con respecto a los recursos que ofrece la FPGA Spartan 3.

Finalmente, se puede concluir que la respuesta de las neuronas y sus variantes son acordes al modelo de Izhikevich. Además, esta propuesta permite controlar el estímulo de entrada para obtener diferentes comportamientos, esto incrementa la flexibilidad para futuras aplicaciones como: circuitos criptográficos, redes neuronales, escalabilidad, optimizando el número de entradas y salidas que dispone el dispositivo.

Reconocimientos

El presente trabajo es apoyado por el Consejo Nacional de Ciencia y Tecnología (CONACYT), la Secretaría de Educación Pública (SEP) y la Universidad de Guadalajara.

6. Bibliografía y Referencias

[1] Diehl P., Zarrela G., Cassidy A., (2016). Conversion of Artificial Recurrent Neural Networks to Spiking Neural Networks for Low-power Neuromorphic Hardware arXiv:1601.04187v1 [cs.NE], 56-58.

[2] Han J., Li, Z., Zheng, W., & Zhang, Y., (2020). Hardware implementation of spiking neural networks on FPGA. Tsinghua Science and Technology, 25(4), 479–486. https://doi.org/10.26599/tst.2019.9010019.

[3] Izhikevich E., (2003). Simple model of spiking neurons. IEEE Transactions on Neural Networks, 14(6), 1569–1572. https://doi.org/10.1109 /tnn.2003.820440.

[4] Izhikevich E., (2004). Which Model to Use for Cortical Spiking Neurons? IEEE Transactions on Neural Networks, 15(5), 1063–1070. https://doi.org/10.1109/tnn.2004.832719.

[5] Izhikevich E., (2021). Eugene M. Izhikevich old (NSI) home page. Retrieved 27 May 2021, from http://www.izhikevich.org.

[6] Kelemen J., (2007). From Artificial Neural Networks to Emotion Machines with Marvin Minsky. Acta Polytechnica Hungarica, Vol. 4 (No. 4), 5–7.

[7] Kousanakis E., Dollas A., Sotiriades E., (2017). An Architecture for the Acceleration of a Hybrid Leaky Integrate and Fire SNN on the Convey HC-2ex FPGA-Based Processor. IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines, 56-63.

[8] Pearson J., Pipe G., Mitchinson B., Gurney K., Melhuish C., Gilhespy I., Nibouche M., (2007). Implementing Spiking Neural Networks for Real-Time Signal-Processing and Control Applications: A Model-Validated FPGA Approach. IEEE Transactions on Neural Networks, 18(5), 1472–1487. https://doi.org/10.1109/tnn.2007.891203.

[9] Lammie C., Hamilton T., Azghadi M., (2018). Unsupervised Character Recognition with a Simplified FPGA Neuromorphic System. IEEE International Symposium on Circuits and Systems. 1-4. doi. 10.1109/ISCAS.2018.8351532.

[10] Rice K., Bhuiyan M., Taha T., (2009). FPGA Implementation of Izhikevich Spiking Neural Networks for Character Recognition. International Conference on Reconfigurable Computing and FPGAs. 451-455.

[11] Rocke P., McGinley B., Morgan F., Maher J., (2007). Reconfigurable Hardware Evolution Platform for a Spiking Neural Network Robotics Controller, Springer-Verlag Berlin Heidelberg, 373-737.

[12] Salamanca A., (2017). Diseño e implementación de una neurona de impulsos en hardware reconfigurable [Tesis de maestría, Universidad de Guadalajara]. Repositorio institucional- Universidad de Guadalajara.

[13] Sandoval Ruiz C., (2017). VHDL Model of configurable neural networks applied to decoding in cognitive radio, Revista Ingeniería UC, Vol. 24, No. 3, 290-301.