CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS

AVANZADOS DEL INSTITUTO POLITÉCNICO NACIONAL

UNIDAD ZACATENCO

DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

SECCIÓN DE ELECTRÓNICA DEL ESTADO SÓLIDO

Prototipo en FPGA de Emulador de Red Neuronal Pulsada

T E S I S

Que presenta

ING. ERIK JONATAN MORALES DE LA ROSA

Para obtener el grado de

MAESTRO EN CIENCIAS

EN LA ESPECIALIDAD DE:

INGENIERÍA ELÉCTRICA

DIRECTORES DE TESIS:

DR. FELIPE GÓMEZ CASTAÑEDA

DR. JOSÉ ANTONIO MORENO CADENAS

Ciudad de México DICIEMBRE, 2019

“A la memoria de mi madre María del Carmen de la Rosa Machorro”

Agradecimientos

A la vida misma que a través de circunstancias y personas, que reflejadas a través de dulces y amargas experiencias han contribuido a formar mi persona trayéndome hasta aquí.

A mi familia, especialmente a:

A mi madre, de la cual tuve la dicha de conocer el más grande amor, por guiarme en la medida que alcanzaron sus fuerzas, por todas sus lecciones y por nunca dejarme desamparado. Por todos sus esfuerzos, no solo para formarme a mí, sino también a mis hermanos. Le estaré eternamente agradecido.

A mi hermana Fabi, por ser como una segunda madre para mí, por su invaluable amistad, por el sabio consejo que nunca falto desde la niñez, por siempre estar a mi lado brindándome todo el amor que cualquier ser humano podría desear, por haberme regalado inmensos momentos de alegría, y por todo el apoyo que sin escatimar siempre me das.

A mi hermana Eli, que siempre nos apoya, brindándonos siempre atención, respeto y amor, pero sobre todo por su amistad.

A mi padre y a mis hermanos, Juan y Joel, por el apoyo recibido y los buenos deseos hacia mi persona.

A mi pequeño sobrino Evan, por brindarme su cariño y respeto, por recordarme que siempre hay quien necesita un modelo a seguir.

A mis asesores

Dr. Felipe Gómez Castañeda y Dr. José Antonio Moreno Cadenas, por su guía, por la confianza, paciencia, enseñanza y por todo el apoyo brindado sin el cual no hubiera sido posible realizar este trabajo.

A mis revisores y sinodales

Dr. Alfredo Reyes Barranca y Dr. Gabriel Romero-Paredes Rubio, por su entera disposición, por su valioso consejo y por el tiempo dedicado a la revisión de este trabajo.

A los profesores de la SEES

A cada uno de ellos por la educación recibida y por la confianza que en su momento depositaron en mí y a los lazos de amistad que sin duda forme, de manera especial al Dr. Alfredo Reyes Barranca, Dr. José Antonio Moreno Cadenas, Dr. Ramón Peña Sierra, Dr. Felipe Gómez Castañeda y M. en C. Luis Martín Flores Nava.

A los auxiliares del laboratorio de VLSI

Un reconocimiento especial al M. en C. Luis Martín Flores Nava, por la disposición, el apoyo, la paciencia, y por las ideas brindadas, que enriquecieron enormemente este trabajo, al Dr. Oliverio Arellano Cárdenas y al Ing. Emilio Espinosa García, que gracias a su labor, hacen posible el quehacer diario de un científico, y en mi caso de un estudiante.

Al personal Administrativo

A todo el personal involucrado en trámites y demás requerimientos, especialmente a Monica Davar Ocegueda por su entera disponibilidad, por su amabilidad y por el empeño que pone en su trabajo, y de manera especial a Yesenia Cervantes Aguirre, que siempre nos apoya y nos da guía.

Al Consejo Nacional de Ciencia y Tecnología (CONACYT)

Por el apoyo económico que recibí durante mis estudios de posgrado, sin los cuales nada de esto hubiera sido posible.

A los amigos y compañeros

Especialmente al compañero Marco Antonio Quezada, el cual me ofreció su amistad desinteresada y sincera, con el cual compartí aula de clases y área de trabajo.

Al Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional (CINVESTAV)

Especialmente a la Sección de Electrónica del Estado Sólido (SEES), por haberme brindado la oportunidad de estudiar en este recinto y haber depositado toda su confianza en mí.

A las personas que de alguna forma u otra han tocado mi vida.

A todos ustedes. Muchas Gracias

Dedicatoria

A las mujeres que más han influido en mi vida:

A mi madre y a mis hermanas, Fabi y Eli.

Y para reconocer el trabajo de una de estas tres grandes mujeres, valiéndome de las palabras del premio Nobel de literatura de 1957 Albert Camus, y que hoy más que nunca resuenan en el fondo de mi ser, tomo prestada la voz con la que le agradeció dicho hombre a su profesor Germain por este galardón, una de las máximas figuras que tuvo en su formación, y al igual literato, con el objetivo de expresar su más profundo agradecimiento, no por los beneficios que pudo obtener de su profesor, sino por la bondad de su ser.

Hoy que no tengo las palabras suficientes para expresarte la inmensa gratitud y el cariño que siento hacia ti, parafraseo y te dedico este pensamiento.

Querida Fabi:

“Espere a que se apartara el ruido que me ha rodeado todos estos días antes de hablarle de todo corazón. He recibido un honor demasiado grande, que no he buscado ni he pedido, pero cuando supe la noticia, pensé en mi madre y en usted. Sin usted, sin la mano afectuosa que tendió al niño que era yo, sin su enseñanza y su ejemplo, no hubiera sucedido nada, pero absolutamente nada de todo esto. No es qué preste demasiada importancia a un honor de este tipo, pero ofrece la oportunidad de decirle a usted lo que ha sido y sigue siendo para mí, y de corrobórale que sus esfuerzos, su trabajo, y el corazón generoso que puso en ello, continúan siempre vivos, y que pese a los años permanecen en este niño agradecido. Te abrazo con todas mis fuerzas.”

Atentamente

Jony

Contenido

Introducción

En años recientes las redes neuronales pulsadas conocidas también como redes neuronales artificiales de tercera generación, han sido el centro de atención en todo el mundo y han dado pie a numerosas investigaciones debido a que se basan en modelos plausibles del comportamiento eléctrico de las neuronas biológicas, elementos principales en el sistema nervioso de cualquier ser vivo. Estas redes neuronales, tienen la capacidad de procesar la información en tiempo real gracias a que incorporan en sus algoritmos plasticidad sináptica dependiente del tiempo (STDP) y comportamiento neuronal “espacio-temporal”. Estos sistemas tratan de emular el comportamiento del sistema nervioso de los seres vivos, tal es el caso de los sistemas neuromórficos, los cuales se basan en el comportamiento de sus contrapartes biológicas, para lo que hacen uso de arquitecturas altamente paralelizadas con capacidad de procesamiento “espacio-temporal” y que en conjunto con la neuro-computación, han dado pie al diseño de sistemas de reconocimiento de patrones tal es el caso de retinas y cócleas artificiales.

A lo largo de tesis, se plantea la teoría necesaria para proponer un algoritmo en STDP no supervisado de reconocimiento de caracteres, que pueda ser simulado vía software y emulado vía hardware, este último, como un sistema neuromórfico digital en FPGA, capaz de procesar en tiempo real tanto la fase de entrenamiento, como la de reconocimiento de caracteres.

Resumen

Esta tesis se enfoca en el descripción de los elementos que conforman una red pulsada o de tercera generación para llevar a cabo la propuesta de un algoritmo de reconocimiento de patrones, específicamente de caracteres simples, para lo cual, se presentan una variedad modelos matemáticos que hacen posible la representación del comportamiento eléctrico de sus elementos principales (neuronas, sinapsis y plasticidad sináptica), y que permiten obtener diversos grados de plausibilidad biológica en las implementaciones que se han reportado al día hoy, tanto en software como en hardware por diversos grupos de investigación alrededor del mundo. Como sistema integrado de información, estas redes usan los modelos neuronales como elemento de computo principal, los cuales tienen la característica de producir pulsos(mayormente conocidos como “spikes”), cuyo comportamiento varia en el tiempo dependiendo del estímulo; las sinapsis como medio de comunicación y modelo de transducción de señales eléctricas entre diversas neuronas, describiendo la conversión de los pulsos generados en corrientes post sinápticas debidos a los neurotransmisores presentes en sus contrapartes biológicas, y finalmente de los modelos de plasticidad sináptica, específicamente de STDP, por sus siglas en inglés *Spike Timing Dependent Plasticity* (resultado de numerosas investigaciones en diversas especies por parte del área de las neurociencias); donde se ocupan las componentes de espacio-tiempo como principal herramienta para realizar la modificación sináptica que viene a determinar la plasticidad cerebral, que no es más que la capacidad del sistema nervioso para depreciar o potenciar las sinapsis (representadas por los pesos sinápticos) entre una neurona y otra, en base a un sistema de asociación de pulsos en tiempo y que efectúan de manera natural las redes neuronales biológicas. Sin embargo, como organizar dichos elementos y estructurar tal proceso para que STDP pueda inducir de manera autónoma el aprendizaje en la red para diversas plataformas es lo que se plantea directamente de este trabajo para que este pueda llevarse a cabo exitosamente.

Específicamente, se programa el algoritmo propuesto, haciendo uso del software Matlab 2018 para que este pueda ser simulado en cualquier computadora personal, sin embargo, implementar el algoritmo propuesto tal cual de una computadora personal a un prototipo en hardware digital, difícilmente se podría llevar a cabo dados los recursos que proporciona un circuito FGPA, por lo que a lo largo del capítulo 3, se realiza la propuesta, descripción e implementación de un sistema de reconocimiento de caracteres con arquitectura neuromórfica, con la finalidad de evitar la retención de la información de todos los componentes temporales de la red, proporcionar paralelismo y procesamiento en tiempo real durante las etapas entrenamiento o inducción de aprendizaje y reconocimiento, para lo cual se realizan adecuaciones a los procesos de integración espacio-temporal, principalmente en sinapsis y en el protocolo de asociación de pulsos en STDP; en el primero agregando los retardos inherentes a las procesos de transmisión sináptica y en el segundo proponiendo un sistema de asociación de pulsos, los cuales tienen fundamento en la teoría de las neurociencias y neuro-computación. Para realizar todo ello, se propone una arquitectura que incorpore y que sea compatible con los elementos principales de las arquitecturas neuromórficas reportadas alrededor del mundo y que en las últimas décadas han sido usadas para plantear y diseñar este tipo de sistemas, en este sentido, se presenta la teoría correspondiente a sistema de representación de eventos por sus siglas en inglés *Address Event Representation* , y la arquitectura FIFO para realizar compensaciones de tiempo entre procesos.

Este trabajo concluye presentando los resultados de las simulaciones realizadas en Matlab 2018 en redes con distintas capacidades de procesamiento, y con la simulación en ModelSim 10 del emulador de red neuronal pulsado con la arquitectura neuromórfica propuesta programada en VHDL, partiendo ambos sistemas de reconocimiento del mismo algoritmo propuesto, diferenciándose únicamente en la plataforma donde se ejecutan y por ende de la modificación de sus procesos para llevar a cabo la misma tarea. Finalmente se presentan las conclusiones, así como el trabajo futuro.

Abstract

This thesis focuses on the description of the elements that make up a pulsed or third generation network to carry out the proposal of a pattern recognition algorithm, specifically of simple characters, for which, a variety of mathematical models are presented that make possible the representation of the electrical behavior of its main elements (neurons, synapses and synaptic plasticity), and that allow obtaining different degrees of biological plausibility in the implementations that have been reported today, both in software and hardware by various research groups worldwide. As an integrated information system, these networks use neuronal models as the main computing element, which have the characteristic of producing pulses (mostly known as spikes), whose behavior varies over time depending on the stimulus; the synapses as a means of communication and model of transduction of electrical signals between different neurons, describing the conversion of the pulses generated in post-synaptic currents due to the neurotransmitters present in their biological counterparts, and finally of the synaptic plasticity models, specifically of STDP, for its acronym in english Spike Timing Dependent Plasticity (result of numerous investigations in various species by the area of neurosciences); where space-time components are used as the main tool to perform the synaptic modification that comes to determine brain plasticity, which is nothing more than the ability of the nervous system to depreciate or enhance synapses (represented by synaptic weights) between a neuron and another, based on a system of association of pulses in time and that naturally effect the biological neural networks. However, how to organize these elements and structure such a process so that STDP can autonomously induce learning in the network for various platforms is what arises directly from this work so that it can be carried out successfully.

Specifically, the proposed algorithm is programmed, making use of Matlab 2018 software so that it can be simulated on any personal computer, however,

implementing the proposed algorithm as such from a personal computer to a prototype in digital hardware, could hardly lead to given the resources provided by an FGPA circuit, so throughout chapter 3, the proposal, description and implementation of a character recognition system with neuromorphic architecture is made, in order to avoid the retention of information from all the temporary components of the network, provide real-time parallelism and processing during the training or induction stages of learning and recognition, for which adjustments are made to the processes of space-time integration, mainly in synapses and in the association protocol of pulses in STDP; in the first, adding the delays inherent to the synaptic transmission processes and in the second, proposing a pulse association system, which are based on the theory of neurosciences and neuro-computation. To do all this, an architecture is proposed that incorporates and is compatible with the main elements of the neuromorphic architectures reported around the world and that in recent decades have been used to propose and design this type of systems, in this sense, we presents the theory corresponding to the system of representation of events by its acronym in english “Address Event Representation”, and the FIFO architecture to perform time compensation between processes.

This work concludes by presenting the results of the simulations carried out in Matlab 2018 in networks with different processing capacities, and with the simulation in ModelSim 10 of the pulsed neural network emulator with the proposed neuromorphic architecture programmed in VHDL, starting both recognition systems proposed algorithm, differentiating only in the platform where they are executed and therefore the modification of their processes to carry out the same task. Finally, the conclusions are presented, as well as future work.

Objetivos

General

Implementar un prototipo funcional en FPGA programado en VHDL de un sistema de red neuronal pulsada.

Particulares

1) Estudio de los elementos necesarios en las redes neuronales tercera generación en algoritmos de entrenamiento “no supervisados” basados en STDP. para proveer de un marco teórico que permita la comprensión del comportamiento de manera general de dichas redes.
2) Basándose en el punto anterior, en base a la teoría encontrada, establecer un algoritmo para llevar a cabo entrenamiento no supervisado de STDP y reconocimiento que pueda ser implementando tanto en software como en hardware. Con esto, se pretende llegar por un lado a nivel software, a una simulación y a nivel hardware, a una emulación.
3) Implementar un sistema de reconocimiento de caracteres basado en STDP no supervisado en software con capacidad para aprender y reconocer caracteres para redes de diversas dimensiones y realizar sus respectivas simulaciones.
4) Proponer e implementar una arquitectura de hardware digital en FPGA, que incorporé las principales características de los sistemas neuromórficos digitales actuales (paralelismo en procesos, retardos, arquitecturas AER y FIFO), permitiendo realizar entrenamiento y reconocimiento de caracteres en tiempo real, en nuestro caso, para una red neuronal de 46 neuronas con el algoritmo que inicialmente fue propuesto en el punto 3.

Organización

La estructura de la tesis es la siguiente:

Capítulo 1

Se presentan los modelos neuronales, sinápticos y de plasticidad sináptica específicamente de STDP conocido por sus siglas en inglés Spike Timing Dependent Plasticity, que actualmente se implementan en software y hardware para llevar a cabo reconocimiento de patrones. Adicionalmente, se incluyen algunas de las técnicas de codificación neuronal usadas para extraer la información de cualquier red neuronal pulsada que tenga un comportamiento plausiblemente aproximado a sus contrapartes biológicas.

Capítulo 2

A partir de la teoría expuesta en el capítulo 1, se propone y se describe un algoritmo general para llevar a cabo el entrenamiento y reconocimiento de caracteres en una red neural de pulsada de “n-dimensiones” cuyo proceso de entrenamiento puede llevarse a cabo de manera auto-organizada o selectiva, para lo cual se define arquitectura y se selecciona el modelo neuronal, sináptico y de plasticidad sináptica, así como método de decodificación empleado para extraer la información una vez entrenada la red. Para realizar todo ello, se describen las implicaciones de llevar a cabo dicho algoritmo en plataformas computacionales y que para su simplificación se eliminan los retardos inherentes en la transmisión sináptica que existen en las redes neuronales biológicas, con lo que se obtiene propiamente una simulación de una red neuronal pulsada.

Capítulo 3

En base al algoritmo genérico de reconocimiento de caracteres presentado en el capítulo 2, se realiza la propuesta y la descripción de toda la arquitectura necesaria para ejecutar los modelos expuestos en el capítulo 1 y seleccionados en el capítulo 2 (modelo neuronal, sinapsis y plasticidad),en un sistema que pueda ser implementado en hardware digital, y que al mismo

tiempo sea compatible y que pueda funcionar con la arquitectura “AER”, pieza fundamental en un sistema neuromórfico convencional. Dado que estos sistemas buscan aproximarse al comportamiento plausible de sus contrapartes biológicas, el sistema ´propuesto a diferencia de la simulación en software, posee los retardos existentes en la transmisión sináptica, y se describe como estos intervienen en los procesos de aprendizaje y de reconocimiento de caracteres, con lo que se obtiene propiamente una emulación de una red neuronal pulsada.

Adicionalmente, en este apartado se describe y explica como un sistema neuromórfico hace posible que la fase de entrenamiento y reconocimiento se lleven internamente y que no sea necesario el pre-procesamiento o post-procesamiento de la información por algún medio externo, para lo cual se expone la teoría necesaria de los elementos principales de una arquitectura neuromórfica que posibilita el procesamiento espacio-temporal de la información y en tiempo real.

Capítulo 4

En referencia al software, se presentan los resultados obtenidos de las simulaciones en Matlab 2018 del algoritmo propuesto para redes de 2 dimensiones, en el primer caso, para una red con capacidad para recibir caracteres de 16 pixeles y aprender 3 de ellos, mientras que en el segundo caso, para una red con capacidad para recibir caracteres de 25 pixeles y aprender 12 de ellos.

En referencia al hardware, se presentan los resultados de las simulaciones realizadas en ModelSim 10 y la cantidad de recursos para implementarlos a nivel temporizado para una red con capacidad para recibir caracteres de 36 pixeles y aprender 10 de ellos.

Finalmente, de manera individual se presentan las conclusiones a las que se llegaron de manera individual en software y hardware y se proporciona una

conclusión general donde se contrastan las diferencias y semejanzas entre la simulación y la emulación, para dar lugar a las expectativas del trabajo futuro.

Anexo

Se colocan los “Register-Transfer Level” (RTL, por sus siglas en inglés), de los módulos principales del sistema de reconocimiento de caracteres en FPGA, resultado de la síntesis en el entorno de desarrollo Vivado 2019.

Apéndice

En la dirección http://www.vlsilab.cinvestav.mx/tesis_fgc.html del laboratorio VLSI CINVESTAV se encuentra los códigos en VHDL de los módulos así como el banco de pruebas (test bench) de la arquitectura propuesta en hardware y los programas que se simulan en software. Adicionalmente, se encuentra el “Material suplementario” donde se expone la teoría de neurociencias de los elementos que conforman las redes neuronales de tercera generación: neurona, sinapsis y plasticidad sináptica específicamente de STDP.

CAPÍTULO 1

“Si he llegado a ver más lejos que otros es porque me subí a hombros de gigantes.”

Sir.Issac Newton, Carta dirigida a Robert Hooke, 1676.

“Las nuevas generaciones se nutren siempre de raíces antiguas y profundas.”

Así habló el ahuehuete, Las lenguas de América: recital de poesía, Carlos Montemayor (compilador). 2005

“La razón pura tiene en ella la causalidad necesaria para producir en la realidad lo que tiene en su concepto; por eso no es posible decir despreciativamente de la sabiduría, que solo es una idea; al contrario, precisamente por esto es la idea de la unidad necesaria de todos los fines posibles y debe servir de toda regla práctica como condición necesaria, aunque restrictiva.”

Immanuel Kant, “Crítica de la razón pura”, Las ideas trascendentales, 1787

1 Redes neuronales artificiales: Modelos para implementar neuronas, sinapsis y plasticidad sináptica.

1.1 Introducción

Las redes neuronales artificiales desde sus comienzos han tenido el objetivo de reproducir el comportamiento biológico, para lo que se han usado modelos mucho más sencillos para describir el comportamiento biológico de las neuronas, las sinapsis que emplean para comunicarse y la plasticidad sináptica con la que modifican las relaciones existentes entre ellas. A lo largo del capítulo se mostrarán las características de las redes neuronales artificiales y los modelos que han sido usados para su implementación así como los métodos por los cuales las redes biológicas codifican la información.

1.2 Generaciones

De acuerdo al trabajo de Sergio Daves en [1], las redes neuronales artificiales se pueden dividir en 3 clases, según las unidades computacionales que usan para procesar la información que emiten y que reciben.

1.2.1 Primera generación

La primera generación se basa en neuronas McCulloch Pitts como unidades principales de procesamiento también conocidas como perceptrones o “compuertas con umbral de disparo”. Con ellas, es posible construir una variedad de redes neuronales tales como perceptrones multicapa, redes Hopfield y máquinas de Boltzman. De acuerdo a Mass en [2], una característica de estos modelos es que solo pueden darnos una salida digital. Cuando llegan entradas hacia la neuronas se configura la sinapsis realizando el producto de las conexiones (w) con el valor discreto que llega de la entrada; el perceptrón al igual que su contra parte biológica integra todos los productos y solo cuando se supera un “umbral de disparo” discreto, a la salida se obtiene un “1 o un “0” [1]

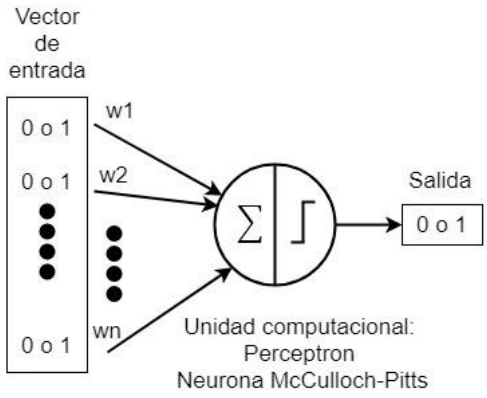

Fig. 1.1.Redes neuronales de primera generación. Extraída de [1].

1.2.2 Segunda generación

Las neuronas funcionales de esta generación están conformadas por dos estados computacionales, por una suma que representa la sinapsis y una función de activación que genera la respuesta de la neurona. La función de activación también conocida como función de transferencia puede tener una forma sigmoidal, lineal, hiperbólica u otra, para definir un rango de salida. La sinapsis se configura realizando el producto de las conexiones “w” con el valor que llega a la entrada, pudiendo ser cualquier número real, y al igual que su contraparte biológica, integrando todos los productos para ser procesados a través de la función de activación; el resultado obtenido, a diferencia de la primera generación, no es un valor discreto, sino una salida analógica [1].

Estas redes neuronales tienen 2 características principales: La primera es que pueden realizar cualquier tipo de cómputo analógico en el sentido de que cualquier función acotada puede ser aproximada bastante bien y la segunda, que usan algoritmos numéricos que se basan en gradiente descendiente tal como el de retro propagación. [2]

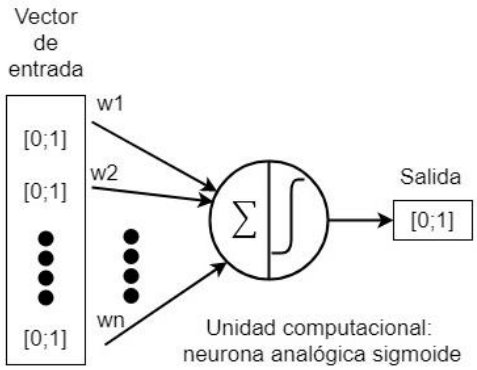

Fig. 1.2.Redes neuronales de segunda generación. Extraída de [1].

1.2.3 Tercera generación

Las redes neuronales de tercera generación son las que mayormente se aproximan al comportamiento de las redes neuronales biológicas. Estas redes neuronales ocupan modelos para reproducir el comportamiento real (peso sináptico, retardo sináptico, función de corriente sináptica, etc.) de las redes biológicas [1]. El estudio de estas redes surgió debido que en los últimos años la evidencia acumulada indicaba que las neuronas biológicas usaban el tiempo de potenciales de acción o “spikes” que generaban, para codificar y procesar la información. Las redes de tercera generación tienen como unidades principales de procesamiento, modelos que replican el comportamiento eléctrico de las neuronas, como en este caso, el “potencial de membrana” que el cual, cuando se supera un umbral definido por el modelo, se genera un pulso o “spike” como respuesta. El instante de tiempo en que se presenta cada “spike” es usado como recurso para computar y comunicar la información [2]. La característica principal es que la entrada y salida son trenes de pulsos que llevan la información determinada por el instante de tiempo en que son generados.

Observaremos que en las redes neuronales de primera y segunda generación el tiempo no determina la respuesta o estado de las neuronas, ni de la información que procesa, ya que son abstracciones más simples de las redes neuronales biológicas a diferencia de las de tercera generación.

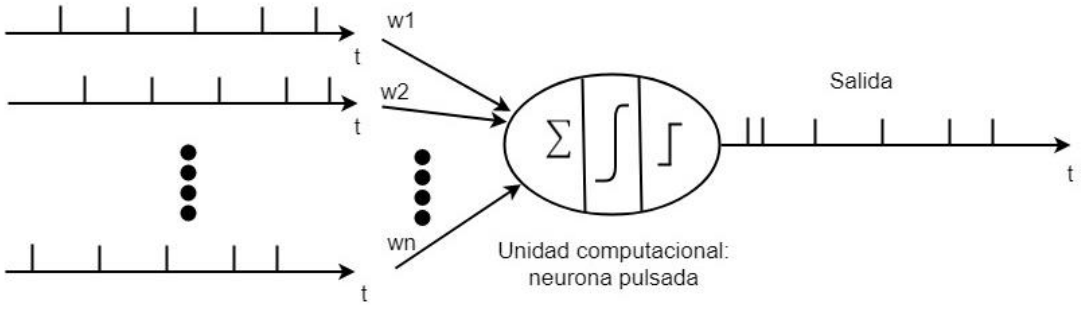

Fig. 1.3. Redes neuronales de tercera generación. Extraída de [1].

1.3 Modelos Neuronales

Existen diversos modelos matemáticos usados para representar el comportamiento eléctrico de la neurona en términos de ecuaciones diferenciales ordinarias (ODE). Aquí se presentan algunos de los modelos usados para realizar implementaciones en hardware y software. [1]

1.3.1 Modelo de Hodgkin-Huxley

En 1952, se definió el primer modelo de neurona que describía el comportamiento eléctrico debido a las propiedades electro químicas de la neurona biológica. El modelo de Hodgkin-Huxley es un modelo basado en la conductancia que se representa como un circuito eléctrico basado en la interconexión de resistencias, fuentes de voltaje y capacitores en serie paralelo. El modelo es descrito a través de la ecuación (1.1) [3]:

$$C_m \frac{dV_m}{dt} = -g_{Na}\, m^3 h(V_m)[V_m - E_{Na}] - g_K\, n^4\, [V_m - E_K] - g_L[V_m - E_L] + I_m \quad \textbf{(1.1)}$$

Donde:

- g_{Na} , g_K y g_L son las conductancias iónicas de sodio, potasio y cloro respectivamente.
- m y n representan las variables de compuerta de los iones de sodio y potasio, respectivamente en la neurona y que son descritas en el trabajo de J. Tranquillo en [3].

- V_m es el potencial de membrana donde se refleja la actividad pulsada de la neurona.
- E_{Na}, E_K y E_L son los potenciales de equilibrio debido a los iones de sodio, potasio y cloro, respectivamente.
- C_m la capacitancia de membrana.
- I_m la cantidad de corriente iónica que experimenta la neurona vía sinapsis o de manera artificial.

De acuerdo Paugam-Moisy y Bohte en [4], el modelo de Hodgkin-Huxley ha tenido éxito en reproducir los resultados experimentales del axón gigante de calamar cuando se eligieron correctamente las constantes del sistema y tiene propiedades significativas respecto al comportamiento registrado en neuronas biológicas: el tiempo de disparo, el periodo refractario absoluto y relativo de la respuesta de su potencial de membrana.

1.3.2 Modelo de Izhikevich

El modelo de neurona Izhikevich, es un modelo neuronal que posee un nivel alto de plausibilidad biológica en comparación con otros modelos y el más aproximado al modelo de Hodgkin-Huxley; el modelo matemático que representa el comportamiento de la neurona biológica está descrito por las ecuaciones (1.2) y (1.3):

$$\frac{dv}{dt} = 0.04v^2 + 5v + 140 + I \quad \textbf{(1.2)}$$

$$\frac{du}{dt} = a(bv - u) \quad \textbf{(1.3)}$$

$$Si\ v = 30\ mV\ entonces\ v = c, u = u + d$$

Donde v es el potencial de membrana y u la variable de recuperación. Las ecuaciones (1.2) y (1.3) tienen 4 variables que nos permiten manipular el patrón de generación de potenciales de acción (“spikes”) de la neurona, las cuales se definen a continuación en base a [1] y al trabajo de izhikevich en [5] y [6], donde:

- a decribe la escala de tiempo de recuperación de la variable u. Valores más pequeños, indican una recuperación más lenta.
- b describe la sensitividad de la variable de recuperación u respecto a las variaciones del sub umbral del potencial de membrana v.
- c describe el valor del potencial de membrana una vez se ha presentado el "reset", es decir el valor desde el cual comienza la despolarización.
- d describe el valor de la variable de recuperación u a causa de un lento sub umbral de disparo, debido a la conductancia de iones de sodio y de potasio.
- I representa la cantidad de corriente iónica que experimenta la neurona vía sinapsis o de manera artificial.

En la figura (1.4), se muestran las formas que reproduce el modelo de Izhikevich, así como sus respectivos valores, las cuales se han observado experimentalmente en el neocortex.

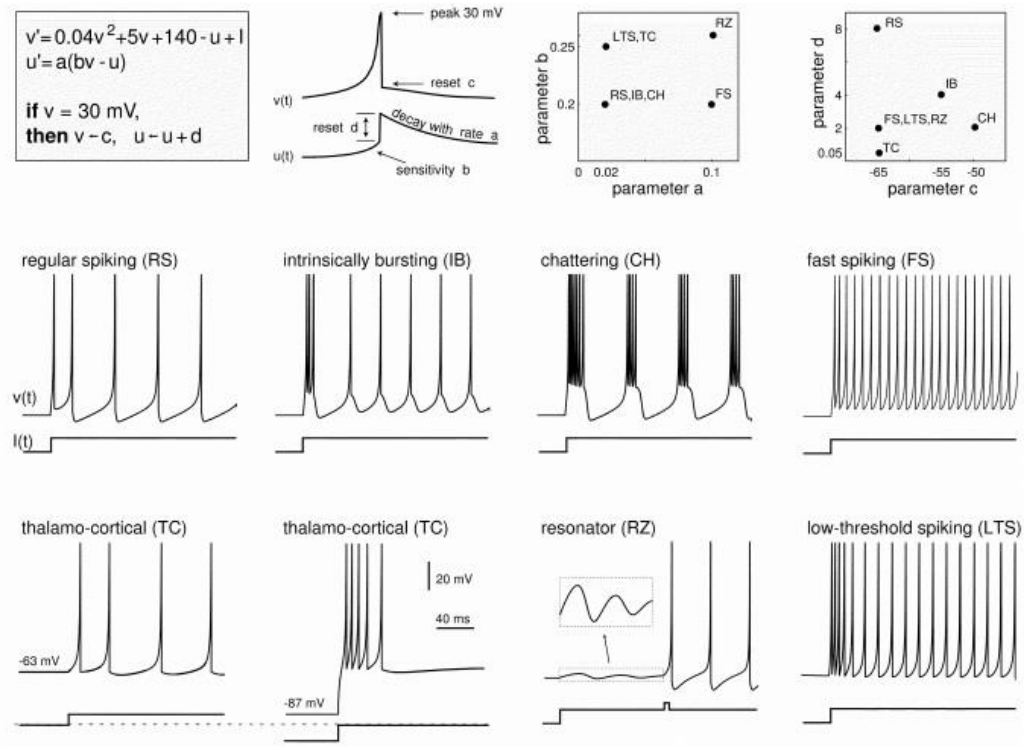

Fig. 1.4 Diferentes formas que reproduce el modelo de Izhikevich extraída de [5].

1.3.3 Modelo Integrate and Fire y variantes.

1.3.3.1 Integrate and Fire

De acuerdo al trabajo de Paugam-Moisy y Bohte en [4], el modelo Integrate and Fire es una simplificación del modelo de Hodgkin-Huxley computacionalmente mucho más tratable, descrito por la ecuación (1.4):

$$C\frac{du}{dt} = -\frac{1}{R}(u - u_{rest}) + I(t) \quad \textbf{(1.4)}$$

Donde

- u es el potencial de membrana.
- R es la resistencia de membrana.
- u_{REST} representa el valor del potencial de membrana cuando la corriente es cero (resting potential).
- I representa la cantidad de corriente iónica que experimenta la neurona vía sinapsis o de manera artificial.

1.3.3.2 Leaky Integrate and Fire

El modelo matemático que representa la neurona es el descrito por la ecuación (1.5) [1] [4]:

$$\tau\frac{dv}{dt} = v_L - v + R_m I(t) \quad \textbf{(1.5)}$$

$$si\ v \geq\ v_{threshold}\ entonces\ v =\ v_{reset}$$

Donde

- v es el potencial de membrana.
- τ es la constante de tiempo de membrana. $\tau = RC$.
- v_L representa el valor del potencial de membrana cuando la corriente es cero (resting potential).
- R_L es la resistencia de membrana.
- I representa la cantidad de corriente iónica que experimenta la neurona vía sinapsis o de manera artificial.

- $v_{threshold}$ es el umbral necesario que el potencial de membrana necesita para que se genere una AP (spike).
- v_{reset} es el potencial al que decae el valor del potencial de membrana una vez que se ha generado un AP.

1.4 Modelos de Sinapsis Química

Los modelos de la sinapsis química presentados en [3] por Joseph Tranquillo, son modelos que contienen un alto nivel de plausibilidad en cuanto al comportamiento experimentalmente observado; sin embargo, estos modelos pueden ser simplificados a fin de obtener el mismo comportamiento biológico pero reduciendo la cantidad de operaciones para procesarlas, tomando en cuenta el nivel de relaciones sinápticas que es necesario implementar vía software o hardware y realizar simulaciones o emulaciones. Para ello será necesario presentar las ecuaciones que describen las secuencias de los potenciales de acción generados por la neuronas, el peso sináptico que las relacionan y el efecto que tendrán sobre la sinapsis.

1.4.1 Respuesta Neuronal

La respuesta de cualquier neurona puede ser expresada como la suma de las funciones delta de Dirac centradas en los pulsos correspondientes a los instantes de tiempo en que se presentan, tal como se muestra en la ecuación (1.6), la cual es presentada en el trabajo Ren, Zhang y Zhao en [7] así como en el de Heeger en [8].

$$H(t) = \sum_{k=0} \delta(t - t_k) \tag{1.6}$$

Donde t_k es el k-ésimo instante de tiempo en que el pulso es generado en la neurona y la función delta de Dirac describe la integración del potencial de acción asociado, tal como se muestra en la ecuación (1.7):

$$\int_{-\infty}^{\infty} \delta(t) = 1. \tag{1.7}$$

Dado que cuando se integra un pulso de amplitud infinita, el resultado es igual a la unidad, la secuencia generada estará definida entonces por pulsos de amplitud unitaria cada vez que se generen potenciales de acción, mientras quesi no los hay, no habrá ninguna contribución tal como se muestra en la ecuación (1.8).

$$\delta(t) = \begin{cases} 1 & si\ exite\ spike\ en\ t = t_k \\ 0 & de\ otra\ forma \end{cases} \quad \textbf{(1.8)}$$

Cuando la secuencia de pulsos interactúa con la sinapsis, esta función tiende a modificar la conductancia eléctrica de la neurona biológica solo sobre instantes de tiempo específicos a diferencia de los modelos observados en [3], que todo el tiempo contribuyen a la modificación de la conductancia eléctrica. Por esta razón, este modelo de transmisión sináptica es considerado un modelo lineal, dado que implica que cada uno de los potenciales de acción (spikes) provoca las mismas características en el cambio de conductancia. En el trabajo descrito por Hegger en [9], se especifica que este modelo en detalle no es correcto por 2 razones: primeramente, porque los efectos entre potenciales de acción (spikes) no son realmente independientes entre ellos biológicamente, y en segundo lugar, porque solo una fracción de los potenciales de acción (spikes) que recibe una neurona provocan un cambio en la conductancia eléctrica y por lo tanto no todos contribuyen a la generación de corriente sináptica, al debilitamiento o al fortalecimiento sináptico dado que no son integradas en el soma. Sin embargo, la secuencia de pulsos descrita por la ecuación (1.6) nos permite aproximar de manera significativa el comportamiento de los potenciales de acción generados y usarla como modelo computacional.

1.4.2 Corriente post sináptica

Dado que cada pulso pre sináptico contribuye a la generación de un pulso post sináptico a través de la corriente iónica que la neurona post sináptica puede integrar de acuerdo a Gerstner y Kistler en [10], la cantidad de corriente post sináptica total es debida a la secuencias de pulsos que recibe, a los pesos

sinápticos de cada una de sus relaciones y a una función que determina la corriente que cada pulso pre sináptico genera, tal como se observa en la ecuación (1.9).

$$I_j(t) = \sum_i w_{ij} \sum_k H(t - t_k^i) * \beta \qquad \textbf{(1.9)}$$

Donde:

- w_{ij} es la "fortaleza sináptica" o peso sináptico que relaciona la neurona pre sináptica "i-ésima" con la neurona post sináptica "j-ésima".
- β es la función que modela la cantidad de corriente post sináptica "j-ésima" a partir de una serie de pulsos generados en instantes de tiempo "k-ésimo" por una neurona pre sináptico "i-ésima".

En este caso, obsérvese que la conductancia (g_{syn}) descrita por J. Tranquillo en las ecuaciones biológicas en [3] que describen la sinapsis química, conductancia debida al par neurotransmisor-receptor, la ecuación (1.9) no aparece; en este caso, este tipo de modelo toma en cuenta la forma más simple de corriente, donde la conductancia es igual a la unidad ($g_{syn} = 1$) y, en dado caso de necesitar una aproximación mucho más plausible, será necesario añadirla a la ecuación (1.9).Por otra parte, la corriente post sináptica puede ser definida de 4 formas para modelar la cantidad de corriente sináptica, únicamente modificando la función "β" la cual, en este caso, podrá estar determinada por el par neurotransmisor-receptor de la sinapsis biológica.

1.4.3 Función de corriente simple

La función beta que modela a la cantidad de corriente es tal como se describe en la ecuación (1.10):

$$\beta = q * f \qquad \textbf{(1.10)}$$

Donde

- q es la carga total que es inyectada a la post neurona vía sinapsis.

- f es la función que modela la forma de la corriente post sináptica biológica encontrada en el trabajo de J. Tranquillo en [3].

En el caso más simple, la función"f" es igual a la unidad, de tal forma que la corriente es constante en cada interacción que tiene la sinapsis con la secuencia de pulsos pre sinápticos, tal como se muestra en la ecuación (1.11).

$$I_j(t) = q \sum_i w_{ij} \sum_k H(t - t_k^i) \qquad \textbf{(1.11)}$$

Además, observamos que la corriente post sináptica "j-ésima" es directamente proporcional al valor del peso sináptico y la cantidad de corriente determinada por la carga (q) que es inyectada. Sin embargo, cabe mencionar que éste es el modelo computacional más simple y plausiblemente el más alejado del comportamiento biológico de la sinapsis química, ya que como tal no hay tiempos de subida y de caída en la corriente sináptica debida a los procesos de difusión y re captación de neurotransmisores.

En la figura (1.5)-A se puede observar el resultado de la interacción entre pulsos pre sinápticos k-ésimos de una neurona "i" y el efecto sobre la corriente "j".

1.4.4 Corriente a partir de la función exponencial "alfa"

La función que determina la cantidad de corriente post sináptica debida al par neurotransmisor-receptor glutamato-AMPA en [3], tiene una duración de tiempo definida para cada pulso pre sináptico. En Gerstner y Kistler, específicamente en [10], se definen respuestas exponenciales con constantes de tiempo "τ_s" cuando interactúan con secuencias de pulsos, tal como se describe en las ecuaciones (1.12) y (1.13).

$$f = \frac{1}{\tau_s} e^{(t - t_k^i)} \qquad \textbf{(1.12)}$$

Sustituyendo (1.12) en (1.10), obtenemos la corriente post sináptica total simplificada, debido al neurotransmisor glutamato-AMPA, tal como se describe en la ecuación (1.13):

$$I_j(t) = \frac{q}{\tau_s} \sum_i w_{ij} \sum_k e^{\left(\frac{t-t_k^i}{\tau_s}\right)} H(t - t_k^i) \qquad \textbf{(1.13)}$$

Por lo tanto, la corriente total es la suma de todas la funciones alfa generadas por diferentes instantes de tiempo, tal como se muestra en la figura (1.5)-B, donde la corriente disminuye conforme transcurre el tiempo, cada vez que la sinapsis ha interactuado con un pulso pre sináptico k-ésimo.

1.4.5 Corriente a partir de función doble exponencial.

Una de las formas más comunes de corriente sináptica de acuerdo a [3] se puede describir por la ecuación (1.14) de manera simplificada, la cual es usada para modelar el comportamiento del par neurotransmisor-receptor glutamato-NMDA:

$$f = \frac{1}{\tau_s - \tau_r} \left[e^{\left(\frac{t-t_k^i}{\tau_s}\right)} - e^{\left(\frac{t-t_k^i}{\tau_r}\right)} \right] \qquad \textbf{(1.14)}$$

Sustituyendo la ecuación (1.14) en (1.10) obtenemos la ecuación (1.15) para describir la corriente post sináptica total “j”:

$$I_j(t) = \frac{q}{\tau_s - \tau_r} \sum_i w_{ij} \sum_k \left[e^{\left(\frac{t-t_k^i}{\tau_s}\right)} - e^{\left(\frac{t-t_k^i}{\tau_r}\right)} \right] H(t - t_k^i) \qquad \textbf{(1.15)}$$

Sin embargo, cuando en este último modelo observamos que $\tau_s = \tau_r$, la ecuación se indetermina por lo que se procede a través de la ecuación (1.16):

$$f = \frac{t - t_k}{{\tau_s}^2} e^{\left(\frac{t-t_k^i}{\tau_s}\right)} \qquad \textbf{(1.16)}$$

Sustituyendo nuevamente (1.16) en (1.10), obtenemos la ecuación (1.17), que resuelve el problema cuando los tiempos de subida y de bajada de la corriente post sináptica son iguales:

$$I_j(t) = q\frac{t - t_k^i}{{\tau_s}^2}\sum_i w_{ij} \sum_k e^{\left(\frac{t-t_k^i}{\tau_s}\right)} H(t - t_k^i) \qquad \textbf{(1.17)}$$

En este caso, la corriente total es la suma de todas la funciones doble exponencial generados por diferentes instantes de tiempo, tal como se muestra en la figura (1.5)-C; la corriente incrementa y disminuye conforme transcurre el tiempo cada vez que la sinapsis ha interactuado con un pulso pre sináptico k-ésimo.

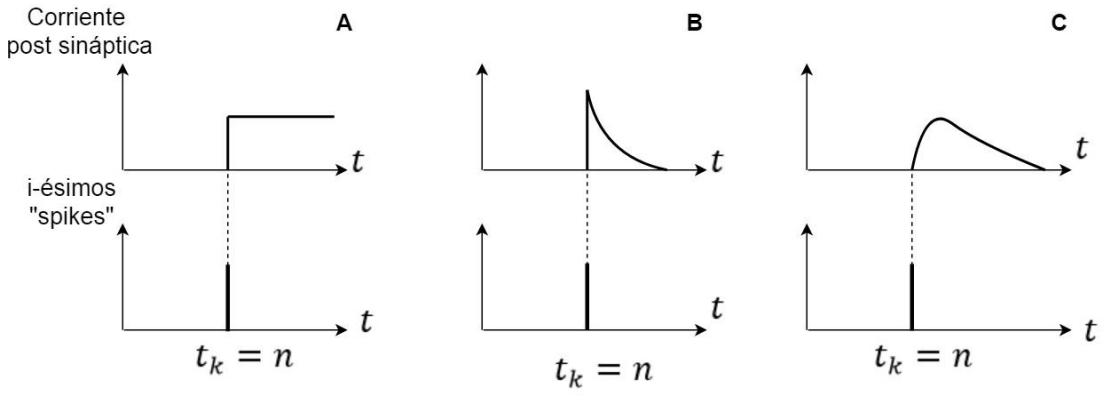

Fig. 1.5. Sinapsis corriente simple, exponencial y doble exponencial respectivamente.

1.4.6 Retardos

Desde el punto de vista computacional y matemático, incluir los retardos hace más complicado el modelado y simulación debido a que los sistemas con retardos no son finitos [6]. Sin embargo, los retardos sobre las sinapsis juegan un papel importante en el procesamiento de la información y se encuentran inherentemente presentes debido a las características fisiológicas de la neurona [3].Si no estuvieran presentes los retardos en las redes neuronales biológicas, como en los modelos computacionales que intentan reproducir su comportamiento, se esperaría una sincronía en los potenciales de acción generados por las mismas cuando un estímulo se le presente, algo deseado

por algunos investigadores en años anteriores; sin embargo, la capacidad de procesamiento de la red se vería reducida e incluso eliminada debido a que toda la red actuaría como si fuera una sola neurona dado que la información viene dada por instantes específicos de tiempo, los cuales, en el caso de que no hubieran retardos serian iguales para toda la red. De acuerdo al trabajo de Izhikevich en [6], el hecho de que existan retardos presentes y que estos sean diferentes en cada unión sináptica provocando que los trenes de pulsos generados se encuentren desfasados unos respecto de otros, permite codificar una mayor información.

Los modelos matemáticos que describen los retardos basados en las características eléctricas del axón prefieren evitarse cuando se trata de incrementar el volumen de las redes neuronales, debido a que los recursos computacionales necesarios son mayores para realizarse simulaciones en software.

El procedimiento para agregar retardos es mediante el desplazamiento de los efectos de las sinapsis individuales en cada unión sináptica de la neurona, gracias a que las neuronas se comportan como circuitos con propiedades activas [3] y, dado que de manera experimental se conocen los rangos de tiempo en que se encuentran estos retardos, es posible incorporarlos en los modelos de sinapsis descritos anteriormente. En la tabla (1.1) se muestran algunos de los retardos encontrados experimentalmente.

conexión	Retardo (ms)	animal
Cortico-cortical	1.7-32	conejo
	1-35	
	1.2-19	
Capa VI-LGN	1-44	gato
Cortico-colicular	0-3	
Capa IV- VB	2	ratón

Tabla 1.1. Retardos extraídos de [6].

De acuerdo al trabajo de Lakymchuk, Rosado-Muñoz, Guerrero-Martínez, Bataller-Mompeán y Francés-Víllora en [11], el tiempo que tarda un pulso en viajar entre 2 neuronas está definido por la ecuación (1.18):

$$t_{ij} = t - t_k^i - d_{ij} \tag{1.18}$$

Donde:

- t_k es el k-ésimo instante de tiempo en que se genera un pulso en la neurona pre sináptica.
- d_{ij} es el retardo que existe entre la neurona pre sináptica "i-ésima" y neurona post sináptica "j-esima".

Basados en [4] (Paugam-Moisy y Bohte), [10] (Gerstner y Kistler) y [12] (Olaf Booij), cuando estos retardos se incorporan a las ecuaciones (1.11), (1.13), (1.15) y (1.17) se convierten en las ecuaciones (1.19)-(1.22), las cuales generan el desplazamiento de la corriente post sináptica, tal como se muestran en la figura (1.6).

$$I_j(t) = q \sum_i w_{ij} \sum_k H(t - t_k^i - d_{ij}) \tag{1.19}$$

$$I_j(t) = \frac{q}{\tau_s} \sum_i w_{ij} \sum_k e^{\left(\frac{t-t_k^i}{\tau_s}\right)} H(t - t_k^i - d_{ij}) \tag{1.20}$$

$$I_j(t) = \frac{q}{\tau_s - \tau_r} \sum_i w_{ij} \sum_k \left[e^{\left(\frac{t-t_k^i}{\tau_s}\right)} - e^{\left(\frac{t-t_k^i}{\tau_r}\right)} \right] H(t - t_k^i - d_{ij}) \tag{1.21}$$

$$I_j(t) = q \frac{t - t_i}{{\tau_s}^2} \sum_i w_{ij} \sum_k e^{\left(\frac{t-t_k^i}{\tau_s}\right)} H(t - t_k^i - d_{ij}) \tag{1.22}$$

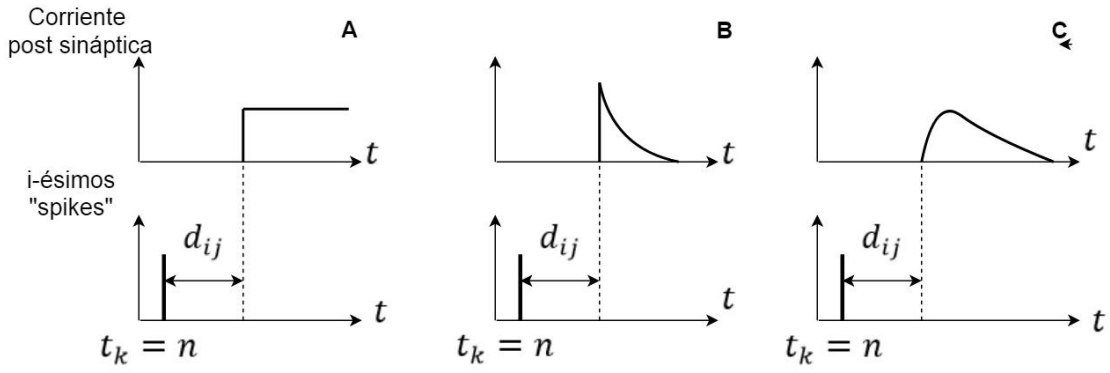

Fig. 1.6. Retardos sobre la sinapsis en su forma simple, alfa y doble exponencial.

1.5 Modelos de plasticidad sináptica: STDP.

De acuerdo al trabajo de Dan y Caporale en [13], de Bi y Wang en [14], así como en el de Bi y Poo en [15], STDP es una forma asimétrica de aprendizaje hebbiana, teórica y experimental, que establece mecanismos de modificación sináptica, basada en la correlación temporal entre los potenciales de acción (spikes) de neuronas pre y post sinápticas con efecto en la memoria de largo plazo. STDP de sus siglas en inglés “Spike Timing Dependent Plasticity” es el resultado de las observaciones en distintos experimentos realizados en diversas regiones del córtex y el hipocampo, en las capas III, IV y V principalmente de neuronas piramidales en diversas especies.

En esta sección, se presentarán diversos modelos inspirados en la forma característica de la ventana de STDP para potenciar o depreciar los pesos sinápticos en una red de tercera generación, cualquiera de los cuales determinarán el cambio del peso sináptico variando el nivel de plausibilidad biológica visto en [13], [14] y [16].

Obsérvese que habrá que determinar el protocolo para asociación de pulsos pre y post sinápticos, ya sea el de pares, triplets o cuadruplets para determinar la diferencia temporal (Δt) y con ello determinar el cambio del peso sináptico (Δw). En nuestro caso, se tomará siempre un aprendizaje causal independiente del protocolo de asociación de pulsos, para lo cual, la diferencia temporal conocida como “spike-timing” entre los pulsos siempre estará dado por la ecuación (1.23).

$$\Delta t = t^{post} - t^{pre} \quad \textbf{(1.23)}$$

Existen varios modelos matemáticos que se aproximan a los resultados en los experimentos biológicos realizados. Sin embargo, computacionalmente hablando, se han investigado diversos modelos inspirados en la ventana de aprendizaje exponencial encontrado en los experimentos realizados en [15], los cuales, han ido desde aquellos que tienen un costo computacional muy bajo en cuanto a la cantidad de operaciones realizadas, hasta aquellos donde el modelo reproduce la forma biológica de STDP con un costo computacionalmente relativamente más alto.

En el trabajo de Ponulak de 2008 especificamente en [17], se presentan diversos modelos para representar la forma estándar de STDP donde:

- Δw es el cambio del peso sináptico.
- Δt es la diferencia temporal entre los pulsos pre y post sinápticos (spike timing).
- A_+ Cambio sináptico máximo en la región de potenciación (LTD).
- A_- Cambio sináptico máximo en la región de depresión (LTP).
- τ_+ Longitud de la ventana de tiempo en potenciación representada en milisegundos
- τ_- Longitud de la ventana de tiempo en depresión representada en milisegundos

Cabe mencionar que A_+, A_-, τ_+ y τ_-son valores constantes positivos y de acuerdo al trabajo de Song, Miller y Abott en [18], siempre que se cumpla la ecuación (1.24), las neuronas involucradas en el aprendizaje de STDP, podrán realizar una clasificación entre los pulsos pre y pos sinápticos correlacionados y los no correlacionados

$$A_-\tau_- > A_+\tau_+ \quad \textbf{(1.24)}$$

1.5.1 Ventana rectangular

El valor del cambio del peso sináptico solo depende de los valores máximos tanto de potenciación como de depresión y como tal, es una regla de asignación, ya que no existe ningún tipo de operación que se realice para procesar la información, tal como se muestra en la ecuación (1.25).

$$\Delta w = \begin{cases} +A_+ & si\ \Delta t\ \in [0, \tau_+] \\ -A_- & si\ \Delta t\ \in [-\tau_-, 0) \\ 0 & de\ otra\ forma \end{cases} \quad \textbf{(1.25)}$$

Si la diferencia temporal entre los pulsos pre y post sinápticos se encuentran en un lapso de tiempo mayor o igual a cero y menor o igual a τ_+ se presentan cambios sinápticos positivos y constantes sobre todo el intervalo, mientras que, cuando la diferencia temporal es negativa y ocurre dentro de un intervalo menor a cero y mayor o igual a $-\tau_-$ se presentan cambios negativos y constantes. Al igual que el modelo biológico, cuando la diferencia temporal (spike timing) de los pares de pulsos está fuera de los intervalos establecidos por la ventana de aprendizaje, no hay modificación sináptica.

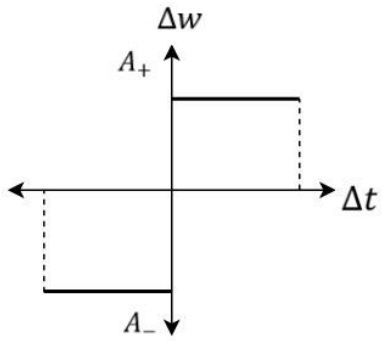

Fig. 1.7 Ventana rectangular.

1.5.2 Ventana Lineal tipo I.

El valor del cambio sináptico varía proporcionalmente respecto a la diferencia temporal de los pulsos. Sin embargo, conforme las diferencias son mayores (en los extremos de las ventanas por derecha y por izquierda) el cambio sináptico es máximo tanto en depresión como en potenciación, sin embargo, cabe mencionar que el comportamiento de esta ventana es opuesto al

encontrado en los resultados experimentales y está descrito por la ecuación (1.26).

$$\Delta w = \begin{cases} +A_+\left(\frac{\Delta t}{\tau_+}\right) & si\ \Delta t\ \in [0, \tau_+] \\ -A_-\left(\frac{\Delta t}{\tau_+}\right) & si\ \Delta t\ \in [-\tau_-, 0) \\ 0 & de\ otra\ forma \end{cases} \quad \textbf{(1.26)}$$

Si la diferencia temporal se encuentran en un lapso de tiempo mayor o igual a cero y menor o igual a τ_+, se presentan cambios sinápticos positivos y proporcionales a esa diferencia temporal, mientras que, cuando la diferencia temporal es negativa y ocurre dentro de un intervalo menor a cero y mayor o igual a $-\tau_-$, se presentan cambios negativos.

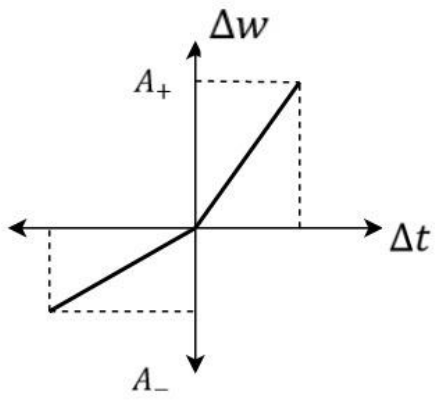

Fig. 1.8 Ventana lineal tipo I

En los experimentos realizados en [17], este tipo de ventana de aprendizaje no garantiza la convergencia del aprendizaje en una red neuronal, con lo que existe una pobre aproximación de los patrones que se desean alcanzar.

1.5.3 Ventana Lineal tipo II.

Esta ventana de aprendizaje de los modelos lineales hasta ahora presentados, es la que puede aproximarse más al comportamiento biológicamente plausible de STDP, una diferencia temporal positiva produce cambios positivos máximos y mínimos, mientras que cuando se observan diferencias temporales

negativas, produce cambios negativos máximos y mínimos, tal como se describe en la ecuación (1.27).

$$\Delta w = \begin{cases} +A_+\left(1-\dfrac{\Delta t}{\tau_+}\right) & si\ \Delta t\ \in [0,\tau_+] \\ -A_-\left(1-\dfrac{\Delta t}{\tau_+}\right) & si\ \Delta t\ \in [-\tau_-,0) \\ 0 & de\ otra\ forma \end{cases} \qquad \textbf{(1.27)}$$

En contraste con la ventana tipo I se tiene un buen rendimiento y estabilidad de aprendizaje y una buena aproximación en cuanto a los patrones que se desean alcanzar; sin embargo, a diferencia de la forma tradicional de STDP, la convergencia del aprendizaje no ha sido formalmente probada. [17]

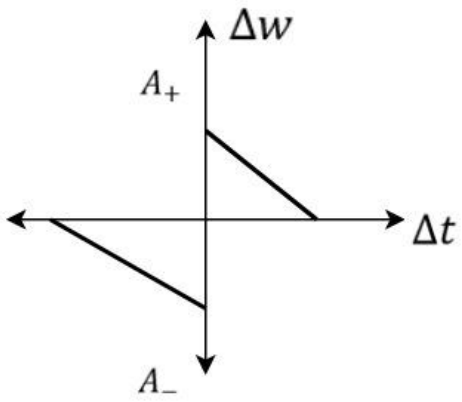

Fig. 1.9 Ventana lineal tipo II

1.5.4 Ventana tradicional de STDP (forma Exponencial)

Ésta es la forma que se encontró específicamente en los experimentos biológicos observados en [14], [15] y [16]; formalmente ha sido descrita a través de la ecuación (1.28), pero no se debe perder de vista que tiene solamente efecto sobre el rango de $[\tau_-, \tau_+]$.

$$\Delta w = \begin{cases} +A_+ e^{\left(\frac{-\Delta t}{\tau_+}\right)} & si\ \Delta t \geq 0 \\ -A_- e^{\left(\frac{\Delta t}{\tau_-}\right)} & si\ \Delta t < 0 \end{cases} \qquad \textbf{(1.28)}$$

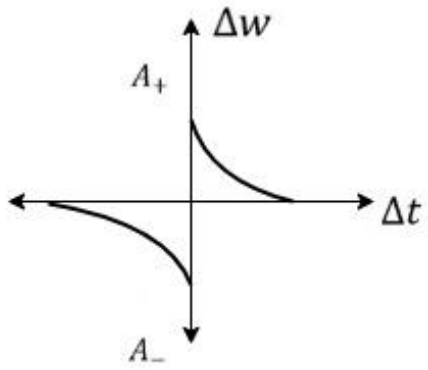

Fig. 1.10 Ventana exponencial de STDP.

1.5.5 Ventana doble exponencial

Esta ventana contiene las constantes τ_{r-}, τ_{r+}y τ_{d-}, τ_{d+}como constantes de tiempo de subida y de bajada respectivamente. Esta forma de la ventana de aprendizaje solo converge en las partes en la que se observa decaimiento, es decir en las zonas que se aproximan a la forma tradicional de STDP; sin embargo, en ocasiones esta ventana de aprendizaje ha sido considerada una buena aproximación biológicamente realista en las ventanas de STDP y que ha sido satisfactoriamente aplicada en métodos de aprendizaje supervisado como ReSuMe ("Remote Supervised Method") si la condiciones de tiempo de la ventana de aprendizaje son satisfechas. La forma de la ventana de aprendizaje está descrita por la ecuación (1.29).

$$\Delta w = \begin{cases} +A_+ \left[e^{\left(\frac{-\Delta t}{\tau_{r+}}\right)} - e^{\left(\frac{-\Delta t}{\tau_{d+}}\right)} \right] & si\ \Delta t \geq 0 \\ -A_- \left[e^{\left(\frac{+\Delta t}{\tau_{r-}}\right)} - e^{\left(\frac{+\Delta t}{\tau_{d+}}\right)} \right] & si\ \Delta t < 0 \end{cases} \quad \textbf{(1.29)}$$

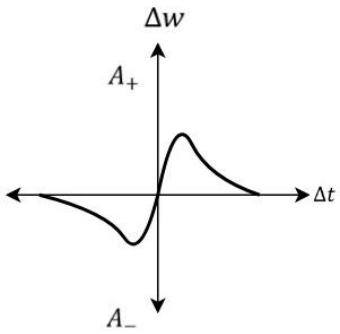

Fig. 1.11.Ventana doble exponencial.

1.5.6 Variación Multiplicativa

Esta ventana de aprendizaje depende de los pesos sinápticos y de la modificación realizada por la diferencia temporal de la ventana de aprendizaje. Esta forma está descrita a través de la ecuación (1.30):

$$\Delta w = \begin{cases} +A_{+}(1-w)^{\mu} e^{\left(\frac{-\Delta t}{\tau_{+}}\right)} & si\ \Delta t \geq 0 \\ -A_{-} w^{\mu} e^{\left(\frac{\Delta t}{\tau_{+}}\right)} & si\ \Delta t < 0 \end{cases} \qquad \textbf{(1.30)}$$

Donde "w" es el peso sináptico de la sinapsis y "μ" es un exponente no negativo; en contraste con la forma tradicional de STDP, en el trabajo de Guetig, Aharonov, Rotter y Sampolinsky en [19] se reportan para esta ventana de aprendizaje pesos sinápticos intermedios y estables, además que el aprendizaje es altamente sensitivo a la distribución de las señales de entrada, de acuerdo al trabajo de Ponulak y Kasinski en [20].

1.5.7 Memoria: Integración de STDP

De acuerdo a los resultados de los experimentos sobre cultivos de neuronas vivas en [14], [15] y [16], STDP solamente determina cambios para cada asociación de pulsos pre y post sinápticos provocando un efecto en la modificación del peso sináptico. Sin embargo, en los experimentos STDP realizados, no se especifica matemáticamente la manera en cómo esto puede realmente afectar a la sinapsis. El caso de integración espacio-temporal de STDP, es descrito a través de la ecuación (1.31), la cual para cada cambio del peso sináptico (Δw) se asocia directamente como el peso sináptico anterior (w_{old}); de esta manera, la memoria es un proceso continuo en el que la experiencia se ve reflejada través del peso sináptico, como se menciona en el trabajo de Lightheart, Grainger y Lu en [21].

$$w = w_{new} = w_{old} + \Delta w \qquad \textbf{(1.31)}$$

1.6 Codificación

Varios experimentos en diversas especies, apuntan a que la información generada a partir de los sentidos (vista, tacto, gusto, oído y olfato), puede ser “leída” a través de la respuesta de neuronas o grupos de neuronas, las cuales para cada estímulo generan secuencias de potenciales de acción que se encuentran relacionadas ya sea respecto a la cantidad, al tiempo o al orden de aparición.

Así por ejemplo, cuando se presente un patrón que visualmente es reconocido, una región específica del cerebro (en este caso la corteza visual), reaccionará a éste, generando un patrón en la actividad eléctrica de las neuronas que permita identificar las características de la imagen tales como posición, inclinación, color, etc; tal como el que se muestra en la figura (1.12), donde un gato que observa un rectángulo sobre una pantalla genera actividad eléctrica relacionada a cada patrón y la cual difiere de una medición a otra.

En este caso, como observador externo ¿cómo podemos “leer” este código que las neuronas emplean para comunicarse y entender el mensaje del patrón de la actividad neuronal cuando la actividad que presenta es tan variada? [10]. Éste es el problema central de la codificación neuronal y una de las tareas fundamentales de las neurociencias.

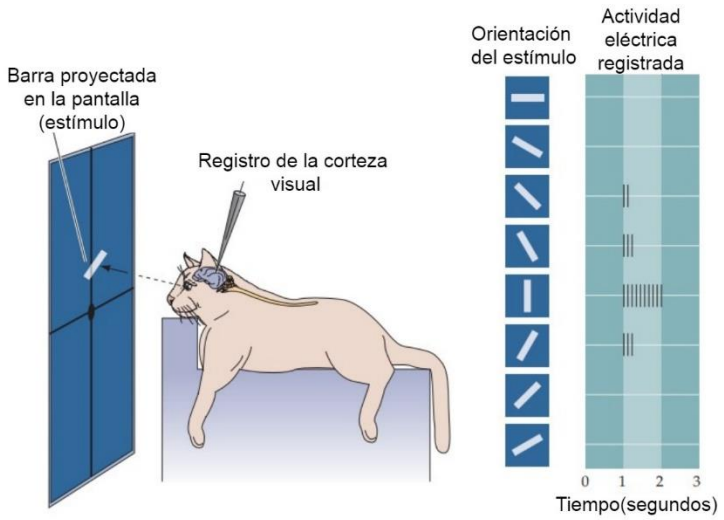

Fig. 1.12. Actividad eléctrica de la corteza visual provocada por un estímulo. Extraído de [22].

Existen 2 métodos para poder “leer” esta información, los cuales son conocidos como codificación por proporción (rate code) y codificación por pulsos (spike code). Basado en el trabajo de Gerstner y Kistler en [10], a inicios de 1920, la codificación por proporción ayudaba a explicar cómo la velocidad de disparo de las neuronas receptoras en los músculos estaba relacionada con la fuerza que se aplicaba sobre estos, por lo que en los siguientes 80 años se ocupó este mismo método para describir las propiedades de otras neuronas corticales y sensoriales. Sin embargo, en años recientes más evidencia experimental sugirió que el método de codificación por proporciones de disparo basado en el promedio temporal de los pulsos, era demasiado simple para describir la actividad del cerebro, ya que no podía explicar otras áreas como la corteza visual del cerebro, donde la codificación basada en pulsos sí permitía determinar los patrones de la actividad eléctrica a través de la posición de los pulsos en espacio y tiempo, por lo que con base en ello, se determinó que cada pulso generado por la red contenía información relacionada con el área donde se generaba (espacial) y el instante de tiempo en que se presentaba (temporal).

Dado que el método de codificación por proporción (code rate) desprecia la posibilidad de que la información esté contenida en los tiempos exactos en que se generan los pulsos, ha sido causa de debates desde sus inicios; sin embargo, cómo determinar qué método aplicar cuando se desea decodificar la información en una red biológica o artificial, es un problema todavía al que se busca solución.

Ambos métodos se ocupan actualmente para decodificar la información contenida en las redes neuronales biológicas y artificiales de tercera generación, y determinar qué método es el más apropiado puede depender de varios factores como las propiedades biológicas, la dinámica neuronal o la arquitectura de la red.

1.6.1 Codificación por proporción de pulsos (Rate code)

1.6.1.1 Proporción como conteo de pulsos (spike)

La primera y más común definición usada como proporción o velocidad de disparo se refiere al promedio temporal de los pulsos. Básicamente es contar la cantidad de pulsos que hay sobre un período de tiempo, tal como describe Troyer en [23], donde la única diferencia radica básicamente en la selección de los períodos de muestreo seleccionados, los cuales oscilan entre T= 100 ms y T=500 ms, sin embargo, la duración usada para determinar la velocidad de disparo puede ser más corta o más amplia. La proporción por conteo de pulsos se encuentra expresada en Hertz y está determinada por la ecuación (1.32).

$$f = \frac{\#\ de\ spikes(potenciales\ de\ acción)}{T}\ [Hz] \qquad \textbf{(1.32)}$$

El procedimiento de codificación por conteo de pulsos se encuentra ilustrado en la figura (1.13).

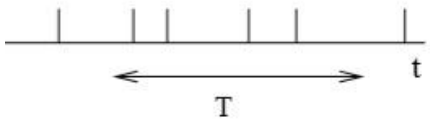

Fig. 1.13. Codificación por conteo de pulsos. Extraído de [23].

1.6.1.2 Proporción como densidad de pulsos (spike)

Se basa en realizar una misma secuencia de estimulación varias veces (diferentes corridas) sobre una neurona para determinar su respuesta con base en un histograma de tiempo de peri-estímulo conocido como “PSTH” por sus siglas en inglés “Peri-Stimulus Time Histogram”. El método es descrito a través de la ecuación (1.33); tal situación se ilustra la figura (1.14):

$$\rho = \frac{1}{\Delta t}\frac{n_K(t; t+\Delta t)}{K}\ [Hz] \qquad \textbf{(1.33)}$$

Donde:

- t es el tiempo desde que inicia el estímulo.
- Δt es una medida de muestreo que va de uno a unos cuantos milisegundos.
- $n_K(t; t + \Delta t)$ es el número de ocurrencias de pulsos (spikes) sumadas sobre todas las repeticiones del experimento.
- K es el número de repeticiones como medida de la actividad típica entre t y $t + \Delta t$.

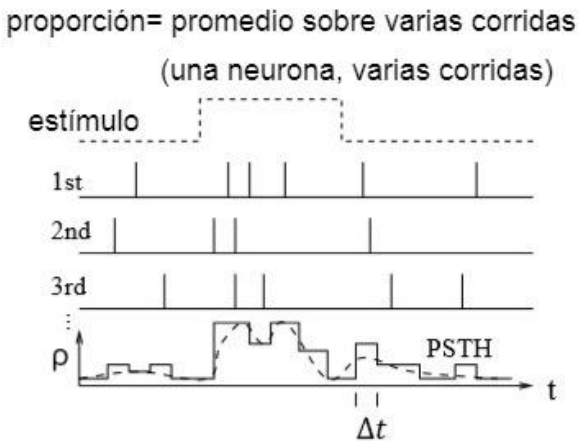

Fig. 1.14. Codificación por densidad de pulsos. Extraído de [23].

1.6.1.3 Proporción como actividad de población

En regiones del cerebro hay agrupaciones de neuronas básicamente con las mismas propiedades y que responden de manera similar al mismo estímulo (como en la corteza visual de gatos y monos). En este caso, se idealiza una situación de neuronas con las mismas propiedades y se toma en cuenta la respuesta de toda la red como se describe en la ecuación (1.34); tal situación se ilustra en la figura (1.15).

$$Am = \frac{1}{\Delta t}\frac{n_{act}(t; t + \Delta t)}{N}\ [Hz] \qquad \textbf{(1.34)}$$

Donde:

- N es el tamaño de la población de neuronas.

- n_{act} es el número de ocurrencias de los pulsos (“spikes”) sumados de todas las neuronas como población.

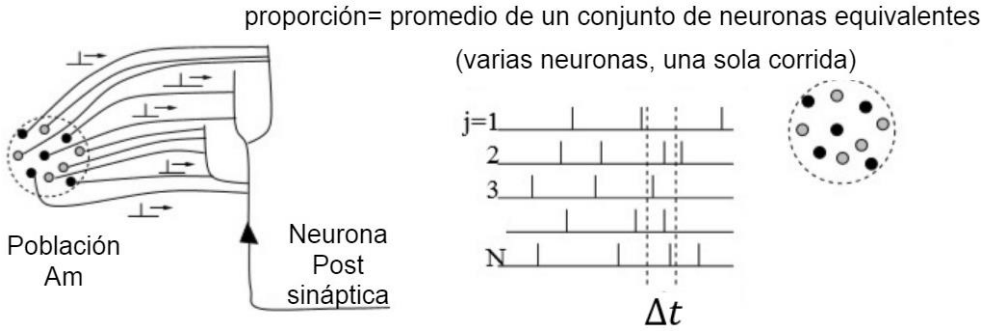

Fig. 1.15. Codificación por actividad de población. Extraído de [23].

1.6.2 Codificación por pulsos (Spike code)

El análisis de la información neuronal es de vital importancia tanto en sistemas biológicos como artificiales basados en las características espacio-temporales de los potenciales de acción individuales (conocidos como “spikes” o pulsos), para determinar la capacidad de procesamiento de la red. Basado en el trabajo de Ponulak y Kasinski en [24], existen 6 estrategias que se han usado para determinar cómo se codifica la información basada en patrones específicos de tiempo y espacio.

1.6.2.1 El primero en disparar

En este modelo, la información viene codificada por la latencia que existe entre el inicio de un estímulo y el primer potencial de acción (spike) de la red neuronal. Este esquema de codificación ha sido encontrado en señales de tacto biológicas como en las puntas de los dedos y ha sido usado para realizar implementaciones de sensores artificiales táctiles y olfatorios. Cabe mencionar que este modelo es el más rápido y más simple de todos los esquemas de codificación, porque permite a la red reconocer el estímulo en unos cuantos milisegundos, ya que solo necesita de una sola de las neuronas que conforman la red neuronal para reconocer algún patrón y es el más fácil de implementar porque solo se necesita inhibir las neuronas vecinas con las que interactúa para prevenir que todas comiencen a pulsar al mismo tiempo. Tal caso, se muestra en la figura (1.16)-A en donde la neurona “n2” fue la que

reconoció el estímulo. En el trabajo de Thorpe, Delorme,y Van Rullen en [25], se considera que la espacio-temporalidad del primer pulso que se genera en los sistemas sensoriales provee suficiente información acerca del estímulo.

1.6.2.2 Codificación basada en el orden de aparición

En este esquema de codificación, la información está basada en el orden de la actividad neuronal, donde cada neurona tiene un lugar en la secuencia que genera toda la red, tal como se muestra en la figura (1.16)-B. Este método ha sido encontrado en experimentos biológicos en la categorización del sistema visual de los primates. Este modelo asume que cada neurona emite solamente un solo spike durante la presentación de una imagen, lo cual es posible implementar con inhibiciones laterales. Este método ha sido usado para categorizar imágenes estáticas con velocidades de procesamiento comparables a las de los seres humanos (menos de 400 ms). [10]

1.6.2.3 Latencia

En este esquema de codificación, se considera que la información está contenida en la diferencia temporal exacta entre un conjunto de “spikes” relativos unos de otros, tal como se muestra en la figura (1.16)-C. Los sistemas basados en latencia son muy eficientes en términos de capacidad de información, lo cual ha sido observado en redes retro propagadas. Sin embargo, el ruido y la inherente dinámica neuronal pueden afectar fácilmente la precisión de la diferencia temporal entre los pulsos.

1.6.2.4 Codificación por ráfaga resonante

En este esquema de codificación, la frecuencia de una ráfaga de pulsos puede determinar cuántas neuronas de la red están activas. En este caso, solo cuando las frecuencias propias del potencial de membrana corresponden a la frecuencia del estímulo, se reconoce algún patrón. Dicha forma de codificación, es la que se considera como mecanismo empleado por las neuronas para comunicarse selectivamente. El esquema de codificación se ve ilustrado en la figura (1.16)-D.

1.6.2.5 Codificación por sincronía.

Este sistema de codificación, se basa en que las neuronas procesan diferentes bits de información del mismo objeto disparando simultáneamente, lo cual sugiere que las neuronas que disparan de manera sincronizada en una población de neuronas, pueden proveer información acerca del significado global del estímulo, tal como se observa en la figura (1.16)-E.

1.6.2.6 Codificación por fase

En este sistema de codificación, los potenciales de acción que son emitidos, son referidos a un punto en el tiempo de una señal periódica tal como se muestra en la figura (1.16)-F. Este comportamiento ha sido observado como medio de codificación, en el hipocampo y el sistema olfatorio. Dicho mecanismo decodificación ha sido implementado en sistemas artificiales de discriminación de olores y en la navegación robótica.

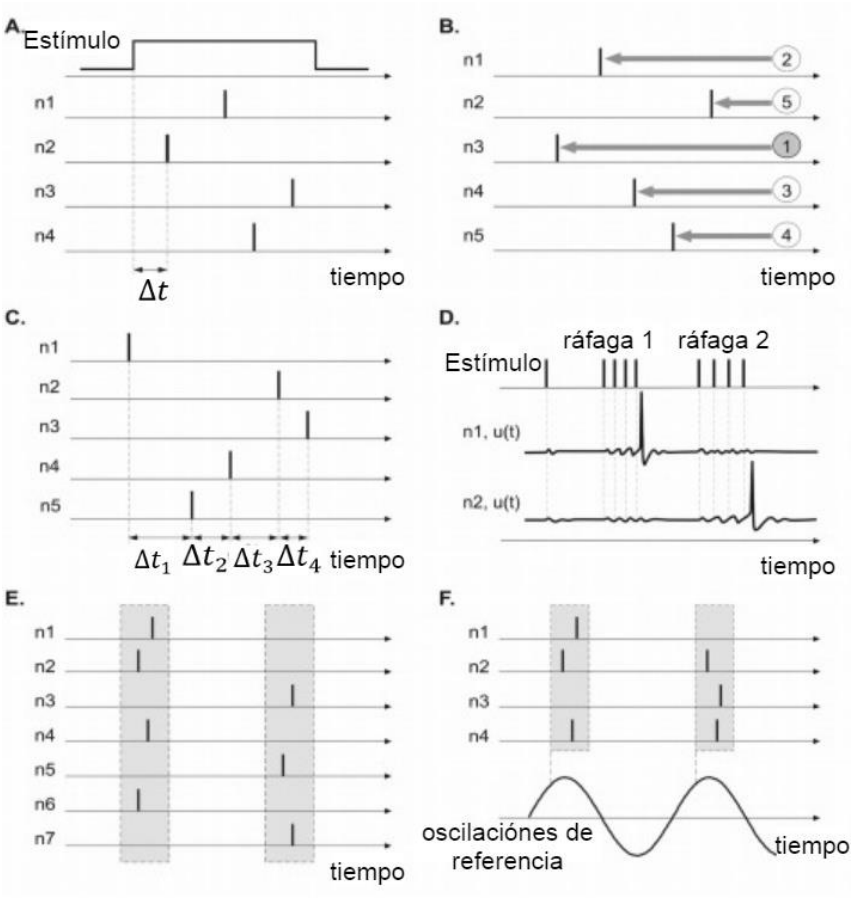

Fig. 1.16 Codificación por pulsos. Extraído de [25]

1.6.3 Conclusión

Las redes artificiales de tercera generación usan como unidades computacionales neuronas pulsadas para procesar la información y existen diversos modelos para aproximar la respuesta biológica del modelo de Hodgkin-Huxley, al igual que la sinapsis y la plasticidad sináptica. A partir de estos tres modelos se pueden construir redes artificiales con diferentes niveles de abstracción y su comportamiento dependerá del modelo neuronal, el tipo de sinapsis, y la ventana de aprendizaje seleccionada; sin embargo, algo común a las redes será la manera de cómo extraer la información y, qué método usar, dependerá de la arquitectura, la aplicación y la dinámica neuronal asociada a los modelos presentados.

1.7 Referencias

[1] S. Daves., *Learning in spiking neural networks.,* Manchester.: Phd Thesis. University of Manchester.

[2] W. Mass, *Networks of Spiking Neurons: The Third Generation of Neural Networks Models.,* Austria: Institute for Theoretical Computer Science Technische Universität Fraz Klosterwiesgasse, 1997.

[3] J. V. Tranquillo, «Quantitative Neurophysiology. Synthesis Lectures on Biomedical Engineering #21,» Morgan & Claypool., 2008.

[4] H. P.-M. &. S. Bohte, *Computing with Spiking Neuron Networks,* Universit de Lyon Laboratoire de Recherche en Informatique - INRIA - CNRS bat. 490, Universit Paris-Sud and Centrum Wiskunde & Informatica, Amsterdam, The Netherlands.

[5] E. M. Izhikevich, «Simple Model of Spiking Neurons,» *IEEE transactions on neural networks.,* vol. 14, nº 6, 2003.

[6] E. M. Izhikevich, «Polychronization: Computation with Spikes,» *Neural Computation MIT,* 2005.

[7] Y. Z. R. W. &. J. Z. Quansheng Ren, «Optical spike-timing-dependent plasticity with weight-dependent learning window and reward modulation,» *Optical Society of America,* vol. 23, nº 19, 2015.

[8] D. J.Heeger, *Poisson Model of Spike Generation,* Lecture Notes Center for Neural Science, New York University, 2005.

[9] D. J.Heeger, *Synaptic Input,* Lecture Notes, Center for Neural Science, New York University, 2005.

[10] W. G. a. W. M. Kistler, «Spiking Neuron Models: Single Neurons, Populations, Plasticity.,» de *Chapter 4. Formal Spiking Neuron Models.*, Cambridge University Press, 2002.

[11] A.-M.-M.-M. a. J. V. F.-V. Taras Lakymchuk, «Simplified spiking neural network architecture and STDP learning algorithm applied to image classification,» *EURASIP Journal on Image and VideoProcessing. Springer open journal.,* 2015..

[12] O. Booij., *Temporal Pattern Classification using Spiking Neural Networks,* M.Sc Thesis. Intelligent Sensory Information Systems Informatics Institute, Faculty of Science,Universiteit van Amsterdam.

[13] Y. D. &. N. Caporale, «Spike timing Dependent Plasticity: A Hebbian Learning Rule,» *Annual Review of Neuroscience,* p. 24, 2008.

[14] G.-Q. B. &. H.-X. Wang, «Temporal asymmetry in spike timing-dependent synaptic plasticity.,» *Elsevier.Physiology & Behavior.,* p. 5, 2002.

[15] M.-m. P. Guo-qiang Bi, «Synaptic Modifications in Cultured Hippocampal Neurons: Dependence on Spike Timing, Synaptic Strength, and Postsynaptic Cell Type,» *The Journal of Neuroscience,* 1998.

[16] D. D. ,. G.-Q. B. Rober C. Froemke, «Temporal Modulation of spike-timing-dependent plasticity.» *Frontiers in synaptic neuroscience.,* vol. 2, nº 19, p. 49, 2010.

[17] F. Ponulak, «Analysis of the ReSuMe Learning process for spiking neural networks.,» *International Journal of Applied Mathematics and Computer Science.(AMCS),* vol. 18, nº 2, 2008.

[18] K. D. M. a. L. A. Sen Song, «Competitive Hebbian Learning through Spike Timimg Depend Synaptic Plasticity,» *Nature Neuroscience.,* 1999.

[19] A. R. R. S. a. S. H. Guetig R., «Learning input correlations through non-linear temporally asymmetric Hebbian plasticity,» *Journal of Neuroscience,* vol. 23, nº 9, 2003.

[20] A. K. &. Filip Ponulak, «Comparison of supervised learning methods for spike time coding in spiking neural networks» *International Journal of Applied Mathematics and Computer Science (AMCS),* vol. 16, nº 1, 2006.

[21] S. G. &. T.-F. L. Toby Lightheart, «Spike-Timing-Dependent Construction» *Neural Computation, MIT Press Journal,* nº 25, p. 2611–2645, 2013.

[22] D. ,. G. J. F. D. H. W. C. M. A.-S. M. J. O. &. W. S. Purves, Neuroscience, La Jolla, California.: Elsevier, Third Edition..

[23] T. Troyer, «An introduccion to computational Neuroscience» Draft Manuscript, Copyright Todd W. Troyer, 2005, 2007.

[24] F. P. &. AndrzejKasinski., «Introducction to spiking neural networks: information processing, learning and aplications» *Acta Neurobiologiae Experimentalis,* vol. 71, pp. 409-433, 2011.

[25] A. D. R. V. R. Simon Thorpe*, «Spike-based strategies for rapid processing» *Elsevier ,* pp. 715-725, 2001.

CAPÍTULO 2

“El hecho de que en las operaciones no aritméticas el cerebro humano, más pequeño y menos rápido, opere con mayor eficacia con los grandes y rapidísimos computadores electrónicos constituye un formidable tributo a la sutileza con que ha sido circuitado y programado nuestro cerebro.”

Carl Sagan, Los Dragones del Edén, 1977.

2 Implementación en software de una red neuronal pulsada con STDP no supervisado para el reconocimiento de caracteres

2.1 Introducción

En este capítulo se planteará y describirá un algoritmo para implementar una red neuronal de tercera generación con la cual podremos realizar un proceso de reconocimiento de caracteres a blanco y negro “simples”, en un programa que pueda ser simulado vía software, con el principal objetivo de proporcionar elementos suficientes para la concepción y comprensión de dicha red y con ello, llevar al sistema a una emulación que pueda ser implementada en hardware.

Para realizar la implementación de un sistema de reconocimiento de caracteres, tendremos que establecer una arquitectura de red neuronal para realizar esta tarea y una vez definida, someter a un proceso de aprendizaje todos los elementos de la red (neurona y sinapsis). Dado que el proceso de aprendizaje es el que permitirá la retención de la información en la red, es necesario determinar la manera en cómo se dirigirá este proceso y con qué método se realizará a la red para inducirlo. En general, el aprendizaje en cualquier red artificial “es un proceso en que los parámetros libres de la red neuronal son adaptados a través de un proceso de estimulación del medio en el cual se encuentra inmersa, y el tipo de aprendizaje estará determinado por la forma en que tiene lugar el cambio de estos parámetros”, tal como describe Haykin en [1]

En nuestro caso, implementaremos el proceso de STDP no supervisado para inducir el aprendizaje, es decir, la red no sabrá a qué resultado llegar dado que no estableceremos ningún patrón objetivo a alcanzar para que la red neuronal pueda guiarse relacionando este valor deseado con la entrada y la salida, tal como lo haría un proceso de STDP supervisado, donde si lo hay y qué en

determinado momento nos pudiera indicar que la fase de aprendizaje y reconocimiento ha terminado. En general, un aprendizaje no supervisado, independientemente que sea o no usado STDP como proceso para inducir el aprendizaje, o bien que no se trate de una red de tercera generación, como la que se plantea en este caso de estudio, si no en general de una red artificial de cualquier tipo, es que el aprendizaje no supervisado intenta generar un único conjunto de pesos sinápticos (w_{ij}) para una sola clase de patrón de entrada y el objetivo del aprendizaje es que esos pesos sinápticos se ajusten de manera autónoma hasta que una condición de equilibrio se dé en el sistema, como describe Nuño Maganda en [2]. Cabe mencionar que, en lo que respecta a esta condición de equilibrio, en las publicaciones referentes a la implementación de sistemas de reconocimiento de patrones equipados con STDP no supervisado, que se muestran a lo largo de este capítulo, se presenta de manera comportamental, es decir, cuando ya no se presentan cambios, ni en los pesos, ni en las corrientes post sinápticas.

El proceso de aprendizaje no supervisado dirigido por STDP es de gran interés principalmente en el área de las neurociencias y neuro-computación porque es el entorno natural en el que se encuentra una red neuronal biológica, es decir, las neuronas no “conocen” de manera inmediata lo que hay en el medio externo ni tienen manera de comparar los fenómenos que ocurren con un conocimiento previo inicialmente. En este sentido, el algoritmo de STDP no supervisado permite estudiar cómo se dan los cambios sinápticos a lo largo del tiempo en las redes. También ayuda a determinar cómo se forman las estructuras a partir del proceso de aprendizaje inducido por STDP y permite el estudio de la codificación espacio-temporal de la información que tiene lugar en estas estructuras, así como la forma en cómo son detonadas las memorias debido a estímulos externos. Por otro lado, en cuanto al área de la ingeniería respecta, lo que se busca es aprovechar estos conocimientos teóricos y experimentales para implementar y desarrollar sistemas de inteligencia artificial.

Para implementar el sistema de reconocimiento de caracteres necesitaremos de diversos elementos para posibilitarlo, los cuales son:

- Un modelo neuronal con una alta plausibilidad biológica como unidad computacional de la red.
- Un modelo sináptico para establecer la forma de generación de corriente y de asociación entre neuronas.
- Una regla de aprendizaje biológica, en nuestro caso de STDP, de la cual será necesario la selección de un protocolo de asociación de pulsos y una ventana de aprendizaje para inducir el cambio de los pesos sinápticos.
- Una regla de asociación de memoria, en nuestro caso, de la regla de relajación como medio para determinar el peso sináptico total que en conjunto con STDP, describirá la función de la memoria a largo plazo.
- Una arquitectura de red.
- Una ventana de trabajo.
- Un algoritmo para determinar la interacción entre todos los elementos de la red y llevar a cabo la tarea de reconocimiento de caracteres.

Tales elementos serán descritos a lo largo del capítulo y nos basaremos en el capítulo 1, para seleccionar los modelos que modelaran la red y en el “material complementario” para fundamentar el comportamiento y la descripción del algoritmo.

2.2 Ventana de trabajo

Las redes neuronales artificiales de tercera generación no son sistemas que puedan simularse de manera infinita en el tiempo. Esto se ve justificado en la manera en cómo se procesa la información en las computadoras actuales (las cuales llevan el procesamiento de las tareas de manera secuencial y no es completamente paralela) así como el que los recursos de memoria de la que disponen es finita.

Una red neuronal que se desee simular durante un periodo muy largo de tiempo se tendría que enfrentar principalmente a las limitaciones de memoria, ya que basándonos en que la información se encuentra codificada espacio-temporalmente en la red, todos estos pulsos tendrían que ser almacenados para posteriormente ser procesados y decodificados para tener algún conocimiento acerca de los que nos está “diciendo” la red; dado que conforme las dimensiones de la red se incrementan, el volumen de la información lo hace también, por lo que dicha proposición no hace factible la simulación de redes que intenten replicar la estructura biológica ni en número ni en recursos. En este sentido, existe una técnica en el área de la neuro-computación que se usa para resolver este problema y se conoce como “ventana de tiempo”, la cual define un periodo de tiempo fijo durante la cual operan todos los elementos de la red neuronal, como describe Troyer en [3]

El tamaño de la ventana de tiempo en general, se determina experimentalmente y debe ser lo suficientemente grande para que se induzca el aprendizaje en la red, en caso contrario, podría necesitarse de varias corridas de programa o en el peor de los casos podría no inducirse el aprendizaje, como nos hace saber de manera puntual Vestbøstad en [4], y Merino Mallorquí y Cosp Vilella en [5].

2.3 Neurona Izhikevich

Basado en el trabajo de Izhikevich específicamente en [6], el modelo neuronal que tiene una mayor plausibilidad biológica es el modelo de Hodgkin-Huxley y precisamente es el modelo que aborda características eléctricas de la neurona. Sin embargo, existen otros modelos matemáticos como los mostrados en la sección (1.3) del capítulo 1 que son los más comunes al momento de realizar implementaciones en software y hardware dado su bajo coste computacional. En la figura (2.1) se puede observar que el modelo de Izhikevich es el modelo que se aproxima más al modelo neuronal de Hodgkin–Huxley en cuanto a su plausibilidad biológica y con el menor coste computacional, lo que nos garantizará un comportamiento aproximado a las

características eléctricas de la neurona biológica tales como: Reposo, activación, re polarización e hiper polarización de sus potenciales de acción así como algunas características específicas de generación en los mismos, como umbral de disparo, período refractario y ruptura de ánodo descritos en el trabajo de Joseph Tranquillo en [7], los cuales, en su conjunto determinan la dinámica neuronal. Esto permitirá obtener características similares cuando se proporcione el estímulo necesario al modelo neuronal de Izhikevich, generando potenciales de acción en instantes de tiempo aproximados tal como la haría una neurona biológica, lo cual es de vital importancia, ya que la información se encuentra contenida en los instantes específicos de tiempo de los potenciales de acción.

A diferencia del modelo de Hodgkin-Huxley donde el potencial de membrana está asociado principalmente a tres tipos de corrientes iónicas: sodio, potasio y cloro (corriente de fuga) como describe Sterrat, Graham, Gillies y Willshaw en [8], el modelo de Izhikevich solo depende de la corriente de sodio que se ve reflejada por la variable “v” (potencial de membrana) y de la corriente de potasio asociada a la variable “u”, conocida como variable de recuperación.

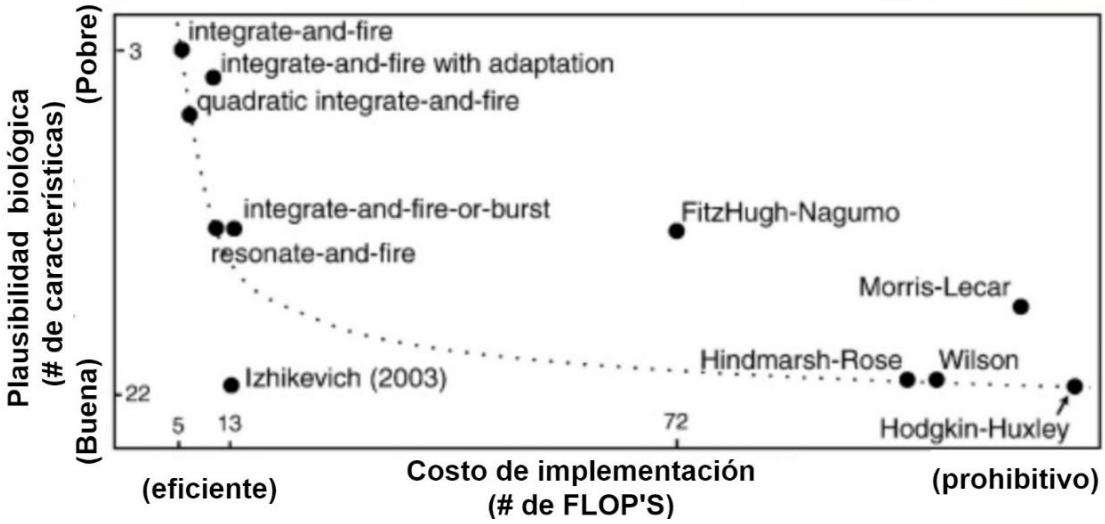

Fig. 2.1.Plausibilidad biológica de diversos modelos neuronales vs Costo computacional. Extraído de [6].

El modelo de Izhikevich, reproduce la actividad de las neuronas biológicas ubicadas en la corteza cerebral, las cuales, en cuanto a su comportamiento,

se dividen en integradores y resonadores. Los modos RS (Regular Spiking), IB (Intrinsically Bursting) y CH (Chattering), tienen comportamiento integrador, el modo FS (Fast Spiking) tiene un comportamiento resonador, mientras que el modo LTS (Low –Threshold Spiking) exhibe comportamiento tanto integrador como resonador. [9]

En nuestro caso, para la implementación del sistema de reconocimiento de caracteres, usaremos la neurona Izhikevich en su modo RS “Regular Spiking” gracias a que se comporta como un integrador, lo cual, le confiere tres características en cuanto a su operación y que describe Izhikevich en [9]:

- Tras su inhibición, la generación de pulsos tiende desaparecer, contrario a lo que sucede en las neuronas que se comportan como resonador, las cuales tras su inhibición podrían tender a la generación.
- En un integrador, los umbrales de disparo se encuentran bien definidos.
- Un incremento de la frecuencia de los estímulos de entrada contribuye a la rápida generación de pulsos.

Específicamente, el modo RS, conforme la amplitud del estímulo incrementa, la frecuencia entre potenciales de acción (spikes) lo hace también; este efecto se ve ilustrado en la figura (2.2), donde una corriente sináptica excitatoria constante cada vez mayor, provoca una mayor cantidad de potenciales de acción. Este comportamiento ha sido observado en neuronas espinosas estrelladas de las capas II a la V de la corteza cerebral.

En cuanto a sus características eléctricas, el modo RS del modelo de Izhikevich tiene un potencial de reposo igual -60 mV, un potencial de disparo de -40 mV , un potencial pico por despolarización de 35 mV y al igual que la neurona de Hodgkin-Huxley tiene un período refractario para que la neurona vuelva a pulsar tras un estímulo y que ronda aproximadamente un período de tiempo de 30 milisegundos, valor que es asociado al parámetro “a” (a=0.03) y a una re polarización por debajo de los -50 mV, la cual es dependiente de la frecuencia y amplitud de la corriente con la que es estimulada, característica que se encuentra asociada al parámetro “c” (c= -65).

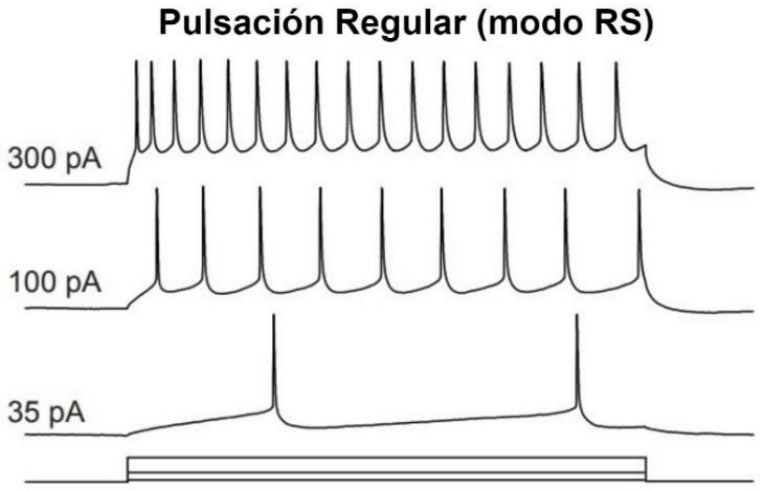

Fig. 2.2 Comportamiento del potencial de membrana del modo RS tras la estimulación de diferentes corrientes sinápticas excitatorias. Extraída de [9].

El comportamiento de la neurona Izhikevich en su modo RS puede describirse a través de su retrato de fase (traducido del inglés “phase portrait”); en el modelo de Izhikevich, las variables “u” y “v” interactúan mutuamente para describir el comportamiento eléctrico de la neurona. El retrato de fase es una representación de todas las trayectorias geométricas de un sistema dinámico, y no es exclusivo para la representación de la actividad neuronal, tema que se aborda en mayor detalle en el trabajo de Nieto, Gómez y Moreno que precedió a esta tesis [10] y que se desarrolla más ampliamente en el trabajo de Izhikevich en [9]. Sin embargo, para nuestros fines bastará saber que el modelo RS conforme se incrementa la corriente sináptica, la actividad eléctrica será más intensa, es decir, la generación de potenciales de acción (spikes) es mayor, mientras que si la corriente sináptica disminuye, la generación de potenciales de acción también será menor. Esta característica será aprovechada para diseñar el algoritmo del sistema de reconocimiento de caracteres. En la tabla (2.1) se presentan los parámetros del modo RS de la neurona Izhikevich, el cual se encuentra descrito por las ecuaciones (1.2) - (1.3) de la sección (1.3.2) del capítulo 1 y que usaremos en nuestra implementación.

Parámetros	Valores
a	0.02
b	0.2
c	-65
d	8
dt	0.005 [ms]

Tabla 2.1.Parámetros del modo RS del modelo Izhikevich para realizar la implementación vía software.

2.4 Protocolo de STDP para inducción del aprendizaje.

2.4.1 Protocolo de asociación de pulsos: Todos contra todos.

STDP necesita de un protocolo de asociación de pulsos para determinar el cambio del peso sináptico entre dos o más neuronas, lo cual desde el punto de vista de los experimentos biológicos realizados por Gerstner y Pfister en [11], muestra asociaciones entre pares de pulsos pre y post sinápticos totalmente aislados, donde “no está claro como STDP puede ser aplicado a trenes de pulsos generados naturalmente, lo que involucra muchos pulsos y muchos posibles casos de “emparejamiento” de estos pulsos; específicamente en este caso, tampoco está bien definido cómo la plasticidad de una sinapsis se puede construir en el tiempo” como se describe por Desai e Izhikevich en [12]. Computacionalmente se propusieron varias alternativas para resolver este problema y en [12] se muestran una de las varias formas para determinar el protocolo de asociación entre pulsos pre y post sinápticos, el cual se conoce como “All to all implementation of STDP” o “implementación de STDP de todos contra todos” y que se encuentra basado en el protocolo de asociación de pares que se encuentra biológicamente en las neuronas.

El protocolo de STDP de “todos contra todos” establece que el cambio sináptico neto es debido a la contribución de pequeños cambios en el peso sináptico debido a todos los casos posibles por asociación de pares pre y post sinápticos, es decir, que cada una de las diferencias temporales (spike timing) entre estos pulsos contribuirán con una porción del cambio en el peso sináptico total. La implementación de este protocolo consiste en que cada pulso pre sináptico se vinculará con todos los pulsos post sinápticos que caen dentro de

la ventana de aprendizaje, calculando las relaciones de tiempo (spike timing) entre los pares de pulsos pre-post sinápticos que contengan, para lo que se hace uso de la ecuación (1.23) de la sección (1.5) del capítulo 1, tomando en cuenta que las diferencias temporales (spike timing) entre un pulso pre y post sináptico, solo tendrán efecto durante la ventana de aprendizaje, es decir, en cada pulso pre sináptico se centrará en una ventana de aprendizaje sobre la cual, solo aquellos pulsos post sinápticos con diferencias temporales que están en el dominio de la región LTP o LTD contribuirán a la modificación del peso sináptico tal como se muestra en la figura (2.3).

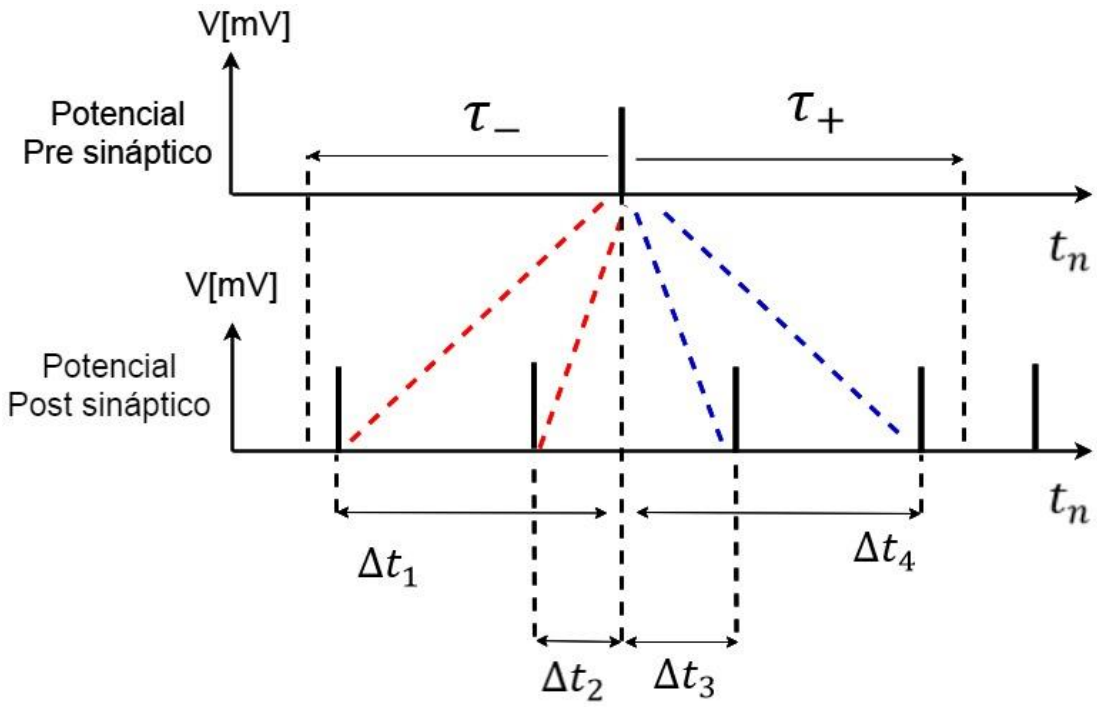

Fig. 2.3 Establecimiento de la ventana de aprendizaje en el protocolo de asociación de "todos contra todos". Líneas punteadas en rojo representan depresión por efecto de la región LTD, mientras que líneas en azul potenciación por efecto de la región LTP.

En este sentido, el protocolo "todos contra todos" toma todos los posibles casos de asociación por pares y no elimina ninguno de ellos a menos que no estén dentro del dominio de la ventana de aprendizaje. Obsérvese que dicho protocolo es ideado para plataformas computacionales, ya que para poder determinar la asociación de pulsos es necesario guardar en la memoria del programa todas las secuencias de pulsos pre sinápticas y post sinápticas, lo

cual es beneficioso al momento de simplificar el algoritmo, ya que elimina la cuestión planteada inicialmente de cómo realizar los pares de pulsos si experimentalmente no se ha planteado un sistema de asociación de pares bien definido.

2.4.2 Selección de la ventana de aprendizaje

Para determinar el cambio en los pesos sinápticos que produce la diferencia temporal individual de cada par de pulsos pre y post sinápticos, es necesario seleccionar una ventana de aprendizaje, de la cuales ya se han presentado varias opciones en la sección (1.5) del capítulo 1. En nuestro caso, seleccionaremos la ventana de aprendizaje clásica de STDP, la cual ha demostrado tener un mayor desempeño en tareas de clasificación; específicamente, en el trabajo de Diehl y Cook en [13] se reporta un 95% de eficiencia en STDP no supervisado, haciendo uso de la ventana de aprendizaje doble exponencial en comparación con la ventana de aprendizaje rectangular que reportó un rendimiento 93.5%.

Sin embargo, cabe mencionar que la manera de efectuar la modificación del peso sináptico por el entorno en que estamos trabajando difiere de lo que naturalmente se podría observar. En este sentido, un entorno computacional nos permite comparar muchos pulsos en una ventana de tiempo guardándolos en la memoria del programa para posteriormente asociarlos y determinar cómo este conjunto de pulsos modifica el peso sináptico en función de una ventana de aprendizaje en particular, mientras que biológicamente sucede de manera instantánea habiendo certeza de ello, ya que en la teoría de asociación de pulsos pre y post sinápticos solo contribuyen los pulsos pre y post sinápticos más cercanos (lo que origina los protocolos de pares, triplet's o cuadruplet's) y en los instantes de tiempo en que ocurre dicha modificación se genera una modificación en la corriente de manera casi instantánea [11]. Esto no quiere decir que la información no se esté procesando de manera espacio-temporal, ya que las componentes espaciales dadas por la localización y las componentes de tiempo dadas por las diferencias temporales (spike timing)

entre los pulsos pre y post sinápticos, se encuentran presentes; lo que quiere decir es que la información no se está procesando en tiempo real y que nos estamos alejando de la plausibilidad biológica en cuanto las características biológicas que deseamos reproducir.

Hacemos mención de lo anterior, porque este protocolo de asociación de pulsos de “todos contra todos”, el cual hemos seleccionado y que normalmente es usado en implementaciones en software, provoca un cambio en las ecuaciones que aplican para el cálculo del cambio del peso sináptico (Δw) que ocurre de manera biológica y que ya hemos mencionado en la sección (1.5) del capítulo 1. En este caso, cada una de las diferencias temporales entre pulsos pre y post sinápticos genera un peso sináptico “W” por cada una de ellas como se describe en la ecuación (2.1) los cuales están en función de la ventana de aprendizaje seleccionada para que, posteriormente, la sumatoria de esos pesos sinápticos genere un cambio en el peso sináptico global (Δw_{ij}) que afecta a esa sinapsis en particular, tal como se describe por la ecuación (2.2), es decir, este conjunto de diferencias temporales contribuye a la modificación de la sinapsis general y no como se describe en los experimentos biológicos en que cada diferencia temporal asociado a un pulso pre y post sináptico dados por el protocolo de pares, genera una modificación en el peso sináptico; estas ecuaciones se basan en [10] (Nieto, Gómez y Moreno) y [14] (Gerstner y Sjöström).

En la ecuación (2.2), las variables t_i^n son los n-ésimos instantes de tiempo en que llegan los pulsos pre sinápticos $n = 1,2,3..N$ y t_j^f son los f-ésimos instantes de tiempo en que llegan los pulsos post sinápticos $f = 1,2,3..N$ mientras que Δw_{ij} es el cambio sináptico global que hay entre la neurona i-ésima que genera los pulsos pre sinápticos y la j-ésima que genera los pulsos post sinápticos y que las relaciona.

$$W(\Delta t) = \begin{cases} +A_{+}e^{\left(\frac{-\Delta t}{\tau_{+}}\right)} & si\ \Delta t \geq 0 \\ -A_{-}e^{\left(\frac{\Delta t}{\tau_{-}}\right)} & si\ \Delta t < 0 \end{cases} \quad \textbf{(2.1)}$$

$$\Delta w_{ij} = \sum_{f=1}^{N}\sum_{n=1}^{N} W(t_j^f - t_i^n) \quad \textbf{(2.2)}$$

Cabe mencionar que en este método pueden darse traslapes entre las ventanas de aprendizaje, por lo que, en un conjunto de pulsos post sinápticos pueden existir pulsos que contribuyan al cambio sináptico en 2 ventanas de aprendizaje diferentes al mismo tiempo, tal como se muestra en la figura (2.4), donde las ventanas de aprendizaje “A” y “B” comparten un mismo pulso, el cual tiene un efecto distinto para cada una de ellas. En la ventana “A”, ese pulso provocará potenciación, mientras que para la ventana “B”, tendrá el efecto opuesto, provocando la depresión de la sinapsis. Por otro lado, puede darse el caso de ventanas que solo contribuyan a la potenciación, como la que se observa en la ventana “C” y en algunas otras donde no se contribuya con ningún cambio como se observa en la ventana “D”. De acuerdo a la ecuaciones (2.1) - (2.2), el efecto de todas esa ventanas de aprendizaje (A, B, C y D) se sumarán para generar un cambio en el peso sináptico que relaciona las neuronas pre sinápticas i-ésimas y las neuronas post sinápticas j-ésimas, por lo que, aunque se tengan ventanas de aprendizaje vacías, pesos sinápticos potenciados o depreciados, al momento de integrarse, dará como resultado una potenciación o depresión en la sinapsis, dependiendo cuál sea el efecto predominante.

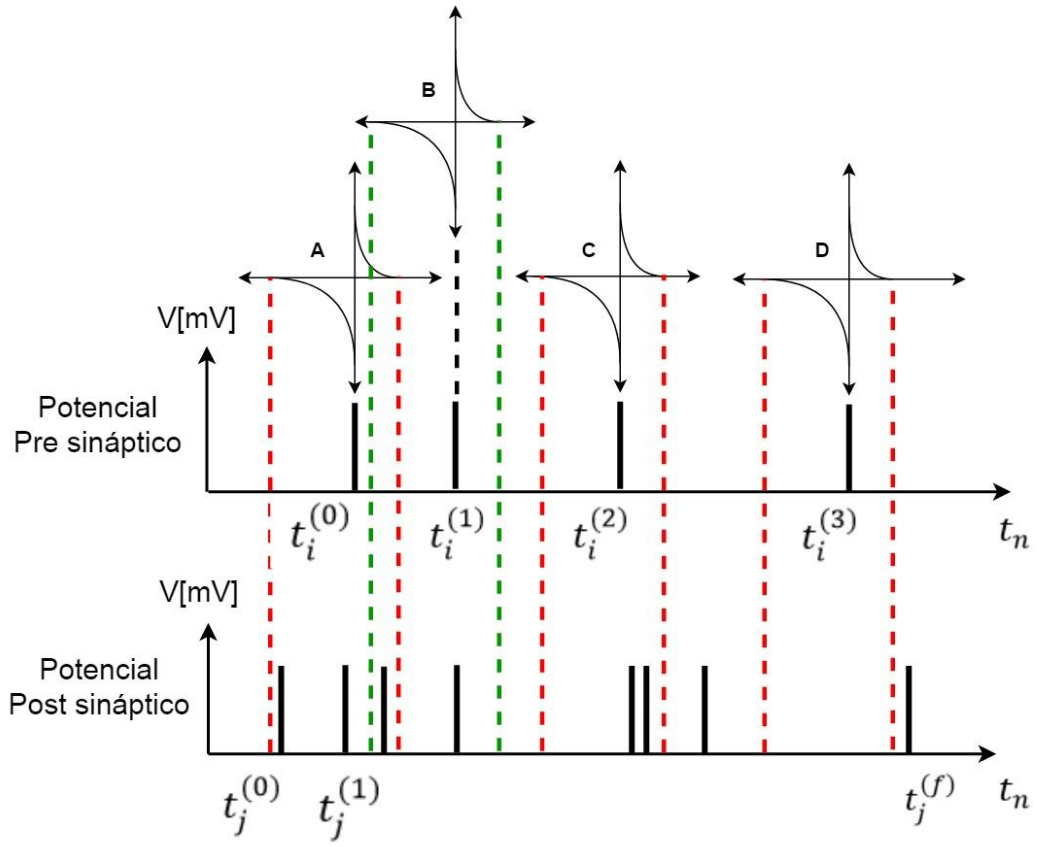

Fig. 2.4 Casos asociados al uso del protocolo de STDP en su conjunto: protocolo de asociación de pulsos y de la ventana de aprendizaje.

Para realizar la implementacion usaremos los parametros presentados por Nobukawa y Nishimura en [15] para la ventana de aprendizaje y que se muestran en la tabla (2.2).

Parámetros	Valores
A_+	0.10
A_-	0.12
τ_+	20
τ_-	20

Tabla 2.2. Parámetros que usaremos para modelar la ventana de aprendizaje de STDP (ventana tradicional).

Obsérvese que los valores que usaremos en la implementacion de la ventana de aprendizaje, las regiones LTP y LTD tienen la misma magnitud en lo que respecta al tiempo y solo difieren ligeramente en la magnitud del cambio sináptico, lo que permite satisfacer la ecuación (1.24) de la seccion (1.5 del

capítulo 1) para garantizar la discriminacion entre pulsos pre y post sinapticos induciendo la potenciacion entre la sinapsis, cuando existe corelación entre ellos y promoviendo la depresión cuando no la hay.

2.5 Regla de asignación de memoria: Regla de relajación.

Para relacionar la forma en cómo STDP afecta al peso sináptico, usaremos la ecuación (2.3), que es una variación de la ecuación (1.31) presentada en la sección (1.5.7) del capítulo 1. Esta variación que ha sido reportada por Humaidi y Kadhim en [16], por Lakymchuk, Rosado-Muñoz, Guerrero-Martínez, Bataller-Mompeán y Francés-Víllora en [17] y por Gupta y Long en [18], se conoce como regla de relajación.

$$w_{new} = \begin{cases} w_{old} + \eta \Delta w (w_{max} - w_{old}) & si\ \Delta w \geq 0 \\ w_{old} + \eta \Delta w (w_{old} - w_{min}) & si\ \Delta w < 0 \end{cases} \qquad \textbf{(2.3)}$$

Donde:

- w_{old} es el peso sináptico que adquiere al inicio de cada época de entrenamiento.
- w_{new} es el peso sináptico que adquiere al final de cada época de entrenamiento.
- η es la velocidad de aprendizaje, encontrándose en un rango de 0 a 1.
- En sinapsis de tipo excitatorias $w_{min} = 0$ y $w_{max} = 1$, mientras que en sinapsis de tipo inhibitorio $w_{min} = -1$ y $w_{max} = 0$.
- Δw es el cambio en el peso sináptico generado por el protocolo de asociación de pulsos seleccionado para trabajar en STDP.

A partir de esta regla de aprendizaje se puede observar que el valor del peso sináptico en sinapsis excitatorias se encontrará en un rango de 0 a 1 ($0 \leq w_{new}, w_{old} \leq 1$) y en las sinapsis inhibitorias en el rango opuesto, de 0 a -1 ($-1 \leq w_{new}, w_{old} \leq 0$). En la sección (2.8) se determinará el tipo de sinapsis que tendrá la red (excitatoria o inhibitoria), las cuales estarán en función de la arquitectura y propiamente del algoritmo reconocimiento de caracteres.

2.6 Modelo sináptico: modelo de corriente simple.

De los modelos presentados en la sección (1.4) del capítulo 1, se elegirá el de menor coste computacional, en este caso, el que describe la ecuación (1.11) que es la función de corriente simple (véase sección (1.4.3)). En este modelo, dado que no se define el período en que tiene efecto la sinapsis, nosotros usaremos un nivel de corriente constante durante toda la ventana de tiempo, tal como se realiza en [18]. En este caso, dado que estamos usando el modelo de Izhikevich y el valor de la corriente se encuentra expresada en unidades de pico Amperes para este modelo neuronal, se usará la ecuación (2.4), donde se sustituye el valor de la variable “q” de la ecuación (1.11) que representa la carga eléctrica total inyectada a la neurona vía sinapsis por la unidad del modelo. Específicamente en [15] (Nobukawa y Nishimura) y [19](Ren, Zhang, Wang, y Zhao) se presenta esta ecuación de manera adimensional, ya que la unidad del modelo sináptico dependerá del modelo neuronal usado o del valor que deseemos darle por defecto a la contribución del peso sináptico individual a la corriente post sináptica.

$$I_j(t) = \sum_i w_{ij} \sum_k H\left(t - t_k^i\right) \ [pA] \qquad \textbf{(2.4)}$$

Obsérvese que la corriente post sináptica que experimentará la neurona post sináptica dependerá del protocolo de STDP así como de la regla de relajación, lo cual a su vez determinará el comportamiento eléctrico de la neurona post sináptica, retroalimentando el sistema.

En nuestro caso, para la implementación en software ninguna de las sinapsis presentará algún tipo de retardo para simplificar la dinámica de la red neuronal.

2.7 Arquitectura de la red

A diferencia de las redes neuronales de segunda generación que necesitan al menos tres capas de neuronas para realizar el reconocimiento de algún patrón y que inclusive, dependiendo la naturaleza del problema, pueden incluir

muchas más (capas ocultas) las redes de tercera generación solo necesitan 2 capas para realizar esta misma tarea. Este tipo de arquitectura de red ha sido reportada en [15], [16], [17], [19] y por Humaidi y Kadhim en [20] la cual ha sido usada específicamente en el reconocimiento de caracteres digitales en blanco y negro consistiendo en una capa de entrada que relaciona los estímulos externos, con una capa de salida que genera una secuencia de pulsos dependientes del estímulo de la primera capa. Específicamente, en la clasificación de imágenes se propone en [16], [20] y por Christophe, Mikkonen, Andalibi, Koskimies,y Laukkarinen en [21], que cada pixel de la imagen sea asociado a una neurona de la capa de entrada y de esta manera, un valor alto en corriente significa negro y un valor bajo de corriente significa blanco. En esta red, dado que lo que se plantea es un aprendizaje no supervisado, lo que se pretende es que exista competición entre las diferentes neuronas de la capa de salida, para que solamente una de las neuronas de salida logre asociarse a cada caracter mostrado, por lo tanto, la arquitectura de la red neuronal tendrá en la capa de entrada tantas neuronas como pixeles contenga la imagen, y tantas neuronas de salida como patrones deseemos que la red neuronal pueda reconocer. En cuanto a las relaciones sinápticas, todas las neuronas de la capa de entrada se relacionarán mediante sinapsis de tipo excitatoria con las neuronas de la capa de salida, las cuales a su vez tendrán sinapsis inhibitorias laterales para relacionarse con sus neuronas vecinas, esto con el objetivo de que las neuronas de la capa de salida reduzcan su actividad eléctrica mutuamente, tal como se muestra en la figura (2.5). Esta arquitectura está íntimamente relacionada en la forma de la ejecución del algoritmo de reconocimiento de caracteres y tendrá una gran relevancia al momento de llevarlo a cabo.

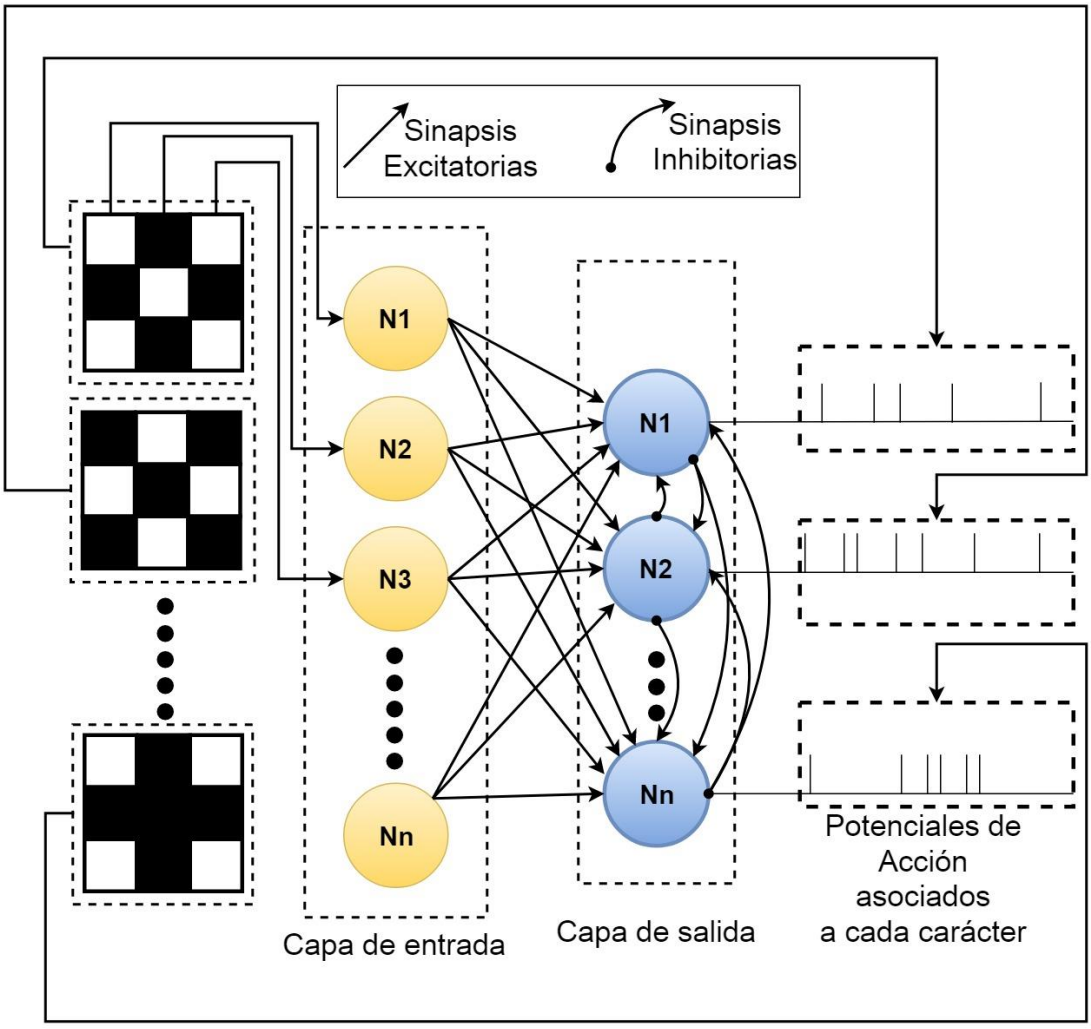

Fig. 2.5. Arquitectura de Red para implementar el algoritmo de reconocimiento de caracteres.

2.8 Algoritmo no Supervisado de STDP

El algoritmo para clasificación de patrones se compone principalmente de dos partes: el entrenamiento o aprendizaje y el reconocimiento. El proceso de aprendizaje tiene como finalidad realizar la modificación de los pesos sinápticos en función de la actividad neuronal asociada y el proceso de reconocimiento tiene la finalidad de evaluar la red con los pesos a los que se ha llegado en la etapa de entrenamiento para clasificar y determinar qué patrón corresponde a su actividad.

En esta sección se describirá en general cómo se genera el proceso de aprendizaje y de reconocimiento, para a partir de ello simplificar algunas de

sus características y generar un algoritmo que pueda ser en ejecutado en una computadora personal.

2.8.1 Descripción del proceso de aprendizaje

Para llevar a cabo de manera exitosa el proceso de aprendizaje como medio para codificar la información de algún estímulo externo y el reconocimiento como una forma de extraerla nuevamente a través de la memoria, es necesario conocer el efecto que tendrán sobre la red las sinapsis al momento de relacionar cada una de sus unidades computacionales: las neuronas.

Basados en el trabajo de Masquelier, Guyonneau y Thorpe en [22], el algoritmo define a todas las neuronas de la capa de entrada como neuronas pre sinápticas i-ésimas y las neuronas de la capa de salida como post sinápticas j-ésimas, unidas mediante sinapsis excitatorias, donde cada una de las neuronas pre sinápticas se relaciona con cada una de las neuronas post sinápticas. Esta estructura tiene 2 propósitos fundamentales:

- Permitir que la actividad de la capa de entrada influya en la actividad de las neuronas de la capa de salida promoviendo la generación de potenciales de acción.
- Establecer el flujo de la información para implementar el protocolo de STDP que tendrá lugar durante el proceso de entrenamiento. En este sentido, todas las sinapsis que relacionan la capa de entrada con la capa de salida y que serán afectadas por el protocolo de STDP, se conocerán como “sinapsis plásticas”, haciendo referencia a que, durante el proceso de aprendizaje, éstas se modificarán como su contraparte biológica descrita por Magee y Johnston en [23].

Por otra parte, dado que este algoritmo es no supervisado, es necesario agregar más elementos a la red neuronal, como hemos mencionado anteriormente; en este caso, relaciones sinápticas inhibitorias para lograr que un solo patrón en la capa de entrada sea aprendido por una sola neurona en la capa de salida. Para lograr este objetivo, es necesario llevar a cabo un proceso conocido como “winer take all” o “ganador toma todo”, el cual consiste

en someter a una competición la actividad eléctrica generada en las neuronas post sinápticas, para que la actividad eléctrica de una sola de ellas (neurona ganadora post sináptica) reduzca la actividad de las neuronas rivales (neuronas vecinas post sinápticas). En este sentido, cada neurona de salida puede establecerse como pre y post sináptica mediante sinapsis de tipo inhibitoria, con lo que cada una de ellas puede ser emisora y receptora de potenciales de acción, dependiendo de las sinapsis que se encuentren activas en ese momento [24] [25]. Cabe señalar que esta estructura solo tiene lugar en la capa de salida y que en ningún momento estas sinapsis inhibitorias, participarán en el protocolo de STDP, razón por la cual, son conocidas como “no plásticas” dado que mantendrán un valor fijo durante todo el procedimiento.

El proceso para establecer el aprendizaje en la red neuronal por lo tanto puede ser resumido y descrito de la siguiente forma:

1. Cuando a la red neuronal se le presenta un estímulo (en nuestro caso una caracter), la primera capa generará los potenciales de acción en función de ese estímulo generado por los pixeles asociados a ellas, codificando la información de manera espacio-temporal, la cual está dada por la relación posición-tiempo de cada uno de los potenciales que conforman los trenes de pulsos de la capa de entrada.
2. Una vez generados los trenes de pulsos de cada una de las neuronas, las sinapsis excitatorias transformarán esos trenes de pulsos en un conjunto de corrientes excitatorias post sinápticas individuales, que llegarán a las neuronas post sinápticas.
3. Una vez generadas las corrientes post sinápticas individuales, las neuronas post sinápticas las integrarán, para generar una corriente post sináptica total, provocando secuencias de pulsos en la capa de salida.
4. Llegado este punto, se someterá a un proceso de competición a las neuronas post sinápticas, ya que alguna de ellas será más sensible al estímulo presentado, por lo que alguna de las neuronas de la capa de

salida tendrá una actividad eléctrica más intensa [22]. Debido a este procedimiento, se puede determinar que una neurona ha sido la ganadora porque se ha presentado una corriente post sináptica excitatoria más intensa que en las demás. Esto es de gran importancia para la red, ya que la actividad neuronal que es más intensa en una sola neurona, por efecto de las sinapsis inhibitorias, provocará que la corriente post sináptica excitatoria (EPSC) en sus vecinas disminuya por efecto de una corriente post sináptica inhibitoria (IPSC) que inyecta la neurona ganadora a partir de sus potenciales de acción, anulando el efecto de las corrientes excitatorias provenientes de las neuronas de la capa de entrada. [24] [25]

5. Una vez establecida la neurona ganadora, tiene lugar el proceso de aprendizaje, en el cual se relacionan todos los pulsos pre sinápticos provenientes de cada una de las neuronas de la capa de entrada con los pulsos post sinápticos de la neurona ganadora mediante el protocolo de STDP. En este proceso, una vez determinada la neurona ganadora, se ignorará la presencia de las neuronas vecinas, realizando tantas repeticiones del algoritmo como sea necesario para inducir el aprendizaje, el cual consistirá en la potenciación y depresión de los pesos sinápticos por las regiones LTP y LTD definidas en nuestro caso por el protocolo de aprendizaje STDP, descrito anteriormente en la sección (2.4).

En general, el proceso para que se pueda inducir el aprendizaje en la red consiste en 4 etapas: 1) codificación del estímulo en la capa de entrada, 2) integración sináptica, 3) selección de la neurona a entrenar por competición y 4) el entrenamiento por el protocolo de STDP para modificar el peso sináptico con base en la actividad de los pulsos pre sinápticos de la capa de entrada y los pulsos post sinápticos de la neurona ganadora cuando estos caen en la ventana de aprendizaje de STDP. Cabe señalar, que el proceso de integración sináptica tendrá lugar durante todo el proceso, ya que sin la presencia de las sinapsis excitatorias no sería posible relacionar los pulsos pre sinápticos de la

capa de entrada con la capa de salida para así generar actividad eléctrica en cada uno de los elementos, ni establecer el protocolo de STPD para el aprendizaje y, sin la presencia de las sinapsis inhibitorias, no se sería posible la selección de alguna neurona por competición, ni aislar su entrenamiento para el reconocimiento de algún caracter.

2.8.2 Descripción del proceso de reconocimiento

El proceso de reconocimiento es similar al proceso de entrenamiento, y la diferencia radica en que el protocolo de aprendizaje de STPD queda inhabilitado y se procede de igual manera que el proceso de aprendizaje. Este procedimiento puede ser descrito en tres partes:

1. Se presenta el patrón a reconocer a la red, de esta manera la primera capa de la red codifica la información para dirigirla nuevamente a la segunda capa.
2. Dado que no existe ya modificación en los pesos sinápticos de las sinapsis y el peso sináptico al que se llegó fue el establecido por el protocolo de STDP, por efecto de integración sináptica de los pulsos provenientes de la primera capa por la sinapsis excitatoria, se generará una corriente post sináptica excitatoria (EPSC) en cada una de las neuronas de la capa de salida y alguna de ellas convergerá a una de las corrientes post sinápticas obtenidas del proceso de aprendizaje.
3. Por efecto de las sinapsis inhibitorias, la neurona asociada genera una corriente post sináptica inhibitoria que tiende a anular el efecto de la corriente post sináptica excitatoria que inyectan las neuronas de la primera capa, anulando la actividad eléctrica de las neuronas post sinápticas vecinas.

Obsérvese, por lo tanto que debido a que la neurona que reconoció el patrón es aquella que tiene la actividad más intensa, el método para extraer la información de la red y determinar qué neurona se asoció a un patrón en específico, será el método basado en codificación por pulsos “el primero en disparar” presentado en la sección (1.6.2.1) del capítulo 1, ya que, por el

modelo neuronal y sináptico, una corriente mayor en la neurona ganadora permitirá que ésta comience a generar pulsos antes que las otras.

2.8.3 Consideraciones para la simplificación del algoritmo.

En [22] se hace énfasis que una red neuronal equipada con STDP se “auto organizará” para cubrir todos los patrones que deseemos que la red “aprenda” y que esta “auto-organización” es lograda basándose en que las sinapsis excitatorias que relacionan la capa de entrada y de salida son las más “adecuadas” a ese patrón, beneficiando a una sola neurona de la capa de salida, la cual supera en generación de potenciales de acción a las vecinas. Esto puede dar lugar a confusiones en la forma en cómo aprende la red, sin embargo, no se debe perder de vista que éste es un proceso de competencia y que este proceso de competencia dependerá de los pesos sinápticos (“w”) con los que se inicializa la red. De esta manera, puede darse el caso en que se presente un patrón y que, debido a los pesos sinápticos de las sinapsis excitatorias que relacionan una neurona de la capa de salida con todas las neuronas de la capa de entrada, se vea beneficiada para que comience un proceso de entrenamiento, pero que cuando se presente otro patrón ocurra lo contrario, que la perjudique y que sea otra neurona la que sea seleccionada para comenzar un proceso de entrenamiento. Este proceso de “auto-organización”, por lo tanto, está definido de manera aleatoria y tiene total dependencia de los pesos sinápticos con los que sea inicializa la red y del patrón que se le presente. Esto abre la puerta para que nuestro proceso de “auto-organización” sea completamente “selectivo”, y que tengamos la oportunidad para seleccionar específicamente a una neurona de la capa de salida y participe en un proceso de aprendizaje.

Por otra parte, en relación con los capítulos anteriores, basado en el trabajo de Dan y Caporale en [26], de manera biológica, en cada sinapsis que se configura de neurona a neurona, puede existir una ventana de aprendizaje diferente en cuanto a la forma, así como en los dominios de tiempo en que las regiones LTP y LTD tienen efecto. Esto quiere decir, en función de nuestra

arquitectura, que si hay “i-ésimas neuronas de entrada”, la “neurona “j-ésima” llevará a cabo “i-ésimas” ventanas de aprendizaje” ya que la ventana de aprendizaje se da por cada estructura sináptica, lo cual computacionalmente incrementa la demanda de recursos de cálculo conforme la red comienza a crecer y hace también impredecible el comportamiento de la red al momento de establecer el proceso de aprendizaje. Dada esta situación, se ha reportado una sola ventana de aprendizaje que ejecuta n-ésimas relaciones sinápticas para llevar a cabo el algoritmo de reconocimiento de patrones.

Biológicamente, basados en [7], [23], [24] y [26] también existen retardos de propagación en los pulsos pre sinápticos durante el proceso de integración sináptica y de retro propagación en los pulsos post sinápticos durante el proceso de STDP, los cuales en nuestro caso serán completamente suprimidos.

Debido a que el proceso para llevar a cabo el algoritmo es de manera secuencial, la integración espacio-temporal tanto en la sinapsis como en el proceso de aprendizaje realizado por STDP, no se realizará de la misma forma a como sucede biológicamente [24], y es la razón del porqué se ejecuta la integración sináptica como el aprendizaje parcialmente sobre una ventana de tiempo, calculando individualmente la corriente post sináptica y después en su conjunto, ya que en caso contrario, implicaría integrar los pulsos “como vayan llegando”. En el caso de implementarse la versión más simple de las características biológicas de los modelos que conforman la red, el algoritmo podría generar más tiempo de cómputo e ineficiencia al momento de ejecutar el código, esto debido a que cada vez que tiene lugar el proceso de STDP, cuando éste asocia un par de pulsos pre y post sinápticos, se tendría que realizar una modificación instantánea en el peso sináptico y por efecto de la integración sináptica, una modificación en la corriente post sináptica instantánea en la capa de salida, y por lo tanto, una modificación en la generación de potenciales de acción que retroalimenta el proceso de STDP en todo momento. Esto nos lleva a la conclusión, que, en todo el proceso de

aprendizaje y reconocimiento, el tiempo solo tiene un significado numérico dentro de la simulación, y todo valor generado por la red no habrá sido generado por un procesamiento en tiempo real.

En resumen, la red que será implementada en software tendrá las siguientes características:

- No existirá ninguna clase de retardos en la sinapsis para la propagación de pulsos pre sinápticos y tampoco en el proceso de STDP, en la retro propagación de pulsos post sinápticos.
- El aprendizaje podrá ser “auto-organizado” o localizado.
- La ventana de aprendizaje de STDP será la misma para toda la red.
- El algoritmo, dada la plataforma que usamos, será llevado a cabo en tiempo no real.
- La integración espacio-temporal de los pulsos será sólo en el sentido numérico.
- Las sinapsis inhibitorias permitirán aislar el entrenamiento a una sola neurona en la capa de salida, lo que permitirá a su vez, que durante el proceso de STDP solo se modifiquen los pesos sinápticos asociados a ella y que durante el proceso de reconocimiento, ésta anule la actividad eléctrica de sus vecinas.

2.8.4 Implementación del algoritmo propuesto para el aprendizaje.

El algoritmo propuesto para el aprendizaje consiste en:

1.- Definir parámetros de la red:

- Definir la ventana de tiempo para tomar los potenciales de acción generados en la red y la cual deberá ser la misma para cada neurona.
- Definir la cantidad de neuronas en la capa de entrada y de salida, la cual deberá corresponder con la cantidad de pixeles de la imagen y la cantidad de patrones que deseemos que la red pueda aprender.
- Establecer los parámetros de las neuronas para que exhiban el modo “Regular Spiking” (RS) del modelo de Izhikevich.

- Establecer los parámetros a usar para modelar la ventana de aprendizaje de STDP tradicional.
- Inicializar aleatoriamente todas la sinapsis excitatorias de la red en el rango de “0 a 1”.
- Definir un número máximo de épocas a realizar. (Cuando nos referimos a las épocas, nos referimos a la cantidad de veces que será necesario ejecutar el algoritmo para inducir el aprendizaje del caracter por la red).

2.-Aplicar un valor alto de corriente en cada una de las neuronas de la capa de entrada que correspondan con pixeles negros y un valor de corriente cero a las que correspondan con los pixeles blancos.

3.-Determinar la actividad eléctrica (potencial de membrana “V”) haciendo uso del modelo de Izhikevich ((1.2) -(1.3) de la sección (1.3.2) del capítulo 1) para i-ésimas neuronas pre sinápticas de la capa de entrada y guardar en la memoria del programa los instantes de tiempo en que se generan los potenciales de acción k-ésimos y que forman los trenes de pulsos i-ésimos provenientes de la capa de entrada.

4.-Determinar las corrientes post sinápticas individuales que generan las neuronas pre sinápticas i-ésimas sobre las neuronas post sinápticas j-ésimas que tienen lugar durante la ventana de tiempo a través del modelo sináptico simple de corriente definida por la ecuación (2.4) y guardarlas en memoria de programa. La corriente post sináptica es el resultado del producto del peso sináptico que las relaciona (w_{ij}) y de la cantidad de potenciales de acción k-ésimos generados por cada neurona pre sináptica i-ésima.

5.- Realizar el proceso de integración sináptica que realizan las neuronas post sinápticas j-ésimas de la capa de salida, lo cual se obtiene integrando las corrientes generadas por el paso anterior, para así obtener la corriente post sináptica excitatoría total que se inyecta a cada una de las neuronas post sinápticas j-ésimas vía sinapsis, la cual está definida también por la ecuación (2.4).

6.- Evaluar cuál de todas las corrientes post sinápticas excitatorías totales que se inyectan a las neuronas j-ésimas de la capa de salida, es la mayor para determinar la neurona ganadora.

7.- Determinar la actividad eléctrica de la neurona post sináptica ganadora/seleccionada, mediante la inyección del valor de corriente post sináptica resultante del paso 5, para así guardar en la memoria de programa los instantes de tiempos en que se generan los potenciales de acción post sinápticos k-ésimos.

8.- Iniciar el proceso de aprendizaje para la neurona ganadora/seleccionada, comparando solamente los pulsos post sinápticos k-ésimos que genera con loa pulsos pre sinápticos k-ésimos de las neuronas i-ésimas, mediante el protocolo de STDP, definido por las ecuaciones (1.23), (2.1) y (2.2) y así determinar el cambio de los pesos sinápticos a través de la interacción de los pulsos pre y post sinápticos, cuya manera de seleccionarse y calcularse ya ha sido descrita en la sección (2.4).

9.-Aplicar la regla de relajación a cada uno de los resultados obtenidos para modificar el peso sináptico a través de la ecuación (2.3) para sinapsis de tipo excitatoria.

10.- Incrementar el conteo de las épocas en uno y evaluar si el conteo ha superado las épocas máximas definidas. En este caso, si el conteo ha superado las épocas máximas, se inicializan nuevamente las épocas en uno, se guardan los pesos sinápticos relacionados a la neurona ganadora/seleccionada, así como la corriente post sináptica total asociada, se incrementa en uno la cantidad de patrones aprendidos, se elimina la neurona ganadora/seleccionada para participar en el próximo proceso de competición-entrenamiento y se prosigue al paso 9. En caso contrario, se repite el proceso desde el paso 4, realizando algunas modificaciones, debido a que ya que se conoce cuál fue la neurona ganadora/seleccionada que se asoció al caracter mostrado en el paso 4, calculando solamente las corrientes individuales post sinápticas i-ésimas asociadas a la neurona ganadora/seleccionada en el paso

5, calculando la integración de corriente correspondiente a ella y descartando el paso 6, con lo que se logra aislar la fase de entrenamiento.

11.- Si la cantidad de patrones aprendidos es igual la cantidad de neuronas en la capa de salida el algoritmo termina, en caso contrario, el proceso continúa y se repite el proceso desde el paso 4.

Por otra parte, en dado caso de hacer selectivo el algoritmo y que no sea “auto-organizado”, será necesario que seleccionemos la neurona post sináptica de la capa de salida que deseemos que aprenda un patrón en específico, por lo que será necesario sustituir el paso 4, determinando solamente la corriente post sináptica individual i-ésima que generan las neuronas pre sinápticas sobre la neurona post sináptica seleccionada en el paso 5, calculando solamente las corrientes post sinápticas i-ésimas asociadas a ella y eliminado el paso 6, procedimiento que realiza el algoritmo de STDP auto-organizado una vez que se conoce la neurona ganadora, y el cual se describe en el paso 10.

Obsérvese que en el algoritmo de aprendizaje descrito, las sinapsis inhibitorias no se procesan formalmente sino de manera simplificada, esto es porque dado que es una sinapsis fija “inhibitoria”, lo que se busca es que ésta tenga un valor máximo, en este caso. Si se procesara formalmente, tendría un valor igual a “-1” para que se genere una corriente inhibitoria en las vecinas que tienda a anular su generación de potenciales de acción; de esta manera, si no hay corriente post sináptica suficiente en las vecinas tampoco hay pulsos y por lo tanto el proceso de STDP no puede proceder. Este proceso es el que nos permite aislar el entrenamiento a una sola neurona simplificando el algoritmo.

Por otra parte, cabe mencionar que los cambios de corriente solo ocurren al inicio del algoritmo de cada época y que son resultado del cambio del peso sináptico de la época anterior, lo que hace referencia a que no se está procesando la información en tiempo real; esto tiene gran relevancia ya que si los cambios en los pesos sinápticos fueran en tiempo real, estos generarían cambios instantáneos en la corriente post sináptica de la red, que a su vez

podrían generar cambios en la información espacio-temporal de los potenciales de acción generados, modificando el proceso de selección si es auto-organizado y el proceso de aprendizaje por la variación de los pulsos en el tiempo.

En la figura (2.6) se ilustra el algoritmo descrito anteriormente, basándose en que existe suficiente corriente en la capa de entrada y de salida, y que se ha determinado la neurona ganadora/seleccionada, lo que nos posiciona en una época mayor a la inicial y en el paso 7 del "algoritmo de entrenamiento", donde se aísla este proceso a un conjunto de neuronas pre sinápticas y a una neurona post sináptica. En este caso, observamos en "A" que se coloca la ventana de aprendizaje en cada uno de los potenciales de acción k-ésimos provenientes de neuronas pre sinápticas i-ésimas dado que la corriente de entrada asociada a los pixeles no cambia durante toda la ventana de tiempo, y dado que existen pulsos en ambas capas, se establece el protocolo de STDP para determinar los cambios en los peso sinápticos "w_{ij}", según la relación de las neuronas pre sinápticas i-ésimas de entrada y la ganadora/seleccionada, lo cual corresponde al paso 8 del algoritmo propuesto. Una vez determinados los cambios sinápticos, se puede observar en "B" que mediante la regla de relajación se determinan los pesos sinápticos a través de las diferencias temporales entre pulsos pre y post sinápticos dando paso a la integración espacio-temporal que realiza STDP y que corresponde al paso 9 del algoritmo. En el paso 10, se concluye con una época y suponiendo que aún no se ha llegado a la época máxima designada dentro del programa, se repite el algoritmo, lo que nos lleva al paso 4, tal como se observa en "C" donde se determina la corriente post sináptica individual que llegara por efecto su sinapsis respectiva y finalmente por la integración espacio-temporal sináptica, que realiza la neurona post sináptica, se determinará la corriente post sináptica total, que retro alimentará el algoritmo y que en la siguiente época, modificará nuevamente los pulsos post sinápticos, tal como se observa en "D" y que corresponde al paso 5 del algoritmo.

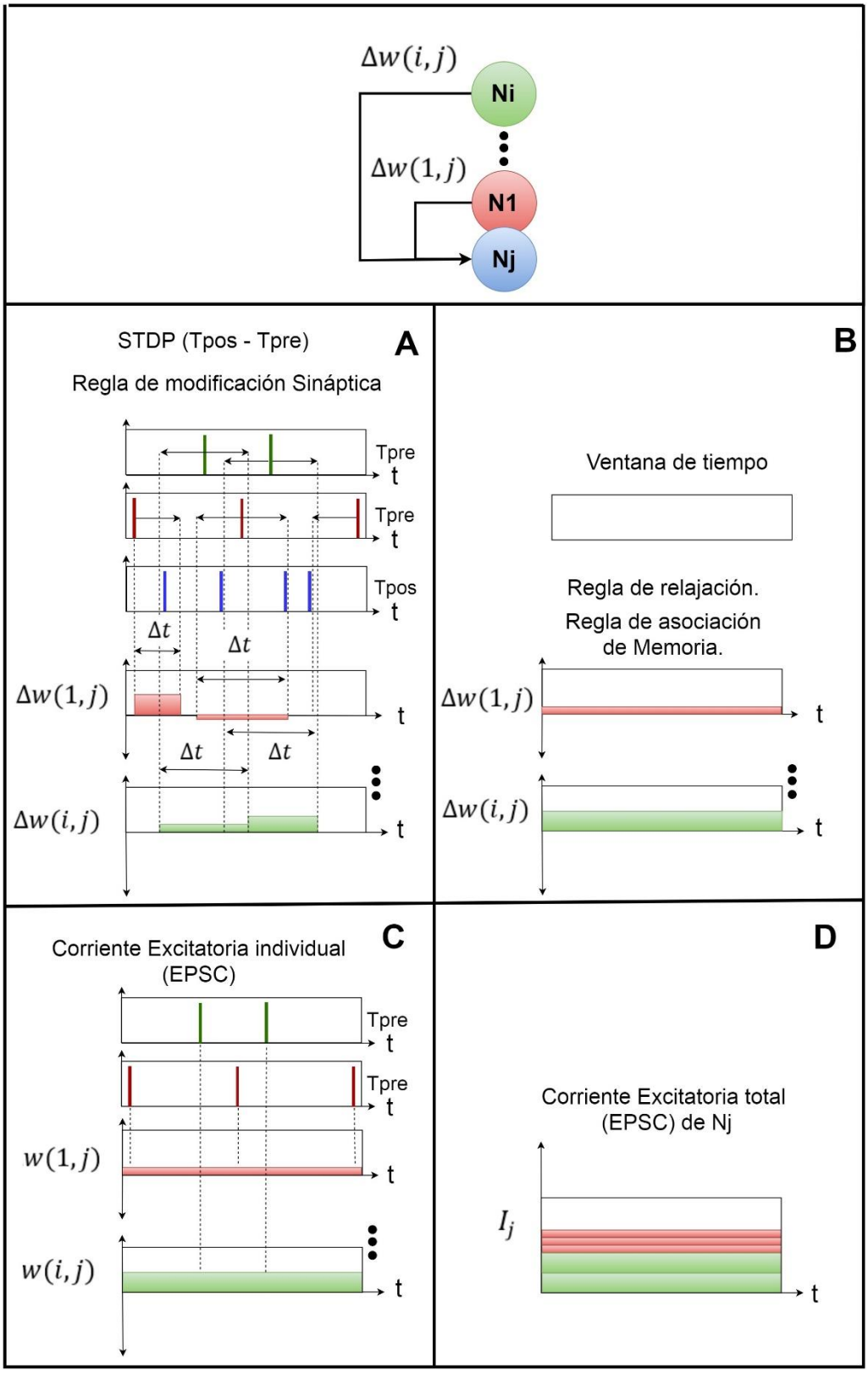

Fig. 2.6. Algoritmo de entrenamiento propuesto basado en STDP no supervisado.

2.8.5 Implementación del algoritmo propuesto de reconocimiento

El algoritmo de reconocimiento es semejante al algoritmo de aprendizaje; sin embargo, difiere en algunos pasos:

1.-Definir los parámetros de la red:

- Generar la ventana de tiempo (la cual deberá tener la misma longitud de la ventana usada en entrenamiento).
- Establecer los parámetros de las neuronas para que exhiban su modo RS.
- Recuperar los pesos sinápticos finales obtenidos del algoritmo de aprendizaje de todas las sinapsis excitatorias y asignarlos correspondientemente.
- Recuperar las corrientes post sinápticas totales finales asociadas a los pesos sinápticos de las neuronas de la capa de salida.
- Inicializar los pesos sinápticos de las sinapsis inhibitorias de la red en un valor máximo igual a w=-1.

2.- Seleccionar la imagen a reconocer.

3.- Aplicar los mismos valores de corriente que se usaron en la etapa de entrenamiento en cada una de las neuronas i-ésimas de la capa de entrada cuando se relacionaron con los pixeles de la imagen.

4.- Determinar las corrientes post sinápticas individuales i-ésimas que generan las neuronas pre sinápticas sobre las neuronas post sinápticas j-ésimas, que tiene lugar durante la ventana de tiempo a través del modelo sináptico simple de corriente, definida por la ecuación (2.4) y guardarlas en memoria de programa.

5.- Realizar el proceso de integración sináptica que realizan las neuronas post sinápticas j-ésimas de la capa de salida, la cual está definida también por la ecuación (2.4).

6.- Evaluar cuál de todas las corrientes post sinápticas excitatorias totales que se inyectan a las neuronas j-ésimas de la capa de salida corresponde al conjunto de corrientes post sinápticas obtenidas del proceso de aprendizaje para determinar la neurona ganadora.

7.- Determinar la actividad eléctrica (potencial de membrana – “V”) de la neurona post sináptica ganadora, guardando en la memoria de programa los instantes de tiempos en que se generan los potenciales de acción post sinápticos.

8.- Calcular la corriente post sináptica inhibitoria a partir de la ecuación (2.4), la cual se genera del producto de los potenciales de acción de la neurona ganadora, por el valor del peso sináptico fijo de las sinapsis inhibitorias que las relaciona con sus vecinas.

9.- Realizar nuevamente el proceso de integración sináptica, para obtener la corriente post sináptica total que se tiene en las neuronas vecinas, la cual es el resultado de la sumatoria de la corriente post sináptica excitatoria e inhibitoria en cada una de ellas, y a partir de esto, inyectar la corriente post sináptica resultante para evaluar el modelo neuronal de Izhikevich para así determinar la cantidad de potenciales de acción que generan.

10.- Repetir el algoritmo cada vez que se seleccione una imagen distinta.

Obsérvese en el algoritmo de reconocimiento, que las sinapsis inhibitorias, si se procesan formalmente durante el algoritmo, con la finalidad de suprimir la actividad eléctrica de las neuronas “perdedoras”, desde ese momento, sabremos qué neurona es afín a cada caracter.

2.9 Consideraciones Finales

El algoritmo propuesto es una simplificación del comportamiento biológico teórico, en cuanto a las sinapsis, neuronas, aprendizaje de STDP, y forma de integración espacio-temporal que éstas realizan de manera independiente.

En cuanto a la arquitectura de la red, ésta puede crecer a “n” neuronas de entrada y de salida; sin embargo, durante el entrenamiento solo las sinapsis que relacionen la ganadora estarán “presentes”, mientras que las demás estarán inactivas tal como se muestra en las figuras (2.7)-(2.8), donde las sinapsis excitatorias e inhibitorias marcadas en azul y rojo respectivamente, se encuentran activas, remarcando que las sinapsis inhibitorias no se procesan formalmente en aprendizaje y sí se procesan en reconocimiento, y así consecutivamente hasta que termine el algoritmo de aprendizaje (sea selectivo o auto-organizado). En la figura (2.7) se observa el inicio del proceso de inducción del aprendizaje (entrenamiento) en la red y en la figura (2.8) el término del mismo, suponiendo que se haya comenzado en Nj =1 y que se hayan concluido en Nj, independientemente del orden.

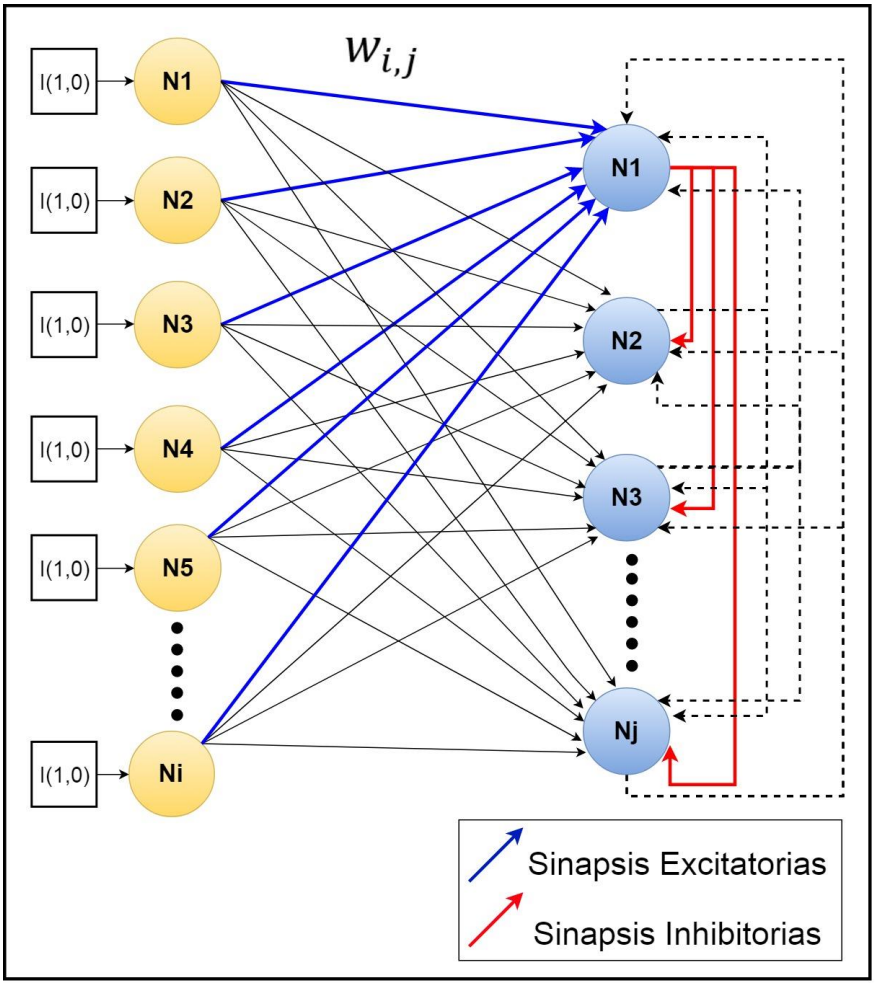

Fig. 2.7. Proceso inicial de aprendizaje basado en el algoritmo propuesto.

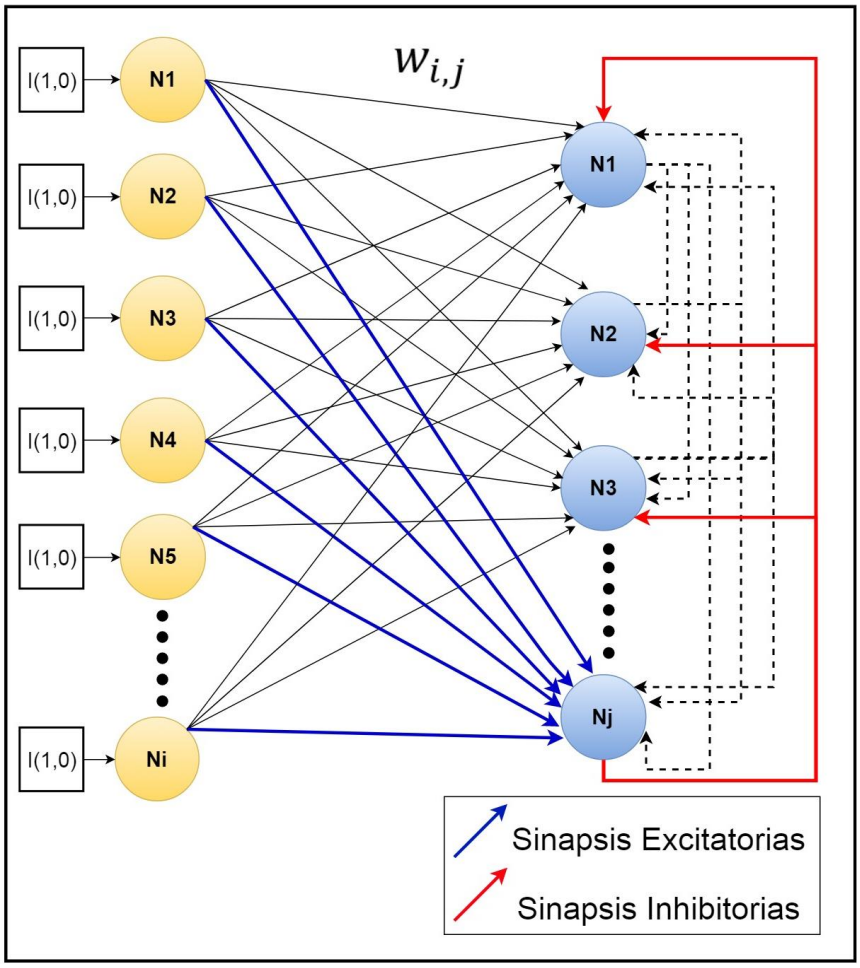

Fig. 2.8. Proceso final de aprendizaje basado en el algoritmo propuesto.

En cuanto a la manera de cómo saber en qué punto ha terminado la fase de aprendizaje, en los artículos citados en su gran mayoría, dado que es un aprendizaje no supervisado, normalmente se recurre a la capacidad “auto asociativa de la red”; esto es que la imagen reconocida (en este caso el caracter reconocido) se ve reflejado en los pesos sinápticos de las neuronas pre sinápticas que se relacionan con la neurona ganadora, las cuales, en su inicio, no mostraban ningún patrón visual reconocible dado que son pesos sinápticos aleatorios. Los pesos finales muestran tendencias hacia los valores máximos y mínimos establecidos por la red, en nuestro caso por la regla de relajación, de unos y ceros, haciendo mención que esta propiedad no es exclusiva de las redes de tercera generación como describe Hagan,Demuth, Beale, y De Jesús en [27]. Así, se propone que se definan épocas máximas al

inicio del programa, para de esta forma realizar, al final del mismo, la evaluación de los pesos sinápticos; en caso dado de que estos no converjan, se debe modificar la longitud de la ventana de tiempo (véase sección 2.2) o modificar la cantidad de corriente con la cual se asocia cada pixel a las neuronas de entrada.

En cuanto a la manera en cómo se determina que una neurona reconoció un caracter, la forma usada para decodificar la información está basada en la codificación por pulsos “primero en disparar”, lo cual corresponde a la idea que, aquella “neurona que es capaz de asociarse con un caracter presentado, es aquella con la actividad eléctrica más intensa y por lo tanto la primera en disparar”. Sin embargo, cabe mencionar que para conocer el potencial de membrana que origina los pulsos, siempre será necesario, dada la secuencialidad con la que se ejecuta el algoritmo, conocer antes la corriente post sináptica que la genera.

En cuanto a los resultados específicamente que tienen que ver con implementaciones en software, se reporta en [18], que la red de tercera generación con aprendizaje realizado por STDP no supervisado, usando la misma forma de la ventana que aquí se presenta, la red falló en el reconocimiento de caracteres que eran casi iguales y específicamente en [21], en términos de porcentaje, solo el 80.3% de los caracteres presentados fueron reconocidos correctamente,12% confundía caracteres y el restante no había aprendido o pulsaba independientemente de la entrada (este último usando otra arquitectura).

2.10 Conclusiones.

El algoritmo propuesto, aunque es una simplificación del comportamiento biológico teórico, la información espacio-temporal se mantendrá numéricamente lo que nos permitirá reservarla en la memoria de programa para después llevar a cabo el proceso de aprendizaje y de reconocimiento. La arquitectura consiste en 2 capas de neuronas, donde la primera tendrá que ver con la cantidad de pixeles que contenga la imagen, mientras que la segunda

dependerá de la cantidad de caracteres a reconocer. Las sinapsis excitatorias relacionarán ambas capas mientras que la inhibitorias solo relacionarán las neuronas de la capa de salida lateralmente, estas últimas, permitirán determinar las neuronas ganadoras resultado de un proceso selectivo o de auto-organización según convenga durante el aprendizaje, pero que durante el proceso de reconocimiento, será necesario conocer la corriente asociada a entrenamiento para determinar qué neurona se asoció a cada patrón e inhibir las “perdedoras”. Obsérvese que la discriminación en software se hace a través de la corriente post sináptica total, no pudiendo ser de otra forma, ya que para poder determinar los potenciales de acción siempre es necesario conocer antes la corriente post sináptica que las genera.

Una ventaja de realizar la implementación del algoritmo de reconocimiento de patrones en software, es que nos permite realizar el entrenamiento de manera estática, es decir sobre una ventana de tiempo, una y otra vez, así como resolver el problema de asociación de pulsos, el cual es factible para determinar cambios en el peso sináptico, calcular corrientes post sinápticas individualmente, y asignaciones de memoria. Sin embargo, cabe remarcar que esto es posible gracias a que no se procesa la información en tiempo.

2.11 Referencias

[1] S. Haykin, Neural networks: a comprehensive foundation, Prentice Hall, 1994.

[2] M. A. N. Maganda, *A High Performance Hardware Architecture for Multilayer Spiking Neural Networks,* Tonantzintla, Puebla: PhD. Thesis INAOE, october, 2009.

[3] T. Troyer, An Introduction to Computational Neuroscience, Draft Manuscript Copyright Todd W. Troyer, 2005, 2007.

[4] K. Vestbøstad, *Spiking Neural Networks for Pattern Recognition.,* Høgskulen på Vestlandet. M.Sc Thesis, June 13, 2017.

[5] E.-G. M. M. ,. J. C. Vilella, *Digital System For Spiking Neural Network,* Spain, Catalunya: Thesis Grade, Universitat Politècnica de Catalunya, June 2017.

[6] E. M. Izhikevich, «"Simple Model of Spiking Neurons"» *IEEE Transaction on Neural Networks,* vol. 14, nº 6, 2003.

[7] J. V. Tranquillo, «Quantitative Neurophysiology. Synthesis Lectures on Biomedical Engineering #21» Morgan & Claypool., 2008.

[8] B. G. A. G. &. D. W. David Sterrat, Principles of computational modelling in neuroscience., New York ,USA.: Cambridge University Press., 2010.

[9] E. M. Izhikevich, «Dynamical Systems in Neuroscience.The Geometry of Excitability and Bursting» MIT Press., 2007, pp. 25-50.

[10] F. G. C. ,. J. A. M. C. Guillermo Nieto Hernandez, *Experimentacion con redes neuronales pulsadas: Evaluación de capacidades*

computacionales, Agosto, 2019: Tesis M.Sc. Centro de investigación y de Estudios Avanzados del Instituto Politécnico Nacional.

[11] W. G. Jean-Pascal Pfister, «Triplets of Spikes in a Model of Spike Timing-Dependent Plasticity,» *The Journal of Neuroscience,* Vol. 26, Nº 38, p. 9673-9682, 2006.

[12] N. S. D. Eugene M. Izhikevich, «Relating STDP to BCM» *Neural Computation. MIT,* vol. 15, p. 1511–1523, 2003.

[13] M. C. Peter U. Diehl, «Unsupervised Learning of Digit Recognition Using Spike-Timing-Dependent Plasticity» *IEEE Transactions in Neural Networks and Learning Systems,* pp. 1-6.

[14] J. &. W.Gerstner, «Spike- timing dependent plasticity» *Scholarpedia, 5(2):1362,* 2010.

[15] H. N. Sou Nobukawa, «Synchronous spike propagation in Izhikevich neuron system with spike-timing dependent plasticity» *2012 Proceedings of SICE Annual Conference (SICE), IEEE,* pp. 453-458, 20-23 Aug. 2012.

[16] A. J. H. a. T. M. Kadhim, «Spiking Versus Traditional Neural Networks for Character Recognition on FPGA Platform» *Journal of Telecommunication, Electronic and Computer Engineering,* Vol. 10, Nº 3, pp. 109-115, July 2018.

[17] A. R.-M. F. G.-M. B.-M. a. J. V. F.-V. Taras Lakymchuk*, «Simplified spiking neural network architecture and STDP learning algorithm applied to image classification» *EURASIP Journal on Image and Video Processing. Springer Open Journal,* 2015.

[18] A. G. &. L. N. Long†, « Character Recognition using Spiking Neural Networks» *IEEE Neural Networks Conference,* August 2007.

[19] Y. Z. R. W. &. J. Z. Quansheng Ren, «Optical spike-timing-dependent plasticity with weight-dependent learning window and reward modulation» *Optical Society of America,* Vol. 23, Nº 19, 2015.

[20] T. M. K. Amjad J. Humaidi, «Recognition of English Characters Using Spiking Neural Networks» *International Journal of Engineering and Technology (IJET),* Vol. 5, Nº 9, Oct-Nov 2017.

[21] T. M. V. A. K. K. T. L. Francois Christophe, «Pattern recognition with Spiking Neural Networks: a simple training method» *Proceedings of the 14th Symposium on Programming Languages and Software Tools. CEUR Workshop Proceedings,* Vol. 1525, Nº , 2015.

[22] R. G. &. S. J. T. Timothée Masquelier, «Competitive STDP-Based Spike Pattern Learning» *Neural Computation, MIT,* Nº 21, p. 1259–1276, 2009.

[23] J. C. M. a. D. Johnston, «A Synaptically Controlled, Associative Signal for Hebbian Plasticity in Hippocampal Neurons» *Science,* Vol. 275, 1997.

[24] D. ,. G. J. F. D. H. W. C. M. A.-S. M. J. O. &. W. S. Purves, «Neuroscience, Third Edition» Sunderland, Massachusetts, U.S.A., Sinauer Associates, Inc. Publishers, 2004, pp. 31-165.

[25] L. R. ,. B. F. E. S. N. C. L. S. ,. A. &. B. D. Squire, «Fundamental Neuroscience» La Jolla, California, Elsevier, Third Edition, 2008.

[26] Y. D. &. N. Caporale, «Spike timing Dependent Plasticity: A Hebbian Learning Rule» *Annual Review of Neuroscience,* p. 24, 2008.

[27] H. B. D. M. H. B. D. J. Martin T. Hagan, «Neural Network Design,» Martin Hagan, 2014, p. 802.

CAPÍTULO 3

“En la naturaleza, las consecuencias siempre van enlazadas con los acontecimientos anteriores de manera ordenada y correspondiente. Esto no es porque su curso sea una serie de circunstancias sueltas y desorganizadas (las cuales fueran ordenadas por la fuerza), sino porque es una enumeración encadenada por motivos justos y por razones fundadas. Así como las cosas están ordenadas en el mundo con la debida proporción y armonía, las consecuencias no son una simple sucesión de las unas a las otras, sino que tienen una afinidad entre sí mismas.”

Marco Aurelio, Meditaciones, Libro IV.170-180

“La síntesis de espacios y tiempos, es lo que hace posible, como formas esenciales de toda intuición, la aprehensión del fenómeno y por consiguiente toda experiencia externa y por consiguiente todo conocimiento de objetos de la experiencia.”

Immanuel Kant, “Crítica de la razón pura”, Axiomas de la intuición, 1787.

“Si quieres fijarte en los pintores, en los arquitectos, en los constructores de barcos, en una palabra, en el obrero que te plazca, veras que en cada uno de ellos pone en cierto orden todo lo que coloca y obliga a cada parte a adaptarse y a sumarse hasta que todo tenga la disposición la forma y la belleza que deben tener.”

Platón, Diálogos, Georgias o de la retórica (Sócrates).

3 Implementación en hardware de una red neuronal pulsada con STDP no supervisado para el reconocimiento de caracteres

3.1 Introducción

El objetivo principal de este capítulo, es implementar un prototipo de un sistema de reconocimiento de caracteres con capacidad de procesamiento y reconocimiento de hasta 10 caracteres de 6x6 pixeles sobre el FPGA “xc7a100t-csg324” (Artix 7) que se encuentra en la tarjeta Nexy’s 4 DDR a nivel temporizado, con la finalidad de plantear una arquitectura neuromórfica funcional aislada, que pueda utilizarse posteriormente en otros sistemas. Para poder desarrollar dicho sistema, se ocupa como entorno de desarrollo VIVADO 2019 y como sistema de simulación Modelsim 10; por otra parte, en cuanto a la descripción del hardware, se emplea lenguaje VHDL estándar cuya estructura de programación se encuentran basada en [1] y [2] con el objetivo de que ésta pueda migrarse a otros FPGA´s

El sistema de reconocimiento propuesto usa el algoritmo de STDP no supervisado, que hemos denominado como aprendizaje “selectivo” descrito en el capítulo 2, con la finalidad de simplificar el control del mismo y llevar a cabo los procesos de entrenamiento-reconocimiento, no valiéndose de otros sistemas para realizar un pre o post procesamiento de la información.

Dadas las características del algoritmo, el diseño plantea la implementación de un sistema “neuromórfico”, para lograr la realización de las tareas de manera paralela y el procesamiento en tiempo real de la información que no es posible llevar a cabo en software, tratándose por lo tanto de una emulación en hardware de un sistema biológico y no de una simulación como la que se realiza en software. En este sentido, existen retardos de propagación y retro propagación entre neuronas pre y post sinápticas, el efecto de generación de corriente a través de la sinapsis ocurre de manera instantánea, al igual que el

de modificación sináptica producto del aprendizaje inducido por STPD, afectando la dinámica neuronal de la red.

A lo largo del capítulo se describirá la operación de sistema, la cual se encuentra fundamentada en el comportamiento biológico teórico de la plasticidad sináptica, comportamiento sináptico y neuronal para aproximar el comportamiento espacio-temporal de una red biológica. Tales cuestiones, se abordan en el "Material suplementario" de esta tesis que se encuentra disponible en la dirección http://www.vlsilab.cinvestav.mx/tesis_fgc.html del Laboratorio VLSI CINVESTAV al igual que los programas que generan los módulos descritos, dado el volumen de la descripción en Hardware.

3.1.1 Sistemas Neuromórficos.

Para llevar a cabo un algoritmo con características espacio-temporales que replique en cierta medida el comportamiento neuronal biológico, es necesario, dadas sus características, repensar incluso los conceptos más básicos y las ideas acerca de ellos, para generar nuevas herramientas que se adecuen a un ambiente computacional, donde el aprendizaje procede de manera diferente a como hacen las redes neuronales de primera y segunda generación, como describe Mass en [3]

Uno de los enfoques usados para resolver estas cuestiones, es mediante la implementación de sistemas conocidos como "neuromórficos", los cuales son sistemas que intentan reproducir las características biológicas del sistema nervioso con distintos niveles de plausibilidad, razón por la que buscan conectar miles o millones de neuronas en un solo chip, y para lo cual, se apoyan de arquitecturas no convencionales basadas en eventos para la transmisión y procesamiento de la información como describe Boahen en [4] e Indiveri, Liu, Delbrück y Douglas en [5].

En cuanto a sus características principales, basado en el trabajo de Schuman, Potok, Douglas, Dean, Rose, y Planck en [6] e Indiveri y Liu en [7], se encuentra el paralelismo de procesos, el manejo de memorias distribuidas, la

escalabilidad, el aprendizaje en tiempo real (online learning), y su capacidad de tolerancia a fallas. Estos sistemas tienen ventajas sobre el cómputo convencional ya que eliminan los cuellos de botella de las arquitecturas Von Neumann de las computadoras actuales, permitiendo un procesamiento más eficiente de la información. Esto posibilita la implementación de las especificaciones provistas por las neurociencias en cuanto a velocidad de operación y características de comportamiento.

El principal interés de los sistemas neuromórficos es la replicación de las no linealidades en el comportamiento eléctrico de las neuronas y demás elementos presentes en las redes biológicas, para la formación de sistemas más complejos que trabajen en tiempo real. Desde que el concepto "neuromórfico" fue acuñado por Carver Mead a mediados los 80's, uno de los objetivos principales de la ingeniería neuromórfica es el desarrollo de circuitos digitales, analógicos o mixtos VLSI para explorar los métodos de procesamiento de la información biológica desde un contexto práctico de la ingeniería eléctrica. [5]

En nuestro caso, para realizar la implementación del sistema neuromórfico propuesto, nos basamos en los trabajos de Partzsch, Höppner, Eberlein, Schüffny, Mayr, Lester, y Furber en [8], Cassidy, Denhamt, Kanoldt y Andreou en [9], Mallorquí y Cosp Vilella en [10] y Garg, Shekhar y Harris en [11], los cuales usan como elementos clave del sistema arquitecturas "FIFO" como sistemas de compensación de tiempo y arquitecturas "AER" para transmisión y procesamiento de los pulsos generados en la red. Tales sistemas, son incorporados en la arquitectura que se describe posteriormente en la sección (3.5.2.2) y (3.5.2.3). El módulo general del sistema se encuentra etiquetado como "complete_SNN" y los módulos que lo conforman se describen a lo largo del capítulo.

3.1.2 Arquitectura de la red

La arquitectura neuronal que se implementa en hardware es la misma arquitectura propuesta en el capítulo 2; sin embargo, dado que el objetivo es

abordar una implementación que nos permita reproducir algunas de las características principales de una red biológica como el cómputo en tiempo real, es necesario que se agreguen retardos de propagación y retro propagación, características vistas en [12] y [13], que influyen directamente en los procesos de sinapsis y de aprendizaje dirigido por STDP, al ser totalmente dependientes del tiempo. En este contexto, los retardos en la arquitectura generan un cambio en la red a implementar, ya que provocan que las distancias entre las neuronas pre y post sinápticas sean diferentes. En la figura (3.1) se muestran los cambios que generan los retardos en la red a implementar.

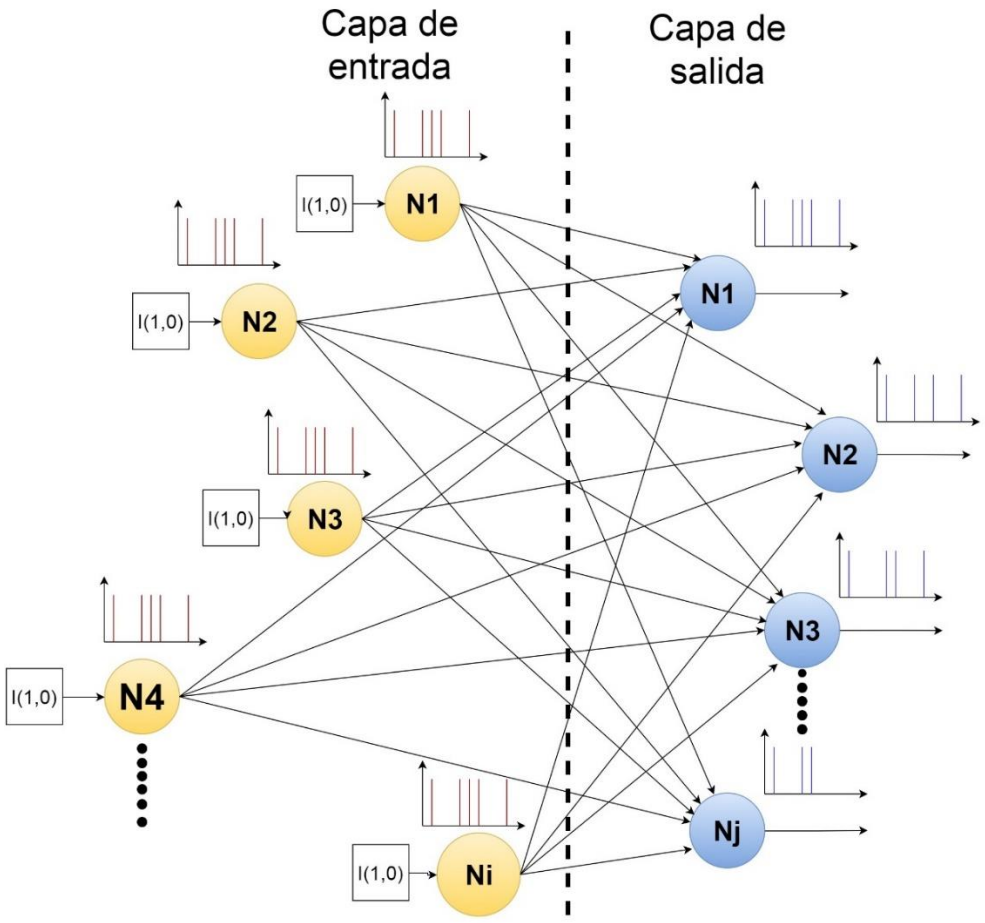

Fig. 3.1. Arquitectura de red implementada en FPGA.

El sistema en general, de acuerdo a esta arquitectura, cuenta con 46 neuronas Izhikevich de las cuales 36 se encuentran en la capa de entrada, 10 en la capa

de salida y 360 relaciones sinápticas donde se refleja el resultado del aprendizaje dirigido por STPD.

Por otra parte, para ubicar visualmente los elementos que conforman a la red una vez llevado a FPGA, puede hacerse una ejemplificación de la equivalencia de la arquitectura de red que se muestra en la figura (3.1) mediante un diagrama a bloques. Éste permitirá ubicar las elementos y por lo tanto ejemplificar su operación. Se parte, por lo tanto, que la arquitectura de la red en FPGA puede verse simplificada en 3 partes: capa de entrada, capa de salida y sistema de control principal, siendo este último donde se implementa toda la lógica de control necesaria para realizar el aprendizaje y reconocimiento de la red. Sin embargo, independientemente de esta separación en cuanto a su arquitectura, el sistema de reconocimiento de caracteres está formado por 6 bloques principales etiquetados como: “Main_Control”, “Entry_Process”, “STDP_Process”, “Neurona_iz_24_b”, “General_Spike_control” y “concentrator”, los cuales tienen lugar en cada una de estas capas antes mencionadas. En ese sentido, cada capa almacena uno o varios módulos principales, los cuales, al mismo tiempo, albergan otros módulos secundarios que le permiten realizar sus procesos. Desde este punto de vista, se ejemplifica y describe la operación general del sistema:

- **Sistema de Control**. Dicho sistema se encarga de dirigir los procesos de aprendizaje y de reconocimiento, por lo que tiene embebida la lógica de control descrita en la sección (2.8) correspondiente a una implementación en software. Sin embargo, dado que el sistema tiene un enfoque neuromórfico, éste fue descrito conservando la estructura que le da forma a estos procesos, pero teniendo en mente un marco de trabajo que posibilita la operación en tiempo real (online-learninig) y paralela. Este sistema se encuentra embebido en “Main_Control”.
- **Capa de entrada**: viene representada por el módulo “Entry_Process”, el cual contiene las 36 neuronas pre sinápticas necesarias para el aprendizaje y reconocimiento de una imagen de 6x6 pixeles y que son

implementadas a través del módulo “Neurona_iz_24_b”. En cuanto a su operación, el módulo “Entry_Process” contiene toda la arquitectura necesaria para replicar los retardos de propagación y retro propagación que se encuentran de manera intrínseca en las neuronas pre sinápticas y post sinápticas respectivamente. Estos últimos son agregados en la arquitectura de este mismo modulo con la finalidad de ubicar y concentrar físicamente todos los retardos de la red. Asimismo, aquí es donde se encuentra embebida la arquitectura “AER” que hace posible el procesamiento de los pulsos pre sinápticos por la capa de salida.

- **Capa de Salida**: viene representada por los módulos “STDP_Process”, y “Neurona_iz_24_b”, los cuales, de manera conjunta, realizan la integración espacio-temporal de una sola neurona biológica post sináptica cuando realizan las operaciones de sinapsis y de plasticidad cerebral dirigida por STDP. Visto de esta manera, el módulo “Neurona_iz_24_b” es quien se encarga de generar los trenes de pulsos característicos del modo RS del modelo neuronal Izhikevich, y el módulo “STDP_Process” es el que se encarga de generar la corriente sináptica. Adicionalmente, este módulo almacena y gestiona los 36 pesos sinápticos debidos a las relaciones existentes entre las neuronas pre sinápticas de entrada a una sola neurona post sináptica; por ende, el módulo en cuestión dispone de toda la arquitectura necesaria para realizar el proceso de aprendizaje dirigido por STDP y sinapsis para una sola neurona de salida. Por esta razón, la capa de salida está formada por 10 módulos “Neurona_iz_24_b” y 10 módulos “STDP_Process”.

Para ejemplificar dicha estructura, se puede observar en el diagrama a bloques de la figura (3.2), los módulos que la componen, así como las secciones de las que forman parte dentro del sistema de reconocimiento de caracteres en FPGA, así como sus principales entradas y salidas.

A partir de la sección (3.2), se describirá individualmente cada una de las capas así como los módulos que lo conforman.

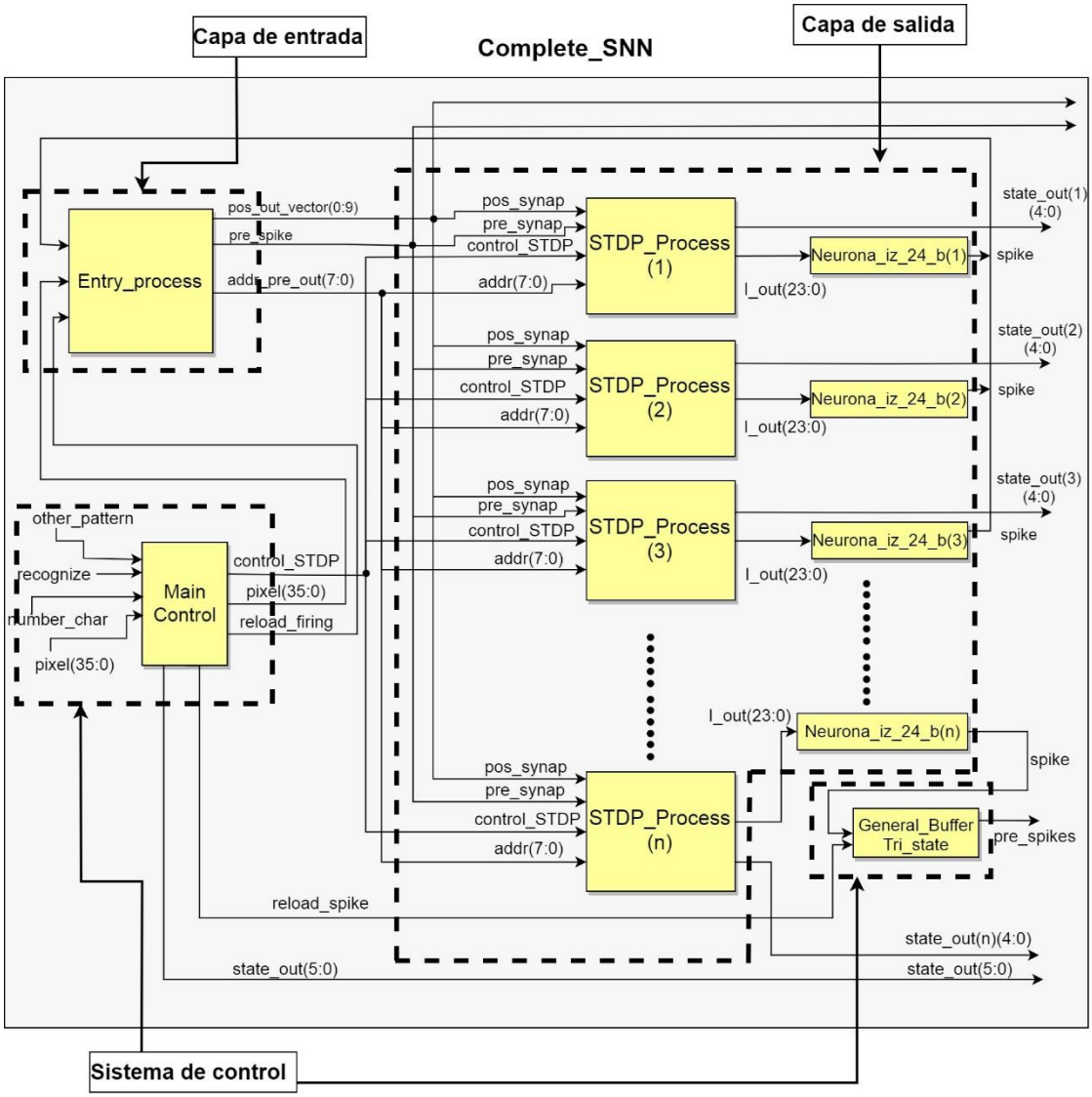

Fig. 3.2. Diagrama a bloques del sistema de reconocimiento de caracteres en FPGA.

3.1.3 Representación Binaria

El sistema propuesto contiene 2 representaciones binarias en punto fijo, las cuales fueron seleccionadas con la finalidad de implementar un sistema con bajos recursos en hardware, manteniendo el desempeño de la red. En ambas representaciones se eligió un ancho de palabra de 24 bits, y lo único que se varió fue la posición del punto decimal, y por ende, la cantidad de números enteros y decimales que podemos representar.

- **Representación neuronal.** En todas las neuronas del sistema se ocupa una representación de 11 bits para representar la parte entera y de 13 bits en la parte fraccionaria. En este caso, se colocaron más bits en la parte entera en comparación con la representación de procesamiento, con la finalidad de incrementar el rango numérico de la corriente sináptica y evitar posibles desbordes, tomando en cuenta que el valor máximo de ésta dependerá del comportamiento dinámico de la red.
- **Representación de procesamiento**: En todos los módulos de "STDP_Process" se ocupa una representación de 6 bits para representar la parte entera y de 18 bits en la parte fraccionaria. En este caso, se colocaron más bits en la parte fraccionaria con la finalidad de incrementar el rango numérico y poder realizar las operaciones necesarias para implementar el protocolo de STDP.

Para que exista coherencia entre los datos de ambas representaciones se realiza una conversión, pasando de una representación de procesamiento a una neurona para representar el mismo número, lo cual se logra reduciendo la exactitud de la primera mediante el módulo "Synapses" que se encuentra internamente en el módulo "STDP_Process". Éste se encuentra descrito en la sección (3.5.3.1.5) y es el punto de convergencia entre ambas representaciones. En resumen, manejar una cantidad mucho más grande de palabra (lo cual sería lo idóneo), implicaría un incremento significativo en recursos de DSP's y LUT's del sistema, lo cual reduciría la cantidad de neuronas que podemos implementar.

3.2 Descripción general del control en la operación del sistema

El control del sistema tiene como fundamento, la operación de dos sistemas de control, que se comunican entre sí y que trabajan de manera simultánea para ejecutar correctamente cada una de las operaciones que realizan. El control puede dividirse en dos:

- **Control Primario o principal:** Es el responsable de controlar los procesos de aprendizaje y de reconocimiento en la red descritos anteriormente en la sección (3.1.2). El Control Primario está formado por dos módulos, los cuales se encuentran descritos en VHDL y etiquetados como “Main_Master_Control” y “Main_Machine State”, los cuales se encuentran internamente embebidos en el bloque “Master_Control”, tal como se muestra en la figura (3.5).
- **Control Secundario:** Este sistema de control se encuentra formado, al igual que el control primario, por 2 módulos descritos en VHDL, etiquetados como “Main_Control” y “Machine_state”, los cuales se encuentran internamente embebidos en cada bloque “STDP_Process” tal como se muestra en la figura (3.31). La finalidad de ambos módulos, es permitir que se lleven a cabo 3 tareas: sinapsis, aprendizaje de STDP y almacenamiento-modificación de los pesos sinápticos de la red.

Esta arquitectura descrita anteriormente, opera en pares de módulos de control con el objetivo de simplificar el control, segmentando la operación en 2 partes, las cuales pueden ser vistas como un “Asignador de estados” y “Administrador de tareas”:

- **Asignador de estados**: En este módulo se encuentran internamente implementadas las secuencias del algoritmo a llevar a cabo mediante máquinas de estado tipo Moore, por lo que su única acción de control reside en responder a las solicitudes que envía el módulo “administrador de tareas” y así generar las transiciones entre los estados que posee en su lógica, tarea que verifica mediante banderas de control. Los módulos que implementan esta función son “Main_Machine State” con una máquina de 34 estados y “Machine_State” con 16 estados.
- **Administrador de tareas:** Cada uno de los “estados” que son enviados desde el módulo “asignador de estados”, requiere tareas a cumplirse para solicitar nuevos estados y, dado que es necesario para

la realización de estas tareas, la intervención de uno o más módulos afines a ese estado en particular, se requiere de un módulo que sirva de mediador entre estos procesos. En este sentido, los módulos que sirven como “administradores de tareas” se encargan de conectarse con los módulos que pueden realizar dichas tareas mediante banderas de control, permitiendo la comunicación entre los módulos necesarios. Cada vez que una operación es completada, el módulo “administrador de tareas” solicita un nuevo estado al módulo que sirve como “asignador de estados”. Los módulos que realizan la función de “Administrador de tareas” son “Main_Master_control” y “Master_Control”

En este contexto, si se viese al sistema de control como un todo, el sistema implementado tiene un “control primario” que posee 36 estados compuestos de múltiples tareas para dirigir todo el proceso de aprendizaje y de reconocimiento, el cual a su vez para lograrlo controla 10 “controles secundarios” que a su vez llevan a cabo múltiples tareas en serie-paralelo sobre las neuronas post sinápticas. En resumen, el sistema de control se basa en la interacción de máquinas de estados, las cuales administran, mediante banderas de control, todos los módulos periféricos, uno reside en el módulo “Main_control” que afecta al sistema de manera general y los demás residen en cada uno de los módulos “STDP_Process”, afectando a cada neurona de salida de manera particular.

3.3 Descripción general del proceso de entrenamiento.

De acuerdo al algoritmo propuesto en el capítulo 2, el entrenamiento puede realizarse de manera selectiva o auto-organizada, en este caso, se realizará de manera selectiva y por las razones descritas en la sección (2.8.3), las sinapsis inhibitorias serán simplificadas, lo que permite que seamos nosotros quienes dictaminemos qué neurona post sináptica inicie un proceso de aprendizaje y sobre todo simplificando los sistemas de control descritos en la sección (3.2). En sí, el proceso de entrenamiento se realiza de manera secuencial dado que solo una neurona post sináptica puede ser entrenada y

no todas de manera simultánea. Sin embargo, gracias a la intervención del protocolo AER, que a menudo se implementa en los sistemas neuromórficos y al “control secundario” descrito en la sección (3.2) (que se encuentra internamente en “STDP_Process”), es que es posible la integración espacio-temporal, permitiendo el procesamiento paralelo de los pulsos pre sinápticos por la neurona post sináptica. En este sentido, cada vez que se presenta una imagen, se inicia el entrenamiento de una neurona post sináptica, proceso que puede ser simplificado de la siguiente forma, viendo al sistema como módulos independientes:

1. “Entry_Process” transforma la imagen en pulsos pre sinápticos que terminan por hacerse llegar, mediante el protocolo AER, a los módulos “STDP_Process” y “Neurona_iz_24_b”.
2. “STDP_Process”, de manera simultánea, modifica los pesos sinápticos mediante la aplicación del protocolo STDP, transformando todo el proceso mediante la sinapsis, en un valor de corriente post sináptica. La integración espacio-temporal de STDP y de la sinapsis, son implementadas en tiempo real, lo que implica que los cambios de corriente y de peso sináptico que se dan de manera instantánea afectan dichos procesos.
3. “Neurona_iz_24_b” de manera simultánea, a través de la implementación del modelo de Izhikevich, responde al valor de la corriente inyectada y a los cambios de ésta, generando potenciales de acción dependientes del tiempo; estos se retroalimentan con su correspondiente módulo “STDP_Process”, no sin antes pasar por el módulo “Entry Process”, donde se agrega el retardo de retro propagación correspondiente.

Cabe mencionar que el entrenamiento de las diversas neuronas post sinápticas, en nuestro caso, ocurre de manera ordenada y ascendente (dado que es selectivo), por lo que la primera imagen que ingrese al sistema se asociará a la primera neurona post sináptica y así sucesivamente, hasta llegar

a la última imagen y a la última neurona, concluyendo el proceso de entrenamiento. Dicho proceso, es controlado por “control primario” que se encuentra embebido en “Master_Control”.

De esta forma, por cada imagen que se someta a entrenamiento, el módulo “Main_Control” mantendrá activo un solo módulo “STDP_Process”, que a su vez mantendrá en estado activo a la neurona post sináptica con la cual se encuentra vinculado. En este sentido, cada proceso de entrenamiento solo ocurre internamente en un solo módulo “STDP_Process” y una sola neurona post sináptica, mientras que el resto de los módulos de la capa de salida permanecerán inactivos.

La operación del algoritmo descrito anteriormente, se puede ver simplificada, en la figura (3.3), donde las flechas verdes indican el flujo de la información de los pulsos pre sinápticos de la red hacia todos los módulos “STDP_Process”. Las flechas azules corresponden a los pulsos post sinápticos generados por uno de los módulos “Neurona_iz_24_b”, y las flechas rojas representan la acción de control que realiza “Main_Control”, permitiendo que se inicie y concluya el proceso de aprendizaje en cada una de las neuronas de la capa de salida.

Por otra parte, cabe mencionar que el sistema no trabaja con ventanas de tiempo para realizar el proceso de aprendizaje como su versión en software, debido a que se está procesando la información en tiempo real. En este caso el protocolo AER, el “control secundario” y la arquitectura interna de “STDP_Process”, nos permite procesar los pulsos provenientes de la capa de entrada de manera simultánea, cada vez que se inicia el entrenamiento de una sola neurona post sináptica. (Véase sección (2.8) del capítulo 2 de esta tesis)

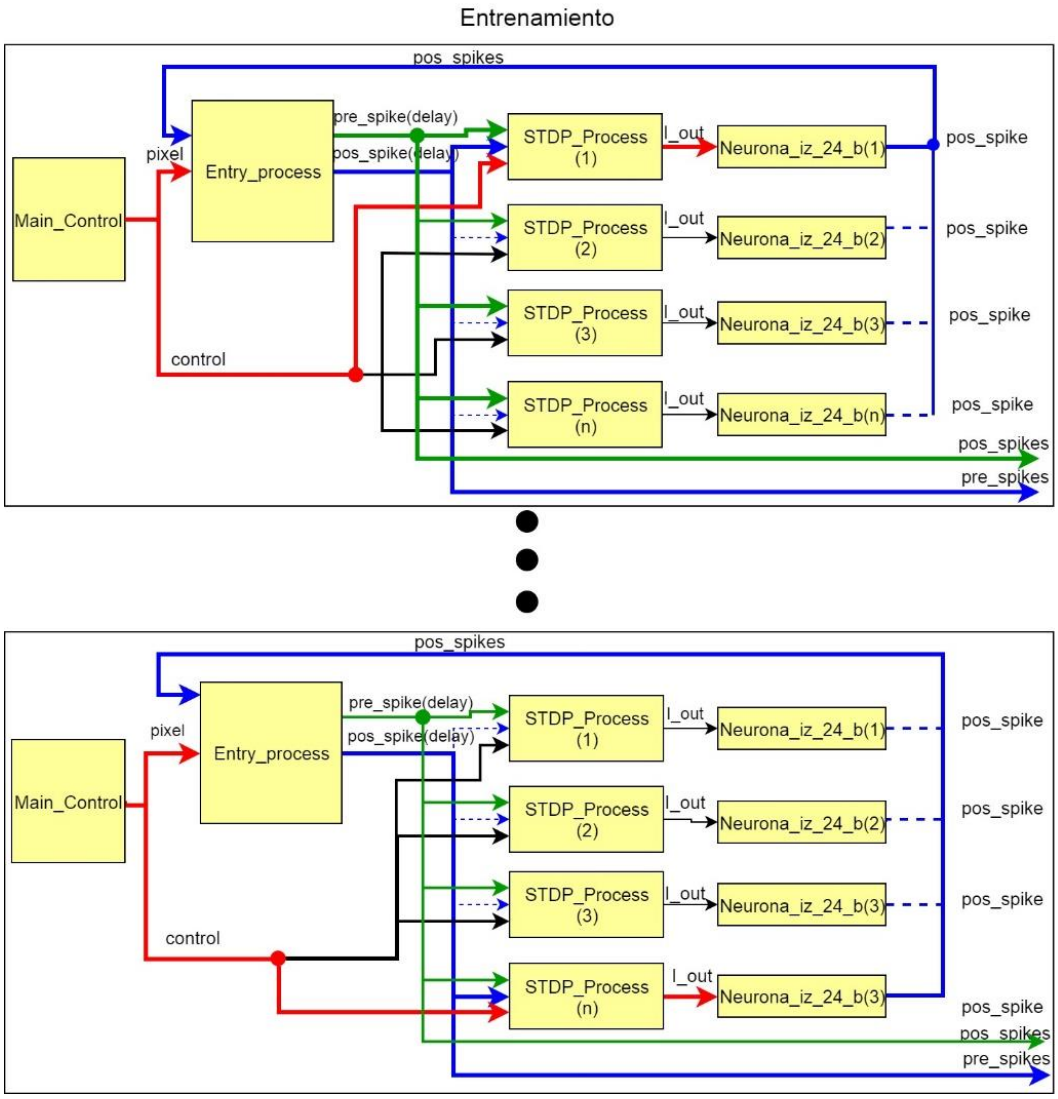

Fig. 3.3. Diagrama a bloques del proceso de entrenamiento de la red en FPGA.

3.4 Descripción general del proceso de reconocimiento.

A diferencia del proceso de aprendizaje, el proceso de reconocimiento se realiza de manera paralela, para lo cual, en este caso es necesaria una ventana de tiempo, debido a que el método de decodificación de la información para determinar la neurona ganadora es el “primero en disparar”. Este procedimiento fue descrito en la sección (1.6.2.1) y (2.8) de los capítulos 1 y 2 respectivamente, de esta tesis, con lo que solo es necesario conocer el primer pulso de una sola de las neuronas post sinápticas durante ese lapso de tiempo, para determinar qué neurona logró asociarse con el caracter.

En este caso, de manera simplificada, el procedimiento de reconocimiento que realiza el sistema cada vez que se le presenta una imagen, viéndolo como un conjunto de módulos independientes, es el siguiente:

1. “Entry_Process”, transforma la imagen en pulsos pre sinápticos que llegan, mediante el protocolo AER, a los módulos “STDP_Process” y “Neurona _iz_24_b”.
2. “Main_Control”, de manera simultánea, habilita un conjunto de señales de control que permiten que todos los módulos “STDP_Process”, procesen la información de los pulsos pre sinápticos de manera paralela por “Entry_process” para que, en conjunto con los pesos sinápticos resultantes del proceso de entrenamiento de cada uno de ellos, mediante la sinapsis, tenga lugar una corriente que se encuentre variando en el tiempo y que termine por afectar a sus respectivas neuronas post sinápticas.
3. “Main_Control”, se encuentra evaluando las salidas de todas las neuronas post sinápticas; en este caso, de todos los módulos “Neurona_iz_24_b”, esperando determinar el primer pulso generado, independientemente de cuál se trate a través de una ventana de tiempo.
4. “Main_Control”, mediante el módulo “General_Spike_control”, permite que solamente la salida de la neurona “post sináptica ganadora” sea visible a la salida del sistema. En ese instante se determina la neurona ganadora, la cual se mantiene pulsando hasta que se ingrese una nueva imagen, repitiéndose todo el procedimiento anterior. Obsérvese que en este caso la sinapsis inhibitoria no genera una diminución de la corriente en las neuronas ”perdedoras”, sino que su actividad eléctrica simplemente es suprimida a la salida, mismo efecto que se obtendría de disminuir la corriente total de éstas, mediante el procesamiento de la sinapsis inhibitoria de la neurona “ganadora”.

En la figura (3.4) se puede ilustrar el procedimiento descrito anteriormente, donde las flechas azules representan los pulso pre sinápticos, las flechas

verdes los pulsos post sinápticos y los flechas rojas las señales de control provenientes del módulo “Main_Control”, con las cuales se realiza la habilitación de los módulos “STDP_Process” de manera simultánea.

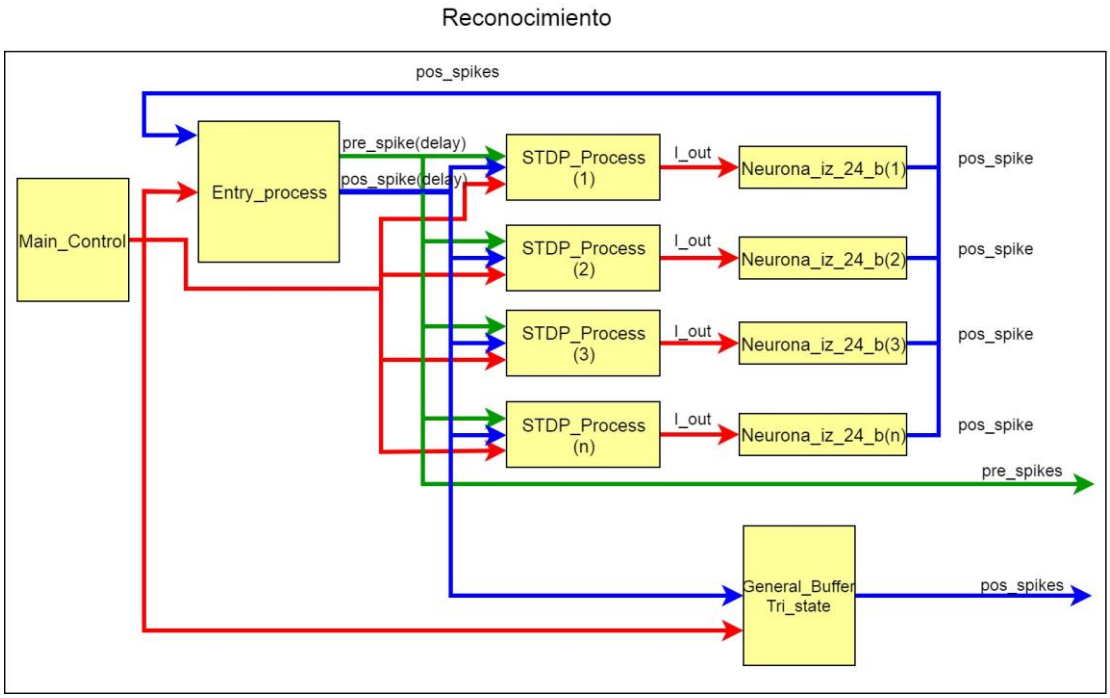

Fig. 3.4. Diagrama a bloques del proceso de reconocimiento de la red en FPGA.

3.5 Descripción del sistema de reconocimiento de caracteres

3.5.1 Sistema de Control.

El sistema de control se basa, como hemos mencionado anteriormente, en 2 módulos: “Main_Control” y “General_Spike_control”, de los cuales “Main_Control” es el principal, mientras que del segundo, la operación es la misma que el módulo descrito en (3.5.2.1.3).

3.5.1.1 Módulo de control

El módulo de control del sistema que se encuentra etiquetado como “Main_Control”, está formado por 4 módulos etiquetados como “Main_Master_Control”, “Main_Machine_State”, “Memoria_RAM_Pattern” y “Work_Window”, de los cuales, los primeros 2 realizan la función de “control primario” descrita en la sección (3.2), y los restantes proporcionan un sistema de memoria RAM para llevar a cabo la carga de imágenes al sistema y una

ventana de tiempo para ejecutar el proceso de reconocimiento. En la figura (3.5), de manera simplificada, se pueden observar los bloques que conforman al módulo, así como las entradas y salidas del módulo por izquierda y por derecha, respectivamente:

En entrenamiento:

- El módulo solicitará la cantidad de imágenes a entrenar a través de “number_char”; si éste es mayor a 10, el sistema solo permitirá el ingreso de 10 imágenes, mientras que si es menor a 10 el sistema solo habilitará las neuronas post sinápticas necesarias para asociarlas.
- El módulo a través de “pixel_out” pasará al módulo “Entry_Process” las imágenes a entrenar, que previamente se guardaron en el módulo “Memoria_RAM_pattern”, para iniciar el proceso de entrenamiento, descrito de manera general en la sección (3.3).

En reconocimiento:

- El módulo a través de los puertos “recognize” y “pixel” manipulará la entrada de imágenes al sistema. Cada vez que se ingrese una imagen, se indicará a través del puerto “recognize” el proceso de reconocimiento descrito en la sección (3.4).
- El módulo a través del puerto “other_pattern”, se prepara para procesar una nueva imagen, repitiendo el proceso nuevamente.

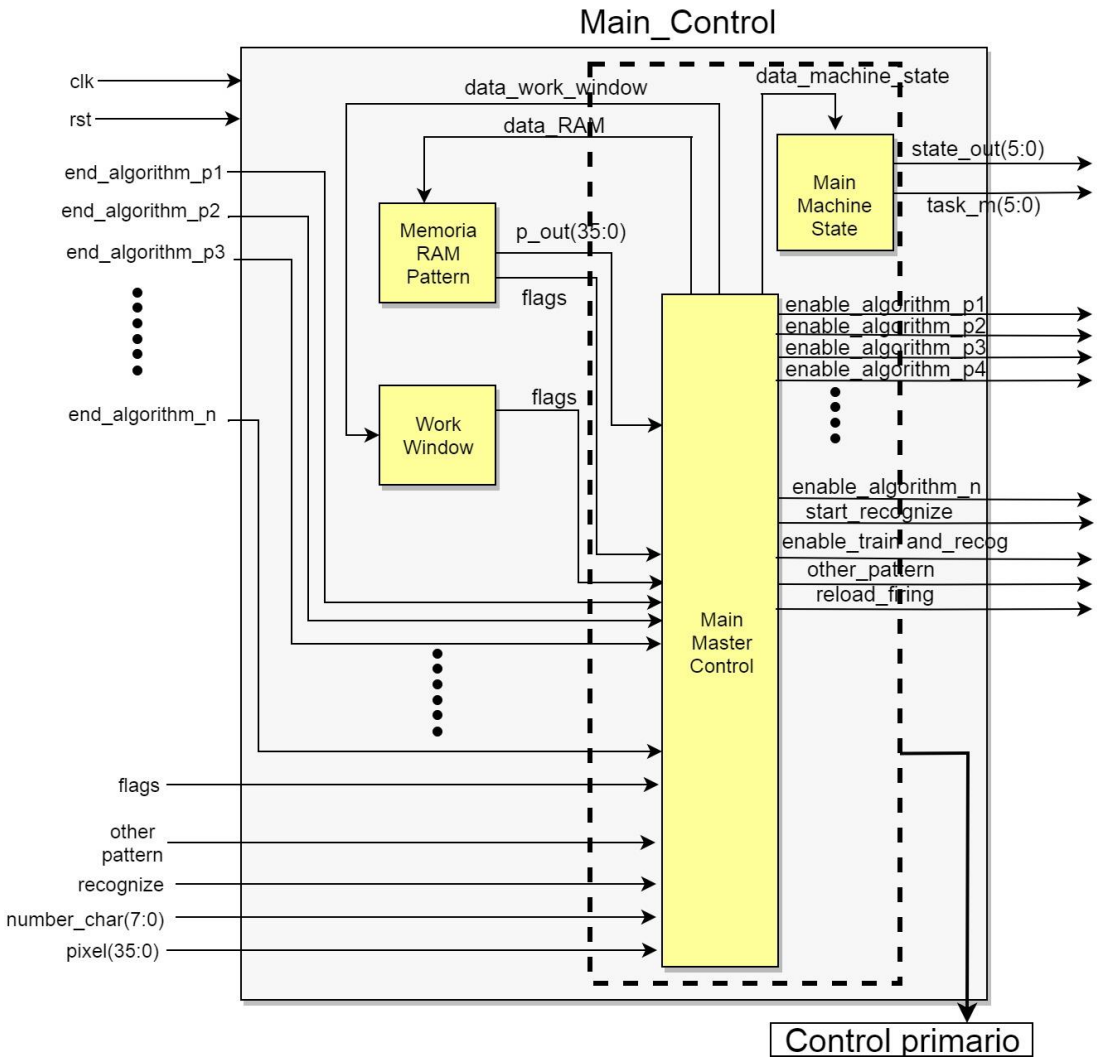

Fig. 3.5. Diagrama a bloques del módulo "Entry_Process".

3.5.1.1.1 Control primario

El control primario se encuentra formado por los módulos "Main_Master_Control" y "Main_Machine_State", los cuales de manera conjunta son el control de módulo "Main_Control" como se describe en la sección (3.2). En este caso, "Main_Master_Control", concentra y administra las banderas de los múltiples "STDP_Process", así como la comunicación con el exterior y "Main_Machine_State permite las transiciones entre estados. De manera conjunta se implementan 34 estados con tareas, en las cuales se pueden distinguir 3 etapas:

- **Carga de patrones**: Se realiza la carga de todos los caracteres, para que se sometan a un proceso de aprendizaje y se alojan dentro de una memoria RAM, para su posterior procesamiento.
- **Etapa de entrenamiento**: Se lleva a cabo el proceso de entrenamiento descrito en la sección (3.3), el cual tiene lugar en cada módulo de “STDP_Process” y se logra pasando una imagen desde la RAM cada vez que termina el entrenamiento de una neurona post sináptica.
- **Etapa de reconocimiento:** Se lleva cabo el proceso descrito en la sección (3.4), el cual se ejecuta de manera paralela por lo que las máquinas de estado que residen en el módulo “STDP_Process”, operan al mismo tiempo. En esta etapa, los caracteres a reconocer son obtenidos desde la RAM, es decir, solo se someten a reconocimiento los caracteres con los que se entrenó la red.

La máquina de estados y las tareas embebidas en “Main_Master_Control” y “Main_Machine_State que forman el “control primario”, se describen a continuación:

1. **Total_pattern:** El sistema solicita la cantidad de imágenes a entrenar. Una vez obtenido este valor, se solicita el estado siguiente; caso contrario, permanece en el mismo estado.
2. **Take_Character**: Se solicita la imagen a entrenar, habilitando el módulo “Memoria_RAM_pattern”, para guardar el valor de los pixeles obtenidos en ella. Si esta operación tuvo éxito se solicita el estado siguiente, caso contrario, permanece en el mismo.
3. **Save_character**: Se realizar y verifica la correcta escritura del módulo “Memoria_RAM_pattern” con la imagen que se tiene a la entrada; en este caso, si se realiza con éxito la tarea, entonces se solicita el siguiente estado, caso contrario, permanece en el mismo.
4. **Prepare_another_p**: Se inhabilita la operación de la “Memoria_RAM_pattern”, y de manera simultánea se valida si la cantidad de imágenes guardadas en el módulo de memoria es igual a

la cantidad de imágenes a entrenar. En este caso, si la cantidad de imágenes almacenadas es igual a la deseada, entonces se solicita el estado siguiente, caso contrario, se solicita el estado **“Take_character”.**

5. **finish_load_RAM**: Se inicializa la variable con la cual se llevará a cabo la administración del proceso de aprendizaje, lo cual una vez hecho, permite la solicitud del siguiente estado.
6. **select_pattern:** Se lleva a cabo la administración del entrenamiento de las neuronas post sinápticas, lo cual se logra mediante distintas banderas de control, las cuales llevan a múltiples estados dependiendo de éstas. El objetivo principal de este estado es realizar la selección de la neurona post sináptica que se ha de entrenar, así como determinar qué neurona ya ha pasado por este proceso. Una vez generadas las banderas para realizar el entrenamiento, éste solicita el siguiente estado, caso contrario, permanece en el mismo estado.

A partir de este el punto, el control primario comienza a relacionarse con el control secundario que existe en cada uno de los módulos “STPD_Process”, en este caso, dependiendo de las banderas activadas, se activará un módulo en particular. Para darnos cuenta qué neurona post sináptica se está entrenando, cada uno de los estados tendrá un identificador numérico que se asocia con ella.

1. **Load_pattern_x**; Se realiza la lectura de la imagen a entrenar almacenada en el módulo “Memoria_RAM_pattern”, para enviárselo a “Entry_process”; una vez realizada esta acción se solicitará el estado siguiente, caso contrario, éste permanecerá en el mismo.
2. **Training_pX_s**: Se inicia el proceso de entrenamiento. En este caso el control primario espera recibir de “STDP_process” la terminación del proceso; si es el caso, se solicita el siguiente estado; caso contrario se permanece en el mismo.

3. **Iddle**: Se envía una solicitud para comenzar nuevamente el proceso de aprendizaje sobre otra neurona post sináptica y solamente, en el caso que aún existan imágenes disponibles para realizar dicho proceso, se solicita el estado **“select_pattern”**, mientras que, en caso contrario, se solicita el siguiente estado.

En este punto, comienza la etapa de reconocimiento, la cual solamente podrá ser afectada a través de los puertos “recognize” y “other pattern”.

1. **Recogize_pattern_s1**: Si los puertos “recognize = 1” y “other_pattern = 0”, entonces, se preparan “Entry_process” y todos los módulos “STDP_process” para entrar en la etapa de reconocimiento, con lo que adicionalmente se lleva a cabo la carga de la imagen a reconocer. Una vez verificados dichos procedimientos, se solicita el siguiente estado, caso contrario, se permanece en el mismo.
2. **Recognize_idle_s2**: Se habilita el módulo “Work_Window”, para abrir una ventana de tiempo sobre la cual tendrá lugar el reconocimiento. Una vez verificado dicho procedimiento se solicita el siguiente estado, caso contrario, se permanece en el mismo.
3. **Recognize_init_w_s3**: Se solicita la habilitación de los módulos “Work_Window, “Entry_process” y “STDP_Process”, tarea que una vez verificada, se solicita el siguiente estado, caso contrario, se permanece en el mismo.
4. **Recognize_init_process_s4**: inicia la etapa de reconocimiento, por lo que se envía la imagen a reconocer a “Entry_process”, de manera simultánea se habilitan todos los módulos “Entry_process” para procesar los pulsos pre sinápticos mediante la sinapsis e iniciar la ventana de tiempo en “Work_Window”. Una vez verificada la operación de dichos módulos, se solicita el siguiente estado, caso contrario, se permanece en el mismo.
5. **Recognize_end_process_s5**: Se espera la terminación de la ventana de tiempo designada en el módulo “Work_Window”, se inhabilita al

módulo “Entry_process”, y termina la sinapsis que realiza “STDP_process”. Una vez verificada la culminación de dichas tareas, el control principal determina qué neurona fue la ganadora e inhabilita las neuronas perdedoras a través de “General_Spike_control”; en este caso se solicita el siguiente estado, caso contrario, se permanece en el mismo.

6. **recognize_other_pattern_s6**: Si los puertos “recognize = 0” y “other_pattern = 1”, entonces el proceso de reconocimiento se repite nuevamente, por lo que se solicita el estado “**recogize_pattern_s1**”, caso contrario, se permanece en el mismo estado.

En la figura (3.6) se puede ilustrar el control descrito anteriormente, donde las etapas de carga de patrones, entrenamiento y reconocimiento se encuentran en color rojo, naranja y verde respectivamente, señalando los estados que pertenecen a cada una, así como las banderas de control que solicitan los cambios entre los diferentes estados.

La operación de la máquina de estados puede simularse en el banco de pruebas etiquetado como ““Main_Machine_State_tb”. En cuanto a los puertos de entrada para la simulación mostrada en las figuras (3.7)-(3.9), estos son etiquetados como “clk”, “rst”, “general_flag_m(4:0)”, mientras que sus puertos de salida son etiquetados como “state_module (5:0)” y “task_m(5:0)”. Sin embargo, es necesario mencionar que “next_state” y “count” son variables, la primera interna y la segunda del banco de pruebas. En este caso “next_state” almacena el valor de los estados y “count” permite la visualización de estos a través de las figuras mencionadas.

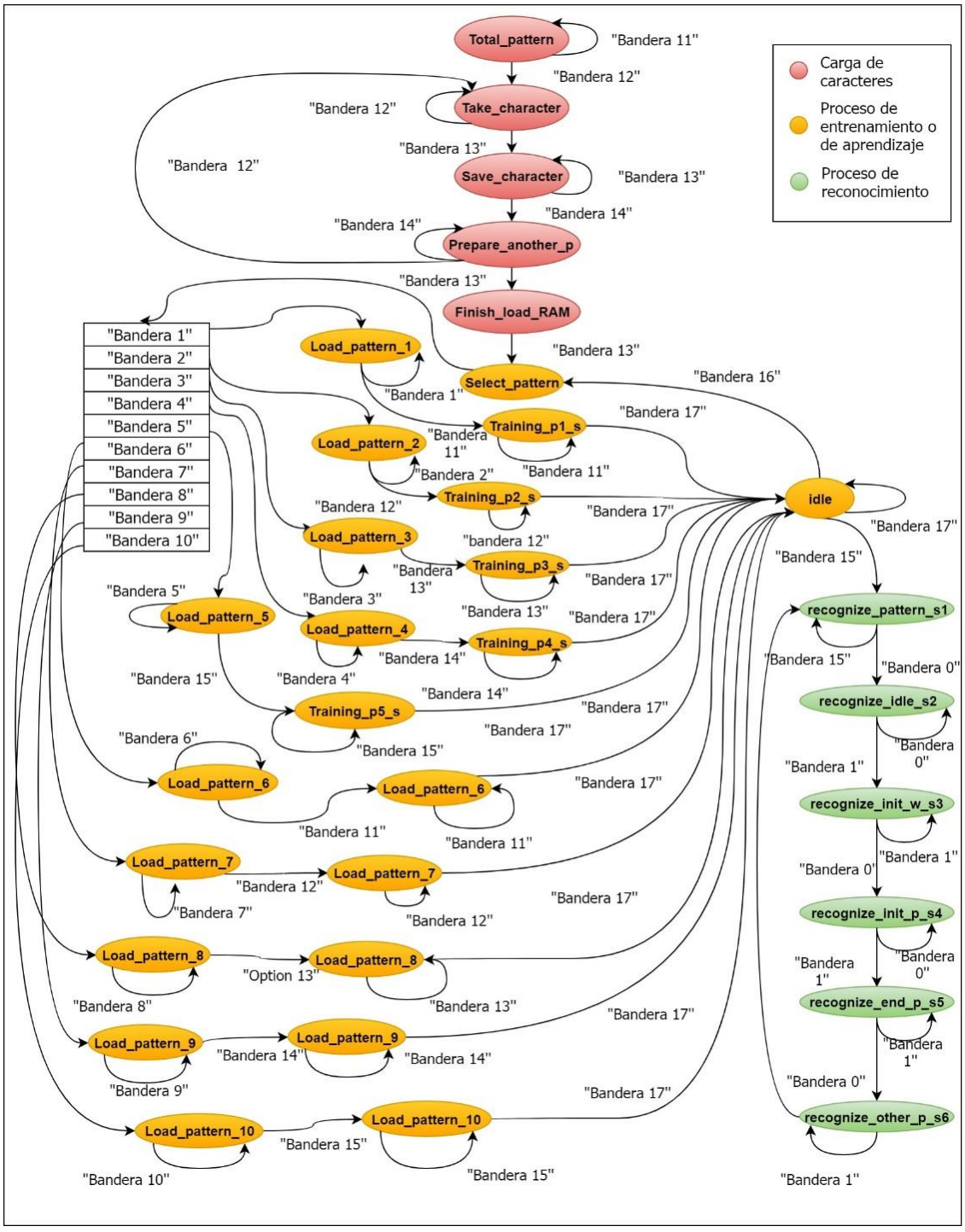

Fig. 3.6 Diagrama de estados que implementa "Main_Master_Control" y "Main_Machine_State" como Control Primario.

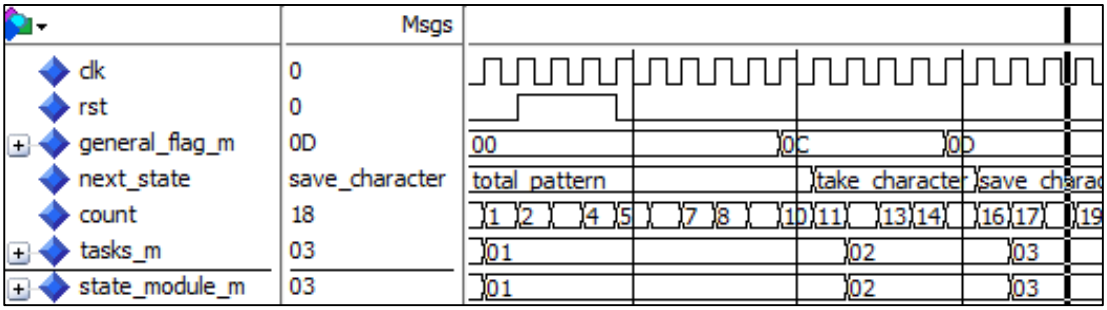

Fig. 3.7. Operación de la máquina de estados en "control primario". Tres primeros estados de la etapa de "carga de patrones".

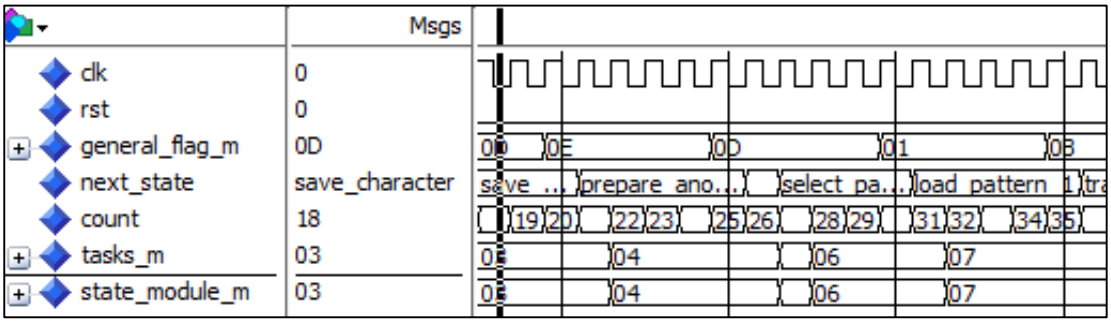

Fig. 3.8. Operación de la máquina de estados en "control primario". Dos últimos estados de la etapa de "carga de patrones" y primeros estados de la etapa del "proceso de entrenamiento o de aprendizaje".

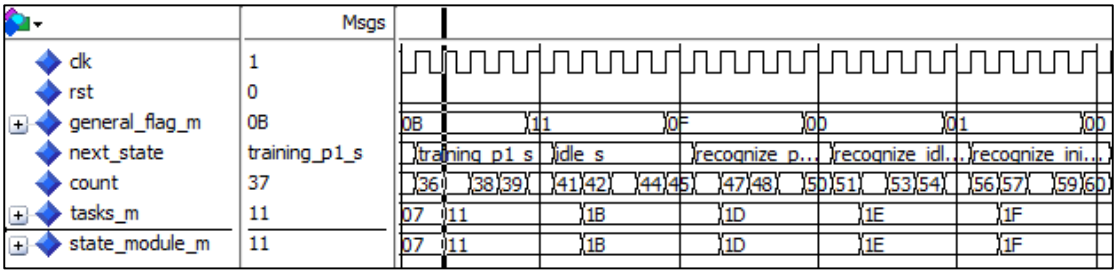

Fig. 3.9. Operación de la máquina de estados en "control primario". Últimos estados de la etapa del "proceso de entrenamiento o de aprendizaje" y primeros estados de la etapa del "proceso de reconocimiento".

3.5.1.1.2 Ventana de tiempo.

El módulo etiquetado como "Work_Window" lleva a cabo la generación de una ventana de tiempo, con el objetivo de establecer un tiempo durante el cual se realiza el proceso de integración sináptica y que éste no se mantenga de manera indefinida durante la etapa de reconocimiento. Esto tiene 2 beneficios: establecer la velocidad de reconocimiento, y evitar que el valor de corriente no supere el ancho de palabra definido por el sistema.

La implementación del módulo consiste en un contador libre ascendente, controlado por el flanco de subida del sistema de reloj general del sistema, así como un conjunto de banderas de control para interactuar con el “control primario”. La longitud de la ventana de tiempo implementada tiene una longitud de 80,500 ciclos de reloj y dado que el ciclo base de sistema representa numéricamente 0.0078125 ms, la longitud de la ventana de tiempo es 628.9 ms; sin embargo, tomando en cuenta la frecuencia de operación del sistema a 10 MHz, la duración de la ventana de tiempo es de 8.05 ms en tiempo real.

El funcionamiento puede verse ilustrado en la figura (3.10), el cual es generado a través del banco de pruebas etiquetado como “Work_Window_tb”, donde los puertos de entrada son “clk”, “enable_w_time” y “reload”. Los puertos de salida son “end_window”, “flag_enable_w_time” y “flag_reload”. Cabe mencionar que “count_w” es una variable interna la cual refleja el conteo de la ventana de tiempo.

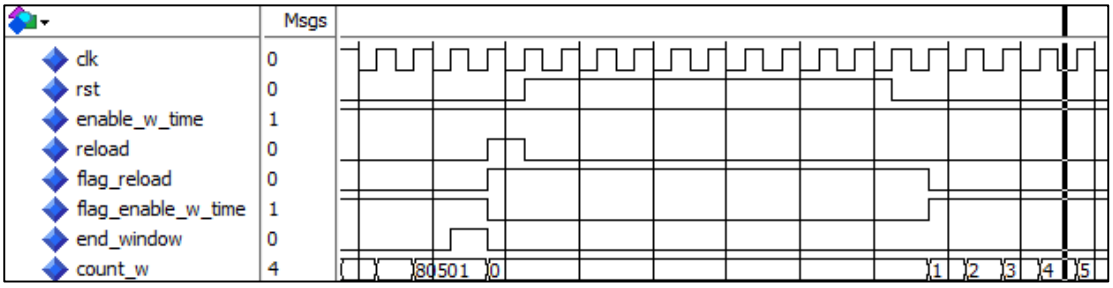

Fig. 3.10. Operación del módulo “Work_Window”.

3.5.1.1.3 Memoria RAM

El módulo de memoria RAM etiquetado como “Memoria_RAM_pattern” está basado en una operación tradicional justo como se describe en [1]. Sin embargo, adicionalmente se agregaron banderas para indicarnos el estado en el cual se encuentra dicho módulo y poder establecer una lógica más eficiente al momento de controlar su operación. El objetivo principal de este módulo será almacenar las imágenes a entrenar por el sistema, por lo que la cantidad de localidades máximas que tiene son 10 (cantidad de neuronas post sinápticas) y la longitud del ancho de palabra es de 36 bits (cantidad de pixeles del caracter).

El funcionamiento puede verse ilustrado en las figuras (3.11) y (3.12), el cual es generado a través del banco de pruebas etiquetado como “Memoria_RAM_pattern_tb”. Los puertos de entrada son: “addr_p(4:0)”, “p_in(35:0)”,” wr_en(1:0)”, “clk”, “enable_RAM_pattern” y “rst”, mientras que los puertos de salida son: “flag_wr(1:0)”, “p_out(35:0)”, “enable_flag_RAM_pattern” y “enable_w_r”. Cabe mencionar que “signal_count” es una señal del banco de pruebas para mejorar la visualización de los procesos mediante un contador y “memory_pattern” es la memoria RAM en sí.

En la figura (3.11) se muestra la escritura de tres caracteres a través de p_in (35:0) y “wr_en” que controla la lectura/escritura, mientras que en la figura (3.12) se muestra el caso opuesto, la lectura.

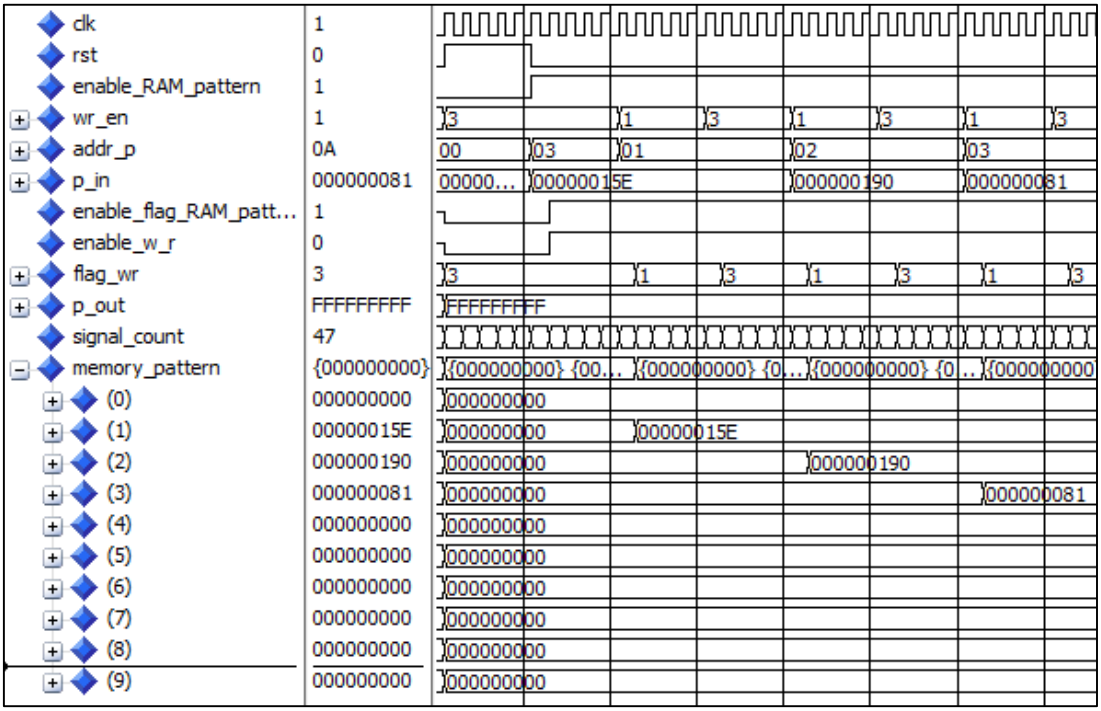

Fig. 3.11.Escritura del módulo “Memoria_RAM_pattern”.

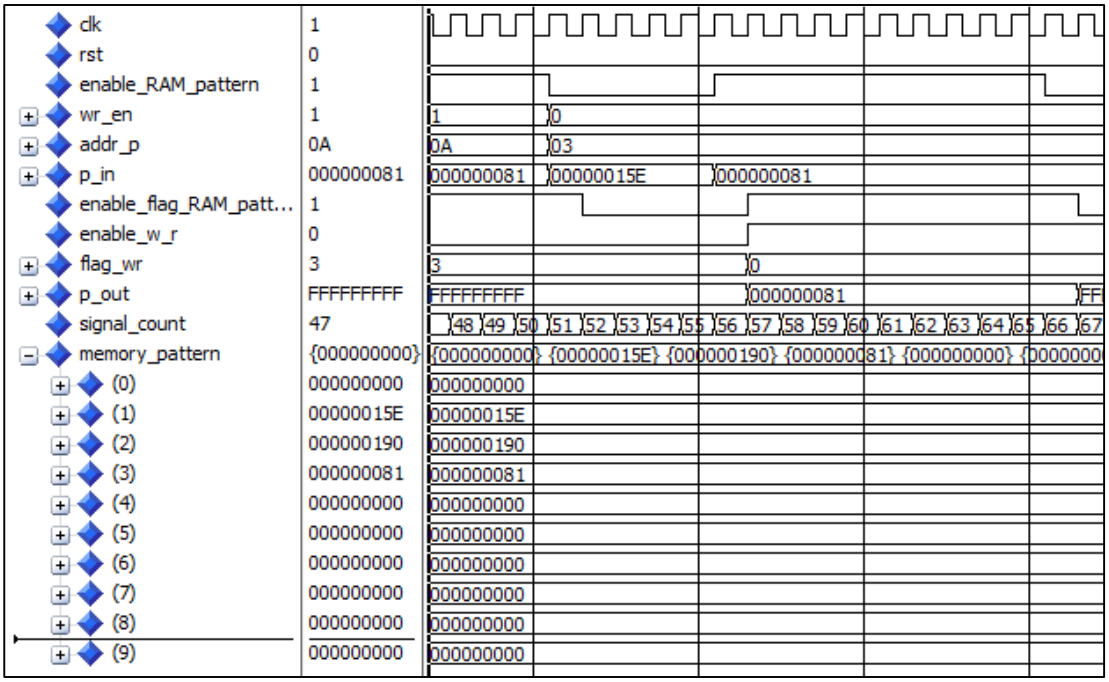

Fig. 3.12.Lectura del módulo "Memoria_RAM_pattern".

3.5.2 Capa de entrada.

El módulo "Entry_process" es un módulo de jerarquía superior que se forma a partir de 3 bloques más simples y que a su vez, engloban a otros módulos para llevar a cabo la operación del mismo. En este caso, el módulo está formado por los módulos "Process_Data_Input", "AER_System" y "FIFO_all_in_one" descritos posteriormente de manera individual en las secciones (3.5.2.1), (3.5.2.2) y (3.5.2.3), respectivamente.

La implementación de estos tres bloques puede verse ejemplificada a través del diagrama a bloques que se presenta en la figura (3.13).

En cuanto a su operación, ésta puede ser descrita como sigue, cada vez que se le presenta una imagen de 6x6:

1. El módulo "Process_Data_Input" genera los potenciales de acción de las neuronas asociadas a cada pixel de la imagen entrante, adicionando un retardo aleatorio para cada conexión sináptica, con lo que a la salida se tendrán 36 secuencias de pulsos generados por cada una de las

neuronas que se encuentran físicamente dentro del mismo y que se envían al módulo “AER_System”.

2. Las secuencias de potenciales provenientes de “Process_Data_Input” llegan al módulo “AER_System”, el cual, asigna direcciones. Cada una de ellas con la finalidad de comunicar la información espacio-temporal de las neuronas que se encuentran en el módulo anterior, y construir con estos una sola secuencia de eventos, por lo que, a su salida, lo único que se obtiene es un tren de pulsos construido con direcciones de todas las neuronas pre sinápticas.
3. El módulo “FIFO_all_in_one” recibe el tren de eventos generados por “AER_System” y agrega un retardo de propagación para el procesamiento posterior de estos eventos en el tiempo, generando de manera conjunta a la salida un pulso cada vez que hay una dirección conocida. Por otra parte, de manera simultánea, el módulo permite la retro alimentación a la red neuronal a través de la concentración de pulsos post sinápticos, agregando una compensación en tiempo para eliminar el tiempo de procesamiento y agregando un retardo de retro propagación por cada neurona post sináptica.

En general, a la salida del módulo “Entry_process”, lo único que observaremos, será un tren de pulsos construido a partir de los pulsos pre sinápticos de 36 neuronas de entrada, las cuales ya contienen un retardo de propagación, y que adicionalmente servirá de medio para agregar retardos de retro propagación en los pulsos post sinápticos de las 10 neuronas de la capa de salida.

El funcionamiento anteriormente descrito puede verse ilustrado en las figuras (3.14)-(3.15) generadas a través del banco de pruebas etiquetado como “Entry_Process_tb”. Los puertos de entrada son “pixel (35:0)”, “pos_in_vector (0:9)”, “clk”, “reload_firing” y “rst”, mientras que los puertos de salida son “add_pre_out (7:0)”, “pos_out_vector(0:9)” y “pre_spike”. Cabe mencionar que “signal_counter”, es una señal del banco de pruebas para mejorar la

visualización de los procesos mediante un contador. Adicionalmente se colocó por fuera del módulo un filtro etiquetado como “spike_X_dir” que realiza el conteo de las incidencias de las neuronas etiquetadas con las direcciones de los eventos, cuando en “pixel” se coloca una palabra completamente en unos (en hexadecimal “FFF...F”), y así observar el comportamiento del mismo; por otra parte, cabe mencionar que en esta simulación no se presentan ningún pulso post sináptico, para simplificar el banco de pruebas.

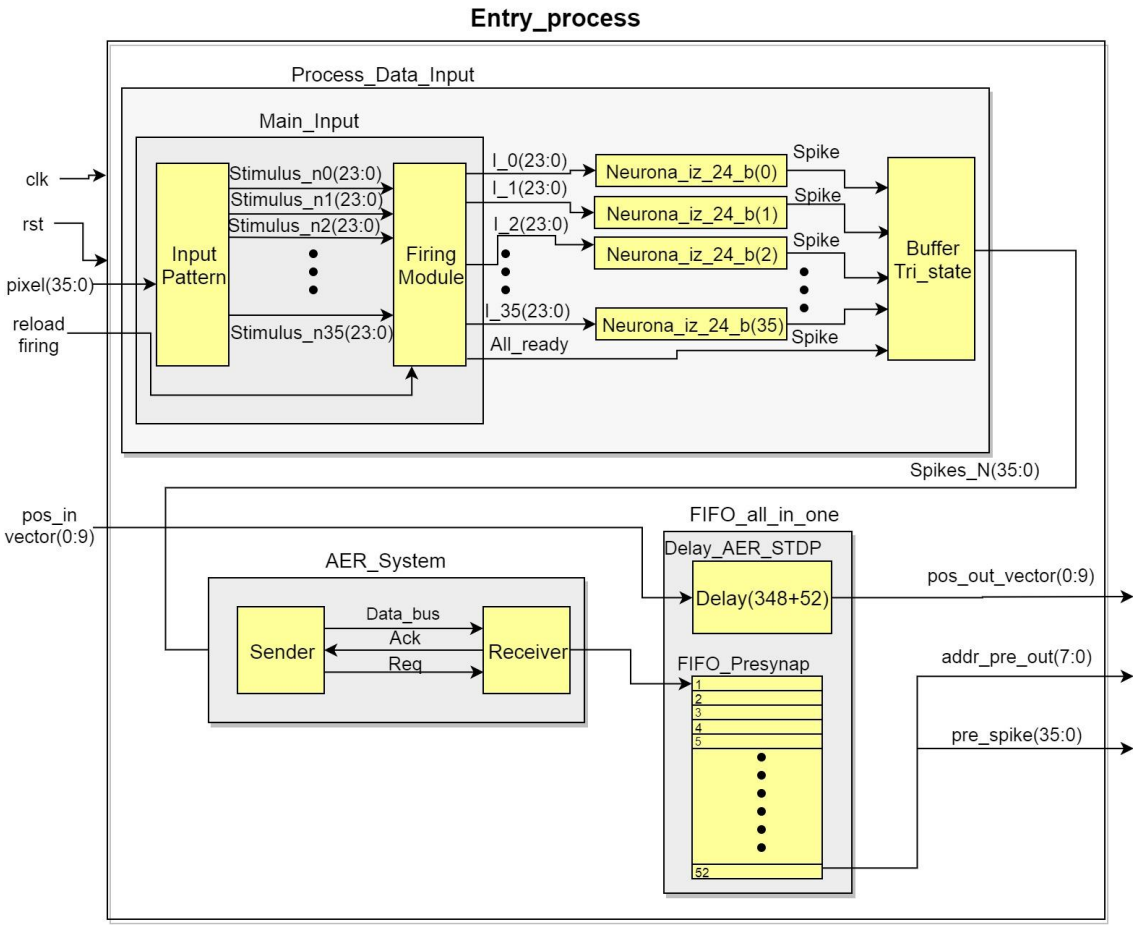

Fig. 3.13. Diagrama a bloques del módulo “Entry_Process”.

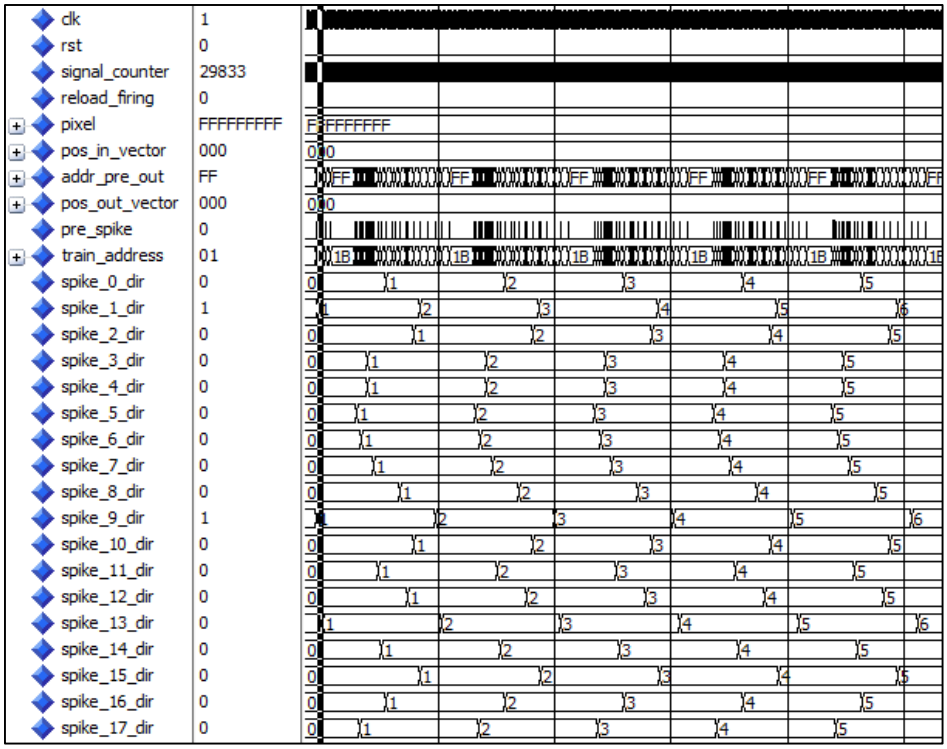

Fig. 3.14. Operación del módulo "Entry_Process". Filtrado y conteo de pulsos pre sinápticos generados por 18 neuronas etiquetadas con las direcciones en el rango de 0 a 17.

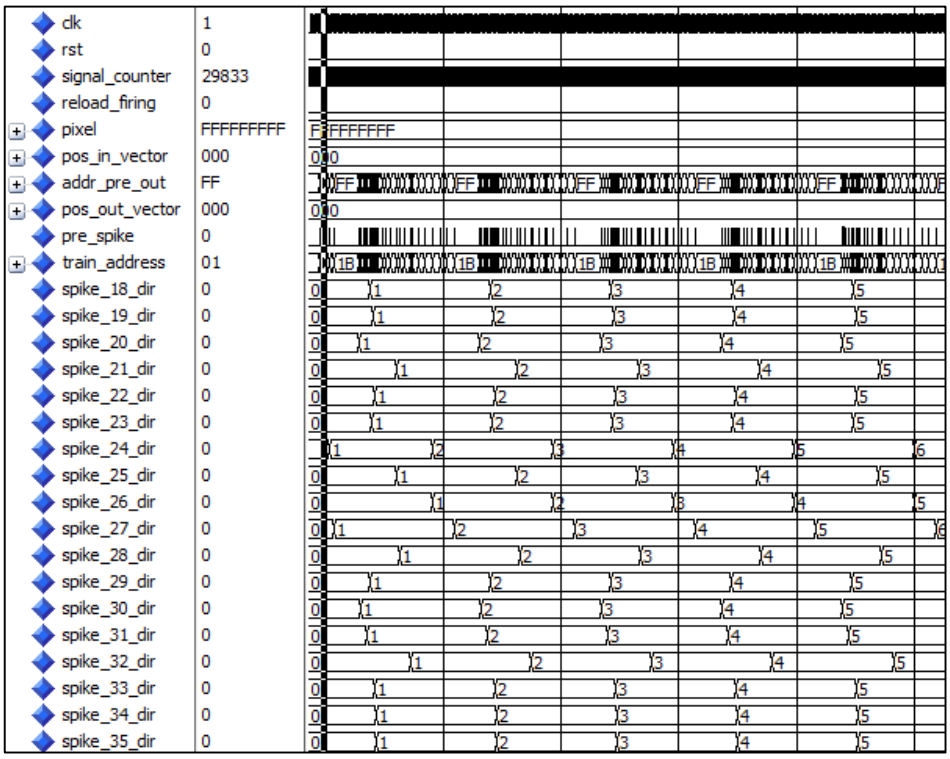

Fig. 3.15. Operación del módulo "Entry_Process". Filtrado y conteo de pulsos pre sinápticos generados por 18 neuronas etiquetadas con las direcciones en el rango de 18 a 36.

3.5.2.1 Módulo de entrada

El módulo de entrada etiquetado como “Process_data input” es un módulo de jerarquía superior que está integrado por los módulos “Neurona_iz_24_b”, “Main_input” y “tri_state_buffer_in”. En su conjunto, realizan la operación de vincular la imagen a blanco y negro a niveles de corriente, adicionando los retardos por desfase de tiempo, y vinculando cada una de estas salidas a una neurona en particular. Dado que el sistema contiene 36 neuronas, a la salida se podrán observar las secuencias de pulsos generados por ellas. En este sentido, las neuronas son implementadas mediante la replicación del módulo “Neuona_iz_24_b” vinculándose con la salida mediante el buffer tri-estado, el cual deja pasar todos los pulsos a la salida, una vez que el módulo “Main_Input” termina por ingresar los retardos. Tal descripción puede verse ilustrada través del diagrama a bloques presentando anteriormente en la figura (3.13).

El funcionamiento antes descrito puede verse ilustrado en la simulación de la figura (3.16), la cual es generada a través del banco de pruebas etiquetado como “Process_Data_Input_tb”, donde los puertos de entrada son “pixel (35:0)”, “clk” , “reload_firing y “rst”, mientras que los puertos de salida son “spikes_N (0:35)”. En dicha simulación, para verificar su comportamiento, se colocó en el puerto “pixel” la palabra en hexadecimal “FFFFF”, para observar que a la salida en el puerto “spikes_N” se generarán los respectivos trenes de pulsos desfasados unos de otros. Obsérvese que, de acuerdo a esta arquitectura, todos los módulos “Neuronas_iz_24b”, trabajan de manera paralela, mientras que el módulo “Main_Input” solo contribuye como módulo de excitación y “Tri_State_Buffer_In” como módulo de aislamiento hacia módulos posteriores.

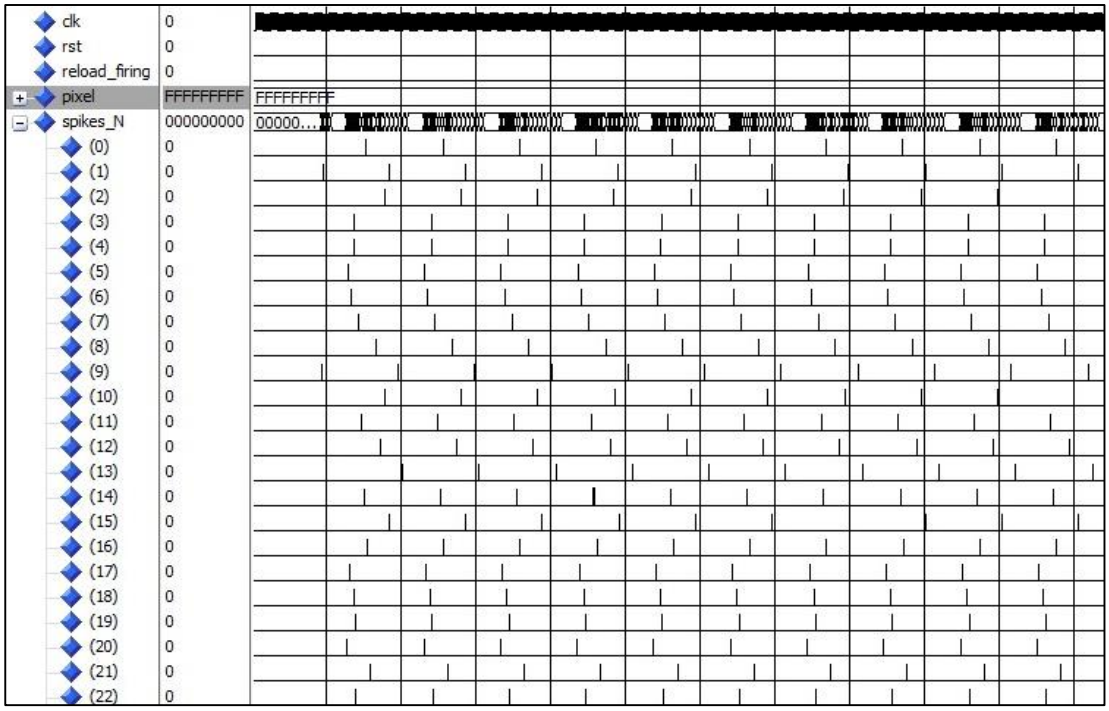

Fig. 3.16.Generación de secuencias de pulsos debido a la estimulación de las neurona pre sinápticas.

3.5.2.1.1 Conversor de pixel en corriente con retardo

Para reproducir el efecto de propagación del potencial de membrana que existe en la sinapsis y el de retro propagación que tiene lugar durante el proceso de STDP, esto se puede realizar agregando retardos en cada unión sináptica. Esta tarea es realizada a través de dos módulos etiquetados como “Input_Pattern” y “Firing_Module”, los cuales forman un módulo de jerarquía superior etiquetado como “Main_Input”. El módulo etiquetado como “Input_Pattern” es un conversor que realiza la acción descrita en la sección (2.8) del capítulo 2, y que consiste en cambiar un pixel negro a un nivel de corriente alta, en nuestro caso a una corriente fija de 8.22 pA, la cual fue seleccionada aleatoriamente, y los pixeles blancos inyectando una corriente igual a cero. El módulo etiquetado como “Firing_Module”, es un módulo temporizado el cual deja pasar, en un intervalo aleatorio de tiempo, la corriente asociada a cada pixel, por lo que una vez dejando pasar todas las corrientes hacia los puertos de salida, éste genera una señal de un bit etiquetada como “all_ready”, la cual notifica que todas las neuronas conectadas a este módulo

ya contienen un retardo; en este sentido, se crea un retardo variándose el instante en que se inyecta corriente a la neurona.

La operación del módulo puede verse ilustrado en la figura (3.17), la cual es generada a través del banco de pruebas etiquetado como “Main_Input_tb”, donde los puertos de entrada son etiquetados como “pixel (35:0)”, “clk”, “reload_firing” y “rst”, mientras que los puertos de salida se encuentran etiquetados como “all_ready” y “s_nx(23:0)”, siendo este último replicado 36 veces e identificado con el número de su respectiva salida.

Cabe mencionar, que sin el módulo “Firing_Module”, la corriente se inyectaría en todas las neuronas al mismo tiempo, con lo que los instantes de tiempo en que tienen lugar los potenciales de acción para las “n” neuronas también serían iguales, esto aplicaría en dado caso que se desee tener un modelo sináptico que no contenga retardos. Finalmente, los retardos aplicados se hacen de manera aleatoria en un intervalo de 30,000 ciclos de reloj de sistema y, dado que éste representa numéricamente 0.0078125 ms, los retardos están en el rango de 234.36 ms numéricamente, mientras que en tiempo real, tomando como base del sistema una frecuencia de operación de 10 MHz, los retardos estarían en un rango de 0 a 3 ms.

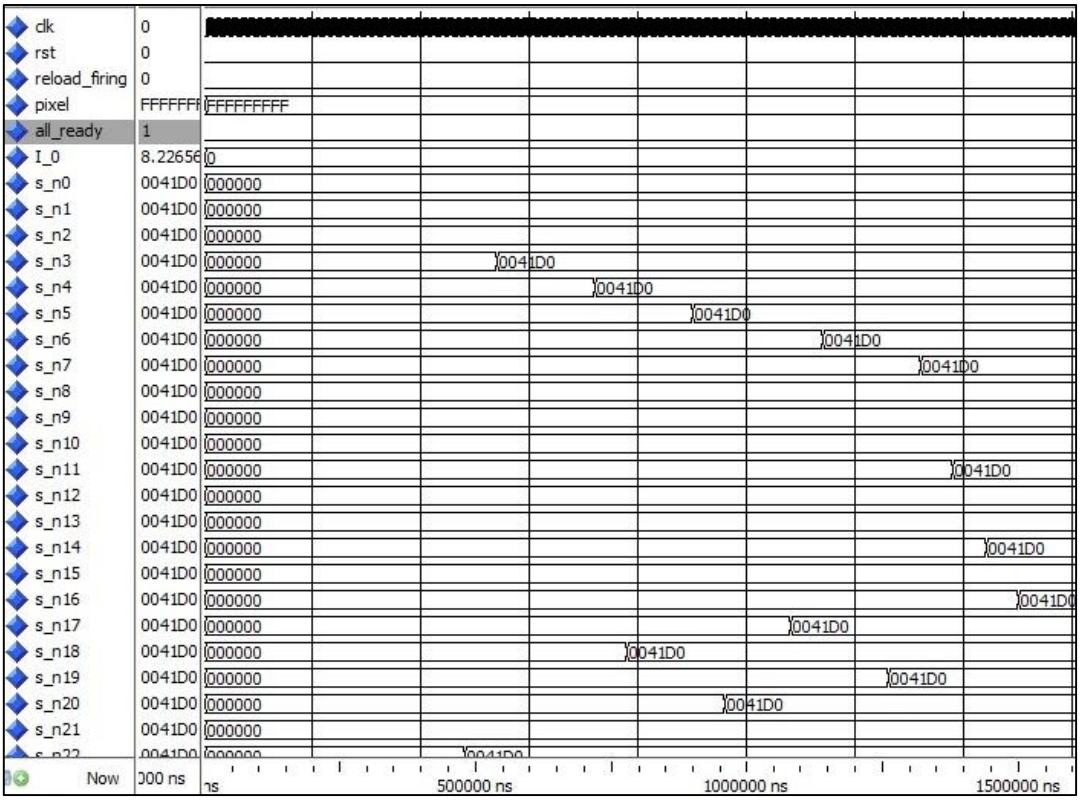

Fig. 3.17. Retardo agregado desde la inyección de corriente sináptica en la capa de entrada.

3.5.2.1.2 Neurona Izhikevich.

Para describir en hardware el modo RS del modelo Izhikevich se usó una representación a 24 bits con 13 bits para la parte entera y 11 bits para la parte decimal como se menciona en la sección (3.1.3). Esta representación también nos permite aproximar al valor decimal de los parámetros designados por Izhikevich y con ello evitar variaciones en la dinámica del modelo, ya que, apegarnos tal cual a esos parámetros, se necesitaría una mayor cantidad de DSP´s y LUT´s para procesar una palabra de mayor longitud, reduciendo la cantidad de neuronas que pudiésemos llegar a implementar.

Los parámetros que se seleccionaron pueden observarse en la tabla (3.1) en su representación decimal y hexadecimal, donde se observa también la unidad de tiempo que integra numéricamente el módulo a través de la variable “dt” en cada ciclo de reloj.

Parámetros	Valor Decimal	Valor Hexadecimal
a	0.015625	000020
b	0.125	000100
c	-65	FDF800
d	8	004000
dt	0.0078125 [ms]	000010

Tabla 3.1.Valores del modo RS de Izhikevich en Hardware.

La neurona de Izhikevich en su modo RS se encuentra en el módulo etiquetado como “Neurona_iz_24_b” y su operación puede simularse en el banco de pruebas etiquetado como “Neurona_iz_24_tb”. En cuanto a los puertos de entrada, estos son etiquetados como “I (23:0)”, “clk” y “rst”, mientras que su único puerto de salida es etiquetado como “spike”. Cabe mencionar que, para observar el comportamiento analógico de la neurona, es necesario mandar a llamar las variables “u” y “v” que se encuentran internamente en dicho módulo.

En cuanto a su operación, en la figura (3.18) se observa una simulación donde se ingresa al modelo de Izhikevich una corriente de 15 pA constante que en su formato hexadecimal corresponde a “007800” y que se ve representado a través de la variable “I”. En este caso, cuando se polariza a la neurona con ese nivel de corriente, el potencial de membrana y la variable de recuperación son representadas a través de las variables “v” y “u” respectivamente, el módulo representa los potenciales de acción a través del puerto etiquetado como “spike” mediante pulsos digitales, los cuales llevan la información temporal de la neurona cada vez que se supera el potencial de disparo.

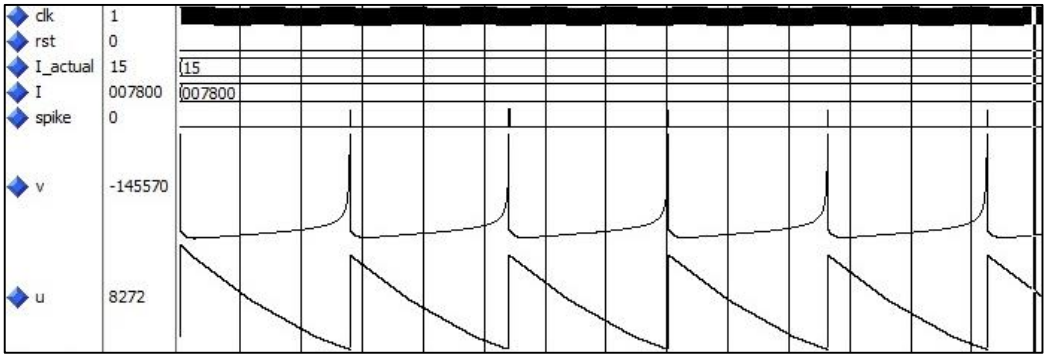

Fig. 3.18.Modelo Izhikevich en hardware polarizada con una corriente constante de 15 pA.

Por otra parte, en cuanto a la operación que tendrá la neurona una vez insertada como parte del sistema de reconocimiento de caracteres, cabe mencionar, que lo que se busca es aproximarnos a la plausibilidad biológica, por lo que durante el funcionamiento, en lugar de reiniciar “propiamente” a la neurona, se inyectarán niveles de corriente igual a cero cuando deseemos que la neurona deje de funcionar, tal como se observa en la figura (3.19), donde se parte desde una corriente “0” hasta niveles de corriente relativamente altos (243 y 228 pA). En tal situación, se puede observar que la variable de recuperación experimenta cambios que pueden afectar en la generación de potenciales de acción posteriores.

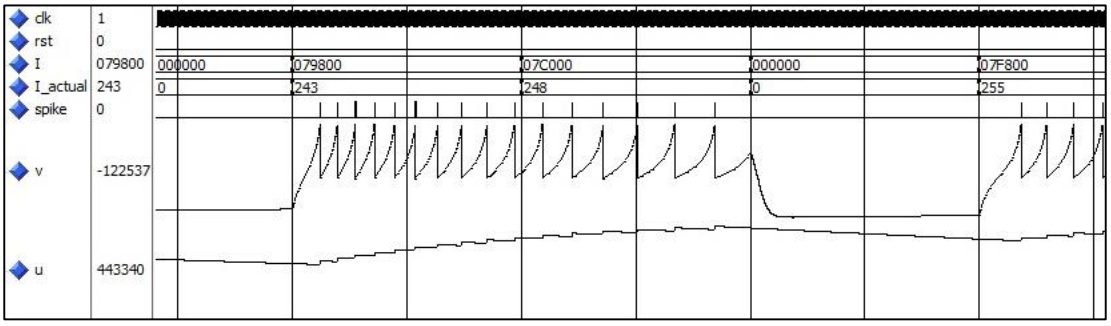

Fig. 3.19. Modelo de Izhikevich en hardware polarizada con diferentes corrientes.

Haciendo énfasis en el “reset”, obsérvese que reiniciar la neurona consiste en poner un valor predeterminado en las variables “u” y “v” como punto de partida sobre el cual la neurona se inicializa; sin embargo, como vemos en la figura (3.20), este “reset” interviene en la dinámica neuronal, ya que somos nosotros quienes posicionamos a la neurona en este punto. La finalidad de evitar la realización propiamente de un “reset” durante la operación y antes de iniciar un proceso de sinapsis o de aprendizaje, es permitir que la actividad neuronal evolucione a lo largo del tiempo, de tal forma que tanto el potencial de membrana como la variable de recuperación sean totalmente dependientes de los cambios de corriente que se dan a lo largo del tiempo.

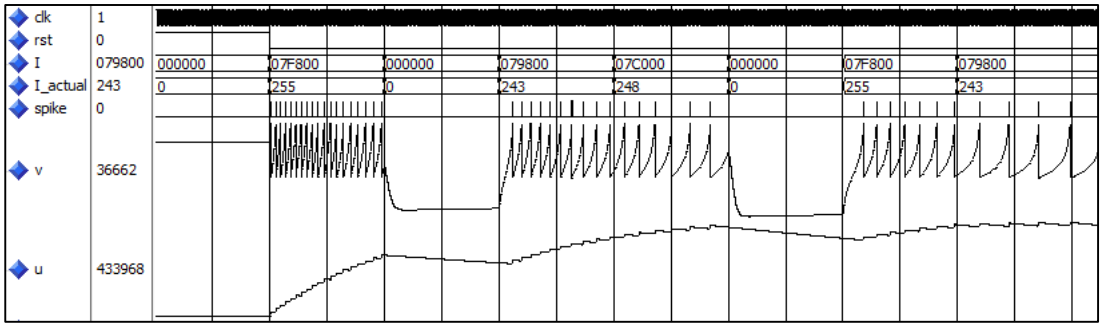

Fig. 3.20. Reset y evolución de las variables "v" y "u" en el tiempo.

3.5.2.1.3 Buffer tri-estado de salida

Un buffer tri-estado, en nuestro caso, nos permitirá implementar un sistema de control, para asegurar que ningún pulso proveniente de las neuronas de entrada se propaguen hacia el exterior hasta que se haya recibido una señal de control. Dicho sistema se encuentra embebido en el módulo etiquetado como "Tri_State_Buffer_in" y su operación puede simularse en el banco de pruebas etiquetado como "Tri_State_Buffer_In_tb". En cuanto a sus puertos de entrada estos son etiquetados como "spike_in (0:35)" y "all_ready", mientras que sus puertos de salida son etiquetados como "spikes_N (0:35)".

En la figura (3.21) se puede observar que los pulsos que se reciben a través del puerto "spikes_in", solo pueden verse reflejados en la salida "spikes_N", cuando "all_ready" se encuentra en estado alto.

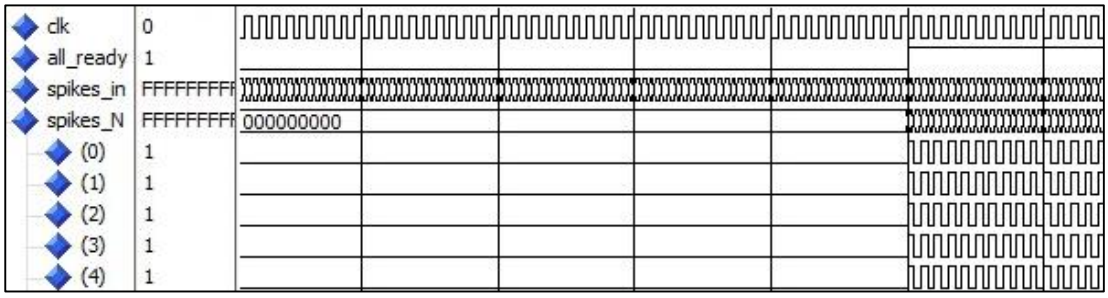

Fig. 3.21. Operación del módulo"Tri_State_Buffer_in.

Obsérvese, que los puertos "spikes_in" y "spikes_N", contienen la misma cantidad de líneas y "all_ready" sirve solo como una compuerta de cierre/apertura.

3.5.2.2 Módulo de representación de eventos (AER)

"AER" por sus siglas en inglés Address Event Representation, es un protocolo de comunicación digital multiplexado y asíncrono, implementado a través de un protocolo de "Handshake" simple, el cual está inspirado en el mecanismo de comunicación de las neuronas biológicas y su finalidad es ser medio de comunicación en circuitos altamente interconectados, preservando la información espacio-temporal generada por cada uno de sus elementos, como describe Gómez en [14]

AER en sus inicios fue creado para resolver la comunicación en retinas artificiales en circuitos analógicos VLSI, ideado inicialmente por Massimo Antonio Sivilotti en 1991 en el Instituto Tecnológico de California (Caltech) descrito en [15], como un "data driven process" donde cada pixel, cuando generaba datos, enviaba una solicitud a un módulo que se encargaba de administrar la información de todos los dispositivos de entrada y esperaba la recepción de su petición, para enviar sus datos y este módulo los transmitiera por un bus compartido de alta velocidad. Sin embargo, fue Misha Mahowald en 1992 (también en el instituto Tecnológico de California), quien desarrollaría en [16], lo que llamó el "interchip communication protocol" usando una técnica que ella misma designó como "Address-Event Representation" y que actualmente da el nombre a dicho protocolo

La idea principal del protocolo de AER es que si hay un arreglo de neuronas que generan pulsos en una ubicación diferente dentro de un circuito, cada una de éstas puede ser etiquetada con una sola dirección digital que la identifica. De esta manera, cada vez que se genera un pulso independientemente de la neurona que lo generó, el sistema pasará la dirección asociada a ella a través de un bus de alta velocidad, evitando que se pierda la información espacio-temporal de los pulsos; así, cuando no hay pulsos, no hay direcciones en el bus. AER permite que varias direcciones pueden ser multiplexadas en el mismo bus, reduciendo la cantidad de líneas físicas de interconexión, sirviendo como un concentrador de nodos conectado a un módulo emisor que envía la

información individual de cada uno de los pulsos de las neuronas a un módulo receptor que identifica la llegada de esta información como un evento etiquetado con la dirección correspondiente, con lo que este último, interpreta que un potencial de acción fue generado en la neurona correspondiente.[16] [4].

“El uso de direcciones digitales para especificar la neurona emisora hace el mapeado de señales pre-sinápticas sobre destinos post-sinápticos extremadamente flexible, porque el evento lleva su posición de origen en sí mismo. La representación AER garantiza el orden temporal de los eventos y el evento puede ser fácilmente decodificado en cualquier ordenación física en el circuito receptor. Alternativamente, el patrón de conectividad se puede cambiar dinámicamente, lo que permite cambiar la estructura de la interconexión de las neuronas artificiales tal y como ocurre en los sistemas biológicos.” [14]

En cuanto a su implementación, el protocolo de comunicación que soporta el sistema AER, está basado en el protocolo de “handshake”, o “apretón de manos”, y como cualquier saludo entre 2 personas, un apretón de manos simple involucra 2 chips, es decir, un emisor y un receptor [16]. El chip que es conocido como emisor inicia el proceso de comunicación, mediante un indicador conocido como “Initiation”, el cual inicia todo un proceso de señalización para la transmisión del evento, que una vez generado, a través del flanco de subida de esta señal, se envía un aviso de envió a través de una línea de transmisión conectada hacia el receptor, que se conoce como “Request”, la cual permite a través de su flanco de subida, cargar y enviar la dirección del evento a través de un bus de alta velocidad ya sea en serie o en paralelo. Una vez que el receptor detecta el flanco de subida del primer bit de la dirección, éste le envía una respuesta conocida como “Acknowledge” a través de un línea también independiente hacia el emisor, con el cual se concreta el “apretón de manos” entre los dos módulos, notificando que la dirección ha sido recuperada y nuevamente, al ser detectado el flanco de

subida de la señal “Acknowledge” por el emisor, la señal “Initiation” vuelve a estado bajo en el emisor, para comenzar nuevamente el proceso de transmisión de la información, desactivado todas la señales a través de su flanco de bajada, tal como se muestra en la figura (3.22) [16].

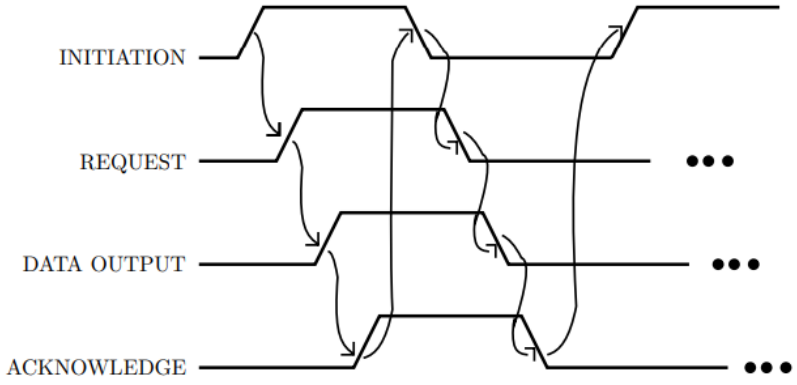

Fig. 3.22.Protocolo de handshake en AER. Extraída de [16].

Como podemos observar, dado que este proceso es asíncrono, el control de la información se basa en los flancos de subida y de bajada de cada una de estas 4 señales: “Initiation”, “Request”, “Data Output” y “Acknowledge”, todas ellas, activadas y desactivadas secuencialmente, cada vez que comienza y termina el “apretón de manos”.

“AER” a nivel modular, puede ser visto como un sistema conformado por 2 módulos, emisor y receptor, de los cuales “Request”, “Data Output” y “Acknowledge” tienen líneas de transmisión físicas tal como se muestra en la figura (3.23).

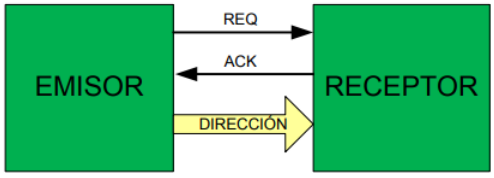

Fig. 3.23. Módulos y líneas físicas que conforman el módulo AER convencional. Extraído de [14].

Sin embargo, obsérvese que “initiation” no tiene línea de transmisión física y esto es debido a que se encuentra embebida dentro del módulo emisor, a través de un circuito interno que la controla conocido como “arbiter”. Este

circuito, se encarga de administrar el tráfico de la información en caso que dos o más neuronas pulsen al mismo tiempo, por lo que será este circuito, el que se encargará de preservar la integridad de la información espacio-temporal en lo más posible, tomando en cuenta que solo una dirección puede pasarse a través del bus de datos. Tal circuito puede verse ilustrado en la figura (3.24).

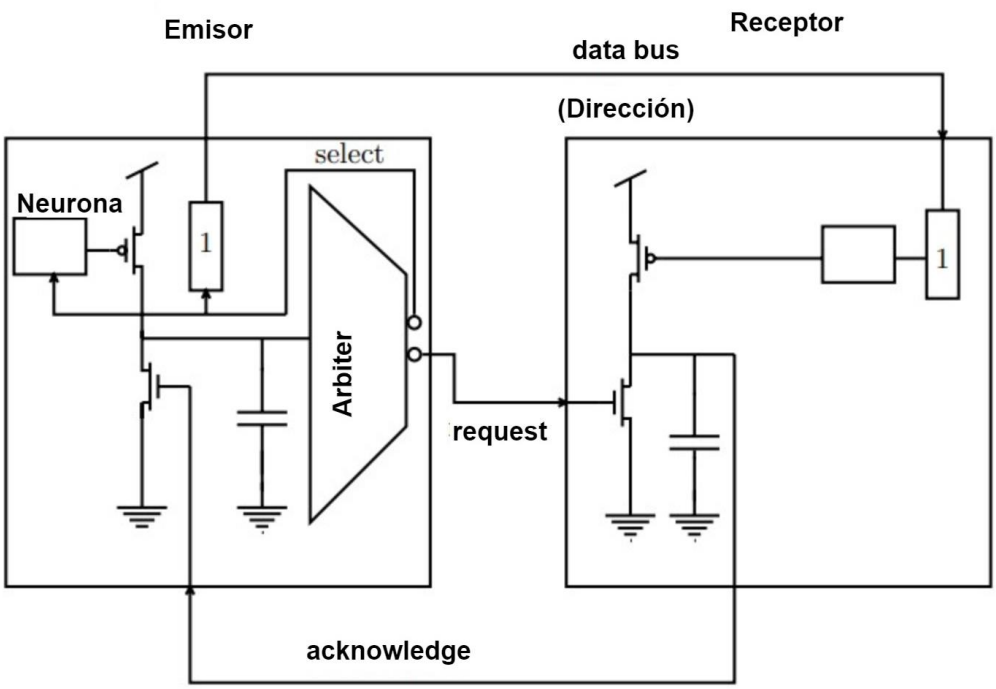

Fig. 3.24.Modulo AER analógico. Extraído de [16].

En general, el sistema AER permitirá computar la información espacio-temporal de la red neuronal en tiempo continuo, y se encargará de resolver colisiones entre pulsos provenientes de diversas neuronas, generando un solo tren de pulsos, construyendo un patrón espacio-temporal específico para cada estímulo. AER en su conjunto, nos permitirá replicar el comportamiento teórico espacio-temporal de una red neuronal biológica y nos ayudará a reproducir dicha integración en STDP y en la sinapsis.

En cuanto a las implementaciones de AER, la terminología para referirse a estos módulos ha cambiado, identificando al emisor como codificador de direcciones o arbitrador (por razones antes expuestas) y al receptor como decodificador de direcciones [14][4], dándole este nombre a este último porque

en recientes implementaciones realiza un operación adicional, la cual es filtrar algunas de esas direcciones, ya sea para darles un procesamiento distinto o bien para enviarlas hacia otros módulos sí así se requiere, existiendo múltiples salidas del sistema AER, tal como se muestra en la figura (3.25).

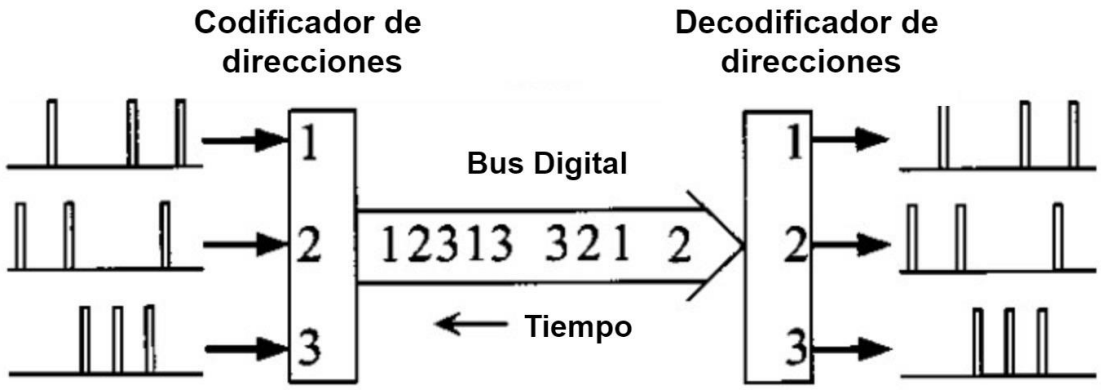

Fig. 3.25. Arquitectura actual de un módulo AER convencional. Extraído de [4].

Basado en el trabajo de Boahen en [4], 8 años después de haber sido propuesto por Mahowald y Siviloti, AER emergió como candidato líder en la comunicación de chips neuromórficos; sin embargo, actualmente uno de los retos más grandes del protocolo AER es estandarizar las especificaciones eléctricas, la velocidad de transmisión, ancho del bus, operación del arbitrador, pérdida máxima de información, nivel de reconstrucción de la información, entre otros aspectos, para hacer compatible el módulo AER no solo con el sistema en el que será insertado, sino con múltiples módulos AER; problema que se hace evidente al momento en que cada uno de los grupos de investigación alrededor del mundo implementan sus propios estándares y por consiguiente haciendo que sus diseños sean incompatibles entre sí [17].

En los últimos años, en la investigación de AER se han desarrollado versiones en hardware reconfigurable FPGA [18], se han implementado sistemas para el control de la transmisión secuencial de las direcciones asignadas mediante nemónicos [19] y a través de máquinas de estado [20], como parte de sistemas de reconocimiento de caracteres de alta velocidad [21] y como sistemas

embebidos en robots móviles [22], pero principalmente en la implementación de retinas artificiales y en el desarrollo del propio protocolo [14] [17] [23].

En nuestro caso, para la implementación de nuestro sistema de reconocimiento de caracteres, implementaremos el sistema AER en su forma original, lo cual, dado que el algoritmo de entrenamiento se realiza de manera aislada y sabemos que todos los pulsos serán dirigidos a una sola neurona, etiquetaremos cada pulso con la dirección espacial de la cual provino, de esta manera, lo que obtendremos será un solo tren de pulsos con las direcciones correspondientes, tal como fue ideado inicialmente.

En cuanto a la operación, surgen dos diferencias, las cuales radican en que el sistema propuesto no es un sistema asíncrono, sino síncrono, el cual es más sencillo de implementar dado que el protocolo de comunicación “handshake” funciona a través de flancos, por lo que intercalando las instrucciones VHDL es posible mediante una señal de reloj en sus flancos de subida y de bajada, reproducir las 4 señales de control en solamente 2 ciclos, para de esta forma mantener la integridad de la información espacio-temporal.

En cuanto a la arquitectura, nuestro sistema mantiene las 3 líneas físicas de AER (Data Output, Request y Acknowledge) y los dos módulos que lo conforman (emisor y receptor), y se usará el enfoque propuesto por Gómez en [14] y por Paz en [17] para reproducir el protocolo handshake, el cual es conocido como el “protocolo AER punto a punto unidireccional”, el cual difiere del protocolo original de señalización ilustrado en la figura (3.22), ya que en el sistema a implementar la señal “Request” y la dirección válida etiquetada como “Address” son enviadas de manera simultánea y solo los flancos de subida y de bajada de “Request” y “Acknowledge” son los que terminan el protocolo. Tal como puede verse en la figura (3.26), esto tiene justificación en la implementación en FPGA, porque este proceso permite asegurarnos que cuando llegue la señal “Request” al módulo Receptor, en el bus ya se tenga la dirección del evento y ésta no quede indeterminada por cuestiones de temporización al momento que la tome del módulo receptor; una vez terminado

el proceso de envío-recepción se pondrá el bus de direcciones en estado de alta impedancia y, en nuestro caso será puesta una dirección completamente en unos (en formato hexadecimal en “FF”).

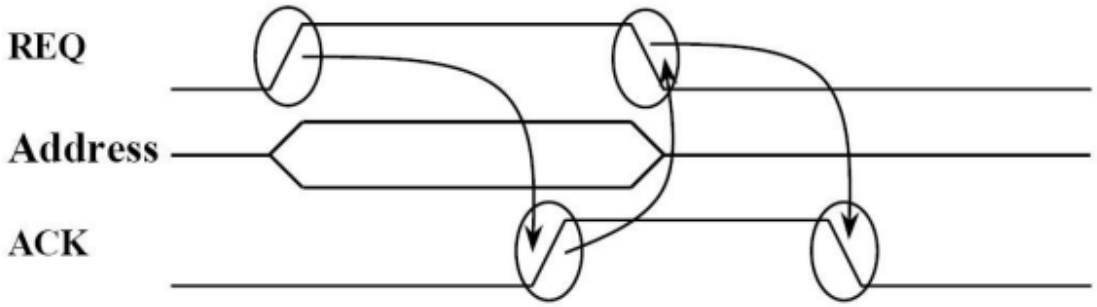

Fig. 3.26. Protocolo handshake AER punto a punto. Extraído de [17].

Por otra parte, en cuanto al desarrollo del sistema de control, es decir del arbitrador, se implementará la señal de inicialización de manera interna en el módulo emisor para que desencadene la generación de la señal “Request” y que envíe la dirección del evento generado, pero sin la capacidad para gestionar la redundancia entre los pulsos, basándonos en que la información más relevante en un mensaje es aquella que se repite menos, y por lo tanto cuando existan pulsos que se presenten en los mismos instantes de tiempo, pero en una región espacial diferente, estos sean eliminados. Esto tiene la finalidad de reducir el tiempo de cómputo, reducir la cantidad de recursos usados, simplificar el control y evitar modificar la dinámica neuronal, ya que si fuese de otro modo, los pulsos pre sinápticos repetidos espacialmente necesitarían almacenarse en algún otro circuito de retención para gestionarlos, ya que solamente una dirección puede usar el bus de direcciones; en este sentido, la lógica de control podría afectar la información temporal de los pulsos que puedan estar ingresando en ese momento de manera directa o indirecta al sistema, modificando los procesos de sinapsis y STDP. De esta manera, a la salida tendremos la dirección “FF” cuando no hay pulsos en la entrada o bien cuando hubo repeticiones, así, a la salida de AER, lo único que obtendremos son las direcciones de los pulsos de las neuronas que proporcionen la mayor cantidad de información relevante.

En este caso, el sistema tiene 2 ventajas:

- AER permite la integración espacio-temporal que realiza la neurona post sináptica biológica; en este sentido, lo que hace AER es definir lo que observa la neurona post sináptica mediante la construcción de una sola neurona que exhibe el patrón de generación de pulsos de todas las neuronas que la constituyen.
- Dado que se integran todos los pulsos no repetidos y se eliminan los redundantes, la generación de pulsos por parte de las neuronas pre sinápticas para una red de bajas dimensiones da tiempo suficiente para poder procesarlos uno a uno de manera secuencial por otro módulo, permitiendo que la información de entrada sea tratada como si fuese paralela.

El módulo AER que implementamos es un módulo de jerarquía superior que está integrado por los módulos etiquetados como “Sender” y “Receiver”, el cual se encuentra embebido en el módulo etiquetado como “AER_System” y su operación puede simularse en el banco de pruebas etiquetado como “AER_System_tb”. En cuanto a sus puertos de entrada, estos son etiquetados como Input_Neurons (0:35), clk y rst, mientras que su único puerto de salida es etiquetado como out_AER (7:0). Cabe mencionar que, para observar el protocolo de AER descrito anteriormente, es necesario invocar las variables "signal_initiation", "signal_Req", "signal_data_bus" y "signal_handshake"” que se encuentran internamente en dicho módulo.

En cuanto a la operación del módulo “AER_System”, éste tiene capacidad para concentrar los pulsos de 255 neuronas de entrada y direccionarlas, es decir, 8 bits para asignar direcciones (0-254), manejando una dirección fuera de rango (255) para designar un estado de alta impedancia como se ha mencionado anteriormente; esto se logra realizando la conexión de las neuronas hacia el puerto de entrada “Input_Neruons”, el cual, ya tiene internamente asignada una dirección para cada conexión. Por otra parte, dado que las neuronas pre sinápticas trabajan de manera paralela, una vez que se generen lo pulsos por

parte de ellas, será el protocolo de handshake punto a punto implementado en el módulo “AER_System”, el que se encargará de administrar y enviar las direcciones, tal como se muestra en la figura (3.27).

Obsérvese que el protocolo “handshake” corresponde al descrito anteriormente y solo cuando hay un pulso generado, éste envía la dirección enviándola fuera del módulo a través del puerto etiquetado como “out _AER” y cuando existe redundancia (como cuando se presenta un pulso generado en las direcciones 1, 5 y 7). En este caso, la salida del módulo AER continúa en alta impedancia, es decir, en una dirección que hemos designado como fuera de rango.

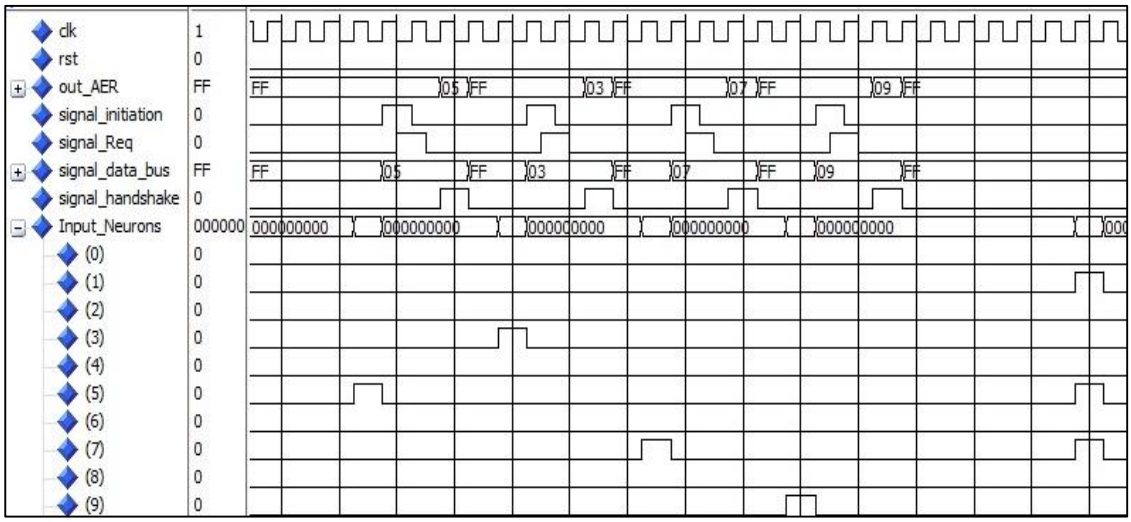

Fig. 3.27. Protocolo de Handshake punto a punto implementado en el módulo “AER_System”.

Finalmente podemos observar que la dirección que sale del módulo “AER_System” tiene una duración de solo un ciclo de reloj, duración que corresponde con el pulso generado en las neuronas pre sinápticas.

3.5.2.3 Módulo de propagación y retro propagación.

El módulo etiquetado como “FIFO_all_in_one” es un módulo de jerarquía superior, el cual se encuentra formado por los módulos etiquetados como “FIFO_pre_synap” y “delay_AER_STSP”, y cumple con el objetivo de concentrar los trenes de pulsos generados por todas las neuronas pre sinápticas y post sinápticas del sistema, incluyendo sus respectivos retardos

de propagación y de retro propagación en un solo módulo, a manera de evitar la dispersión de estas secuencias y tener una mejor apreciación de las mismas para su control.

3.5.2.3.1 FIFO: Desfase de tiempo pre sináptico

FIFO es un tipo de buffer que se usa a menudo en circuitos electrónicos como componente aconsejable en sistemas que trabajan a diferente velocidad o con cierta irregularidad, sirviendo como medio de almacenamiento durante periodos de tiempo con el objetivo de incrementar o reducir el tiempo de tránsito de la información entre componentes y con ello compensar la diferencia de velocidad. FIFO, por sus siglas en inglés “First In First Out”, hace referencia a un sistema de “primeras entradas primeras salidas”, que en cuanto a su implementación, la arquitectura de estos sistemas se basan en pilas de memoria dedicadas y sistemas de control para la lectura /escritura de los registros, los cuales pueden ser implementados vía software o hardware.

Por otra parte, en cuanto a su operación, existen variaciones de este buffer, como LIFO, que por sus siglas en inglés “Last In First Out”, hace referencia a un sistema de “últimas entradas primeras salidas” opuesto al sistema FIFO, pero que se basa en el mismo sistema de memoria donde solo el sistema de control de lectura/escritura varía y por ende la selección entre uno u otro dependerá de las necesidades de la aplicación. [24]

Dado que los circuitos FIFO normalmente están construidos sobre arreglos de memoria RAM y sistemas de control de lectura/escritura en tiempo real, para la implementación de nuestro módulo nos basaremos en sistemas FIFO que usan registros de corrimiento, “shift register’s”, los cuales hacen referencia a la forma en que se accede a los datos almacenados, en este caso, haciendo desplazamientos entre registros o rotaciones. En este tipo de sistemas necesariamente debe existir sincronía entre las operaciones de lectura/escritura de memoria para que los datos que están sobre el módulo puedan ser leídos /escritos al mismo tiempo, es decir, sobre un mismo ciclo de reloj. [24]

Por esta razón, con la finalidad de simplificar la implementación de un sistema FIFO que introduzca en su control desplazamientos para la lectura/escritura operando en un solo ciclo de reloj, usaremos un sistema FIFO síncrono donde la arquitectura del mismo, se basa sobre un módulo de memoria de RAM y una lógica de control en “anillo” tal como se describe en el trabajo de Zhang, Yi, Wang y Jinye Zhang en [25], para tener acceso a la información de la memoria.

La operación de la lógica de control en anillo, consiste en que en cada ciclo de reloj se realiza la lectura de la dirección de memoria y posteriormente la escritura de la misma, todo esto controlado por un contador ascendente que apunta sobre una dirección de memoria cada vez que concluye un ciclo de reloj. De esta forma, en cada ciclo de reloj, el proceso se realiza una y otra vez pero sobre direcciones de memoria diferentes, repitiéndose sucesivamente el proceso hasta que el contador supera la cantidad de direcciones que posee el módulo de memoria RAM, reiniciando el contador y posicionándose en la primera dirección de memoria del programa nuevamente y repitiendo el proceso mientras el sistema se encuentre activo, tal como se muestra en la figura (3.28).

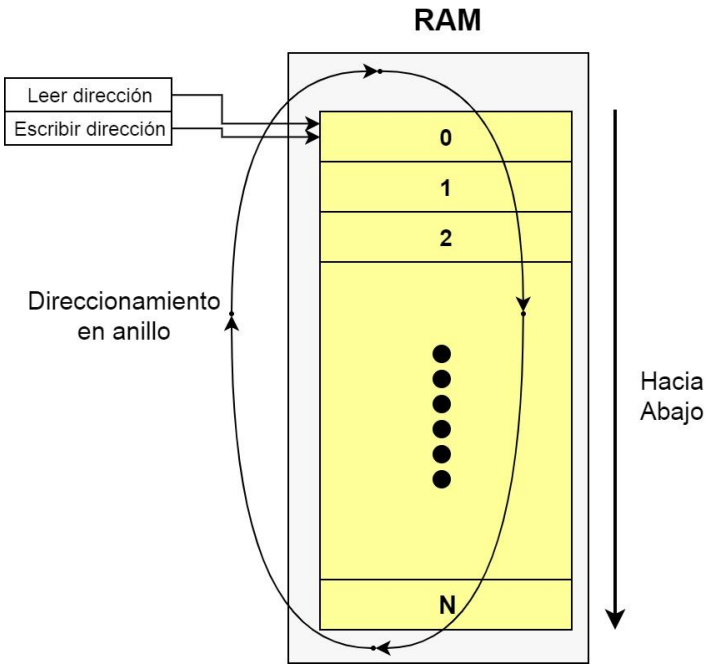

Fig. 3.28. FIFO con lógica de control en Anillo. Extraído de [25]

Como se observa, esta forma de control permite que siempre el primer elemento que entra sea el primero en salir. En nuestro caso, el módulo FIFO tendrá el objetivo de generar el tiempo necesario para realizar el posterior procesamiento de los pulsos en procesos de sinapsis y de aprendizaje STPD, reservando en memoria las direcciones de los eventos y preservando su información espacio-temporal, y con ello evitar agregar retardos adicionales por la propagación de los pulsos pre sinápticos (provenientes de "AER_System") que llegan a la neurona post sináptica.

Adicionalmente a ello, el sistema FIFO implementado, dado que será usado para formar parte de un sistema más grande y tendrá una fuerte relación principalmente con el módulo "AER_System", éste tendrá una salida adicional, la cual estará en alto cada vez que exista una dirección válida a la salida y, por otra parte, generando un desfase de tiempo de 52 ciclos de reloj en la salida, para que tenga lugar en ese lapso de tiempo, el procesamiento realizado por "STDP_Process".

El sistema FIFO descrito anteriormente, su objetivo principal es evitar agregar retardos pre sinápticos en operaciones subsecuentes y conservar la información espacio-temporal; dicho módulo se encuentra etiquetado como "FIFO_pre_synap" y su operación puede simularse en el banco de pruebas etiquetado como "FIFO_presynap_tb". En cuanto a sus puertos de entrada estos son etiquetados como "addr_FIFO_in (7:0)", "clk" y "rst", mientras que sus puertos de salida son etiquetados como "addr_FIFO_out" (7:0) y "pre_spike".

Tal implementación, puede observarse en la simulación de la figura (3.29), sin embargo, cabe mencionar que el módulo se describió como un sistema completo y no como un módulo de control y una memoria RAM separados para ser unidos posteriormente. Obsérvese, que el módulo genera un pulso cada vez que hay una dirección conocida, lo cual posibilita el procesamiento posterior de los pulsos generados ya que sabemos de cuál neurona provino y el instante de tiempo en que tuvo lugar.

Fig. 3.29.Operación del módulo "FIFO_pre_synap".

3.5.2.3.2 Flip Flop´s tipo D: retardos de retro propagación.

La finalidad del módulo tiene 2 objetivos: el primero es eliminar el desfase de tiempo generado por el módulo etiquetado como "FIFO_pre_synap" y adicionalmente insertar un retardo de retro propagación en los trenes de pulsos provenientes de las neuronas post sinápticas; de esta manera, el objetivo general es eliminar el tiempo requerido para llevar a cabo el proceso de STDP que ocurre en el módulo "STDP_Process" , y que este proceso se lleve a cabo de manera "instantánea", permitiendo que solo los retardos de propagación insertados por el módulo "Process_Data_Input" (descrito anteriormente en la sección (3.5.2.1.1)) y los retardos de retro-propagación agregados por este módulo, sean los que den forma al proceso de aprendizaje, lo cual hace referencia a que el sistema opera de manera ideal (aunque no hay prueba de que este proceso de manera biológica consuma un período de tiempo definido o variable).

Para realizar la implementación, basándonos en el enfoque propuesto en la implementación en software, es necesario tener una neurona post sináptica por cada imagen a reconocer, por lo que será necesaria la asignación de 10 retardos de retro propagación. Para lograr esto, dado que solo se requiere desplazar los pulsos generados por las neuronas post sinápticas, se puede configurar un sistema de flip flop´s en serie tipo D, donde el número de ellos está en relación directa con la magnitud del retardo que deseamos implementar y el cual se establece con el número de ciclos de reloj. En este caso, para eliminar el proceso de aprendizaje que dirige STDP, es necesaria la implementación de 52 de ellos en serie, mientras que para asignar el retardo de retro propagación, hemos designado un número aleatorio, el cual es 348. Basado en tal operación, la arquitectura del módulo consiste en la replicación

de esta cantidad de flip flop´s en serie y posteriormente replicándolos 10 veces, de tal forma que el módulo generado reciba los pulsos provenientes de todas las neuronas post sinápticas, tomando en cuenta que el retardo de retro propagación es el mismo para todas ellas. En este sentido, los retardos de retro propagación aplicados se hacen en 348 ciclos de reloj de sistema, y dado que este representa numéricamente 0.0078125 ms, el retardo de retro propagación es de 2.718 ms, mientras que, en tiempo real, tomando como base de sistema una frecuencia de operación de 10 MHz, los retardos son de 34.8 μs.

El sistema generado es un módulo de jerarquía superior etiquetado como “Delay_AER_STDP”, el cual contiene la implementación de los flip flop´s en serie y que se encuentra etiquetado como “FFD_row”, a su vez, contiene la unidad del sistema y se encuentra etiquetado como “FlipFlopD”.

El sistema de retardos descrito anteriormente se encuentra etiquetado como “Delay_AER_STDP” y su operación puede simularse en el banco de pruebas etiquetado como “Delay_AER_STDP_tb”. En cuanto a sus puertos de entrada, estos son etiquetados como “pos_in_vector (0:9)”, “clk” y “rst”, mientras que sus puertos de salida son etiquetados como “pos_out_vector (0:9)”.

En cuanto a su operación, ésta puede verse ilustrada en la figura (3.30), donde a través del puerto etiquetado como “pos_in_vector”, se generan pulsos, los cuales terminan siendo desplazados 400 pulsos de reloj a la salida del módulo, a través del puerto etiquetado como “pos_out_vector”.

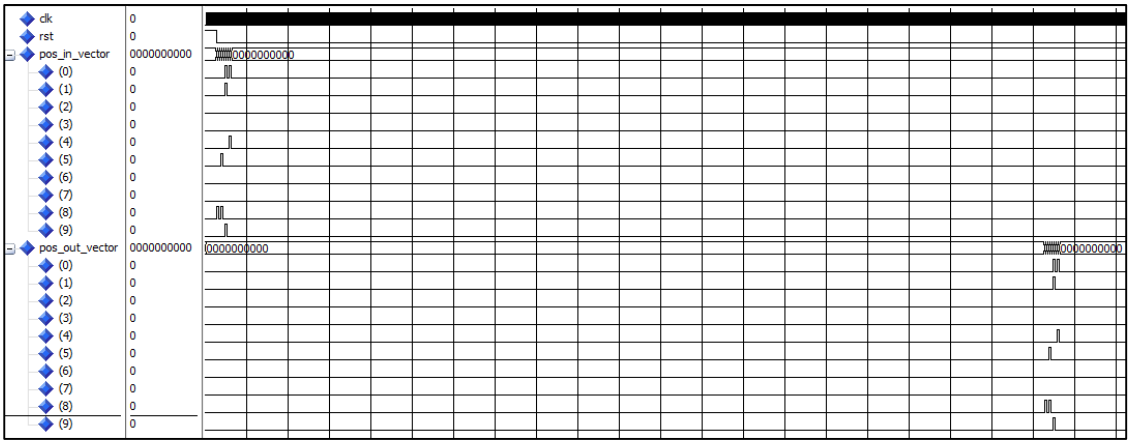

Fig. 3.30. Operación del módulo "Delay_AER_STDP".

3.5.3 Capa de salida

La capa de salida está formada formado por los módulos etiquetados como "STDP_Process" y "Neurona_iz_24_b", este último ya ha sido descrito en la sección (3.5.2.1.2) de este capítulo. En este caso, el módulo que se ha de describir es el módulo que posibilita el aprendizaje y reconocimiento de algún patrón por la red: el módulo "STDP_Process".

3.5.3.1 Módulo de aprendizaje y reconocimiento localizado.

La parte medular del sistema recae en el módulo etiquetado como "STDP_Process", el cual es un módulo de jerarquía superior que se encuentra formado por 6 módulos etiquetados como "Modulo_STDP", "Memory_RAM", "LFSR", "Relaxion_Rule", "Address_Low_Current", "Synapses", "Machine_State" y "Main_control". En su conjunto, permiten la generación de los 360 pesos sinápticos en la red, su modificación a través del proceso de aprendizaje dirigido por STDP y su retención para llevar a cabo el reconocimiento de algún patrón. Por otra parte, es necesario saber que la arquitectura del sistema está basada en eventos, es decir, en todas sus operaciones necesitará tanto de una dirección como de un valor asociado a la operación que realiza. En este caso, un sistema compatible que pueda operar de manera conjunta con el módulo "AER_system" visto en la sección (3.5.2.2), para procesar la información espacio-temporalmente y en tiempo real. En la figura (3.31) se ilustra el diagrama a bloques de los módulos que lo conforman:

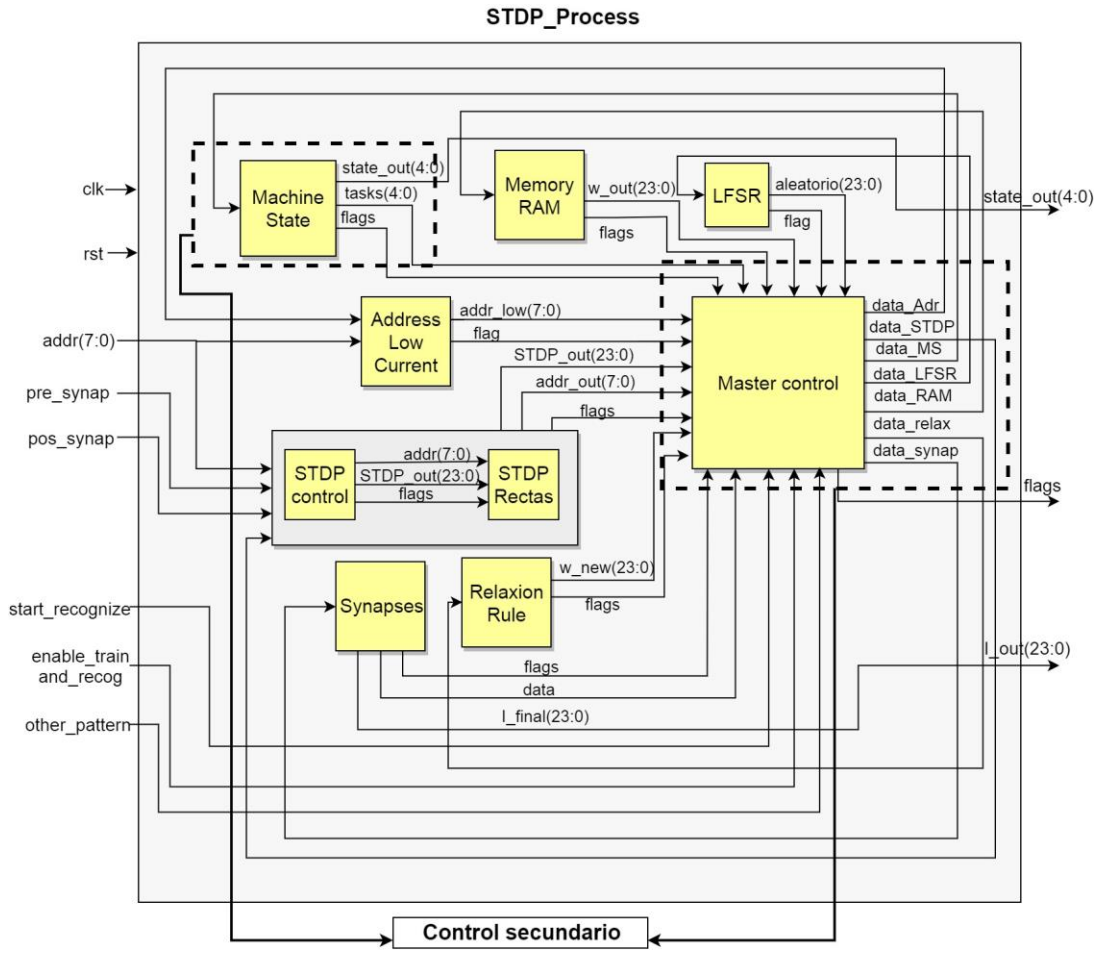

Fig. 3.31.Diagrama a bloques del módulo "STDP_Process".

3.5.3.1.1 Control Secundario

El control secundario se encuentra formado por los módulos "Main_Control" y "Machine_State", los cuales, de manera conjunta son el sistema de control del módulo "STDP_Process" como se describe la sección (3.2). En este caso, "Master_Control", concentra y administra las banderas de los múltiples módulos periféricos y permite la comunicación con el módulo de control primario, mientras que "Machine_State" permite las transiciones entre estados. De manera conjunta, se implementan 16 estados con tareas, de las cuales se pueden distinguir 5 etapas, las cuales son dirigidas por el "control primario" descrito en la sección (3.5.1.1.1):

- **Inicialización de sistema**: Se inicializa el "control secundario" a partir de la comunicación que realiza con el control primario.

- **Generación y asignación de pesos sinápticos**: Se realiza la generación de números aleatorios de la red así como su almacenamiento en la memoria RAM del módulo.
- **Integración de baja corriente**: Se genera un proceso de sinapsis previo al entrenamiento, con la finalidad de generar suficiente corriente en la neurona post sináptica vinculada al módulo “STDP_Process”, estimulando la generación de pulsos y por lo tanto que sea posible realizar un proceso de aprendizaje posteriormente.
- **Proceso de aprendizaje:** Se lleva a cabo el proceso de aprendizaje dirigido de STDP, mediante la comparación de los pulsos pre sinápticos y post sinápticos provenientes del módulo “Entry_process”, siendo los pulsos post sinápticos previamente generados desde el módulo “Neurona_iz_24_b”, que al mismo tiempo se encuentra asociado al módulo “STDP_Process”. En esta sección se realiza la modificación de los pesos sinápticos mediante la acción conjunta del protocolo STDP y la sinapsis.
- **Proceso de reconocimiento**: Mediante la acción conjunta del control primario, se lleva a cabo la paralelización del proceso de integración sináptica en los módulos “STDP_Process”. De esta forma, en las neuronas post sinápticas se ejecuta de manera individual la sinapsis con la finalidad de que el “control primario” lleve a cabo una etapa de decodificación que se basa en el método “primero en disparar” (véase capítulo 1 sección (1.6.2)) y con ello determinar qué neurona post sináptica finalmente pudo reconocer alguna imagen.

La máquina de estados y las tareas embebidas en “Machine_State” y “Master_Control” que conforma el “control secundario” se describen a continuación:

1. **S0:** Se verifica que “Master_Control” se encuentre inhabilitado; si éste se encuentra en ese estado y hay confirmación desde el “control

primario”, se solicita el siguiente estado, caso contrario se permanece en el mismo.

2. **S1**: Se habilita la operación del módulo “Master_Control” desde el “control primario”; si se tuvo éxito, se solicita el estado siguiente, caso contrario, permanece en el mismo.
3. **S2**: Se inicializan todos los módulos que integran al módulo “STDP_Process”; una vez realizada esta operación, se solicita el estado siguiente, caso contrario, permanece en el mismo estado.
4. **S3**: Se habilita y verifica que el módulo “LFSR” se encuentre en operación. En este estado “Master_Control” recibe un número aleatorio el cual una vez confirmado, se solicita el siguiente estado, caso contrario se permanece en el mismo.
5. **S4**: Se habilita la operación del módulo “Memory_RAM” y se administra la escritura del número recibido en el estado anterior, para que éste sea escrito sobre una de sus localidades de memoria, dejando en estado inactivo “Memory_RAM”. Una vez terminado el proceso, se solicita el siguiente estado, caso contrario se permanece en el mismo.
6. **S5**: Se evalúa si las 36 localidades de memoria están “llenas”, si es el caso, se solicita el siguiente estado y se apaga el módulo LFSR, caso contrario, se solicita el estado **“S3”.**
7. **S6**: Se habilita la operación de los módulos “Address_Low_Current” y “Synapses”, colocando a este último en la fase de integración de baja corriente. Una vez terminado el proceso, se solicita el siguiente estado, caso contrario se permanece en el mismo.
8. **S7**: Se lleva a cabo la sinapsis, lo cual se logra vinculando el módulo “Memory_RAM”, “Synapses” y “Address_Low_Current” , los cuales, en conjunto llevan a cabo 6 procesos:

- El módulo “Address_Low_Current” recibe por cada pulso pre sináptico una dirección asociada, la cual envía a “Master_Control”
- El módulo “Master_Control” obtiene el valor del peso sináptico de esta dirección mediante la lectura del módulo Memoria RAM,

- El módulo “Master_Control” envía dirección y peso sináptico asociado al pulso pre sináptico al módulo “Synapses”.
- El módulo “Synapses” realiza la integración de los pesos sinápticos en tiempo, generando una corriente de salida.
- El módulo “Master_Control” verifica que el nivel de corriente generada esté por encima de un umbral definido en la lógica interna del módulo “Synapses”, si éste es el caso, se apaga el módulo “Address_Low_Current” y se solicita el siguiente estado, caso contrario, éste permanece en este mismo estado, repitiendo los pasos anteriormente descritos.

9. **S8:** Se inicia el proceso de entrenamiento, lo cual se logra mediante la acción conjunta en este estado, de los módulos “Memory_RAM”, “Synapses” y el “Modulo_STDP”, en este sentido, la operación puede describirse en 3 partes, colocando previamente antes de comenzar al módulo “Synapses” en fase de aprendizaje:

- El módulo “Master_Control” se mantiene a la espera de que aparezca un cambio en el peso sináptico (Δw) al igual que una dirección asociada a éste. Si los hay, éste se encarga de retener estos valores para ser usados posteriormente.
- El módulo “Master_Control” realiza la lectura de la dirección obtenida en el paso anterior, para acceder al valor del peso sináptico actual, que se encuentra en una localidad de memoria del módulo “Memory_RAM”, reteniendo su valor para usarlo posteriormente como la variable de “w_old”.
- Si el módulo realiza con éxito los pasos anteriores, el módulo solicita el estado siguiente, en caso contrario permanece en el mismo estado.

10. **S9:** Se realiza la modificación del peso sináptico para obtener el nuevo peso sináptico mediante la operación del módulo “Relaxion_Rule”, lo cual se logra mediante 4 procedimientos.

- El módulo “Master_Control” envía valores del cambio de peso sináptico (Δw) y la variable “w_old” (obtenida en el estado “s8”) a “Relaxion_Rule” para su procesamiento.
- El módulo “Master_Control” recibe el peso sináptico resultante del módulo “Relaxion_Rule”.
- El módulo “Master_Control” habilita la escritura de la dirección de la localidad de memoria con el peso sináptico recibido y sobrescribe en esa dirección el nuevo valor del peso sináptico.
- Si se realizaron con éxito todos los procedimientos antes descritos, se solicita el estado siguiente; caso contrario, permanece en el mismo estado.

11. **S10**: Se lleva a cabo la sinapsis durante el proceso de entrenamiento a través del módulo “Master_Control”, el cual se logra enviando el peso sináptico resultante del estado “S9” y su respectiva dirección hacia el módulo “Synapses”. Si el proceso se lleva a cabo correctamente, se solicita el siguiente estado, caso contrario, se permanece en el mismo.
12. **S11:** Este proceso lo lleva a cabo únicamente “Master_Control” y sirve como medio de verificación y retroalimentación para el sistema. Este proceso se encarga de llevar a cabo un conteo de la cantidad de pares de pulsos que han tenido lugar en la región de potenciación (LTD), por lo que si este conteo ha superado el umbral de 100 pares de pulsos potenciados, éste solicita el estado **“S12”**, caso contrario, se solicita el estado **“S8”,** repitiendo todo el proceso de aprendizaje.
13. **S12:** Se concluye la etapa de entrenamiento. En este caso, “Master_Control” envía una petición a “Synapses” para que éste entre en fase de reconocimiento. Una vez verificada esta acción se solicita el estado siguiente, caso contrario se permanece en el mismo.
14. **S13:** El sistema de control secundario se encuentra a la espera de la iniciación de la etapa de reconocimiento por parte del sistema del “control primario”, si éste recibe señalización, se solicita el estado siguiente, caso contrario, permanece en el mismo.

15. **S14:** Se inicia la etapa de reconocimiento, realizando nuevamente la sinapsis, para lo que hace uso nuevamente de los módulos “Memory_RAM”, “Synapses” y “Address_Low_Current”, el cual realiza de manera conjunta 4 procesos.

- Cada vez que se presenta un pulso pre sináptico, “Address_Low_Current” envía la dirección asociada a “Master_Control”.
- “Master_Control” usa la dirección asociada del pulso para leer al valor del peso sináptico obtenido del proceso de aprendizaje desde el módulo “Memory_RAM”.
- Una vez obtenido el peso sináptico referente a esa dirección, éste se envía junto con la dirección hacia el módulo “Synapses”.
- Una vez realizadas las acciones descritas anteriormente, “Master_Control” se vale del “control primario” para saber si es necesario mantenerse en este estado. Si no existe una solicitud de terminación, entonces repiten los procesos descritos anteriores, caso contrario solicita el estado siguiente.

16. **S15:** El control secundario verifica a través del “control primario” si es necesario analizar otra secuencia de pulsos, si es así, entonces solicita el estado **“S12”** repitiendo todo el proceso de reconocimiento, caso contrario, permanece en el mismo estado.

Cabe mencionar, que durante todo el proceso, “Master_Control” como parte del sistema de “control secundario”, se encarga de realizar el correcto funcionamiento de los bloques evitando tomar valores intermedios entre los diferentes estados y diferentes módulos. La operación del “control secundario” puede observarse en la figura (3.32).

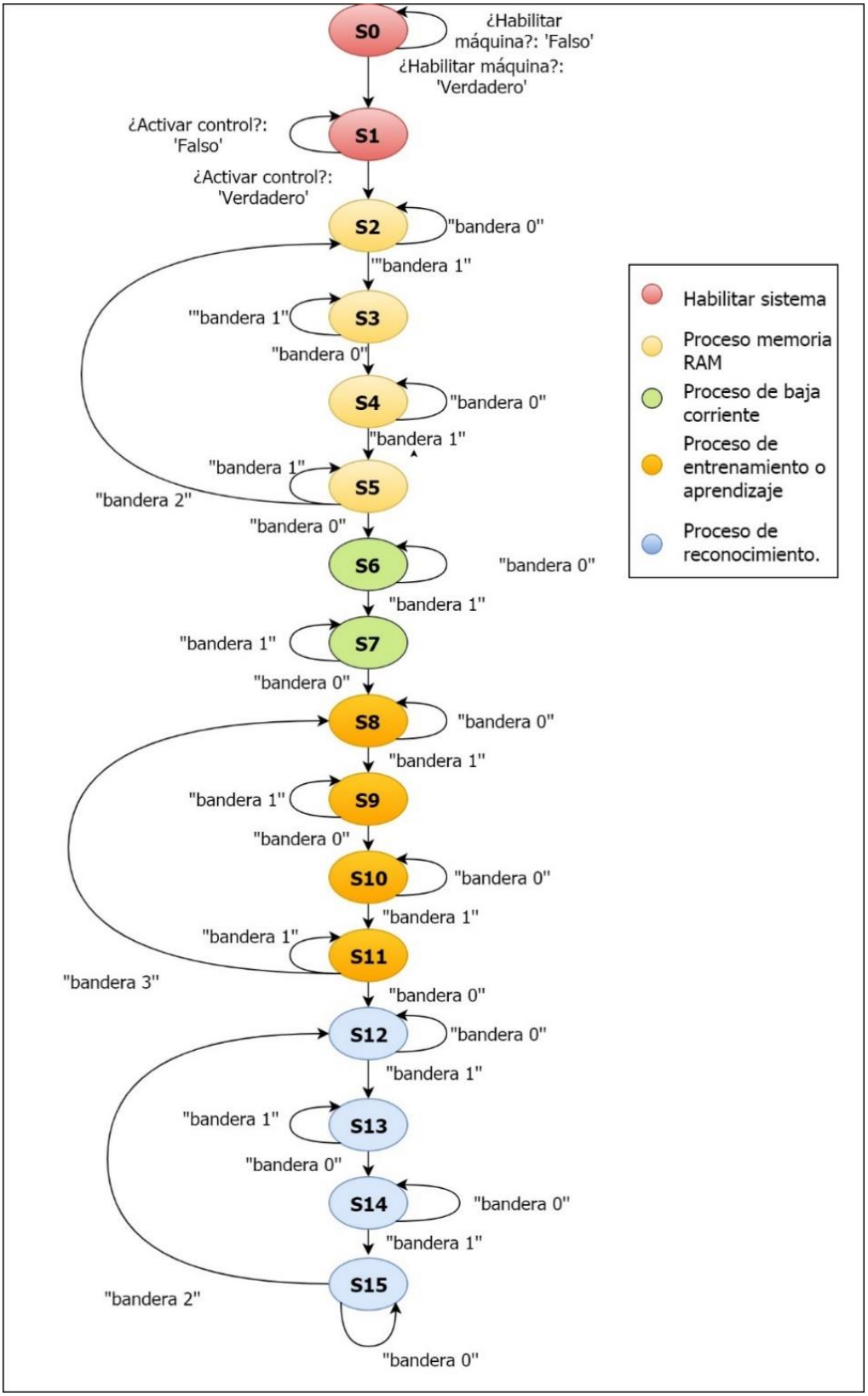

Fig. 3.32. Diagrama de estados que implementa "Master_Control" y "Machine_State" como Control Secundario.

La operación de la máquina de estados que se lleva a cabo en el “control secundario” puede simularse en el banco de pruebas etiquetado como “Machine_State_tb”. En cuanto a los puertos de entrada para la simulación mostrada en la figuras (3.33) y (3.34), estos son etiquetados como “general_flag_m (4:0)”,”clk” y “rst”, mientras que sus puertos de salida son etiquetados como “state_module (5:0)” y “task_m(5:0)”. Sin embargo, es necesario mencionar que “next_state” y “signal_count” son variables, la primera interna y la segunda del banco de pruebas. En este caso “next_state” almacena el valor de los estados y “count” permite la visualización de los estados en ambas figuras. Adicionalmente a ello, cabe mencionar que existen puertos de entrada y salida adicionales que sirven para interactuar con el “control primario”, que por fines prácticos, han sido omitidos aquí.

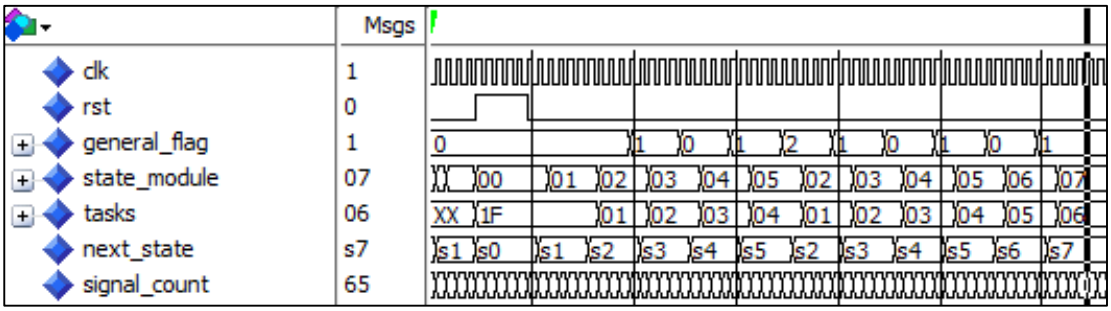

Fig. 3.33.Operación de la máquina de estados en “control secundario”. Estados correspondientes a las etapas de “Habilitar sistema”, “Proceso de memoria RAM” y “Proceso de baja corriente”.

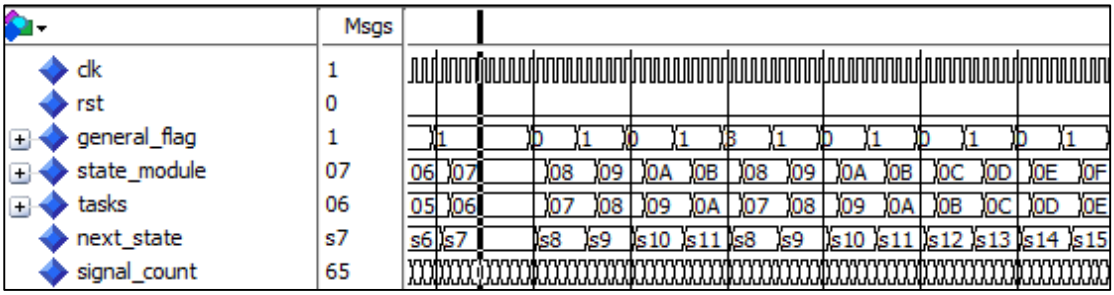

Fig. 3.34. Operación de la máquina de estados en “control secundario”. Estados correspondientes a las etapas de “Proceso de entrenamiento o aprendizaje” y “Proceso de reconocimiento”.

3.5.3.1.2 Protocolo de STDP

Como hemos visto en la sección (2.4), para la implementación de STDP en software, es necesaria la selección de un protocolo de asociación de pulsos y una ventana de aprendizaje [26]; en este caso, ambas tareas son realizadas

por el módulo jerárquico superior etiquetado como “Modulo_STDP”, el cual integra a su vez los módulos “STDP_rectas” y “STDP_control”.

En cuanto a su operación, ésta puede ser descrita de manera secuencial aunque estos trabajen paralelamente, lo cual consiste en lo siguiente: cuando “STDP_control” genera una asociación de pares de pulsos, éste envía la dirección del evento así como la diferencia temporal (Δt), al módulo “STDP_rectas”, el cual, si detecta una dirección conocida y una diferencia temporal definida, calcula el cambio del peso sináptico (Δw) asociado al evento.

El “Modulo_STDP” es la parte medular del algoritmo de aprendizaje, ya que es el que se encarga precisamente de generar los pares de pulsos pre y post sinápticos, provenientes del módulo “Entry_process” y con ello determinar la magnitud del cambio en el peso sináptico. Para el desarrollo de dicho módulo nos hemos basado en el trabajo de Jin, Rast, Galluppi, Davies, y Furber en [26], de Frenkel, Indiveri, Legat, y Bolen en [27] y de Belhadj, Malot, N’Kaoua, Bornat y Renaud en [28], que son implementaciones del sistema STDP “neuromórfico”; sin embargo, dado que las reglas de asociación de pulsos no son suficientemente explicitas, se optó por la propuesta de un módulo de asociación de pulsos aislados y de otro propiamente como la ventana de aprendizaje STDP.

El funcionamiento del módulo se puede simular a través del banco de pruebas etiquetado como “Modulo_STDP_simple_tb”. En este caso, solo se muestra uno de los casos de asociación en la figura (3.35). En dicha simulación, los puertos de entrada son etiquetados como “clk”, “rst”, “addr_in(7:0)”, “pre_synap” y “pos_synap”, mientras que los puertos de salida son etiquetados como “STDP_out (23:0)” y “addr_out (7:0)”. Cabe mencionar que las señales etiquetadas como “check_result” y “signal_conteo_entre_spike” provienen del banco de pruebas, siendo el primero usado como un conversor de formato binario a decimal del puerto “STDP_out” y así visualizar el valor del cambio

sináptico (Δw);el segundo es un contador de los pulsos entre los pulso pre y post sinápticos.

Adicionalmente, cabe mencionar que existen puertos de entrada y salida adicionales que sirven para interactuar con el "control secundario" que por fines prácticos, han sido omitidos aquí. Durante la simulación, obsérvese que solo el pulso pre sináptico que se ha "emparejado" con un pulso post sináptico termina siendo procesado y por lo tanto contribuyendo a la plasticidad sináptica del sistema.

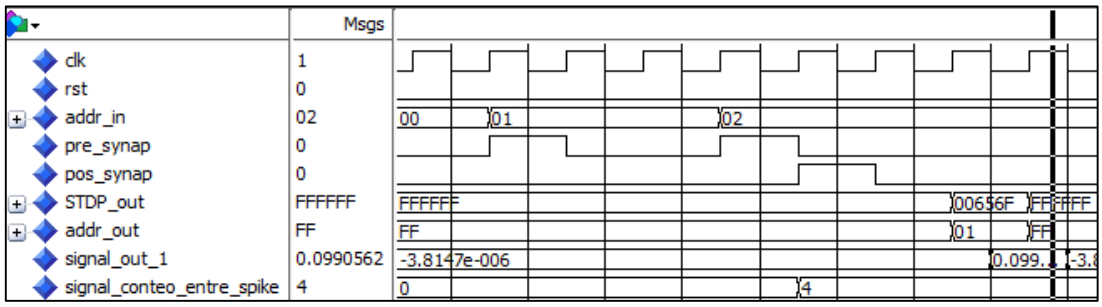

Fig. 3.35. Operación de "Modulo_STDP".

3.5.3.1.2.1 Protocolo de asociación.

Para el correcto funcionamiento del sistema de inducción de aprendizaje, STDP necesita de un protocolo de asociación de pulsos como base del sistema. Sin embargo, como se ha mencionado en la sección (2.5) del capítulo 2, no está claro cómo STDP puede asociar diferentes pulsos en un sistema donde las secuencias de pulsos se presentan en tiempo real y "cómo aplicar las reglas de pulsos a trenes de pulsos más realistas que pares de pulsos pre – post sinápticos aislados, siendo objeto de gran interés en la investigación" [29]. Inclusive, los protocolos teóricos todavía difieren en cuanto a la forma en cómo se pueden agrupar los pulsos entre neuronas pre y post sinápticas, como se describe en el trabajo de Pfister y Gerstner en [30]. Sin embargo, para afrontar esta problemática, usaremos una variación de los enfoques usados por Standage y Trappenberg en [29] y por Davies, Rast, Galluppi y Furber en [31], para obtener un método que nos permita implementar el protocolo de

pares en FPGA y que sirva de medio para dirigir el aprendizaje en tiempo real (“online learning”).

Basados en [31], ha habido implementaciones de variantes de las reglas de asociación de STDP, donde se toman en cuenta series de tres y cuatro pulsos (triplet’s y cuadruplet’s), pero que al momento de implementarse resultan ser más complejas que la regla de asociación de pares estándar de STDP, esto debido a que se necesitan rastrear diferentes patrones a través de la secuencias de pulsos pre y post sinápticos.

El protocolo de pares conocido como el “vecino más cercano”, tiene como regla vincular solo aquellos pulsos que se encuentran más cercanos el uno del otro, para realizar la modificación del peso sináptico, eliminando aquellos que no intervienen. Lo anterior ha demostrado tener validez al menos al reproducir con cierto grado de plausibilidad el comportamiento biológico. Sin embargo, en contraste con lo que se presenta en [29], cuando se ha tratado de implementar el más “sencillo” de los protocolos, el de pares, se han presentado problemas al tratar de elucidar la forma de cómo asociarlos usando el método del “vecino más cercano”, ya que en ciertas secuencias de pares, éste queda indeterminado; en lugar de éste, se han propuesto esquemas basados en el “par más cercano”, donde solamente los pulsos pre sinápticos más cercanos a los post sinápticos serán aquellos que podrán ser vinculados. En la figura (3.36) se expone la forma en cómo estos dos enfoques abordan el mismo problema, sin embargo, cómo asociar todos los posibles casos de interacciones no se resuelve en ambos.

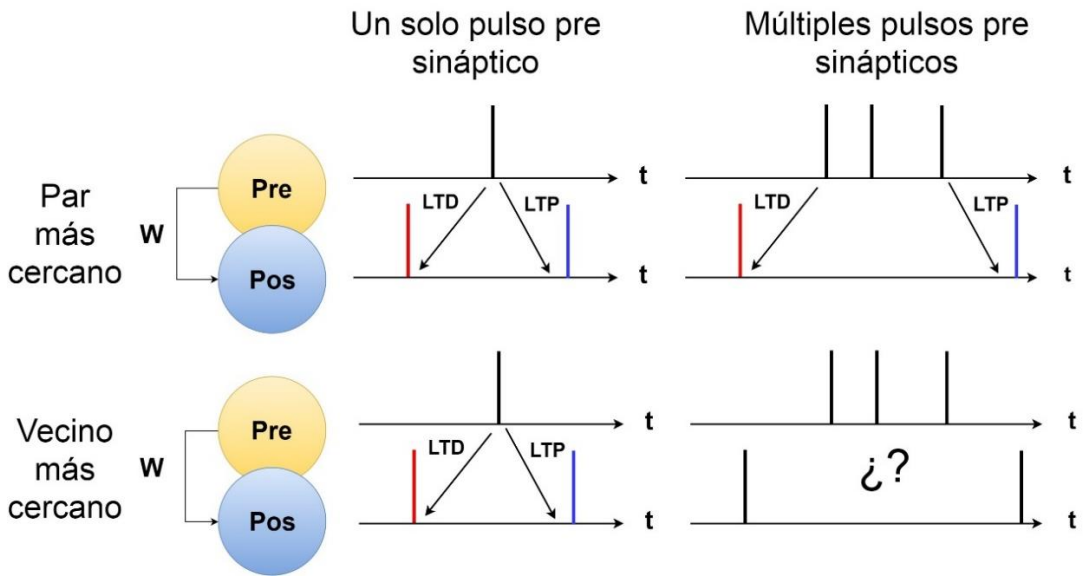

Fig. 3.36 Comparación entre sistemas de asociación de pulsos. Extraído de [29].Pulsos en rojo inducen depresión y los azules potenciacion sináptica.

Dado que en ambos enfoques se eliminan los pulsos que no cumplan dichas reglas de asociación para contribuir en la plasticidad sináptica, en nuestro caso, proponemos un sistema de control que nombramos como el "primero en disparar". El primer problema para implementar dicho sistema de control, es responder qué sucede cuando tenemos un pulso que está presente en el tiempo, ¿Cómo se podría saber en qué medida "está cerca en el tiempo" al interactuar con otro pulso? ¿Si elegimos un pulso dentro de una secuencia de pulsos, ya sea si proviene de una neurona pre o post sináptica, cómo encontraremos una pareja para él en la secuencia opuesta, al considerar que éste también se encuentra moviéndose en el tiempo? En nuestro caso, estas preguntas son respondidas a través de la regla de asociación propuesta que otorga nombre a dicho método, y el cual consiste en vincular el primer pulso pre o post sináptico que aparezca en la secuencia con su opuesto (si fuese pre sináptico buscar el post sináptico y viceversa), el cual también deberá ser el primero en disparar, usando como medida de más cercano la longitud de la ventana de tiempo, es decir, mediante la evaluación de la diferencia temporal (Δt) existente entre pulsos entre τ_- y τ_+ , siendo los pares formados los únicos que contribuyan a la plasticidad sináptica. Para determinar dicha diferencia temporal (Δt) se hará uso del esquema tradicional de STDP y por lo tanto se

implementará el sistema a través de la ecuación (1.23) para determinar la diferencia temporal (spike timing) y dirigir un aprendizaje causal.

El esquema de control propuesto, da respuesta en el sentido de generar la lógica de control de dos trenes de pulsos que cambian en el tiempo, dotando al sistema con la capacidad de formar pares de pulsos independientemente de quién aparezca primero, y evaluando si estos pueden contribuir o no con la plasticidad.

Dadas las características del sistema de control propuesto, se usará para implementar dicho sistema un contador ascendente y descendente unido a la lógica de control necesaria para formar pares de pulsos pre y post sinápticos, la cual cubre estando en operación, todos los posibles casos. La manera en cómo se puede realizar la interacción entre los pulsos se describe mediante una serie de reglas las cuales se presentan a continuación:

1. Si un pulso pre sináptico aparece antes que un pulso post sináptico, un contador comienza a contar en el sentido de las agujas del reloj hasta que aparece un pulso post sináptico, formando un par de pulsos pre-post y terminando el conteo. El contador se reinicia en valor cero y espera el siguiente pulso.

   Por otro lado, si aparece primero un pulso post sináptico, entonces un contador comienza a contar en el sentido contrario a las agujas del reloj hasta que aparece un pulso pre sináptico, formando un par de pulsos post – pre y terminando el conteo. El contador se reinicia en cero y espera un nuevo pulso.
2. Si aparece un pulso pre o post sináptico una vez que se ha iniciado el conteo, todos los pulsos provenientes de las neuronas pre o post sinápticas, dependiendo de quién haya sido la que haya disparado, el contador continuará en operación y los pulsos serán rechazados hasta que encuentre su par, es decir, si el que inició el contador fue un pulso pre sináptico, éste esperará un pulso post-sináptico y viceversa. Una

vez que se ha formado un par de pulsos pre–post o post-pre, el contador se reinicia en el valor cero y espera un nuevo pulso.

3. Si y solo si, no existen pulsos pre o post sinápticos que hayan disparado el contador, de tal forma que aparecieran un pulso pre y post sináptico de manera simultánea en el tiempo, la diferencia temporal entre ambos es cero. Para el caso en que el contador se haya activado para cualquier pulso pre o post sináptico, el sistema espera a que se encuentre su par, de modo que incluso cuando vengan dos pulsos pre o post sinápticos al mismo tiempo, la diferencia temporal entre ambos no será cero, ya que se tomará en cuenta el pulso que disparó el conteo.
4. En cualquiera de las reglas presentadas anteriormente, cuando el contador supera la ventana de aprendizaje (-20 ms a 20 ms), incluso si existe un conteo presente debido a cualquier pulso pre o post sináptico, el sistema de control se reiniciará, con lo que el contador volverá a cero y esperará un pulso pre o post que inicie el conteo nuevamente.
5. Si no existe ningún pulso pre o post sináptico que llegue al módulo, el conteo no existe, es decir, el sistema se mantiene en estado de espera o inactivo.

El sistema de reglas puede verse ilustrado en la figura (3.37), donde los pares formados en potenciación se encuentran en azul, los pares formados en depresión se encuentran en rojo, los pulsos en violeta son aquellos con diferencia temporal “0” y los pulsos en negro son aquellos que no contribuyen en ninguna medida a la plasticidad sináptica.

Por otro lado, tómese en cuenta que la base numérica del reloj del módulo es de 0.0078125 ms al igual que el módulo “Neurona_iz_24_b” (véase sección (3.5.2.1.2)), por lo que en cada incremento de reloj, éste sumará indefinidamente hasta que llegue al dominio máximo generado por la ventana de aprendizaje, es decir, la amplitud de la ventana de tiempo en pulsos de reloj es de 2560 para la región de potenciación y depresión, respectivamente, por lo que en tiempo real abre ventanas LTD y LTP de 256 μs cada una.

El sistema en general, solo recibirá trenes de pulsos pre sinápticos etiquetados y pulsos post sinápticos del módulo “Entry_Process” y su única función será la de obtener pares de pulsos para obtener la diferencia temporal asociada (Δt) así como la dirección de la sinapsis que afectará.

El módulo que contiene toda la lógica interna para realizar el procedimiento antes descrito se encuentra etiquetado como “STDP_control” y su funcionamiento puede verse ilustrado en las figuras (3.38)-(3.40) las cuales ilustran las primeras 3 reglas de asociación propuestas y que son generadas a través del banco de pruebas etiquetado como “STDP_control_tb”. Los puertos de entrada son etiquetados como “clk”, “rst”, “addr_in(7:0)”, “pre_synap” y “pos_synap”, mientras que los puertos de salida son etiquetados como “STDP_out (23:0)” y “addr_out (7:0)”. Cabe mencionar que en dicha simulación, “signal_out_1”es la representación decimal del puerto de salida STDP_out (23:0), “check_result” es la diferencia temporal entre pulsos pre y post sinápticos esperada y “signal_conteo_entre_spike” la cantidad de pulsos generados por el reloj de sistema entre pares de pulsos, todos generados en el banco de pruebas para mejorar la visualización de los procesos.

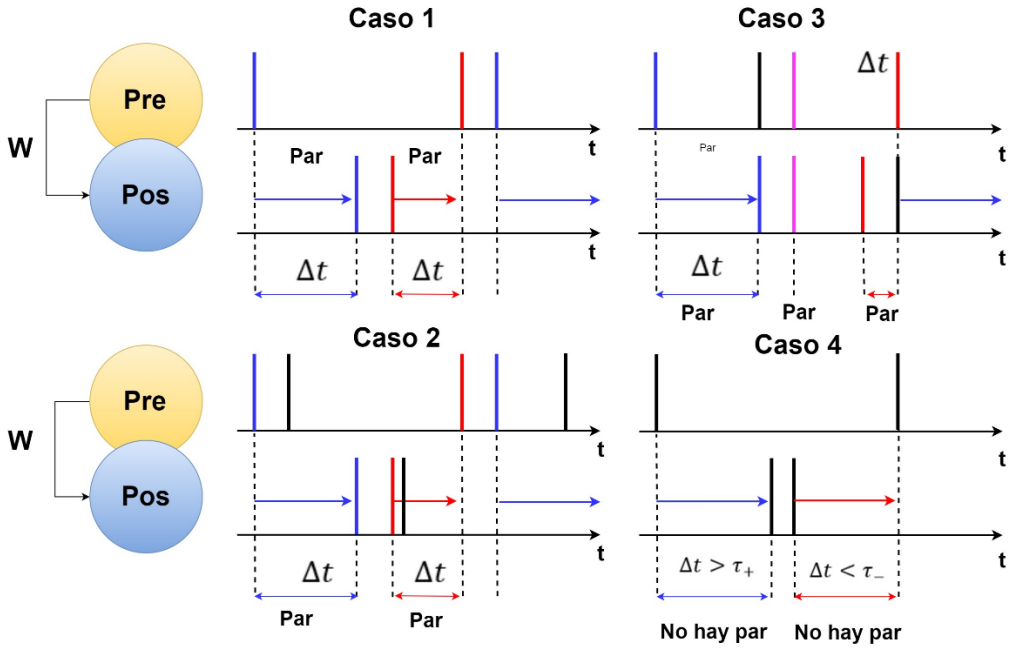

Fig. 3.37.Reglas de asociación de pares basado en el “primero en disparar”. Pulsos en rojo inducen depresión y los azules potenciacion sináptica

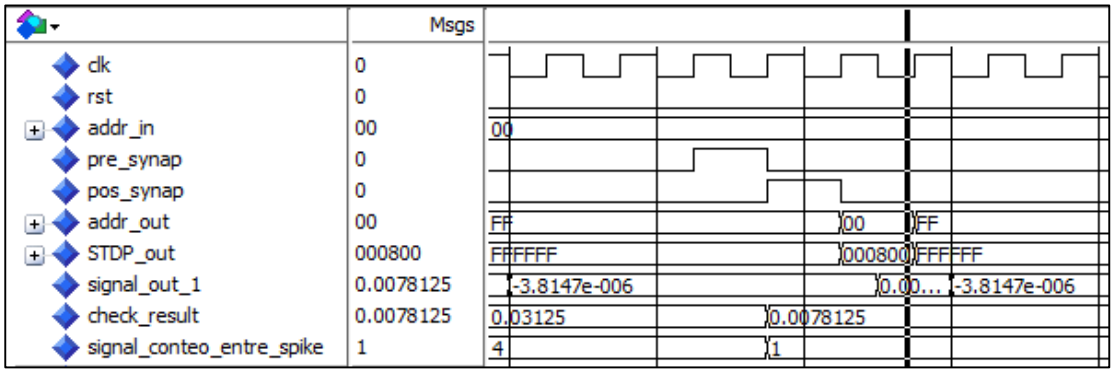

Fig. 3.38. Caso 1 de las reglas de asociación de pares propuesta.

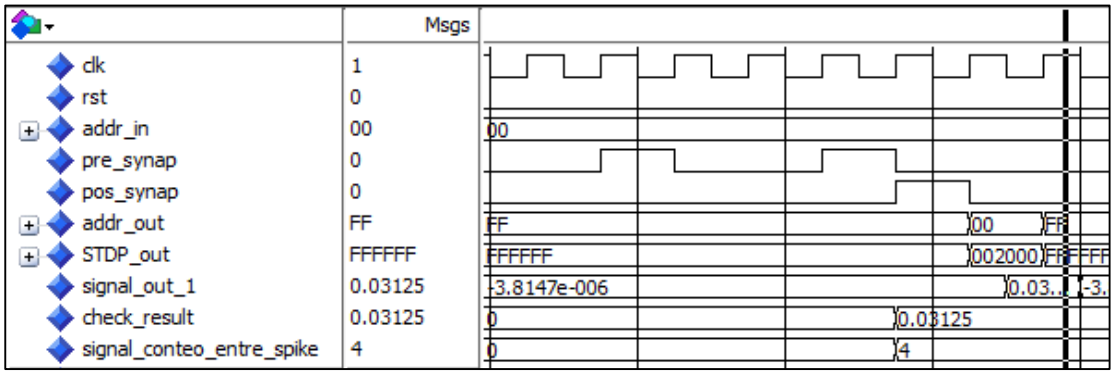

Fig. 3.39. Caso 2 de las reglas de asociación de pares propuesta.

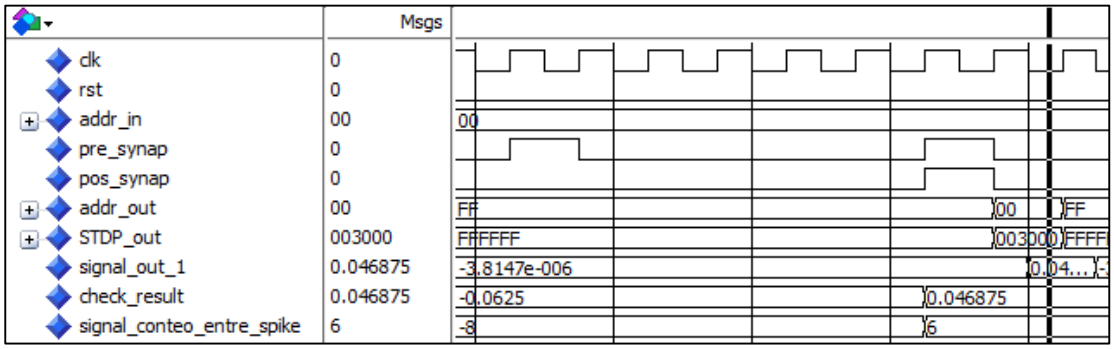

Fig. 3.40. Caso 3 de las reglas de asociación de pares propuesta.

3.5.3.1.2.2 Ventana de aprendizaje.

En este caso, para dirigir el aprendizaje se usará como ventana de aprendizaje la ecuación (1.28) usada en la sección (1.5) del capítulo 1, manteniendo los mismos valores para las constantes A_+, A_-, τ_+ y τ_+ que se usaron en software y que se encuentran en la tabla (2.2) del capítulo 2. Para obtener dicha

ecuación en hardware, nos basamos en el trabajo de Tornez, Gómez y Moreno en [32], donde se presenta un método para aproximarse a cualquier función a implementar en FPGA, y la cual consiste en colocar varias rectas tangentes sobre dicha función y así obtener una aproximación numérica al modelo, reduciendo el costo computacional (en este caso de DSP's). Sin embargo, cabe mencionar que existen 2 problemas a considerar, de los cuales, el primero es determinar la cantidad de líneas tangentes a colocar sobre la función, ya que si son muy pocas, estaremos lejos de obtener una buena aproximación y el segundo, es que la arquitectura basada en este enfoque no es fácilmente reprogramable, ya que cualquier variación sobre los parámetros de la función, cambiará la disposición así como el número de las rectas tangentes, aunque se trate de la misma función.

Por otra parte, el beneficio es que solo se necesitan dos constantes para modelar la ecuación de la recta tangente (m y b), donde m representa la pendiente de la recta y b representa la ordenada al origen; "Δt" representa la diferencia temporal entre pulsos y "Δw" el cambio sináptico, tal como como se muestra en la ecuación (3.1).

$$\Delta w = m(\Delta t) + b \qquad \textbf{(3.1)}$$

Para reproducir el valor numérico de la función exponencial de STDP, se colocaron 20 líneas tangentes sobre la función, para así obtener 40 constantes que representan las literales m y b de la ecuación (3.1) y así generar 10 pendientes para la sección LTP y LTD, respectivamente.

En cuanto a la operación, ésta puede ejemplificarse a través de módulo que aparece en la figura (3.41), el cual contiene en su lógica interna, las constantes de las pendientes que se aproximan a la función STDP; en este sentido, cada vez que entra una diferencia temporal al módulo, se evalúa esta diferencia temporal (Δt) y dependiendo de la posición dentro del eje, éste seleccionará las constantes correspondientes, para posteriormente realizar la operación

descrita por la ecuación (3.1), obteniendo el cambio del peso sináptico (Δw), así como su dirección.

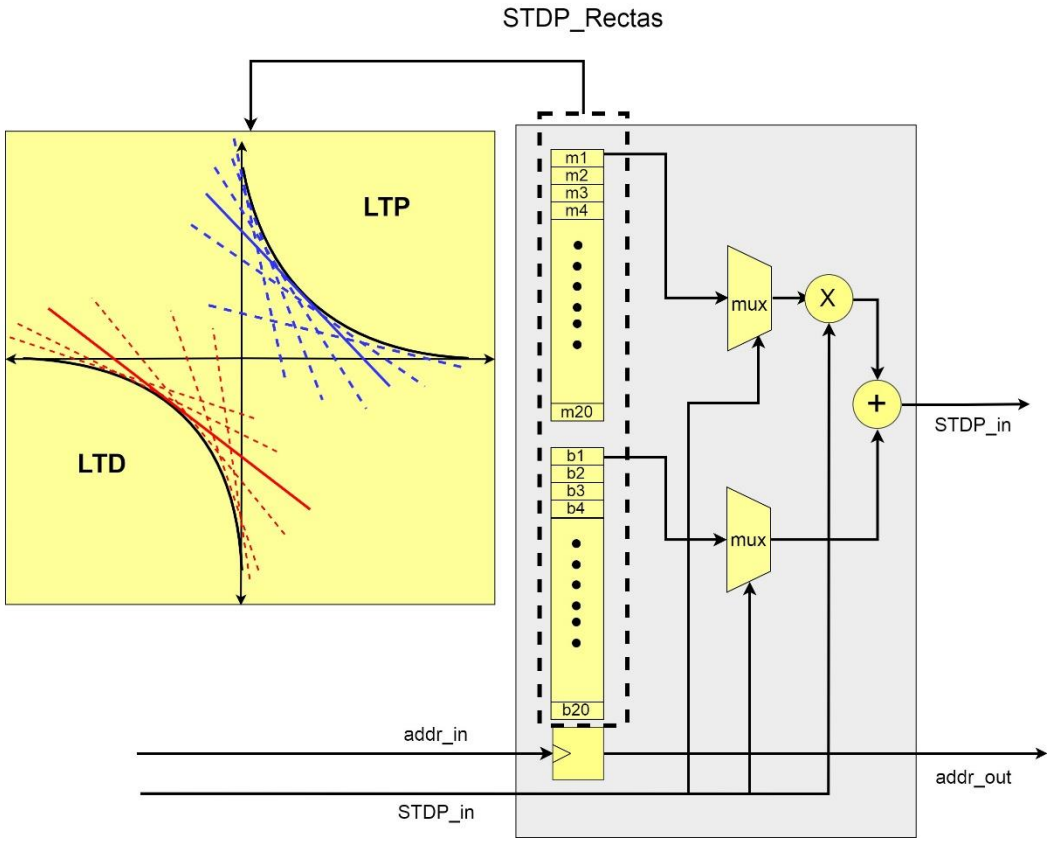

Fig. 3.41. Diagrama de operación del módulo "STDP_rectas".

La ventana de aprendizaje tradicional de STDP se encuentra en el módulo etiquetado como "STDP_rectas" y su operación puede simularse en el banco de pruebas etiquetado como "STDP_Rectas_tb". En cuanto a los puertos de entrada estos son etiquetados "addr_in (7:0)", "STDP_in (23:0)", "clk" y "rst", mientras que los puertos de salida son etiquetados como "addr_out (7:0)" y "STDP_out (23:0)".

Dicha operación puede verse ilustrada en la figura (3.42), donde se hace un barrido de la ventana de aprendizaje, de -20 ms a 20 ms a través de "STDP_in", para obtener la ventana doble exponencial de STDP a través de "STDP_out".

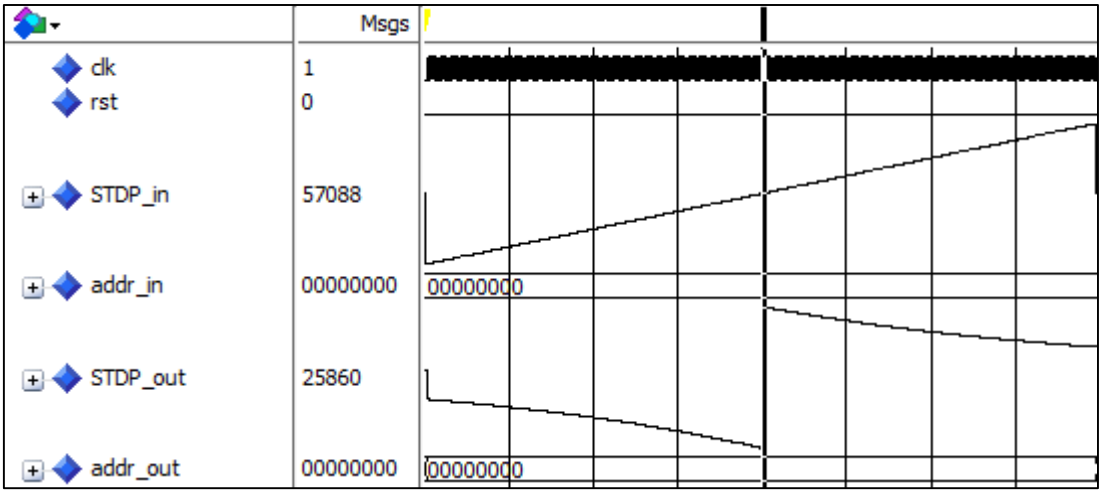

Fig. 3.42.Operacion del módulo "STDP_rectas".

El propósito general del módulo es proveer solo la dirección del evento y suministrar al sistema un cambio sináptico (Δw) a partir de una diferencia temporal (Δt). Por otro lado, es necesario mencionar que existen otro tipo de ventanas de STDP que requieren menos recursos en hardware. Sin embargo, son formas lineales o simplificadas del aprendizaje en STDP, como la que se aborda en el trabajo de Cassidy, Andreou y Georgiou en [33].

3.5.3.1.3 Memoria RAM.

El módulo de memoria RAM etiquetado como "Memoria_RAM" opera de la misma forma que el módulo de memoria de la sección (3.5.1.1.3). Sin embargo, el objetivo principal de este módulo es almacenar los pesos sinápticos asociados a cada neurona post sináptica, por lo que la cantidad de localidades máximas que tiene son 36 y la longitud del ancho de palabra que puede almacenar es 24 bits.

El funcionamiento puede verse ilustrado en las figuras (3.43) y (3.44). Esto es generado a través del banco de pruebas etiquetado como "Memoria_RAM_tb", donde los puertos de entrada son "addr_p (7:0)", "wr_en (1:0)", "wr_in (23:0)", "clk", "enable_RAM" y "rst", mientras que los puertos de salida son etiquetados "flag_wr (1:0)", "w_out(23:0)", "enable_flag_RAM" y "enable_w_r". Cabe

mencionar que “signal_count” es una señal del banco de pruebas para mejorar la visualización de los procesos mediante un contador; la variable “memory” es la memoria RAM en sí.

En la figura (3.43) se muestra la escritura de una imagen a través de “w_in” y “wr_en”, mientras que la figura (3.44) se muestra el caso opuesto, la lectura.

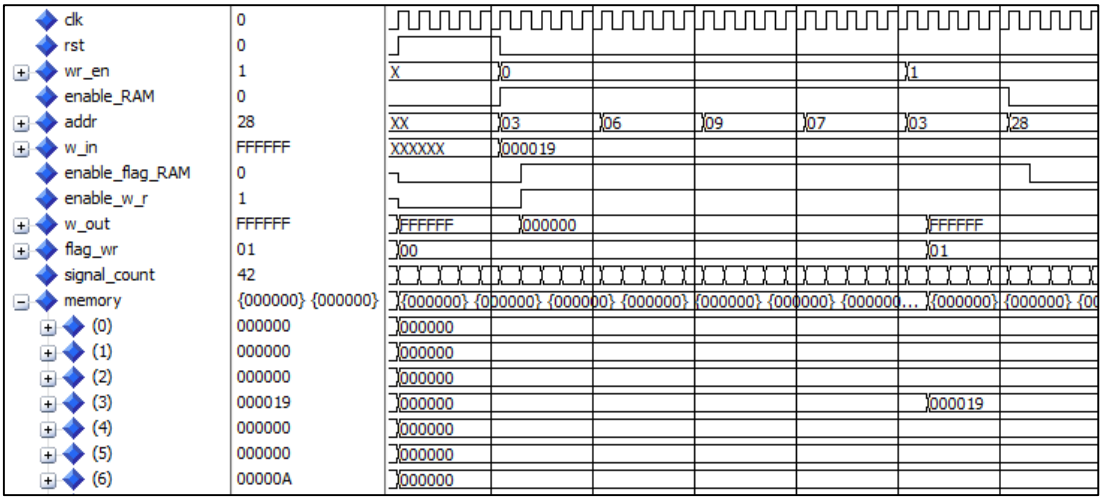

Fig. 3.43. Escritura del módulo “Memoria_RAM”.

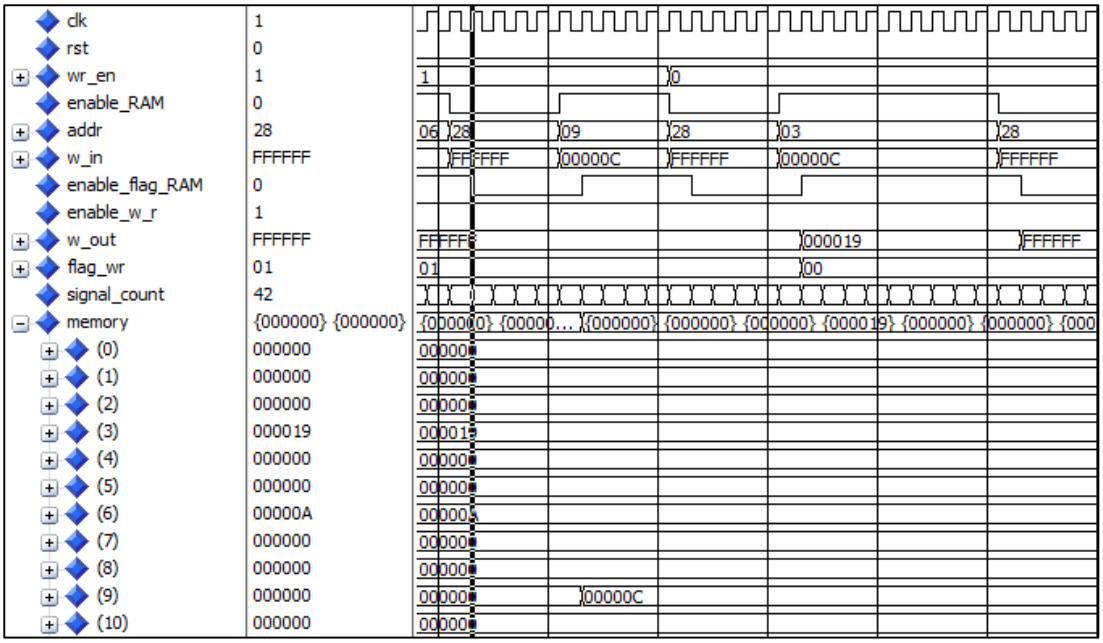

Fig. 3.44. Lectura del módulo “Memoria_RAM”.

3.5.3.1.4 Regla de relajación.

El método usado para obtener el peso sináptico total “w”, se genera a partir de la ecuación (2.3) en su forma excitatoria, usada en la sección (2.5) del capítulo 2, cuya implementación se puede ejemplificar a través de la figura (3.45). Obsérvese que las únicas constantes a almacenar en dicho módulo son “Wmax = 1”, “Wmin =0” y la velocidad de aprendizaje, la cual tiene un valor decimal de 0.5 (rate=0.5).

La regla de relajación se encuentra en el módulo etiquetado como “Relaxion_Rule” y su operación puede simularse en el banco de pruebas etiquetado como “Relaxion_Rule_tb”.

En cuanto a los puertos de entrada, estos son etiquetados como “total_delta_w (23:0)”, “w_old (23:0)”, “clk”, y “rst”, mientras que los puertos de salida se encuentran etiquetados como “w_new (23:0)”. Cabe mencionar que las señales “sal_w_old”, “sal_w_new”, “sal_total_delta”, son provistas por el banco de pruebas, las cuales convierten la representación binaria de los puertos “w_old (23:0), w_new (23:0)” y “total_delta_w (23:0)” en formato decimal respectivamente; adicionalmente a ello, la señal “signal_cuenta” es un contador de pulsos que permite una mejor visualización de la simulación.

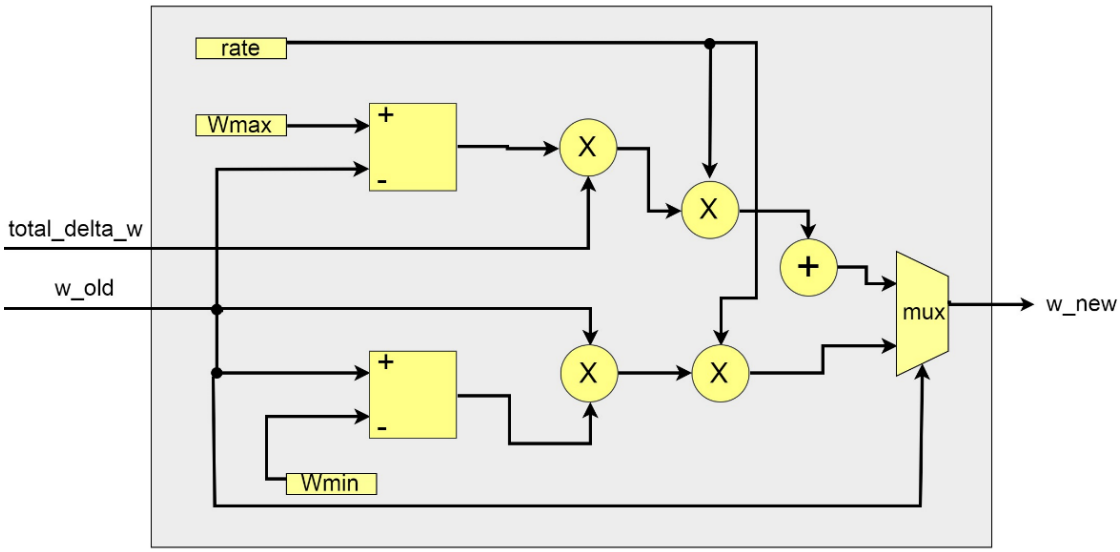

Fig. 3.45. Diagrama a bloque del módulo “Relaxion_Rule”.

En la figura (3.46) puede observarse que siempre a partir de un cambio sináptico positivo existirá un incremento en el peso actual.

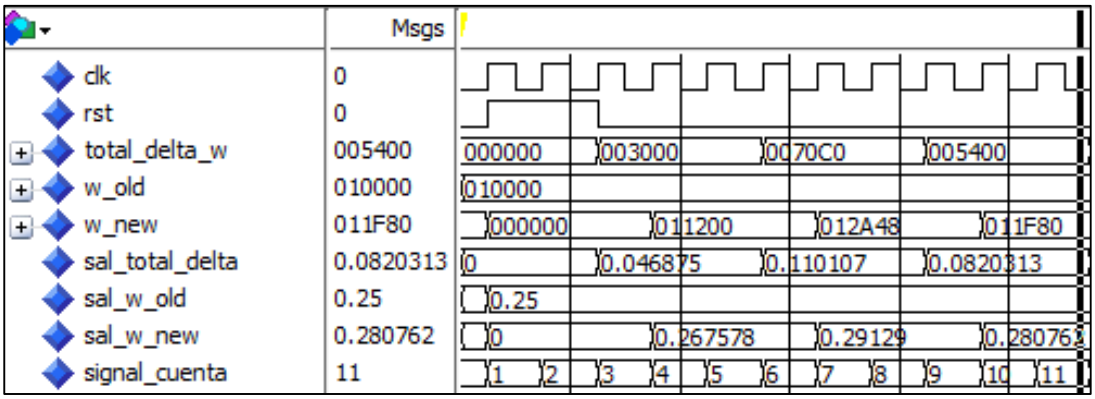

Fig. 3.46. Operación del módulo "Relaxion_Rule".

Por otro lado, en la figura (3.47) puede observarse que siempre a partir de un cambio sináptico negativo existirá una disminución en el peso actual.

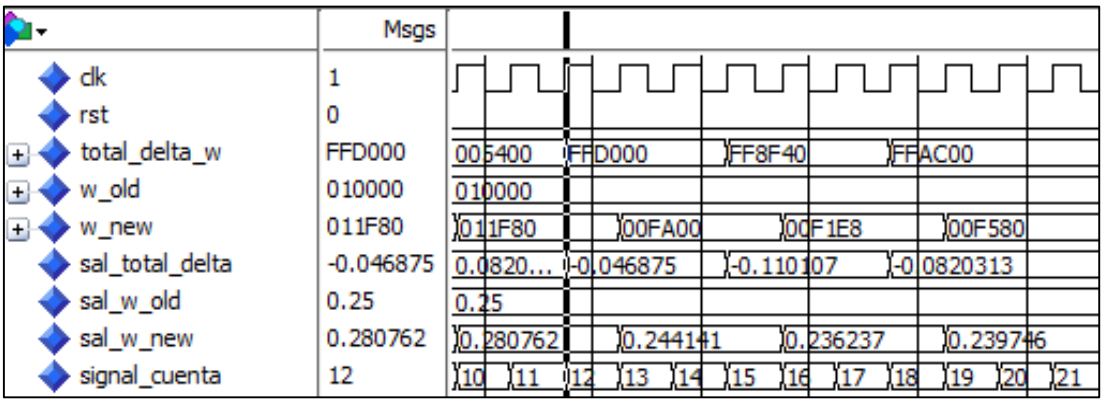

Fig. 3.47. Operación del módulo "Relaxion_Rule".

3.5.3.1.5 Sinapsis

El proceso de sinapsis se lleva a cabo mediante la implementación de la ecuación (1.19), de la sección (1.4.6) del capítulo 1, que integra el modelo más simple usando retardo en sinapsis, teniendo como base una unidad de corriente (1 pA), como en el modelo sináptico en software (véase sección (2.6)) a través de módulo de la sección (3.5.2.1.1), cuya acción conjunta en ambos simplifica la implementación de la sinapsis en hardware. En el módulo encargado de llevar a cabo la sinapsis se implementa un control, así como una serie de registros que permiten procesar de manera paralela y secuencial la

información. En cuanto su operación, ésta se lleva a cabo en 3 fases, las cuales son controladas por el “control secundario” (véase sección 3.5.3.1.1):

- **Fase de integración de baja corriente**: Durante esta etapa se convierten los pesos sinápticos que se reciben a la entrada en una corriente post sináptica mediante la suma del valor existente en un registro dedicado para almacenar la corriente, figura (3.48)-A. En este proceso, el módulo se encuentra evaluando en todo momento, si hay suficiente corriente para iniciar un proceso de aprendizaje, el cual tiene como valor umbral 25.35 pA.
- **Fase de entrenamiento**: Durante esta etapa el módulo toma las direcciones y los pesos sinápticos enviados desde el control secundario con la finalidad de convertirlos en corriente post sináptica. Esto se logra primero sumando el valor del peso en un registro que tiene etiquetada la dirección recibida y, posteriormente en ese mismo ciclo de programa sumando todos los registros con las direcciones restantes en conjunto con la corriente final que se obtuvo en el proceso anterior. De esta manera, las 36 sinapsis terminan siendo sumadas, y por lo tanto, la corriente post sináptica total será la contribución de los pasados, presentes y futuros pesos sinápticos integrados. Este procedimiento se ilustra en el diagrama a bloques de la figura (3.48)-B.
- **Fase de reconocimiento**: Durante esta etapa se realiza la misma tarea que en el proceso de baja corriente. Sin embargo, durante esta etapa no existe un umbral mínimo para que el módulo deje de integrar. En su lugar, la operación es detenida por el “control primario”. Cabe mencionar, que esta etapa sirve como medio de reinicio de la neurona post sináptica cada vez que se repite la fase, evitando propiamente un “reset” y en lugar de ello inyectando una corriente igual a cero, evitando así alterar la dinámica neuronal de la red como se describe en la sección (3.5.2.1.2). Este procedimiento se ilustra en el diagrama bloques de la figura (3.48)-C.

Cabe mencionar, que los procesos antes descritos, se realizan cada vez que existe una dirección y peso sináptico a la entrada, es decir, cada vez que exista un pulso, este módulo lo integrará y cuando no se tenga ninguno, el módulo mantendrá la corriente previa independientemente en qué etapa se encuentre.

Los procesos sinápticos descritos anteriormente se encuentran embebidos en el módulo etiquetado como “Synapses” y su operación puede simularse en el banco de pruebas etiquetado como “Synapses_tb”. En cuanto a los puertos de entrada para la simulación mostrada en la figura (3.49), estos son etiquetados como “rst”, “clk”, “addr_synap(7:0)”, “w_old_synap(23:0)” y “w_new_synap(23:0)”, mientras que sus puertos de salida son etiquetados como “flag_umbral” e “I_final (23:0)”. Sin embargo, es necesario mencionar que el módulo cuenta con puertos de entradas y salidas adicionales usados para establecer tareas de control y monitoreo por parte del control secundario, que por fines prácticos, han sido omitidos aquí. En su lugar, para verificar su operación es necesario invocar las variables “I_4”, “I_8” e “I_umbral” que residen en su lógica interna.

La simulación realizada muestra las dos primeras fases de operación del módulo, las cuales se describen a continuación:

1. El sistema inicia el proceso de baja corriente, por lo que se envía a través de “w_old_synap” un peso sináptico, el cual se almacena en la variable “I_umbral”, la cual se ve reflejada en “I_final”. Una vez terminado, este proceso se visualiza a través la bandera “flag_umbral”.
2. El sistema inicia el proceso de aprendizaje. En este caso, se envía el mismo peso sináptico a través de “w_new_synap”, pero esta vez con diferentes direcciones. Obsérvese que esta vez se almacenan en distintos registros, pero al momento de integrarse de manera paralela, esto se ve reflejado en el puerto de salida “I_final”.

Finalmente, cabe mencionar que el proceso para enviar correctamente la información de procesamiento a la representación neuronal (véase sección (3.1.3)), se realiza agregando 6 bits por derecha a los pesos sinápticos, los

cuales, tras ser sumados, se toman de izquierda a derecha los bits que corresponden a la representación neuronal.

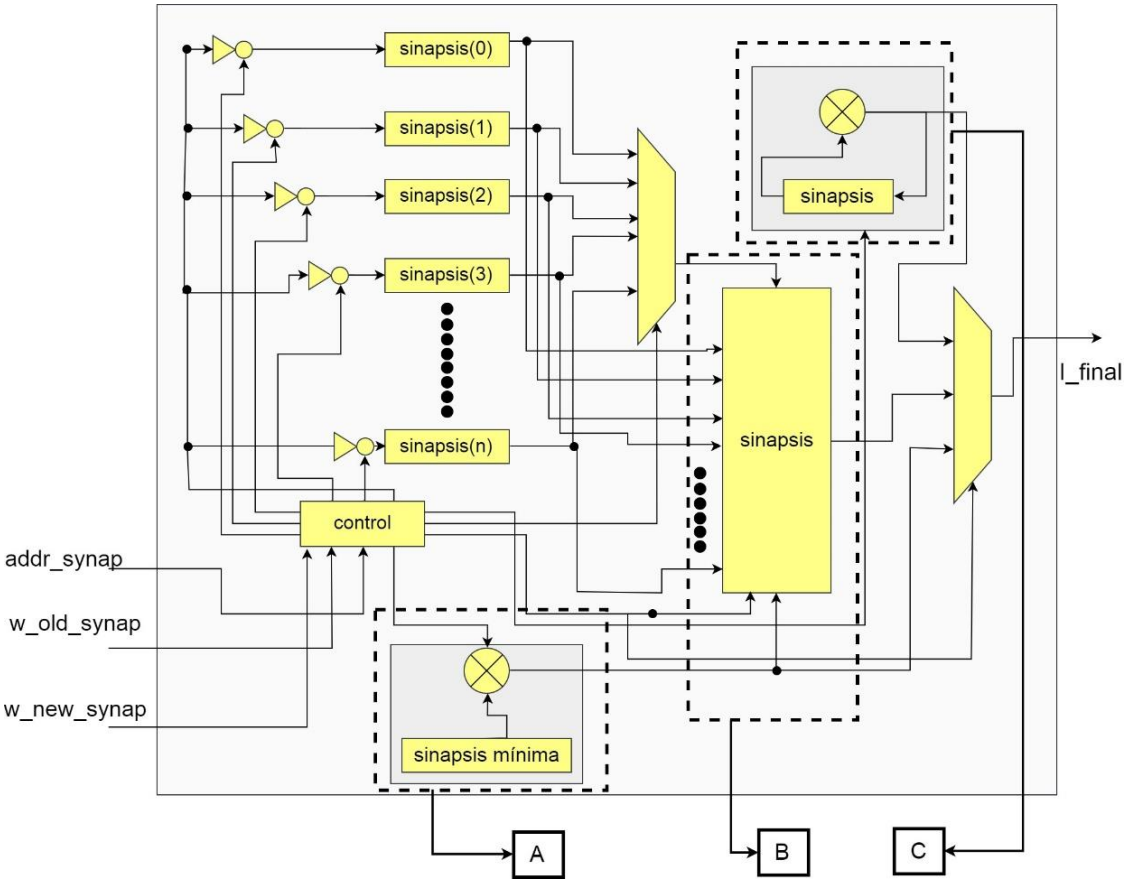

Fig. 3.48.Diagrama a bloques del módulo "Synapses".

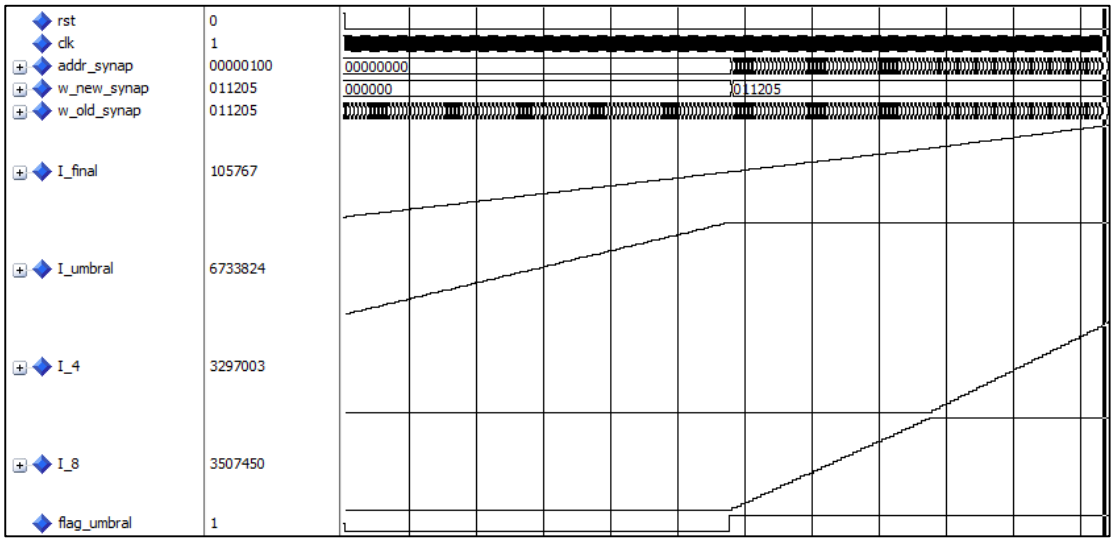

Fig. 3.49. Operación del módulo "Synapses".

3.5.3.1.6 Integrador de baja corriente.

El integrador de baja corriente es simplemente un módulo que se encarga de vincular las direcciones provenientes del módulo “Entry_process” (véase sección (3.5.2)), durante las etapas de “integración de baja corriente” y “reconocimiento” que realiza el “control secundario”, esto con la finalidad de aislar el proceso de sinapsis del proceso de aprendizaje dirigido por STDP, el cual se lleva de manera conjunta únicamente en la etapa de entrenamiento. Dicho módulo se encuentra etiquetado como “Address_Low_Current” y su operación puede simularse en el banco de pruebas etiquetado como “Address_Low_Current_tb”. En cuanto a los puertos de entrada para la simulación mostrada en la figura (3.50), estos son etiquetados como “rst”, “clk”, “address_pre_synap (7:0)”, mientras que el puerto de salida es etiquetado como “address_low_c (7:0)”. Sin embargo, es necesario mencionar que el módulo cuenta con puertos de entradas y salida adicionales, usados para establecer tareas de control y monitoreo por parte del control secundario, que por fines prácticos, han sido omitidos aquí.

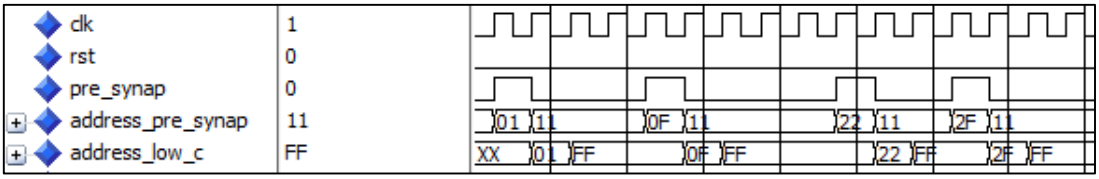

Fig. 3.50.Operación del módulo “Address_Low_Current”.

3.5.3.1.7 LFSR: Generador de números aleatorios

Con la finalidad de inicializar los pesos sinápticos de la red a implementar, se usará un “Linear Feedback Shift Register” el cual se traduce como “registro de desplazamiento de retroalimentación lineal” conocido por sus siglas en inglés “LFSR”. El “LFSR” es un registro de desplazamiento que permite generar números aleatorios y que basados en su operación de acuerdo al trabajo de Ward y Molteno en [34] los podemos dividir en dos tipos:

- Fibonacci: en el cual a varios registros se les aplica la operación lógica XOR para proporcionar el bit de entrada de desplazamiento.

- Galois: donde al bit de salida del registro desplazado se le aplica la operación XOR respecto a la entrada de ese mismo registro.

En nuestro caso, se implementa un módulo LFSR tipo Galois basándonos en la operación que se presenta en [34] y [35], en los cuales, la generación de números aleatorios se encuentra fundamentalmente basada en el ancho de palabra y los “tap’s” seleccionados para generarlos. Los “tap’s” son las posiciones de los bits donde se encuentran las compuertas XOR que permiten, a través de la rotación de un registro, dada su retroalimentación, generar distintos números. “Un tap en la posición i en un LFSR de n-etapas indicaría que, en cada iteración, la salida desplazada del primer registro le sería aplicada la operación XOR con la salida de la i-ésima posición del registro y alimentada a la entrada de la siguiente posición (i-1)” [34].

Dado que el objetivo principal de la implementación de un “LFSR” es solo para proveer al sistema de un generador simple de números aleatorios para inicializar los pesos sinápticos, en nuestra representación en punto fijo, implementaremos un LFSR de 11 bits para que solamente se generen cantidades por debajo de 0.5 en formato decimal. Para llevarlo a cabo, basado en las recomendaciones realizadas en [34], para implementar los “tap’s” en un registro de 11 bits y generar un “LFSR Galois” que nos provea de números aleatorios, “aunque éste no sea de ciclo máximo”, serán necesarios solamente 2 “tap’s” ubicados en el bit 11 y 9. Obsérvese que dado que se está usando una longitud de palabra de 24 bits y nuestro generador está generado a partir de una longitud de 11 bits dentro del bloque LFSR, se concatena este ancho de palabra con un vector de bits en ceros, para que a su salida pueda ser expresado en decimal, el valor máximo de generación de “0.5” y que no exista ningún problema con la representación.

Dicho módulo se encuentra etiquetado como “LFSR” y su operación puede simularse en el banco de pruebas etiquetado como “LFSR_tb”. En cuanto a los puertos de entrada para la simulación mostrada en la figura (3.51) estos son etiquetados como “clk”, “rst”, “enable_LFSR”, mientras que el puerto de

salida es etiquetado como “aleatorio (23:0)”. Sin embargo, es necesario mencionar que el módulo cuenta con puertos de entradas y salida adicionales, usados para establecer tareas de control y monitoreo por parte del control secundario, que por fines prácticos, han sido omitidos aquí. Adicionalmente, se ha incluido la señal “salida” como un conversor de números binario a decimal y facilitar la visualización de las secuencias de módulos aleatorios generadas por el módulo. Dicha simulación consiste en habilitar el módulo a través de “enable_control” para que se comiencen a generar números aleatorios.

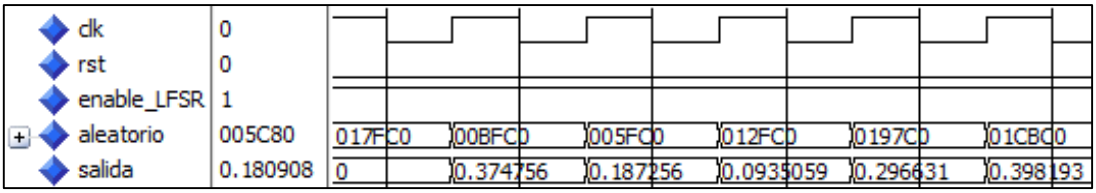

Fig. 3.51.Operacion del módulo “LFSR”.

3.6 Conclusiones

El sistema propuesto es un sistema de pequeñas dimensiones con arquitectura neuromórfica, con capacidad para realizar el entrenamiento y reconocimiento de hasta 10 caracteres, sin la necesidad de un sistema adicional de pre procesamiento o de control, usando como medio de aprendizaje STDP no supervisado y selectivo. El sistema, por sus características basadas en [6], es un sistema bio-inspirado, ya que intenta replicar el comportamiento de una red neuronal biológica sin necesariamente llegar a la plausibilidad, aunque se esté ocupando el módulo neuronal más aproximado; esto debido principalmente a 3 razones:

1. La dinámica neuronal de una red se ve influenciada por el número de neuronas que se integra en su red. En nuestro caso, como hemos mencionado, es de bajas dimensiones.
2. El modelo sináptico usado es el más simple de todos y en comparación con los modelos expuestos en el capítulo 1, tiene un nivel bajo de plausibilidad biológica, cuya elección fue realizada con la finalidad de

ahorrar recursos en hardware y tener una mayor cantidad de neuronas disponibles.

3. El sistema de asociación de pares que dirige el protocolo de STDP fue propuesto de acuerdo a la teoría expuesta en el “Material suplementario” de esta tesis, y por lo tanto, es una forma teórica más de cómo realizan las neuronas biológicas la asociación de pulsos, el aprendizaje y sinapsis en tiempo real.

En general, el sistema permite la integración espacio-temporal simultánea de los pulsos de la todas las neuronas pre sinápticas, gracias a que la arquitectura provista es totalmente compatible con la arquitectura neuromórfica conocida como “AER”, permitiendo que el sistema funcione por eventos. En este sentido, durante el aprendizaje, cada vez que se tiene un evento de asociación de pares, se obtiene un evento de modificación de peso sináptico (A), el cual a su vez genera un evento de asociación de memoria (B) y que a su vez genera un evento de sinapsis (C). Lo anterior, afecta en tiempo los pesos sinápticos calculados a los que se llegan y por ende a la corriente post sináptica, tal como se observa en la figura (3.52); en la etapa de reconocimiento ocurre algo semejante, solamente que en esta ocasión cada pulso pre sináptico genera solamente un evento de sinapsis.

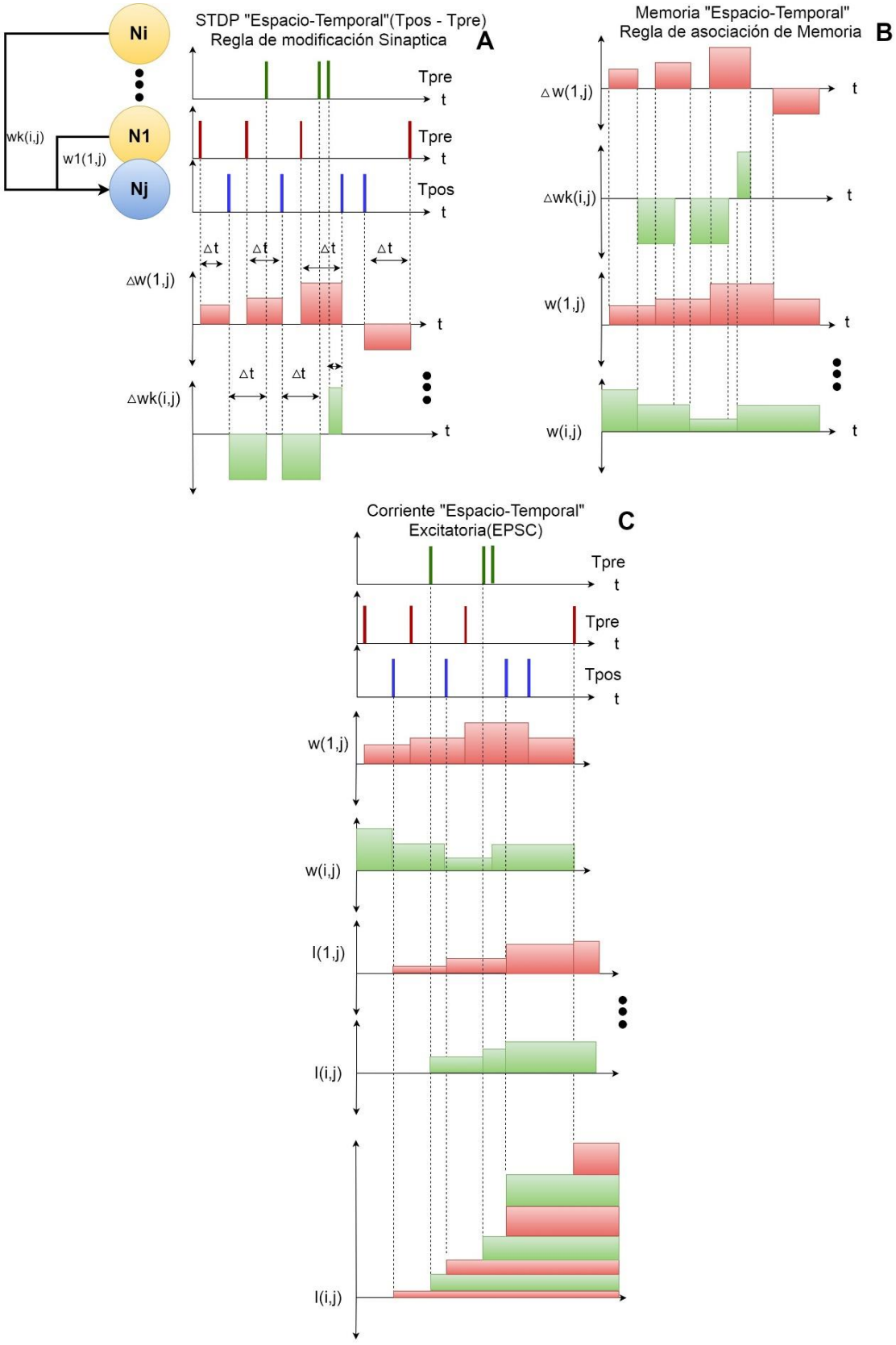

Fig. 3.52. Sistema de eventos durante el entrenamiento del sistema neuromórfico propuesto.

3.7 Referencias

[1] V. A. Pedroni, "Circuit Design with VHDL," MIT Press, 2004.

[2] D. P. Morton, *Hardware Modeling and Top-Down Design Using VHDL,* Boston, E.U.A.: M.Sc. Thesis. Massachusetts Institute of Technology, June 1991.

[3] W. Maass, "On the relevance of time in neural computation and learning" *Theoretical Computer Science, Elsevier.,* p. 157–178, 2001.

[4] K. A. Boahen, "Point-to-Point Connectivity Between Neuromorphic Chips Using Address Events" *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing,* Vol. 47, No. 5, pp. 416-434, 2000.

[5] S.-C. L. T. D. R. D. G Indiveri, "Neuromorphic Systems" *Elsevier,* 2009.

[6] T. E. P. B. E. D. S. R. J. S. P. Catherine D. Schuman, "Computer Science (Cornell University) >Neural and Evolutionary Computing. A Survey of Neuromorphic Computing and Neural Networks in Hardware", 19 May 2017. [Online]. Available: https://arxiv.org/abs/1705.06963. [Accessed June 2019].

[7] S.-C. L. Giacomo Indiveri, "Memory and Information Processing in Neuromorphic Systems" *Proceedings of the IEEE,* vol. 103, no. 8, pp. 1379 - 1397, August 2015.

[8] S. H. M. E. S. C. M. R. L. S. F. Johannes Partzsch, "A Fixed Point Exponential Function Accelerator for a Neuromorphic Many-Core System," *2017 IEEE International Symposium on Circuits and Systems (ISCAS),* pp. 1-4, 2017.

[9] S. D. P. K. ,. A. A. Andrew Cassidy, "FPGA Based Silicon Spiking Neural Array" *IEEE Biomedical Circuits and Systems Conference,* pp. 75-78, 2007.

[10] J. C. V. Eduard-Guillem Merino Mallorquí, *Digital System for spiking neural network emulation,* Grau en Enginyeria Electrònica Industrial i Automàtica.Universitat Politècnica de Catalunya, June 2017.

[11] R. S. J. G. H. Vaibhav Garg, "Spiking Neuron Computation With the Time Machine" *IEEE Transactions on Biomedical Circuits and Systems ,* Vol. 6, No. 2, pp. 142 - 155, 2012.

[12] D. ,. G. J. F. D. H. W. C. M. A.-S. M. J. O. &. W. S. Purves, "Neuroscience, Third Edition.," Sunderland, Massachusetts, U.S.A., Sinauer Associates, Inc. Publishers, 2004.

[13] J. V. Tranquillo, "Quantitative Neurophysiology. Synthesis Lectures on Biomedical Engineering #21," Morgan & Claypool, 2008.

[14] F. d. A. G. Rodríguez, *Análisis, diseño e implementación de sistemas neuromórficos basados en pulsos para el procesado de información de retinas artificiales,* Universidad de Sevilla: Ph.D. Thesis, 2011.

[15] M. A. Sivilotti., *Wiring Considerations in Analog VLSI Systems, with Application to Field-Programmable Networks,* Pasadena, California, USA: Ph.D.Thesis. California Institute of Technology, 1991.

[16] M. Mahowald, *VLSI Analogs of Neuronal Visual Processing: A Synthesis of Form and Function,* Pasadena, California: Ph.D. Thesis. California Institute of Technology, 1992.

[17] R. P. Vicente, *Una aportación al procesamiento de la información visual mediante técnicas bioinspiradas,* Sevilla, España: PhD. Thesis. Escuela Técnica Superior de Ingeniería Informática . Universidad de Sevilla, 2008.

[18] J. J. R. P. D. T. A. B. J. R. R. B. Susana Ortega Cisneros, "Space-Time AER Protocol Receiver Asynchronously Controlled on FPGA" *11th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE) ,* 2014.

[19] J. M. K. J.M. Moreno, "Synchronous Digital Implementation of the AER Communication Scheme for Emulating Large-Scale Spiking Neural Networks Models" *IEEE NASA/ESA Conference on Adaptive Hardware and Systems,* pp. 189-196, 2009.

[20] P. H. Hans Kristian Otnes Berge, "High-Speed Serial AER on FPGA" *IEEE International Symposium on Circuits and Systems,* pp. 857-860, 2007.

[21] T. S.-G. C. S.-G. B. A. a. B. L.-B. J. A. Pérez-Carrasco, "High-Speed Character Recognition System based on a complex hierarchical AER architecture" *IEEE International Symposium on Circuits and Systems,* pp. 2150-2153, 2008.

[22] R. P.-V. M. R. A. L.-B. G. J. A. C. A. Jiménez-Fernández, "AER-based robotic closed-loop control system" *2008 IEEE International Symposium on Circuits and Systems,* pp. 1044-1047, 2008.

[23] C. Z. Ramos, *Modular and scalable implementation of AER neuromorphic systems,* PhD. Thesis. Universidad de Sevilla, 2011.

[24] Texas Instrumens, *FIFO Architecture, Functions, and Applications,* Texas Instrumens, SCAA042A, November 1999.

[25] C. Y. W. Z. Yanjun Zhang, "Asynchronous FIFO Implementation Using FPGA" *IEEE International Conference on Electronics and Optoelectronics (ICEOE 2011) ,* Vol. 3, pp. 207-209, 2011.

[26] A. R. F. G. S. D. F. Xin Jin, "Implementing Spike-Timing-Dependent Plasticity on SpiNNaker Neuromorphic Hardware" *IEEE 2010*

International Joint Conference on Neural Networks (IJCNN), pp. 2302-2309, July 2010.

[27] G. I. J.-D. L. D. B. Charlotte Frenkel, "A Fully-Synthesized 20-Gate Digital Spike-Based Synapse with Embedded Online Learning" *IEEE International Symposium on Circuits and Systems (ISCAS),* pp. 1-4, 2017.

[28] J. T. O. M. G. N. Y. B. ,. R. Bilel Belhadj, "FPGA-based Architecture for Real-time Synaptic Plasticity Computation" *2008 15th IEEE International Conference on Electronics, Circuits and Systems,* pp. 93-96, 2008.

[29] D. S. a. T. Trappenberg, "The Trouble with Weight-Dependent STDP" *IEEE. Proceedings of International Joint Conference on Neural Networks. Orlando, Florida, USA.,* 2007.

[30] W. G. Jean Pascal Pfister, "Triplet´s of Spikes in a Model Spike Timing-Dependent Plasticity," *Journal of Neuroscience,* Vol. 26, No. 38, pp. 9673–9682, 2006.

[31] A. R. F. G. a. S. F. Sergio Davies, "A forecast-based biologically-plausible STDP learning rule" *IEEE Proceedings of International Joint Conference on Neural Networks, San Jose, California, USA,* pp. 1810-1817, 2011.

[32] G. M. T. Xavier, *Desarrollo en FPGA de un emulador de panel fotovoltaico,* México, D.F.: M. Sc. Thesis. Centro de investigación y de Estudios Avanzados del Instituto Politécnico Nacional, Enero 2014.

[33] A. G. A. J. G. Andrew Cassidy, "A Combinational Digital Logic Approach to STDP" *2011 IEEE International Symposium of Circuits and Systems (ISCAS),* pp. 673-676, 2011.

[34] T. C. M. Roy Ward, *Technical Reports from the electronics group at the University of Otago. Table of Linear Feedback Shift Registers,* Dunedin, New Zealand, 2012.

[35] D. C. G. J. A. C. M. O. B. L.-B. A. Linares-Barranco, "Poisson AER generator: Inter-Spike-Intervals Analysis," *IEEE International Symposium on Circuits and Systems,* pp. 3149-3152, 2006.

CAPÍTULO 4

“*La belleza más noble no es la que nos deslumbra al instante, la que nos seduce por asaltos tempestuosos y embriagadores, sino aquella que se insinúa lentamente, la que uno lleva dentro de si dentro del pensamiento, y que un día, soñando se vuelve a ver adelante, y que por fin, después de haberse circunscrito con modestia en nuestro corazón, toma posesión completa de nosotros, llena nuestros ojos de lágrimas y nuestro corazón de deseo”. La lucha lenta de la belleza*

“Es para todo escritor sorpresa enojosa y siempre nueva que su libro, desde que se separa de él viva con vida propia. Quizá la olvidara, quizá se elevará por los conceptos que en él ha depositado, quizá ni lo entenderá ya y habrá perdido el alto vuelo a que se remontara para concebirlo; más el libro es buscado por los lectores, produce dicha o desdicha, es causa de nuevas obras, se hace el alma de principios y de acciones: en una palabra, vive como un ser provisto de espíritu y de alma pero no es un hombre. Es una dicha para el autor decir que lo que en él existía de ideas y de sentimientos creadores de vida, fortificantes, edificantes, esclarecedores, vive en sus obras y que él mismo no es más que la ceniza gris, mientras que el fuego ha sido conservado y propagado por todas partes. Si se considera, pues, que toda acción de un hombre, y no solamente de un libro, sirve por algún motivo de acción a otras acciones, decisiones y pensamientos, que todo lo que hace está anudado a lo que se hará, tendremos que reconocer verdadera inmortalidad existente, la del movimiento”. El libro troncado casi en un hombre.

Friedrich Nietzsche, “Humano, demasiado humano”, 1879.

4 Resultados, conclusiones y trabajo futuro.

4.1 Resultados en Software

4.1.1 Introducción

Para realizar la implementación del algoritmo descrito en el capítulo 2, se usó Matlab 2018 para escribir el código fuente para redes de diferentes dimensiones y por lo tanto con diferentes capacidades para almacenar y recuperar la información referente a un conjunto de caracteres. En específico, el primer caso consta de una red neuronal para realizar aprendizaje auto-organizado de caracteres de 16 pixeles (4x4) y reconocimiento de 3 caracteres, y el segundo, consta de una red neuronal para realizar aprendizaje selectivo de caracteres de 25 pixeles (5x5) y reconocimiento de 12 caracteres.

Los valores tanto del modelo neuronal del modo RS de Izhikevich y de la ventana de aprendizaje que realiza STDP, es el mismo para ambas arquitecturas, las cuales se pueden encontrar en las tablas (1.2) y (2.1) del capítulo 1 y 2 de esta tesis, respectivamente.

Ambos programas se encuentran en el apéndice, disponibles en la dirección http://www.vlsilab.cinvestav.mx/tesis_fgc.html del Laboratorio VLSI CINVESTAV etiquetados como “r_4x4_STDP_auto.m” y “r_5x5_STDP_selectivo.m”.

4.1.2 Software de aprendizaje y reconocimiento auto-organizado

El código fuente usado para implementar el aprendizaje auto-organizado y de reconocimiento para una red neuronal de tercera generación, como se describe en el capítulo 2, se encuentra etiquetado como “r_4x4_STDP_auto.m”. Dicho software describe una red neuronal que posee 16 neuronas en la capa de entrada y 3 en la capa de salida, con capacidad de recibir caracteres de 4x4 pixeles y realizar reconocimiento de hasta 3 patrones,

para lo cual, la red implementada está formada por 19 neuronas y 48 pesos sinápticos. Dado que la auto-organización se realiza por la competición entre las neuronas de la capa de salida, se usó el método burbuja para determinar este proceso y organizar el aprendizaje en función de la actividad neuronal más intensa.

Adicionalmente, la velocidad de aprendizaje en la regla de relajación (ecuación (2.3)) de la sección (2.5) del capítulo 2 es de “0.1” ($\eta = 0.1$) y en los procesos tanto de entrenamiento como de reconocimiento se estableció una longitud en la ventana de trabajo de 250 ms, para desarrollar la actividad eléctrica de todas las neuronas. En cuanto al número de épocas máximas definidas en el programa, se establecieron “3,000” épocas para alcanzar la convergencia en el aprendizaje de la red neuronal.

En cuanto a cómo se asociaron los pixeles a la corriente que recibe la primercapa; a los pixeles negros se les asigno un valor de corriente constante de 5 pA (I=5 pA), mientras que a los pixeles blancos una corriente constante de 0 pA (I= 0 pA).

4.1.2.1 Proceso de aprendizaje auto-organizado

En el proceso de aprendizaje, dado que nuestra red solo tiene capacidad para reconocer 3 imágenes, seleccionamos los caracteres “A”, “F” y “C”, los cuales son mostrados en la figura (4.1).

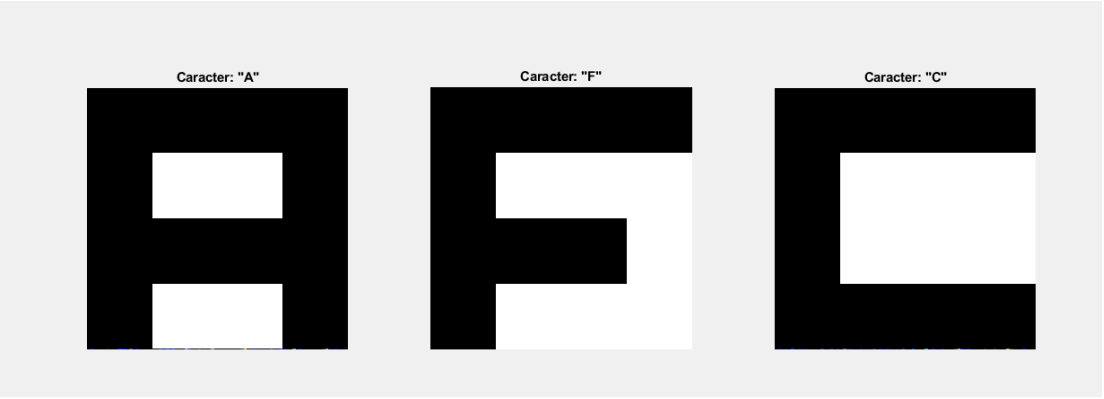

Fig. 4.1.Caracteres a aprender y reconocer por la red.

Cada uno de estos caracteres fueron pasando uno a uno a la red neuronal hasta concluir las épocas máximas definidas, es decir, se colocó el caracter “A” para que la red lo “memorizara” durante un período de 3,000 épocas, con lo que se concluía el aprendizaje del primer patrón y así sucesivamente hasta llegar al último.

Durante este proceso de aprendizaje, en la época número “1” los pesos sinápticos “w” son asignados de manera aleatoria y en la época final se obtienen los pesos sinápticos finales que se fueron modificando durante el proceso de entrenamiento. En las figuras (4.2)-(4.4), se observan en escala de grises, los pesos sinápticos al inicio y al final del entrenamiento, para cada neurona de salida y corroborar la capacidad auto asociativa de la red descrita en la sección (2.9) del capítulo 2. El blanco corresponde al valor “1”, y el negro al valor “0” del peso sináptico.

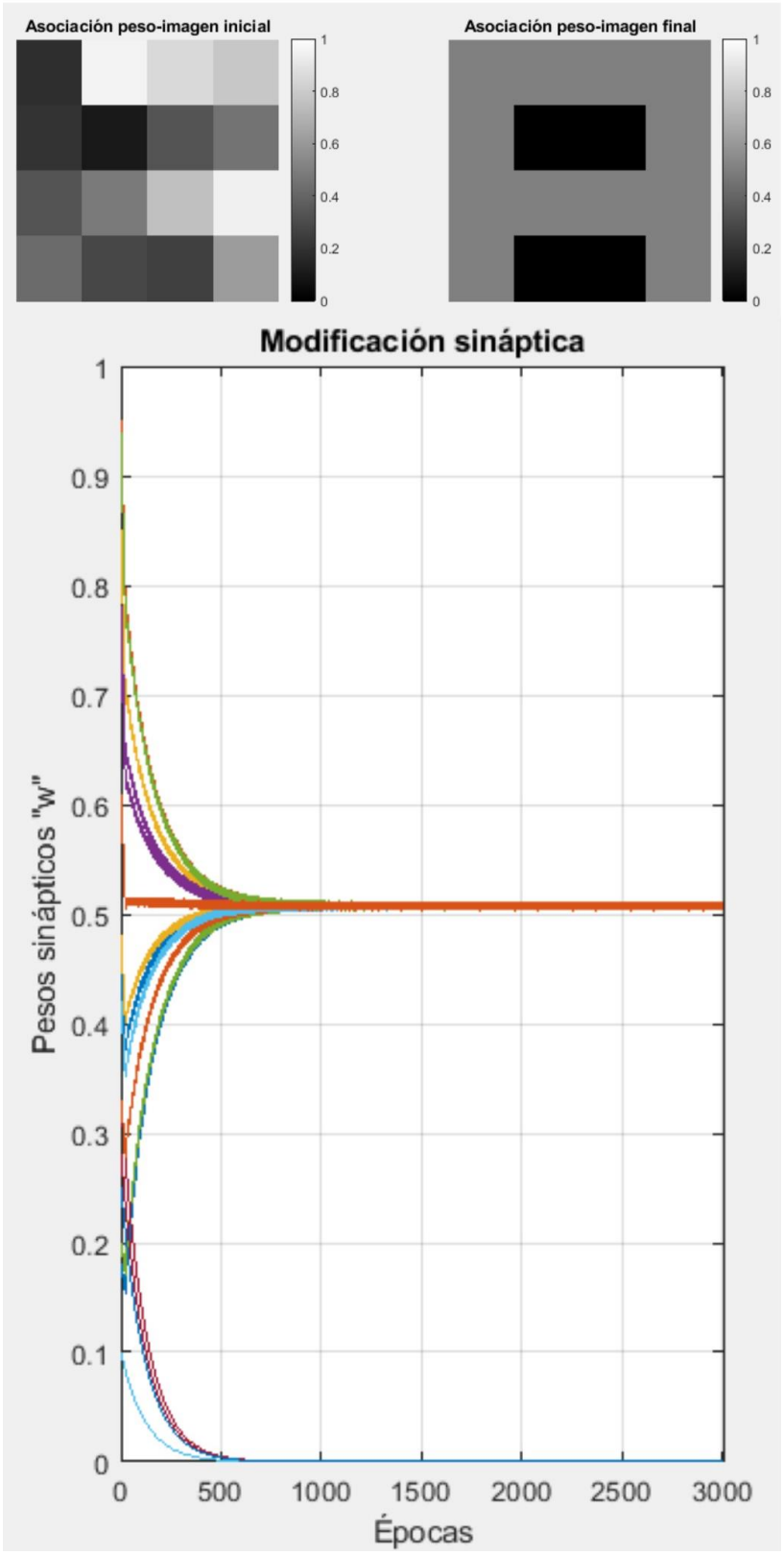

Fig. 4.2. Evolución de los pesos sinápticos asociados al caracter "A" durante el aprendizaje.

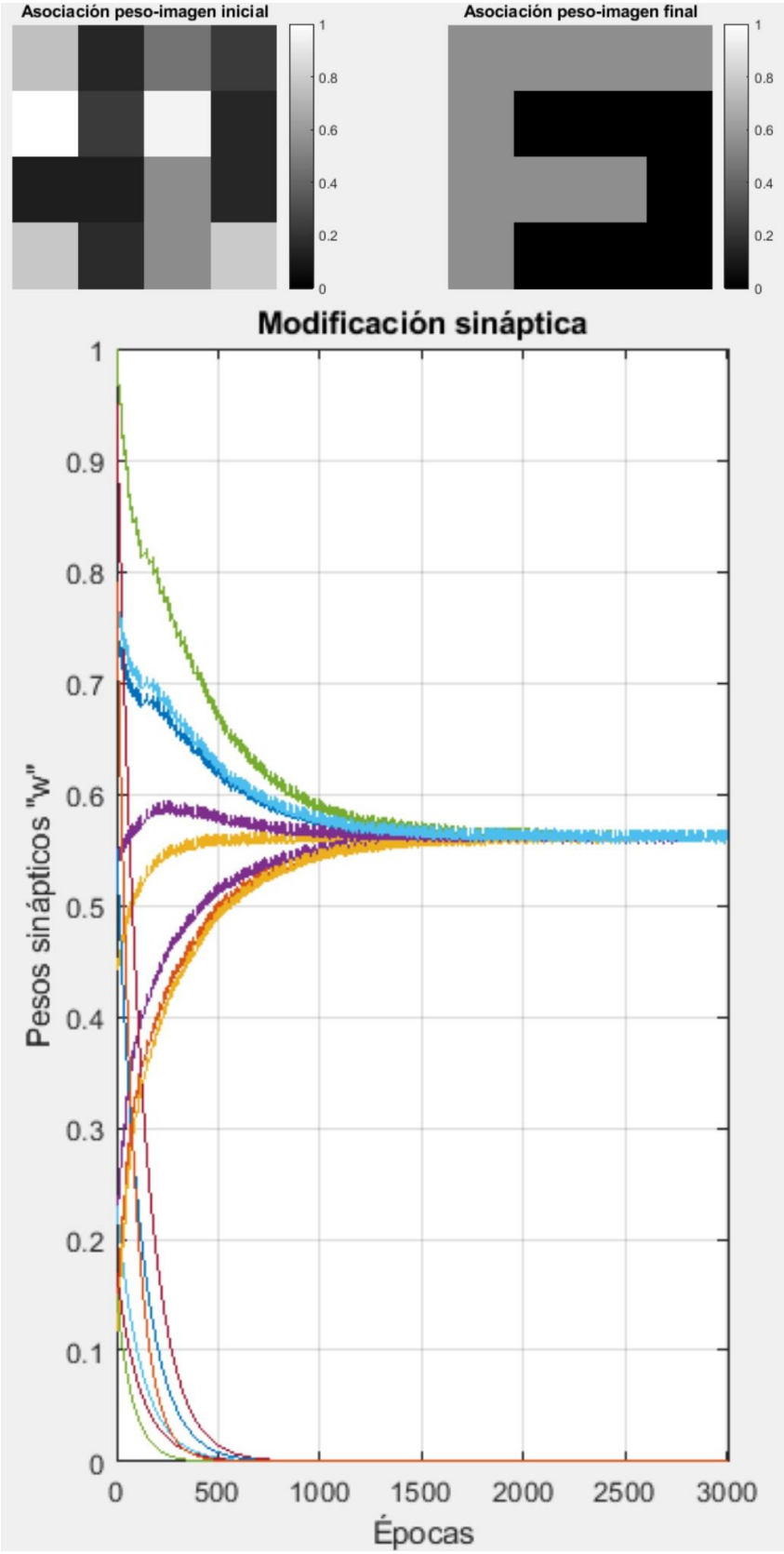

Fig. 4.3. Evolución de los pesos sinápticos asociados al caracter "F" durante el aprendizaje.

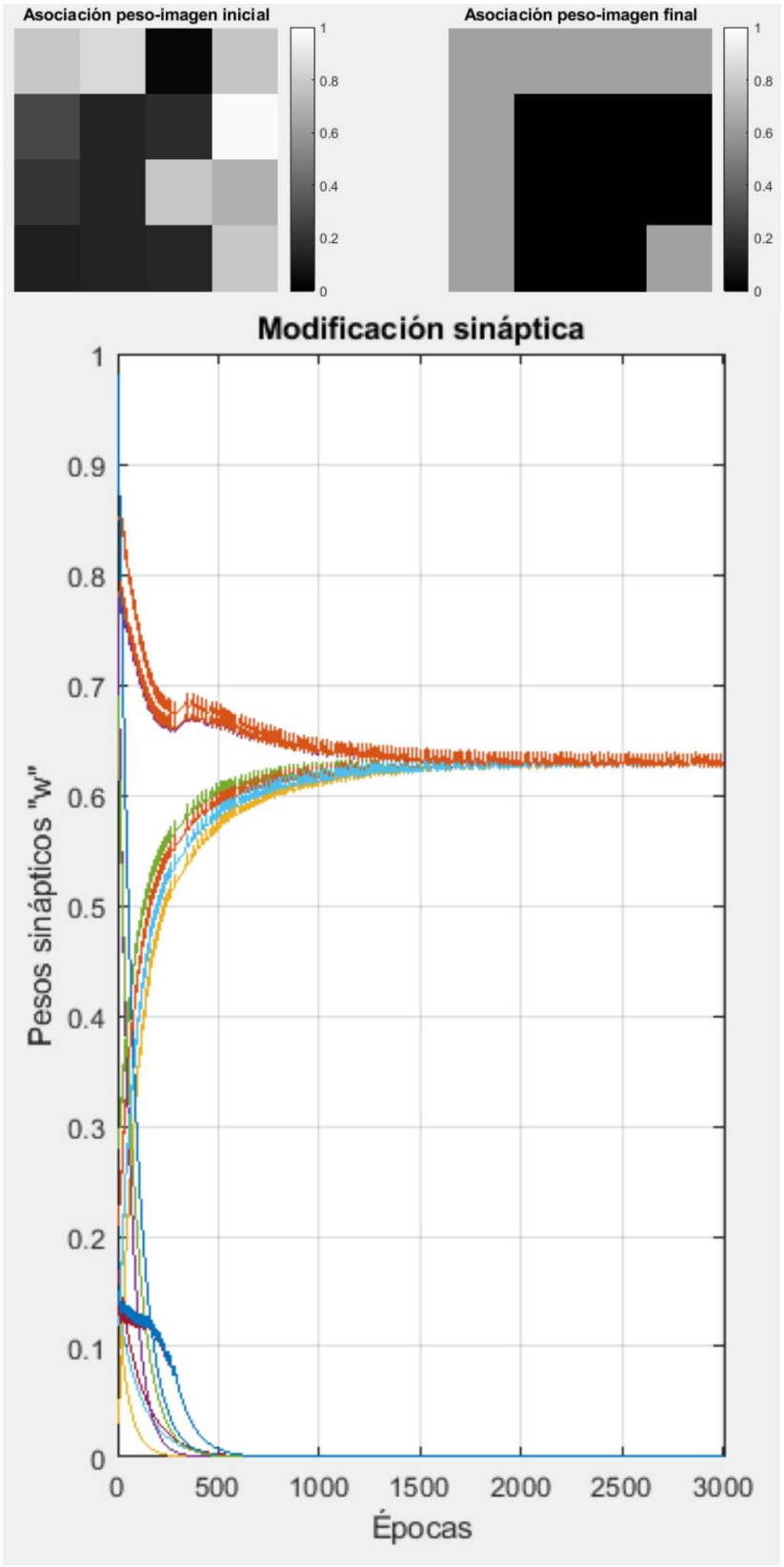

Fig. 4.4. Evolución de los pesos sinápticos asociados al caracter "C" durante el aprendizaje.

A partir de los resultados obtenidos, podemos observar que los pesos sinápticos, independientemente de qué caracter se haya asociado a la neurona de capa de salida, cuando la imagen es reconocida y memorizada, los pesos sinápticos reflejan el patrón aprendido. El único caso en el cual el aprendizaje no se llevó a cabo al 100 %, fue en el carácter “C”. Sin embargo, dado que el reconocimiento se basa en la actividad de toda la red, no debería existir ningún problema al momento de llevar la fase de reconocimiento.

Adicionalmente, el programa genera una matriz tipo “string” etiquetada como “resultado” donde se resume qué neurona post sináptica se asoció a cada patrón, lo que permite conocer el valor de corriente post sináptica excitatoria (EPSC), de cada una de las neuronas post sinápticas cuando reconocen el caracter; tales resultados se muestran en la tabla (4.1).

Neurona Post sináptica	Caracter aprendido	EPSC [pA]
1	A	18.13
2	C	15.14
3	F	15.05

Tabla 4.1. Resultados obtenidos del algoritmo de entrenamiento.

Como vemos, las neuronas asociadas al carácter “C” y “F” tienen una corriente semejante y que incluso puede verse esta característica en la capacidad auto-asociativa de estas 2 neuronas; en este caso, la asociación de imagen por el peso sináptico individual de cada una de ellas solo depende de los pixeles con los cuales se entrenaron.

Pese a esto, dado que la corriente post sináptica a la que se llega es debido precisamente a los pesos sinápticos que tiene asociados, es que existen estas diferencias, y por lo tanto la red neuronal no tiene problemas para poder discernir entre los 3 caracteres.

4.1.2.2 Proceso de reconocimiento.

El proceso de reconocimiento se basa en el algoritmo descrito en la sección (2.8.2). Este proceso consiste en presentar los caracteres a la red neuronal

aprendidos durante la fase de entrenamiento, para determinar cuál de todas las neuronas de la capa de salida es capaz de asociarse por el valor de corriente post sináptica obtenida en el proceso anterior, siendo, que una vez aplicada a la neurona ganadora, los potenciales que genere tendrán un efecto inhibidor sobre las demás, obteniendo una respuesta eléctrica más intensa.

En este caso, cuando se le presentó a la red previamente entrenada, el caracter “C”, se observa en la figura (4.5) que la neurona que se logró asociar fue la segunda neurona de la capa de salida. Cuando se le presentó el caracter “F”, se ve en la figura (4.6), que la neurona que se logró asociar fue la tercera y finalmente cuando se le presentó el carácter “A”, en la figura (4.7) se ve que la neurona que se logró asociar fue la primer neurona.

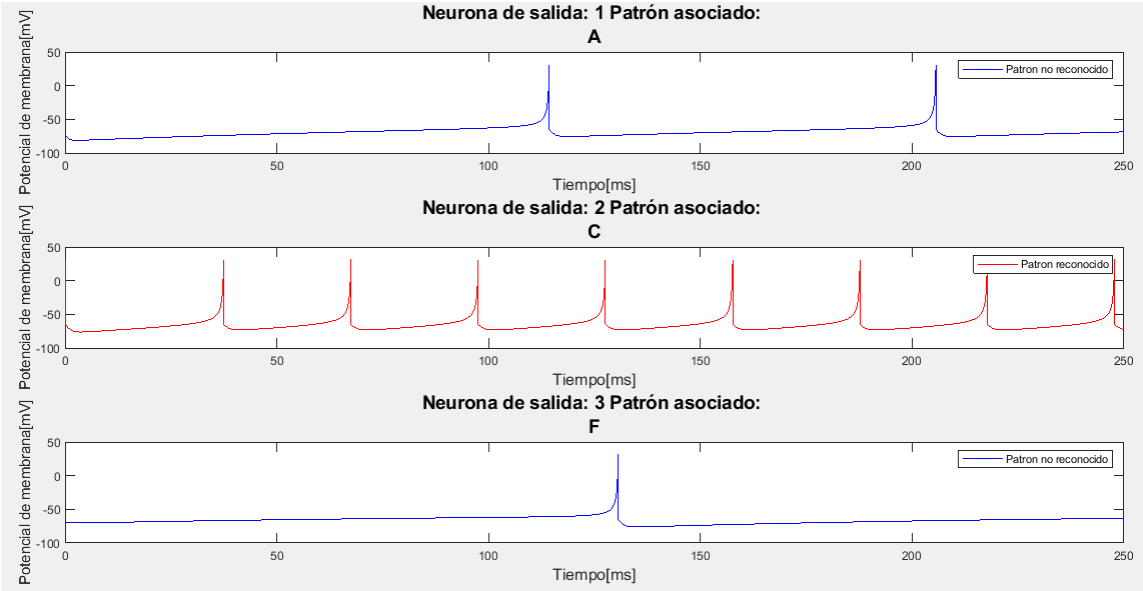

Fig. 4.5.Etapa de reconocimiento. Caracter “C”.

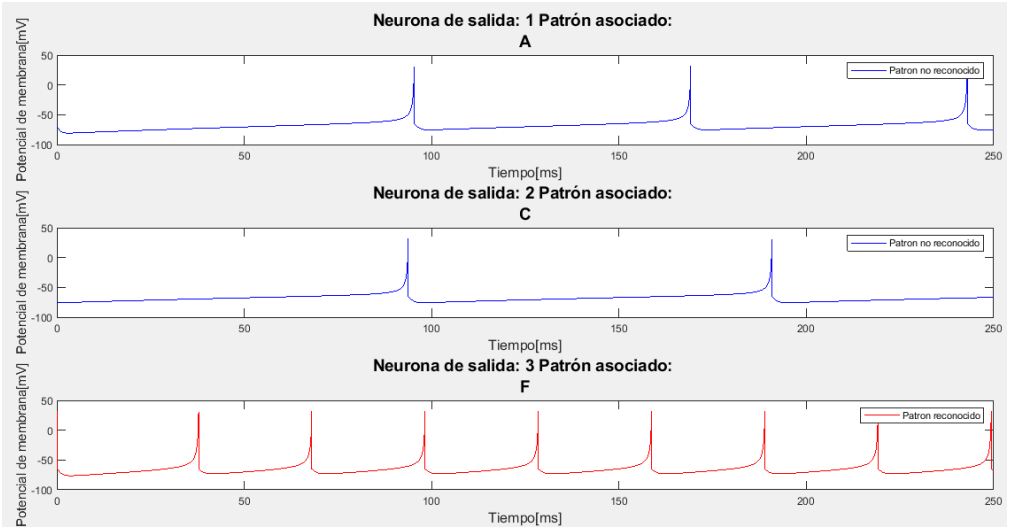

Fig. 4.6.Etapa de reconocimiento. Caracter "F".

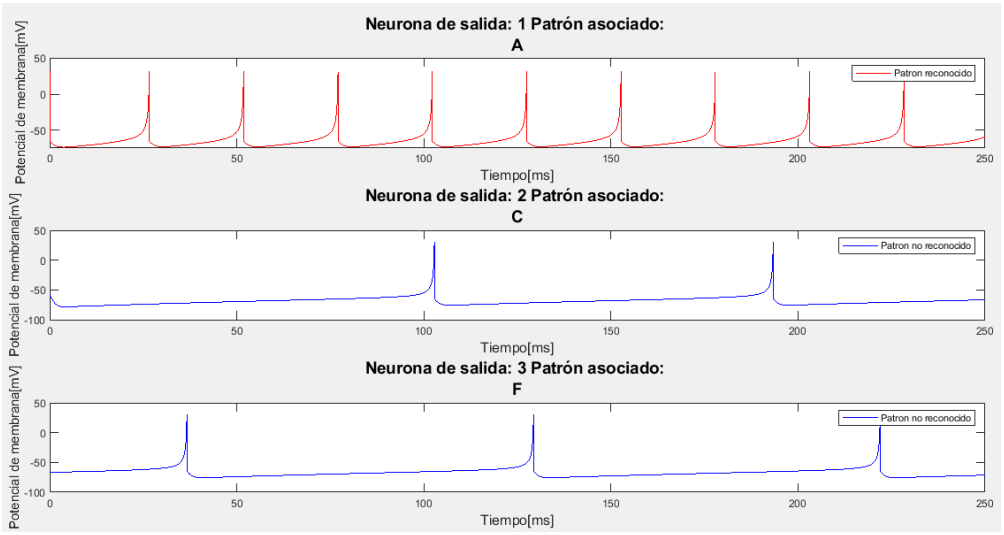

Fig. 4.7.Etapa de reconocimiento. Caracter "A".

Obsérvese, que si analizáramos el "code rate" de las neuronas de salida de acuerdo a la ecuación (1.32) de la sección (1.6.1.1), del método por conteo de pulsos, se obtiene la tabla (4.2).

Neurona en capa de salida	Frecuencia en "A"[Hz]	Frecuencia en "C"[Hz]	Frecuencia en "F"[Hz]
1	40	8	12
2	8	36	8
3	12	12	36

Tabla 4.2.Codificación por conteo de pulsos

Como podemos ver, la frecuencia más alta nos permite identificar qué caracter se asoció a cada neurona de la capa de salida, la cual, en todos los casos fue menor o igual a 40 Hz cuando el patrón fue reconocido independientemente de cuál se tratara. En este caso, la red tuvo una capacidad de reconocimiento del 100%.

Por otra parte, si ocupamos el método de codificación basado en “el primero en disparar” presentado en la sección (1.6.1.2), el sistema de decodificación es mucho más eficiente, ya que no se necesita realizar ninguna operación para determinar cuál fue la ganadora. En todos los casos, el primer pulso de la neurona “ganadora” apareció posicionado antes que los primeros pulsos de las neuronas “perdedoras”.

4.1.3 Software de aprendizaje y reconocimiento selectivo

El código fuente usado para implementar el algoritmo de aprendizaje selectivo y de reconocimiento para una red neuronal de tercera generación, como se describe en el capítulo 2, se encuentra etiquetado como “r_5x5_STDP_selectivo.m”. Dicho software, describe una red neuronal que posee 25 neuronas en la capa de entrada y 12 en la capa de salida, con capacidad de recibir caracteres de 25 pixeles y realizar reconocimiento de hasta12 patrones, para lo cual, la red implementada está formada por 37 neuronas y 300 pesos sinápticos. Dado que el aprendizaje es selectivo, los caracteres fueron sometidos a un entrenamiento en el cual las neuronas post sinápticas aprendían en un orden ascendente, con lo cual, el carácter 1 se asociaba con la neurona post sináptica “1” y así sucesivamente hasta que terminara el entrenamiento de la última neurona de salida. En cuanto a los valores de la ventana de aprendizaje de STDP, la forma de vincular la corriente a las neuronas de la capa de entrada, el número de épocas y la velocidad de aprendizaje fueron los mismos que para la red de aprendizaje auto-organizada, variando únicamente la ventana de trabajo, la cual fue en este caso de 100 ms.

4.1.3.1 Proceso de aprendizaje selectivo.

En el proceso de aprendizaje, dado que nuestra red tiene capacidad para reconocer 12 caracteres, seleccionamos los mostrados en la figura (4.8).

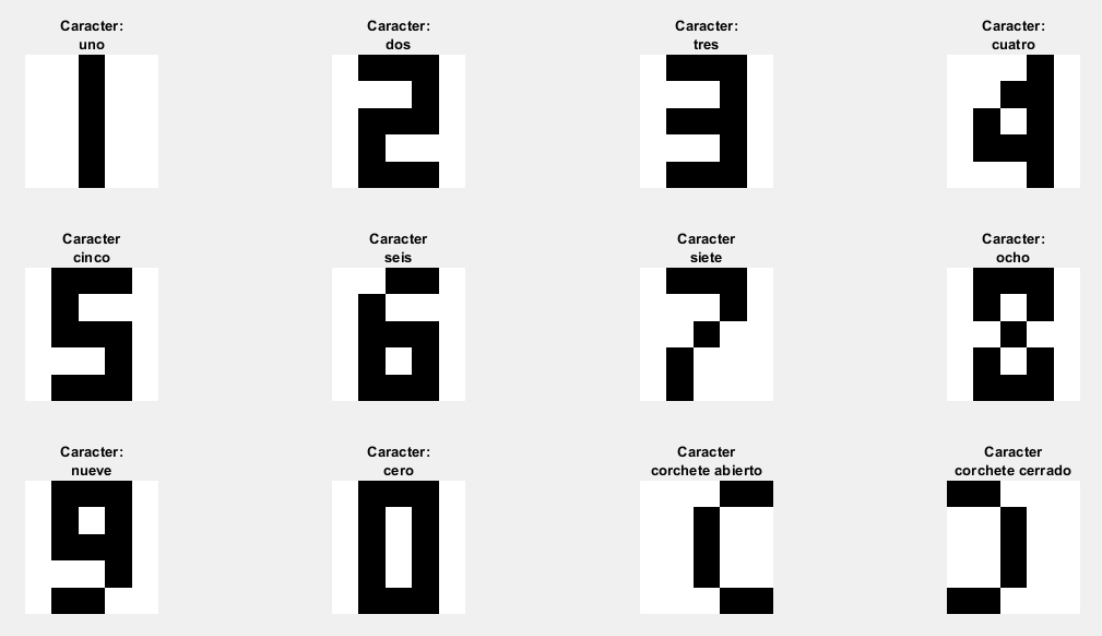

Fig. 4.8. Caracteres a aprender y reconocer por la red.

Al igual que el proceso de aprendizaje auto-organizado, en la época número "1", los pesos sinápticos "w" son asignados de manera aleatoria y, en la época final, se obtienen los pesos sinápticos finales que se fueron modificando durante el proceso de entrenamiento. En las figuras (4.9)-(4.20), se observan en escala de grises los pesos sinápticos al inicio y al final del entrenamiento para corroborar la capacidad auto asociativa de la red descrita en la sección (2.10) del capítulo 2 para cada neurona de salida. El blanco corresponde al valor 1, y el negro al valor 0 del peso sináptico.

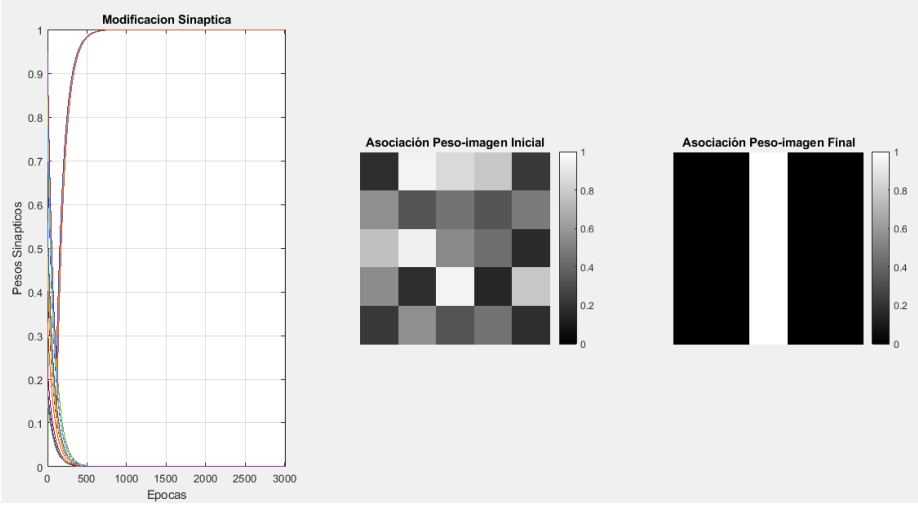

Fig. 4.9. Evolución de los pesos sinápticos asociados al caracter "1" durante el aprendizaje.

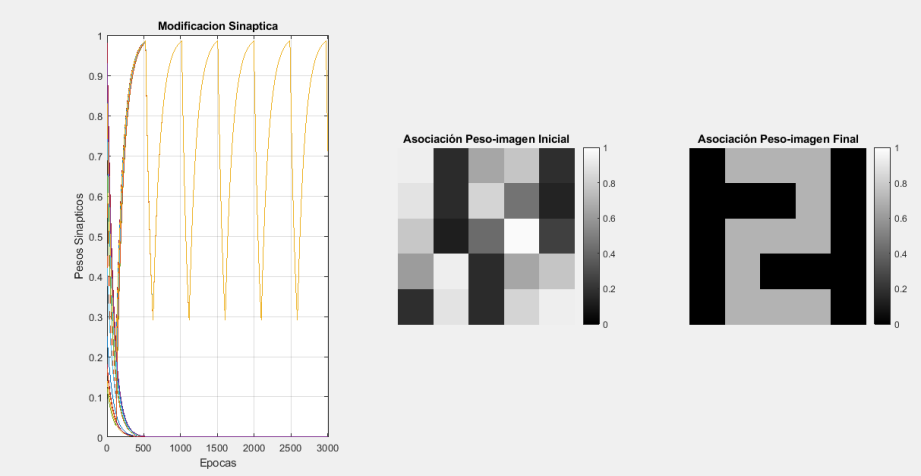

Fig. 4.10. Evolución de los pesos sinápticos asociados al caracter "2" durante el aprendizaje.

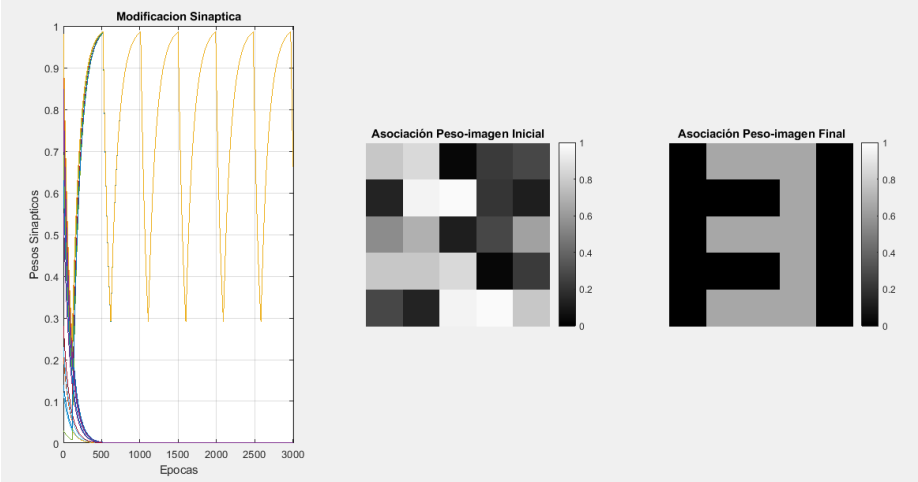

Fig. 4.11. Evolución de los pesos sinápticos asociados al caracter "3" durante el aprendizaje.

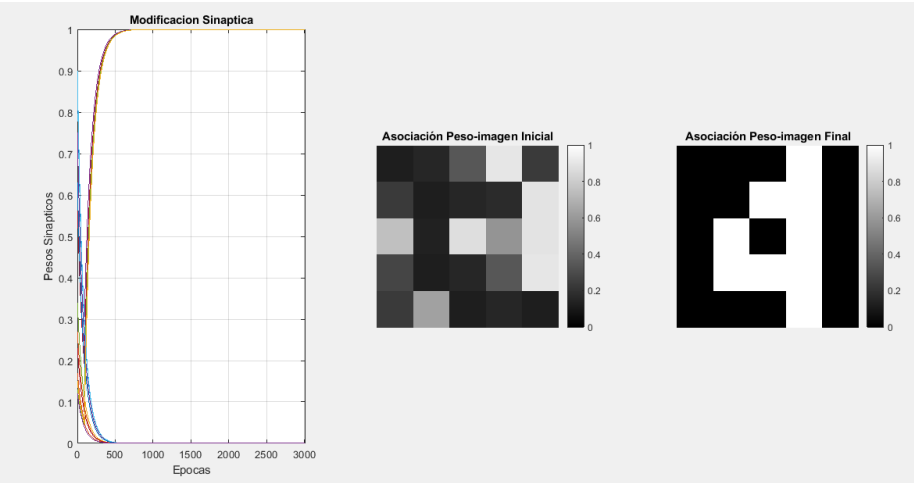

Fig. 4.12. Evolución de los pesos sinápticos asociados al caracter "4" durante el aprendizaje.

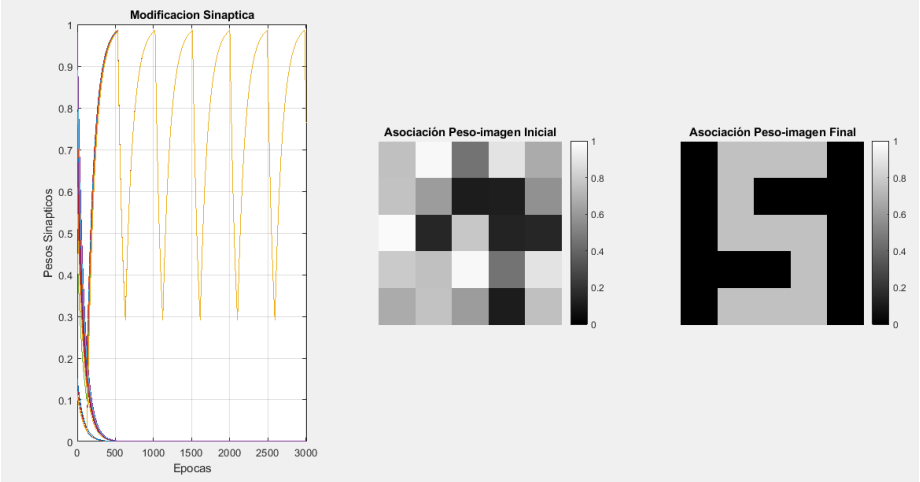

Fig. 4.13. Evolución de los pesos sinápticos asociados al caracter "5" durante el aprendizaje.

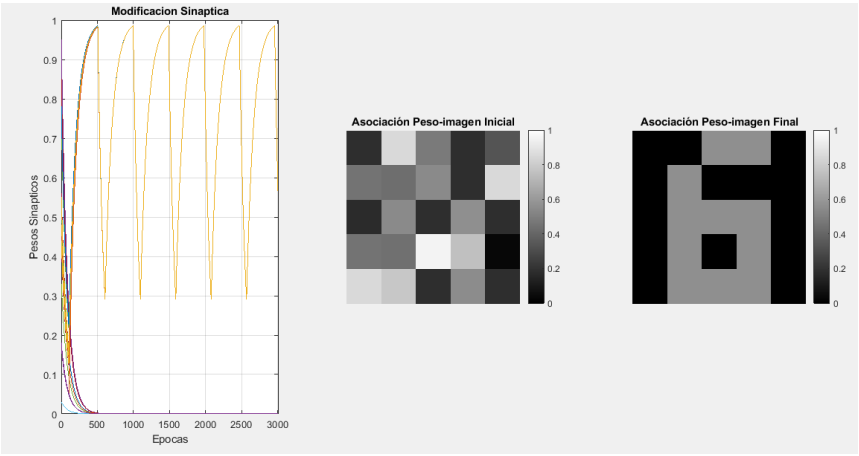

Fig. 4.14. Evolución de los pesos sinápticos asociados al caracter "6" durante el aprendizaje.

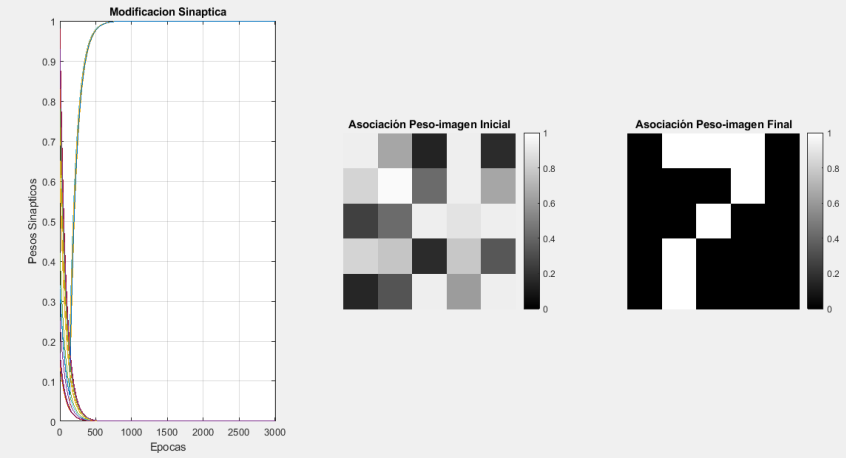

Fig. 4.15. Evolución de los pesos sinápticos asociados al caracter "7" durante el aprendizaje.

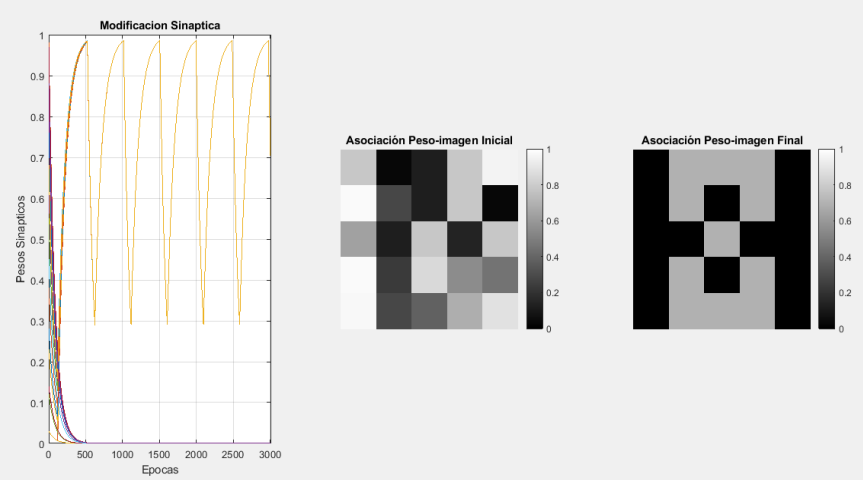

Fig. 4.16. Evolución de los pesos sinápticos asociados al caracter "8" durante el aprendizaje.

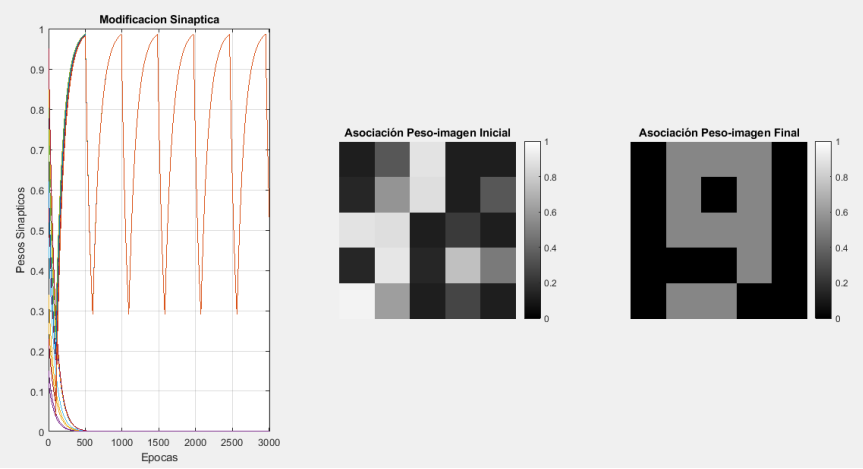

Fig. 4.17. Evolución de los pesos sinápticos asociados al caracter "9" durante el aprendizaje.

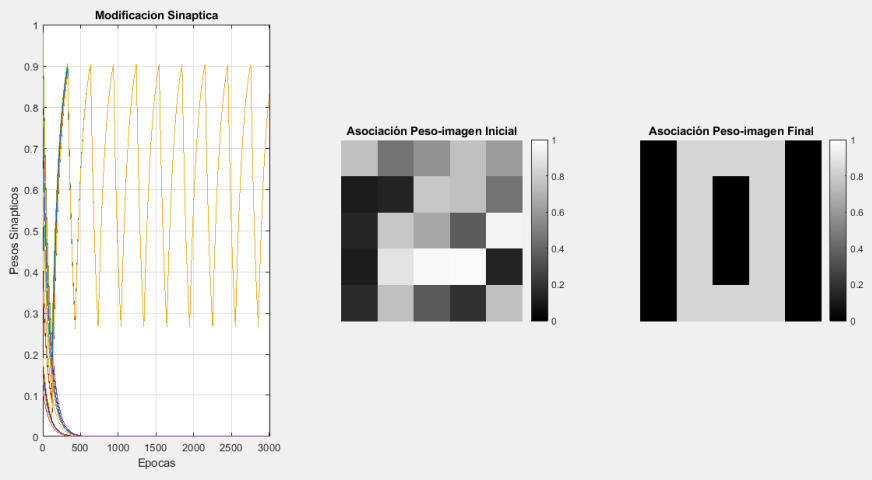

Fig. 4.18. Evolución de los pesos sinápticos asociados al caracter "0" durante el aprendizaje.

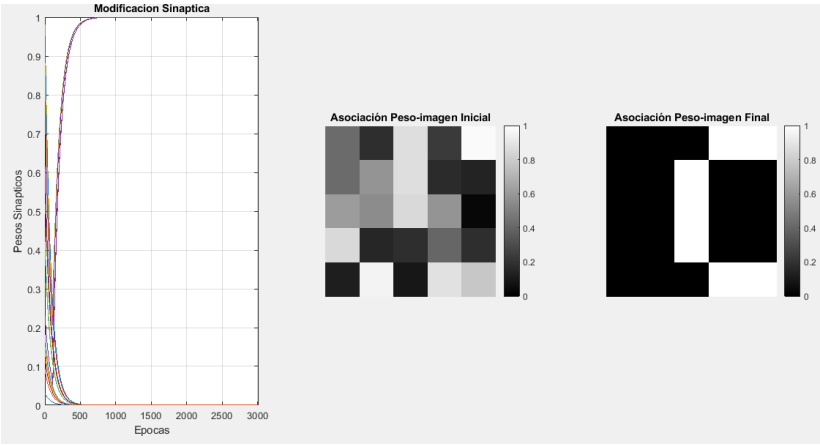

Fig. 4.19. Evolución de los pesos sinápticos asociados al caracter "(" durante el aprendizaje.

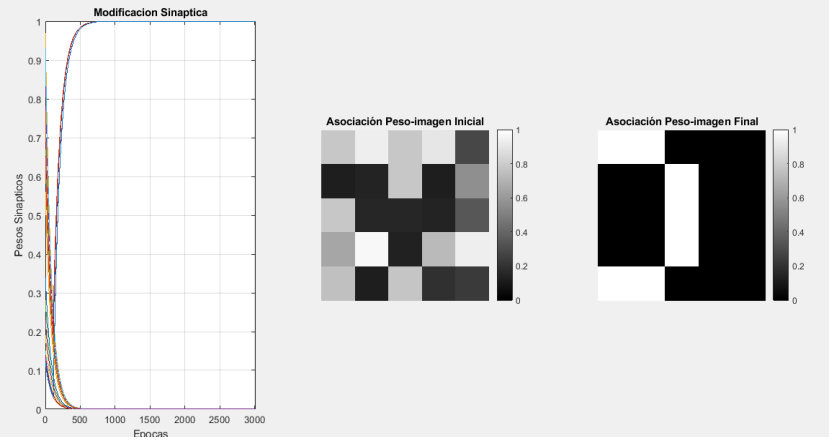

Fig. 4.20. Evolución de los pesos sinápticos asociados al caracter ")" durante el aprendizaje.

Adicionalmente, el programa genera una matriz tipo "string" etiquetada como "resultado", donde se resume qué neurona post sináptica se asoció a cada

patrón y que permite conocer el valor de corriente post sináptica excitatoria (EPSC) de cada una de las neuronas post sinápticas cuando reconocen el caracter; tales resultados se muestran en la tabla (4.3).

Neurona Post sináptica	Carácter aprendido	EPSC [pA]
1	"1"	5
2	"2"	7.73398
3	"3"	7.1937
4	"4"	9
5	"5"	8.31483
6	"6"	6.14908
7	"7"	7
8	"8"	7.6412
9	"9"	5.78897
10	"10"	10.012
11	"corchete abierto"	7
12	"corchete cerrado"	7

Tabla 4.3. Resultados obtenidos del algoritmo de entrenamiento.

Obsérvese que en este caso, dado que las corrientes finales debidas a los pesos sinápticos, en ciertos casos es la misma, la red fallará indudablemente en el reconocimiento de los caracteres “7”, "corchete abierto" y "corchete cerrado".

4.1.3.2 Proceso de reconocimiento.

El proceso de reconocimiento ocurre solamente presentando los caracteres con los que se entrenó previamente a la red neuronal al igual que como sucede en el proceso de reconocimiento de la red de la sección (4.4.2), recordando que el orden de entrenamiento es la única diferencia. El reconocimiento realizado en función del potencial de membrana de cada neurona sináptica individual, se presenta en las figuras (4.21)-(4.29). Cabe mencionar que, dado que la corriente post sináptica que se usa para el discernimiento entre los caracteres “7”, "corchete abierto" y "corchete cerrado" es la misma, no se puede determinar qué neurona fue la “ganadora” de acuerdo al método

mostrado, ya que los potenciales de acción generados ocurren en los mismos instantes de tiempo y por ende no podemos determinar el valor de las sinapsis inhibitorias que afectarán.

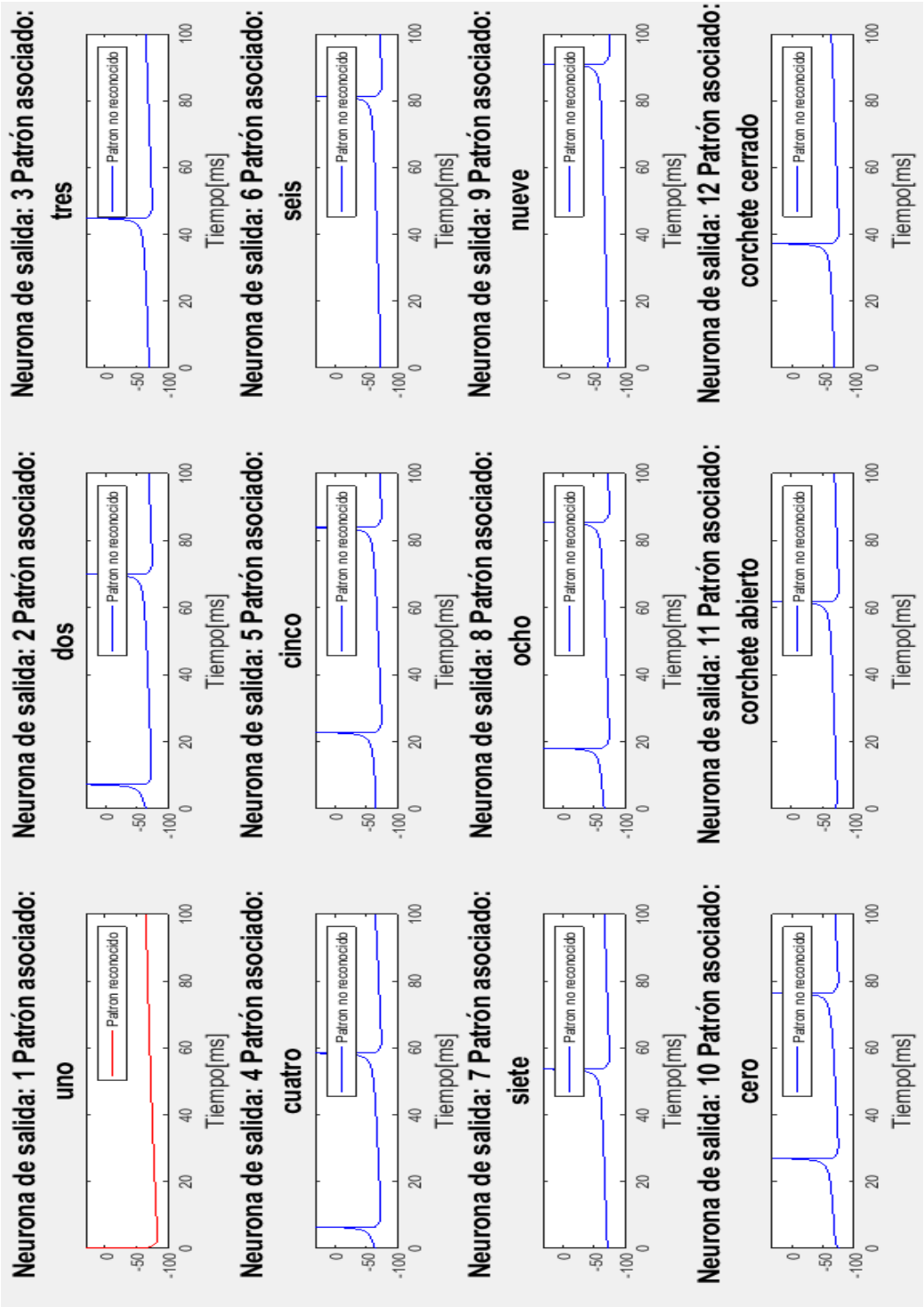

Fig. 4.21. Etapa de reconocimiento. Caracter "1".

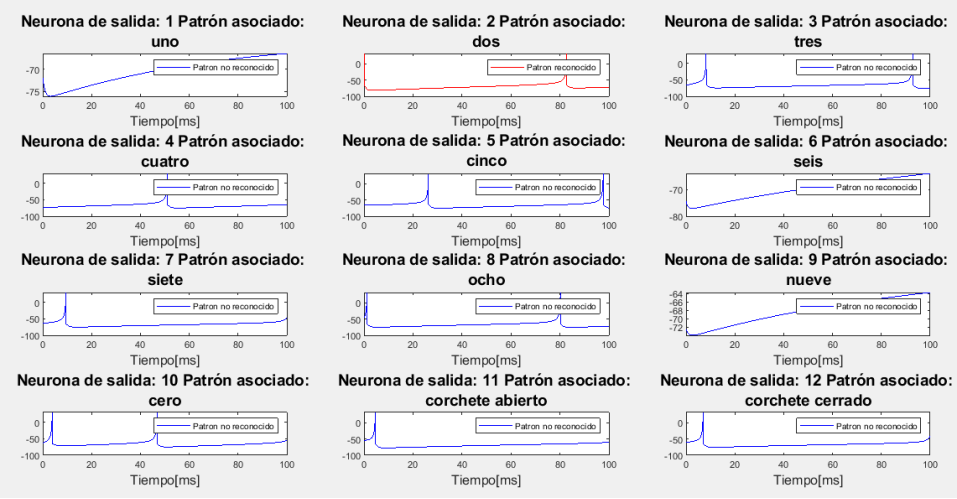

Fig. 4.22. Etapa de reconocimiento. Caracter "2".

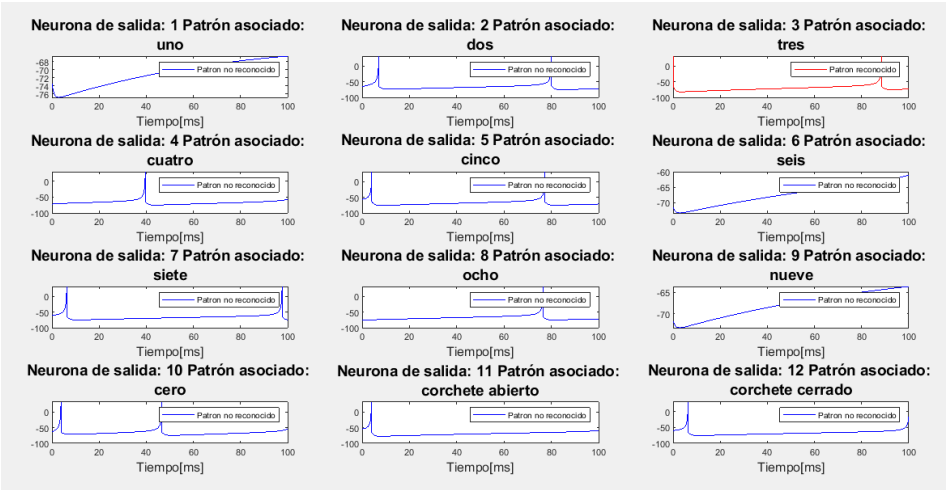

Fig. 4.23. Etapa de reconocimiento. Caracter "3".

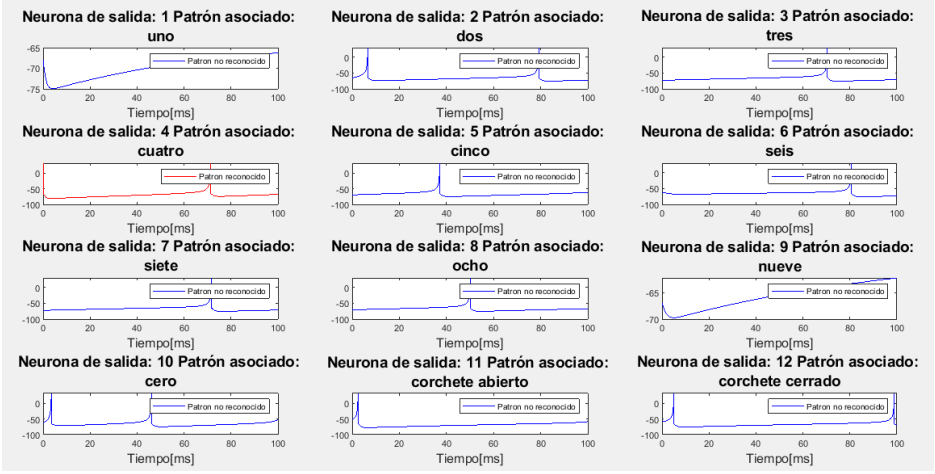

Fig. 4.24. Etapa de reconocimiento. Caracter "4".

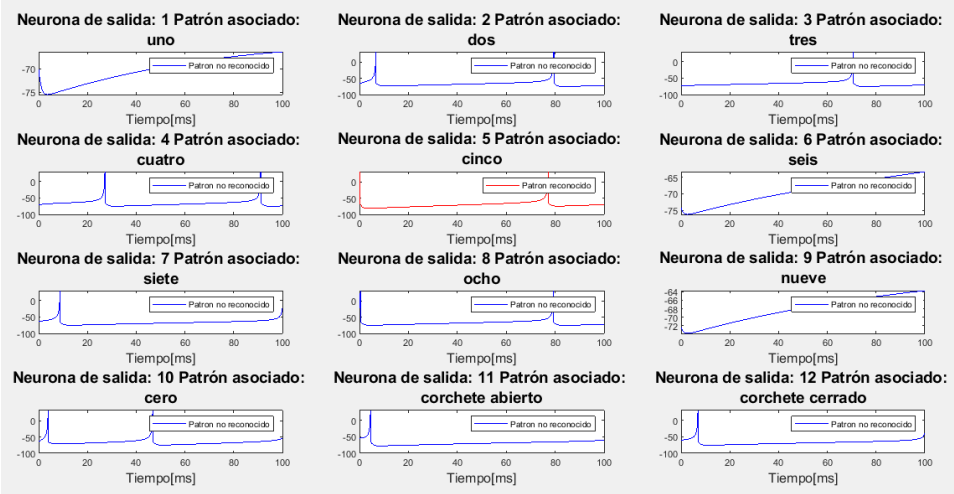

Fig. 4.25. Etapa de reconocimiento. Caracter "5".

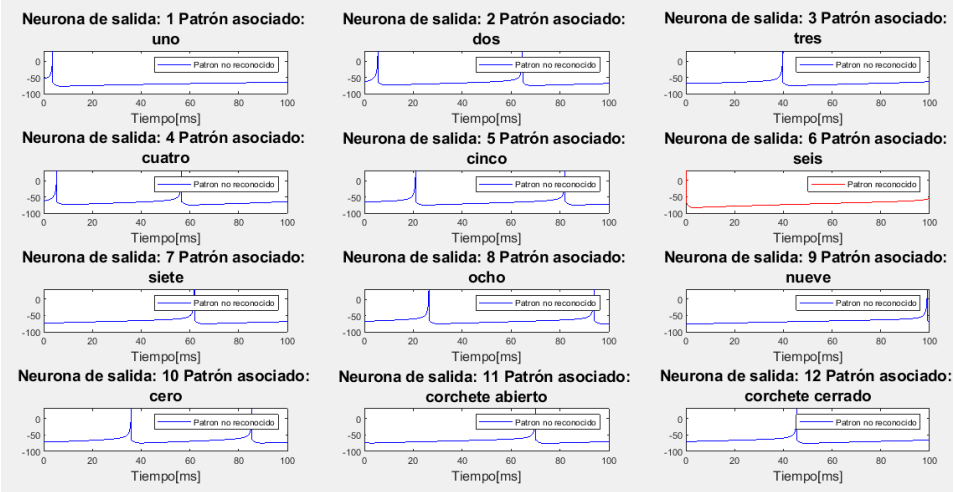

Fig. 4.26. Etapa de reconocimiento. Caracter "6".

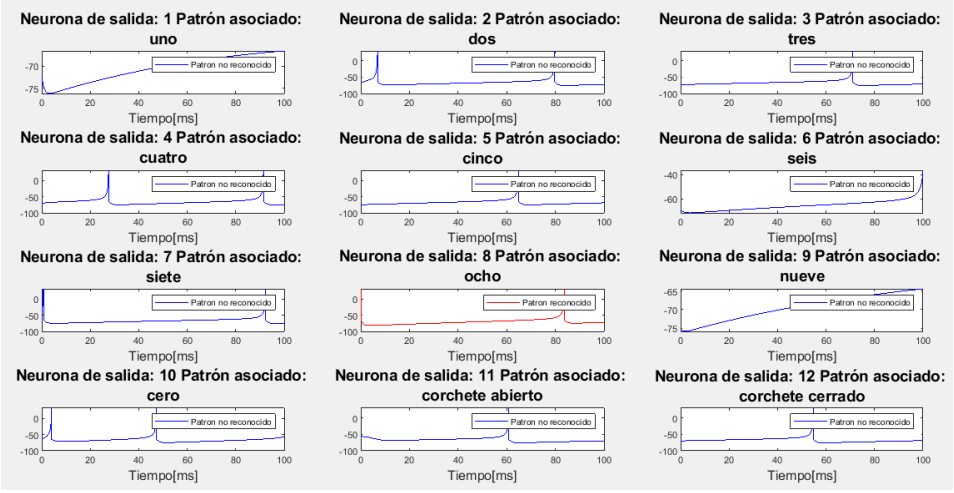

Fig. 4.27. Etapa de reconocimiento. Caracter "8".

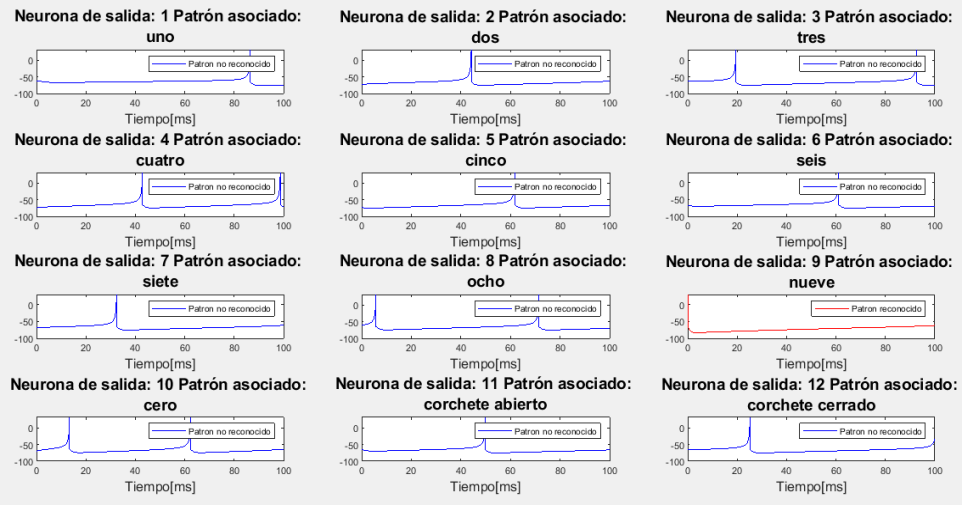

Fig. 4.28. Etapa de reconocimiento. Caracter "9".

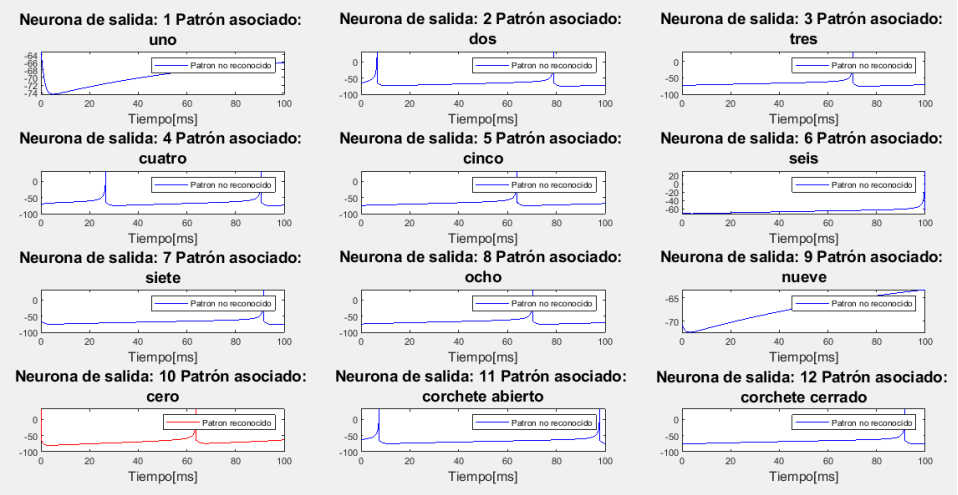

Fig. 4.29. Etapa de reconocimiento. Caracter "0".

Como podemos observar, en este caso para determinar una frecuencia de operación en etapa de reconocimiento y determinar así la neurona "ganadora" (como lo hicimos en la sección (4.1.2.2)), se generan mayores problemas para su discernimiento en función de la actividad del potencial de membrana, ya que hay más coincidencias en el número total de "spikes" dentro de la ventana, por lo que su posición en el tiempo tiene una mayor relevancia que la cantidad de ellos. Por consiguiente, basados en el esquema de codificación por pulsos "el primero en disparar", podemos discernir a la "neurona ganadora" de las "perdedoras" como aquella que pulsó primero y que en el tiempo se mantiene pulsando, o bien, si existe alguna semejanza entre las secuencias generadas,

aquella donde los pulsos que la forman aparecen antes. En este caso la red tuvo una capacidad de reconocimiento del 75%.

4.1.4 Conclusiones Software

La parte fundamental del aprendizaje dirigido de STDP es el protocolo de asociación de pulsos y la ventana de aprendizaje. En su conjunto, proporcionan un sistema para determinar la diferencia temporal entre los pulsos pre y post sinápticos como una sumatoria de cambios sinápticos parciales de todos los pares formados, que una vez integrados generan un solo cambio del peso sináptico total, el cual, una vez procesado a través de la regla de relajación permite la asignación de peso sináptico en cada época y, generando cambios de corriente post sináptica, retroalimentando a todo el sistema.

El método de entrenamiento antes descrito es satisfactorio en redes neuronales de distintas dimensiones, independientemente de la forma del aprendizaje ya sea “auto-organizado” o “selectivo”, siempre y cuando la proporción de la región LTD sea mayor que la región LTP como sugiere la ecuación (1.23) de la sección (1.5). Sin embargo, el aprendizaje no supervisado dirigido en STDP en software, dado el entorno computacional en el que se desarrolla, tiene desventajas al tener que trabajar con ventanas de tiempo, lo que implica que la convergencia del aprendizaje de STDP es dependiente de las dimensiones, arquitectura, tipo de neuronas y modelo sináptico que usa. Lo anterior es principalmente porque dependiendo de la amplitud temporal de la ventana de tiempo seleccionada, se determina la cantidad de pulsos pre y post sinápticos que participan tanto en el aprendizaje dirigido por STDP como los que se integran en sinapsis. Esto se ve reflejado en los experimentos realizados en las redes implementadas en software que, a pesar de tener los mismos parámetros de la ventana de aprendizaje, protocolo de asociación, velocidad de aprendizaje y sinapsis, requirieron de longitudes diferentes en la ventana de tiempo para lograr la convergencia del aprendizaje.

Cabe mencionar, que a pesar de que exista convergencia del aprendizaje no supervisado (el cual se ve reflejado en los pesos sinápticos a los que llega a la red) no garantiza la correcta clasificación de los caracteres, debido principalmente a que la red no contiene retardos, los cuales al no estar presentes, independientemente del pixel que inyecte la corriente a las neuronas de entrada, al ser la misma, siempre se provocarán que éstas tengan el mismo patrón de generación de pulsos, afectando de la misma forma la sinapsis, tal como se menciona en la sección (1.4.6) del capítulo 1. En este caso, una red con la misma frecuencia de operación disminuye su capacidad para disociar la información, ya que, si los potenciales de acción generados ocurren en los mismos instantes de tiempo, y solo varía la componente espacial dada por “su posición en la red”, provoca problemas en etapa de reconocimiento como el presentado en la sección (4.1.3.2).

En cuanto a la etapa de reconocimiento, en el método para determinar la neurona ganadora, dada la secuencialidad de la plataforma, siempre será necesario conocer previamente la corriente post sináptica total de cada una de las neuronas de salida (con la cual incluso ya podríamos diferenciar los caracteres) y así determinar la cantidad de potenciales que genera cada una de ellas y con ello aplicar un método para extraer la información de la red.

El método más apropiado para determinar la neurona ganadora de una red con las características mostradas y en función de la dinámica del potencial de membrana, es el método basado en pulsos, que es aplicable a cualquiera de los procesos de reconocimiento realizados, ya que en éste, la información espacio-temporal de los pulsos post sinápticos es más importante. Lugar y tiempo son más efectivos que un conteo de pulsos, que en varias situaciones puede generar el mismo resultado.

En cuanto el procesamiento espacio-temporal de la información, la información temporal se mantiene únicamente en el sentido numérico, y la espacial, en el sentido que sabemos qué neurona o neuronas están afectando los procesos de sinapsis y aprendizaje; en dicha situación, los cambios en corriente y de

modificación sináptica no se dan hasta que se da el inicio de una nueva época, por lo que el tiempo en que aparecen los pulsos en dichos procesos, no interviene de manera instantánea en la ventana de tiempo en la cual se está procesando, sino en una posterior.

Finalmente, cabe mencionar que el método de asociación de pulsos “todos contra todos” y al esquema de trabajo por “ventanas de tiempo” que estamos usando, se puede hacer gracias a la plataforma que estamos usando, ya que nos permite guardar las secuencias de los pulsos pre y post sinápticos, las cuales varían de acuerdo a la cantidad de corriente que se da entre época y época de entrenamiento. Así, migrar tal cual este esquema a FPGA simplemente no es viable, al menos con las características usadas, porque los recursos de memoria en cierto punto independientemente de ser grandes, conforme incrementan las dimensiones de la red no pueden ser estimables (refiriéndonos a la cantidad de pulsos post sinápticos que deben ser almacenados en memoria para procesar posteriormente). En efecto, los pulsos post sinápticos son dependientes de una corriente en particular y sus variaciones, por lo que la dinámica neuronal afecta la cantidad de registros de memoria que el sistema podría llegar a manejar.

4.2 Resultados en Hardware

4.2.1 Introducción

El sistema descrito en VHDL para implementar el reconocimiento de caracteres en FPGA, se basa en una arquitectura neuromórfica para realizar entrenamiento y reconocimiento no supervisado, con capacidad de procesamiento espacio-temporal en tiempo real a diferencia de su implementación en software. El sistema de reconocimiento de caracteres consta de una red neuronal para realizar aprendizaje selectivo de caracteres de 36 pixeles (6x6) con retardos y reconocimiento de hasta 10 caracteres, por lo cual, la red implementada está formada por 46 neuronas y 360 pesos sinápticos.

A largo de esta sección se muestran cada una de las etapas que dirigen el control primario y secundario y que permiten identificar cada una de las etapas de operación del sistema como tal. En este caso se ocupa el banco de pruebas etiquetado como “SNN_complete_tb”, para realizar la simulación temporizada.

Se concluye la sección con la cantidad de recursos necesarios para implementar la red en FPGA así como los módulos principales que intervienen en sus procesos.

Finalmente, los programas correspondientes a la implementación en hardware se encuentran en el apéndice disponible en la dirección http://www.vlsilab.cinvestav.mx/tesis_fgc.html del Laboratorio VLSI CINVESTAV; en la sección (4.2.6) se encuentra una guía para facilitar la comprensión y organización de los mencionados.

4.2.2 Carga de patrones

Durante esta etapa, se llena la memoria RAM del módulo etiquetado como “Main_Control” con los caracteres a usar en los procesos de entrenamiento y reconocimiento. En este caso, los caracteres a procesar se muestran en la figura (4.30).

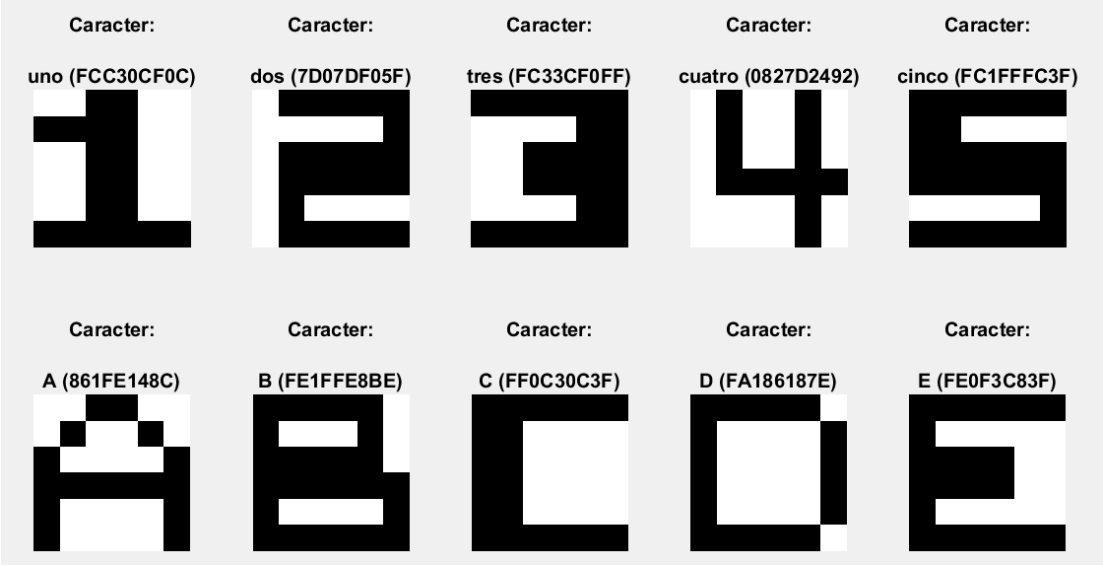

Fig. 4.30. Caracteres a aprender y reconocer por la red neuronal en FPGA. En los paréntesis se encuentra la representación binaria de los caracteres en formato hexadecimal.

La operación de carga se ilustra en la figura (4.31) a través de la variable “memory_pattern”.

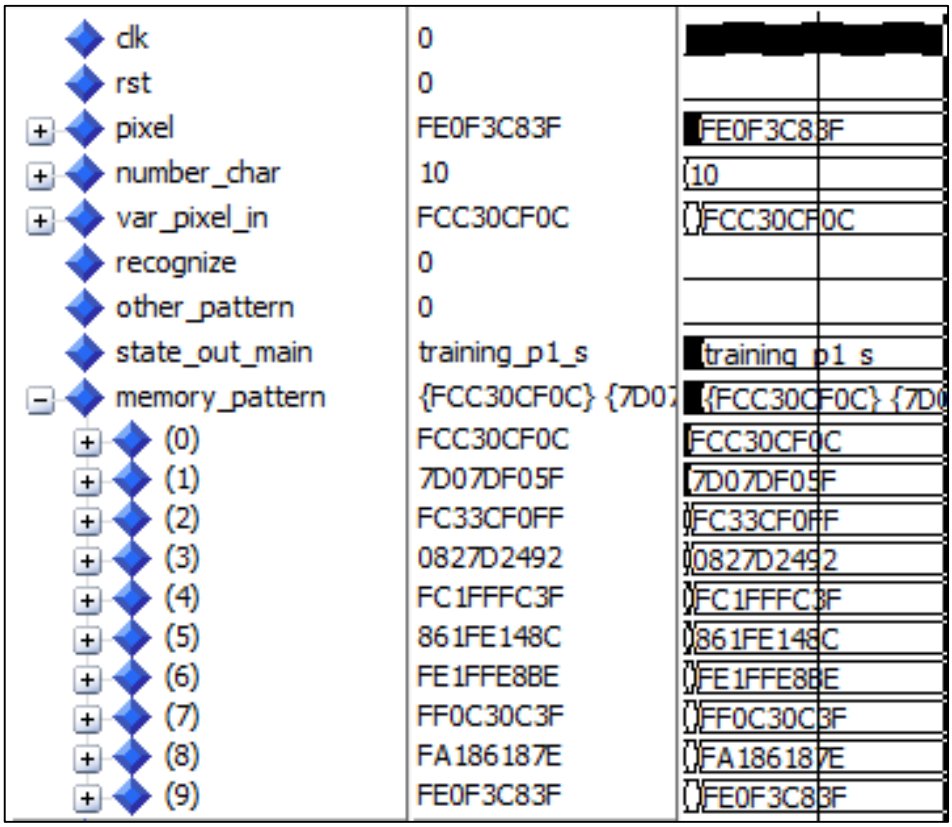

Fig. 4.31.Carga de caracteres en la red.

4.2.3 Etapa de entrenamiento

En la etapa de entrenamiento se lleva a cabo la modificación de los pesos sinápticos inducidos por el protocolo de STDP, así como el proceso de sinapsis para dar lugar a la corriente post sináptica, por lo que, para llevarlo a cabo, son necesarios los procesos de integración de baja corriente y aprendizaje.

4.2.3.1 Integración de baja corriente

En la etapa de integración de baja corriente, se llega a un valor de corriente umbral necesario para iniciar una fase de aprendizaje, lo cual es logrado a través de la arquitectura neuromórfica que permite el procesamiento por eventos; en este caso, cada vez que se genera un evento de pulso pre sináptico, se genera un evento de sinapsis en tiempo real, hasta superar unacorriente umbral, tal como se muestra en la figura (4.32). A través del

puerto “spike_from_aer” se puede ver el tren de pulsos generado por la contribución de todas las neuronas pre sinápticas, en el puerto “spike_neuron_x” la respectiva salida de cada neurona post sináptica. La señal “I_in_NX” nos permite observar la modificación de corriente post sináptica asociada a cada neurona de la capa de salida, la variable “memory” las variaciones de los 36 pesos sináptico en la memoria RAM, la acción de “control primario y secundario” a través de las variables “state_out_main” y “state_out_pat_x”, mientras que el carácter con el que se está realizando el reconocimiento o entrenamiento por la red a través de la variable “var_pixel_in”.

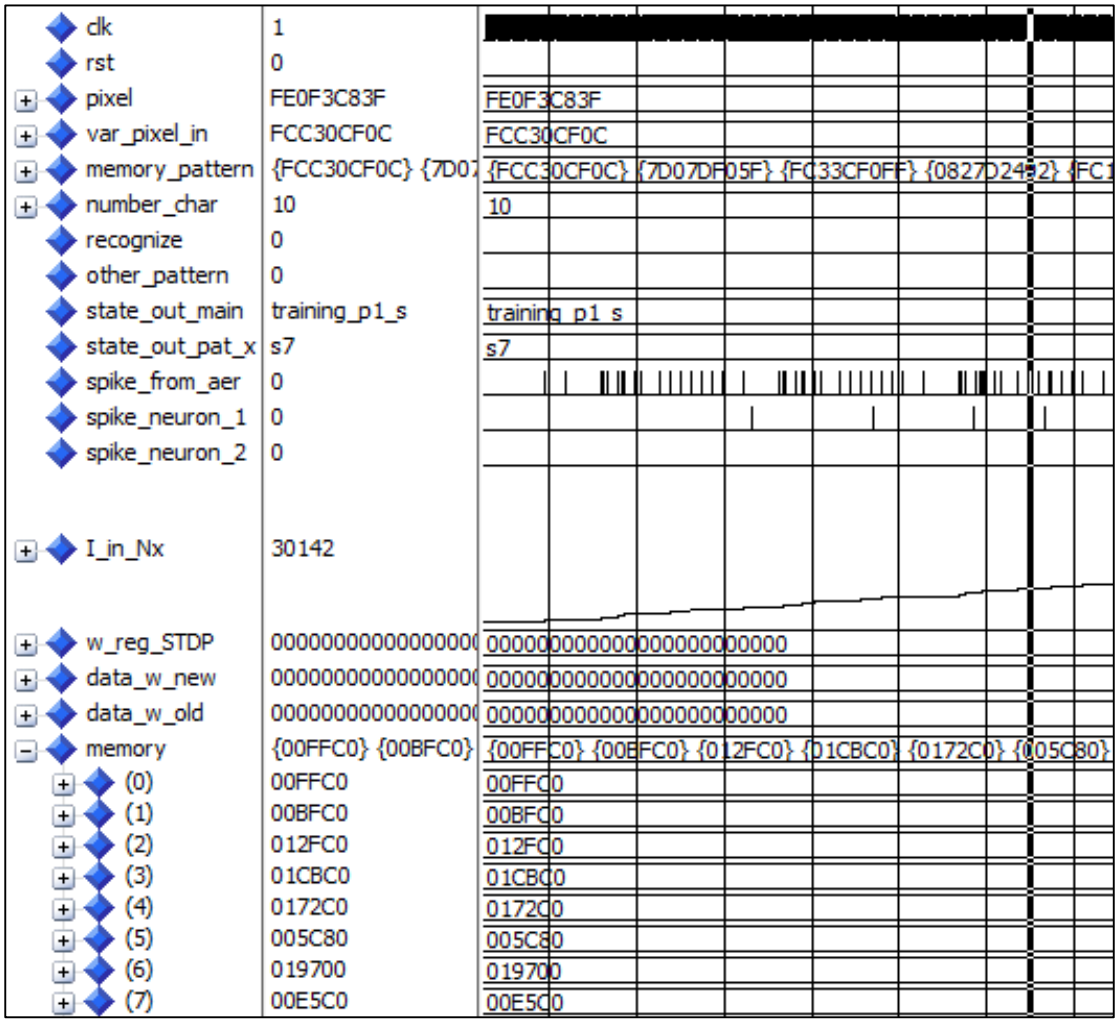

Fig. 4.32. Etapa de integración de baja corriente en FPGA.

4.2.3.2 Aprendizaje.

En las figuras (4.33) y (4.34) se muestra la operación de la arquitectura neuromórfica durante el entrenamiento para una sola neurona en FPGA; en este caso, cuando se inserta en el sistema el carácter a entrenar, cada vez que se genera un evento de asociación de pulsos, se genera un evento de diferencia temporal, que a su vez genera un evento de modificación sináptica, que a su vez genera un evento de asignación de memoria para que finalmente se genere un evento de sinapsis en tiempo real que termina por afectar la generación de potenciales de acción de la neurona post sináptica. En este caso las variables “w_reg_STDP”, “data_w_old” y “data_w_new” son variables que nos facilitan la observación de dicha operación.

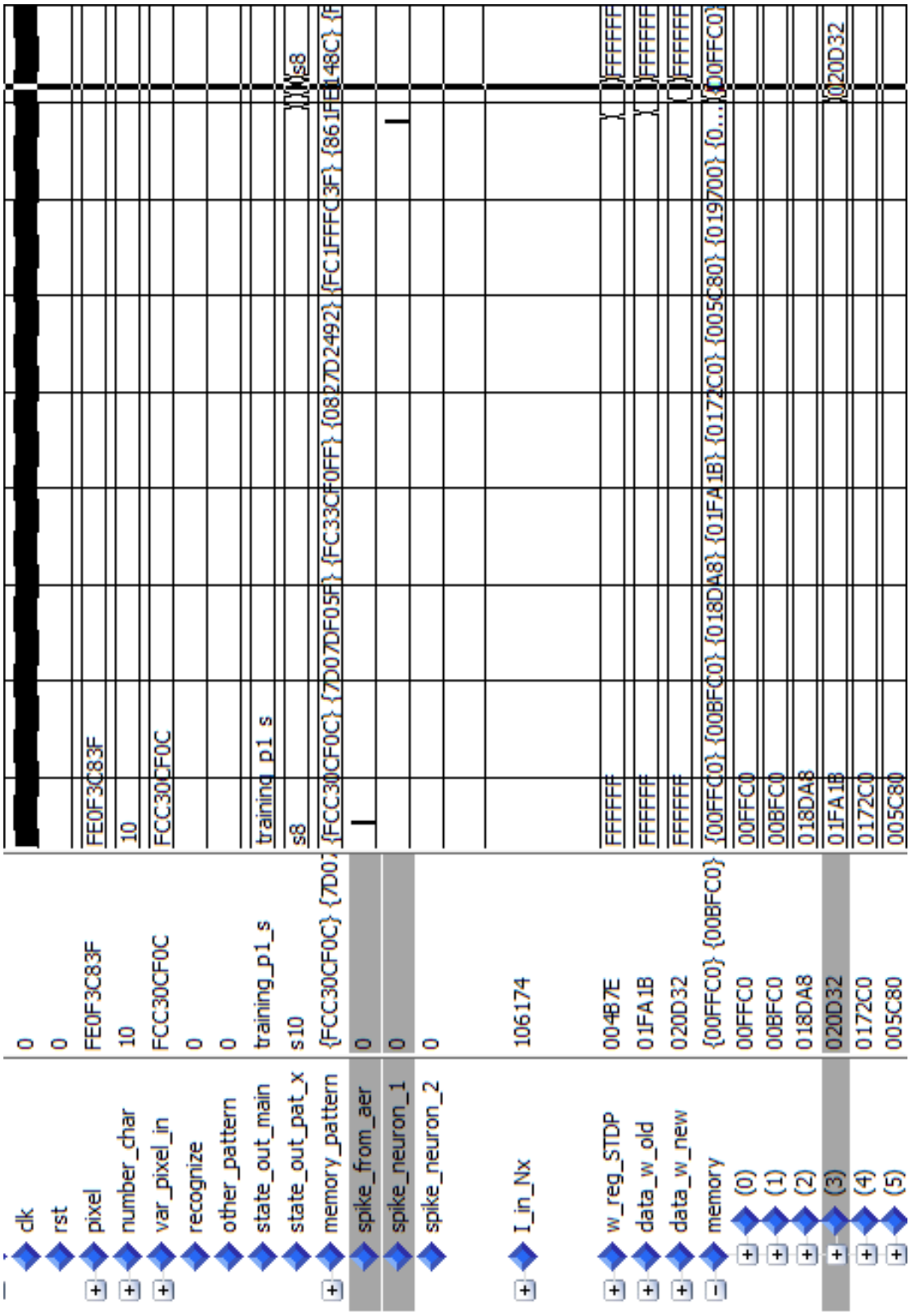

Fig. 4.33. Procesamiento por eventos de la arquitectura neuromórfica. La variable "w_reg_STDP" contiene el cambio del peso sináptico después de haberse formado un par de pulsos; el sistema obtiene el peso sináptico anterior referente a la dirección donde se ubica la sinapsis para depositarlo en la variable "data_w_old", y con ello posteriormente modificar el peso sináptico en la RAM con el valor obtenido de la variable "data_w_new".

Fig. 4.34.Proceso de entrenamiento en FPGA para un solo caracter, modificación de las primeras posiciones de los pesos sinápticos.

Todo esto llevándose a cabo hasta que se supere la cantidad de pares de pulsos potenciados necesarios (100 pares) para determinar que la red ha aprendido el carácter. Cabe mencionar, que durante este proceso se observó que la inducción del aprendizaje en cada neurona post sináptica se logró a través de corrientes post sinápticas relativamente elevadas en comparación con las corrientes con las que se polarizaron las neuronas de entrada, y que a diferencia de la corriente de estas últimas, la corriente post sináptica total en cada neurona de salida fue resultado de las modificaciones sinápticas que se encontraban potenciadas a lo largo del tiempo, y que cuando fueron integradas se llegaron a valores de corriente de 500 pA o más. En cuanto a la duración del entrenamiento de cada neurona post sináptica, éste tuvo una duración de 179,400 ciclos de reloj a partir de que se presentó el primer pulso potenciado. Dado que la frecuencia de operación es de 10 MHz y la unidad numérica que se integra en cada ciclo de reloj es de 0.0078125 ms, se tendrá una duración en cada fase de entrenamiento de aproximadamente de 17.94 ms en tiempo real y numéricamente de 1.401 segundos. Sin embargo, hacemos énfasis de que la duración de la fase de entrenamiento dependerá del proceso de integración de baja corriente, del caracter presentado y de la dinámica neuronal asociada. Por lo tanto, la corriente, así como el tiempo necesario para lograr la inducción del aprendizaje será variable para cada neurona post sináptica entrenada. En las figuras (4.35) y (4.36) se observan los cambios sinápticos de 2 módulos de memoria RAM, la corriente post sináptica asociada a la neurona de salida, el conteo de pares de pulsos potenciados a través de “train_epoch” y la señal “signal_count” proporcionada por el banco de pruebas, que nos muestra la cantidad de pulsos de reloj a partir del primer pulso potenciado, hasta que se concluyen los pares potenciados solicitados. Una vez terminado el proceso de entrenamiento de cada neurona post sináptica, se obtienen valores máximos y mínimos en los pesos sinápticos, que dan lugar a la corriente post sináptica de cada una de ellas, repitiéndose este proceso para cada neurona entrenada como se observa en la figura (4.37).

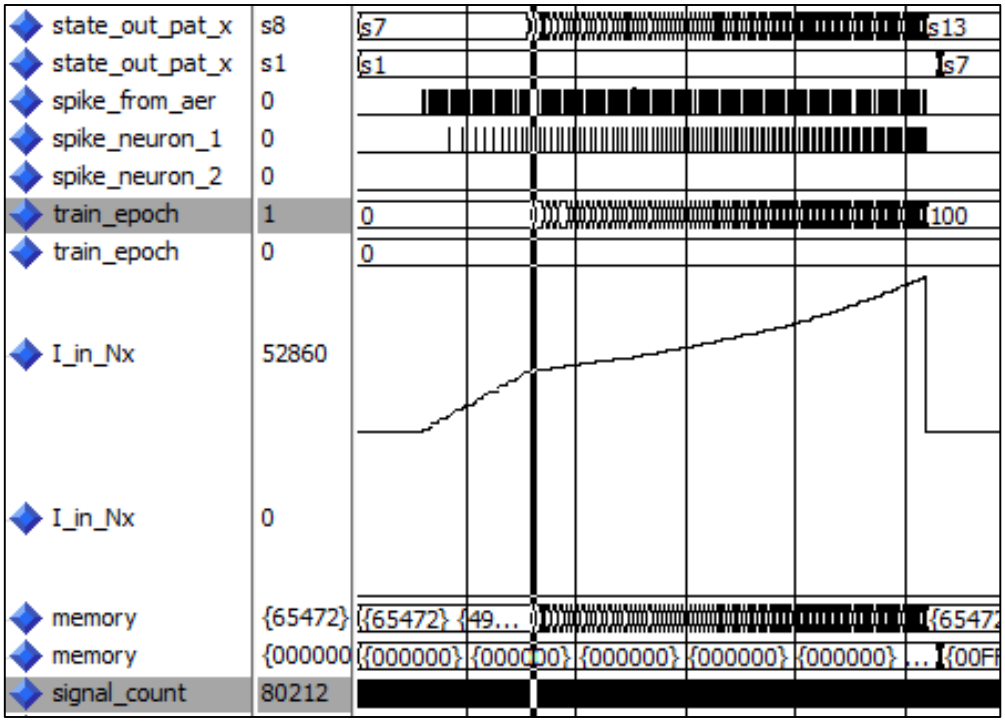

Fig. 4.35. Inducción del aprendizaje por STDP en FPGA. Cursor fijado en la época número uno (train_epoch=1) que corresponde al primer pulso potenciado con la cual comienza la inducción de aprendizaje en la red neuromórfica.

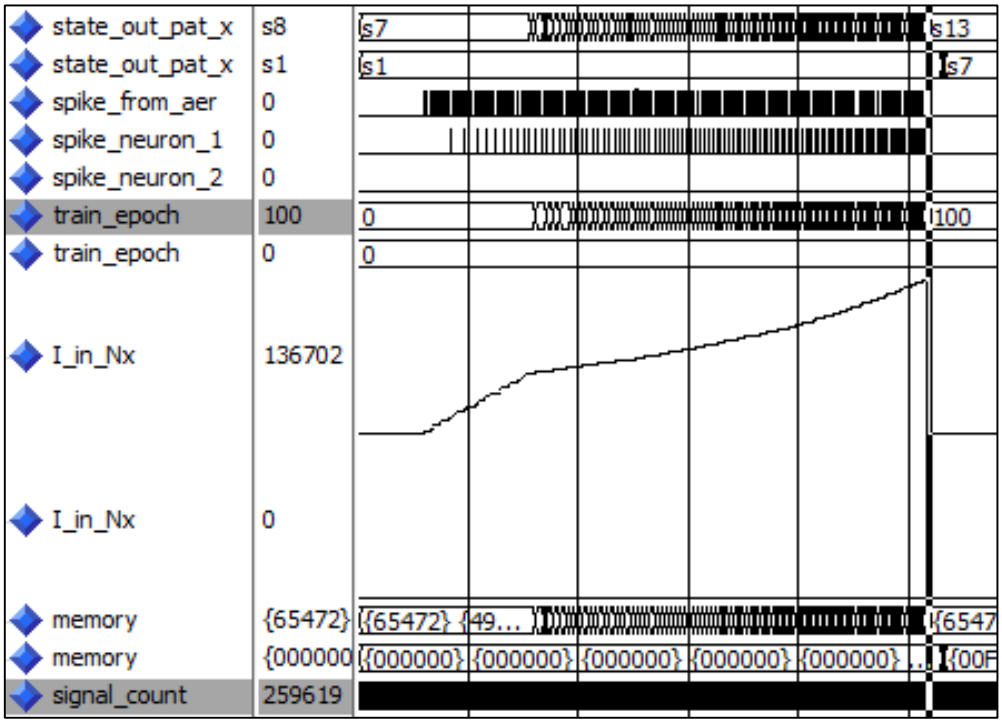

Fig. 4.36 Inducción del aprendizaje por STDP en FPGA. Cursor fijado en la época final designada (train_epoch=100) que corresponde al último pulso potenciado necesario para finalizar la inducción de aprendizaje en la red neuromórfica.

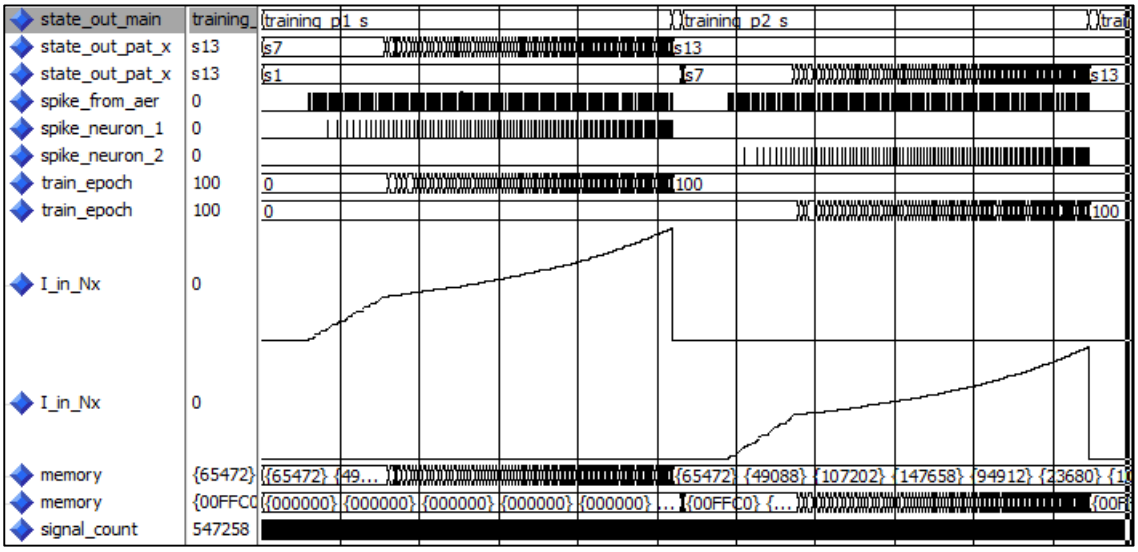

Fig. 4.37.Terminación del entrenamiento entre una neurona y otra por conteo de pares potenciados en FPGA.

En la figura (4.38) se puede observar el orden secuencial de entrenamiento de 10 neuronas para reconocer 10 caracteres; el orden corresponde con la posición de memoria en que se ubica dicho caracter, es decir, la neurona en el registro de memoria uno se asocia con la neurona número uno y así sucesivamente. En cuanto a la duración de dicho proceso, tomando en cuenta solamente las etapas de entrenamiento a partir del primer pulso potenciado, y en el supuesto que la duración de la etapa de inducción de aprendizaje sea aproximadamente la misma y despreciando las etapas de integración que son muy cortas en comparación de las primeras, el proceso de entrenamiento de 10 caracteres con la arquitectura actual, tardaría aproximadamente 179.4 milisegundos en tiempo real y 14 segundos numéricamente.

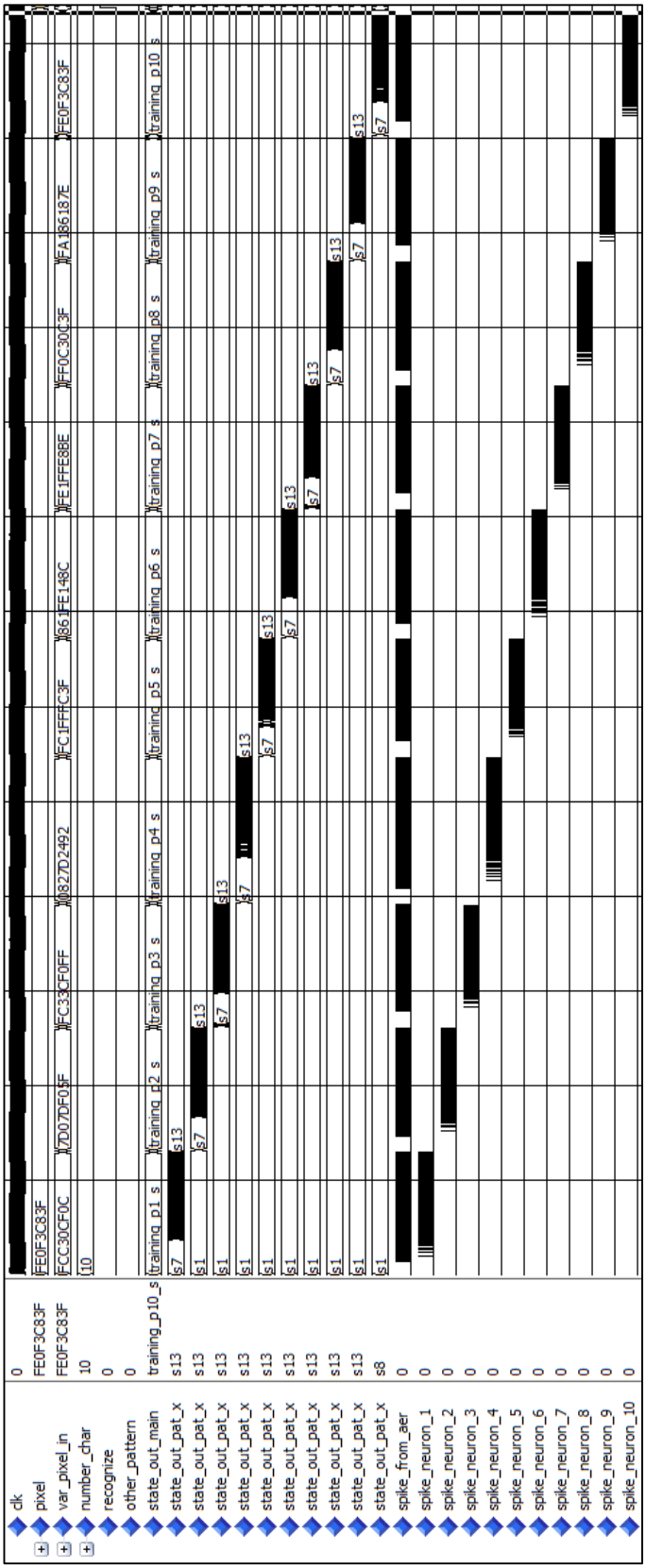

Fig. 4.38.Secuencia de entrenamiento para 10 caracteres.

4.2.4 Etapa de reconocimiento

El proceso de reconocimiento es similar al de la etapa de baja corriente; en este caso, cada vez que existe un pulso pre sináptico se genera un evento de sinapsis con lo cual solamente se integra el valor del peso sináptico obtenido del entrenamiento en dicha dirección. Esto termina una vez que el sistema detecta el primer pulso post sináptico a la salida, tal como se observa en las figuras (4.39)-(4.42), donde la neurona ganadora será siempre la primera en disparar y por lo tanto la que haya reconocido el caracter.

En cuanto a la duración del proceso de reconocimiento, éste tardará solamente unos cuantos nanosegundos ya que el sistema se basa en el primero en disparar y requerirá por lo tanto, un tiempo mínimo para alcanzar una corriente mínima para que la neurona ganadora comience a generar potenciales de acción (valores a partir de 3 pA).

En dicha simulación, cabe mencionar que solamente los caracteres que participaron en el proceso de entrenamiento fueron reconocidos correctamente y en los casos donde los caracteres no participaron en dicho proceso, el sistema falló en realizar el reconocimiento (caracteres desconocidos (FFF…FF) y (801801C00) en las figuras (4.39) y (4.42) respectivamente). En este sentido, el sistema propuesto de reconocimiento de caracteres cuando se seleccionaron 10 caracteres, con retardos de propagación y de retro propagación en la red, no se tuvieron inconvenientes para clasificar los caracteres previamente entrenados En este caso, la red tuvo una capacidad de reconocimiento del 100%.

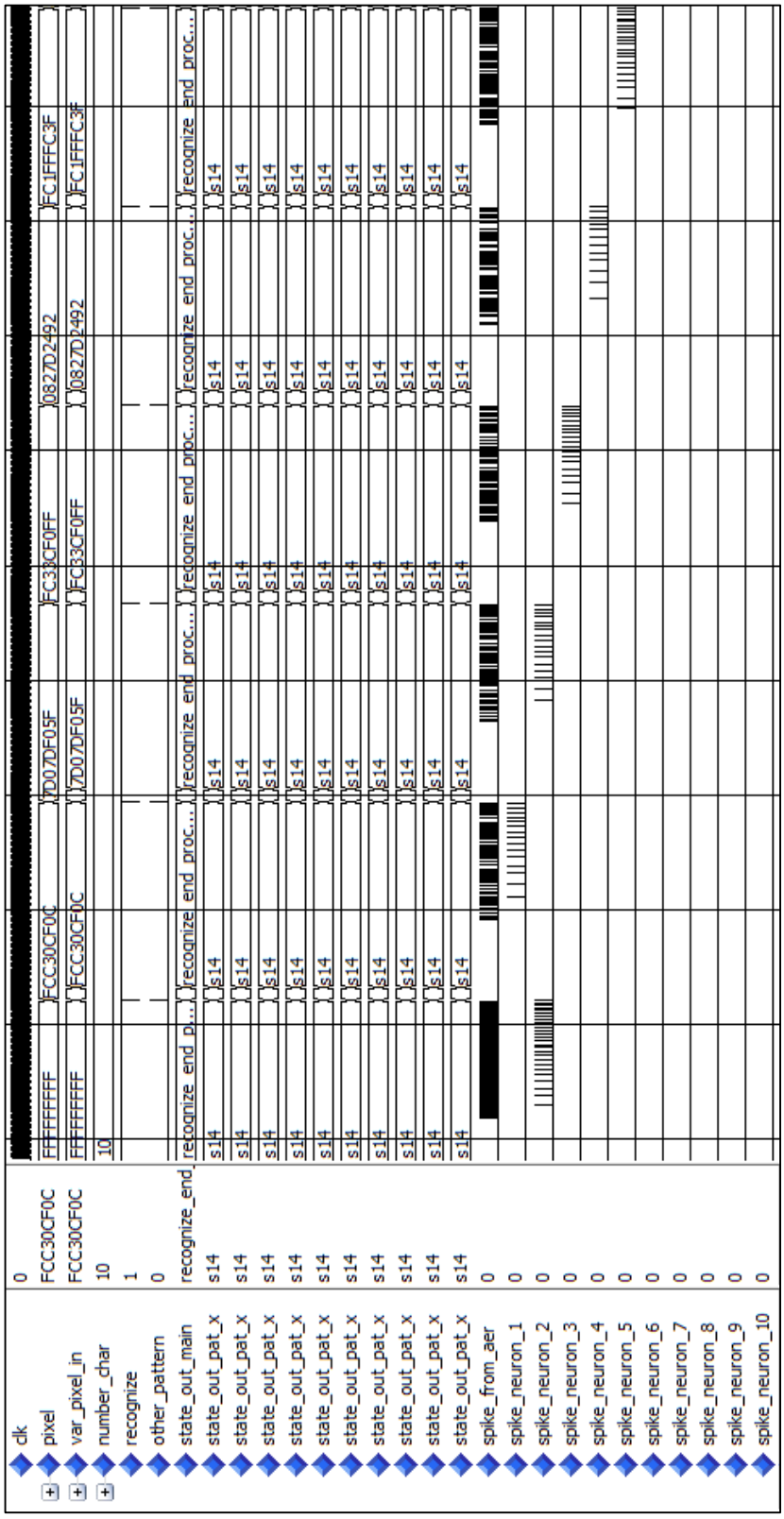

Fig. 4.39. Reconocimiento de caracteres en FPGA. Secuencia de reconocimiento: “caracter desconocido (FFFF...F)”, "uno (FCC30CF0C)"; "dos (7D07DF05F)";"tres (FC33CF0FF)", "cuatro (0827D2492)”, y "cinco (FC1FFFC3F)".

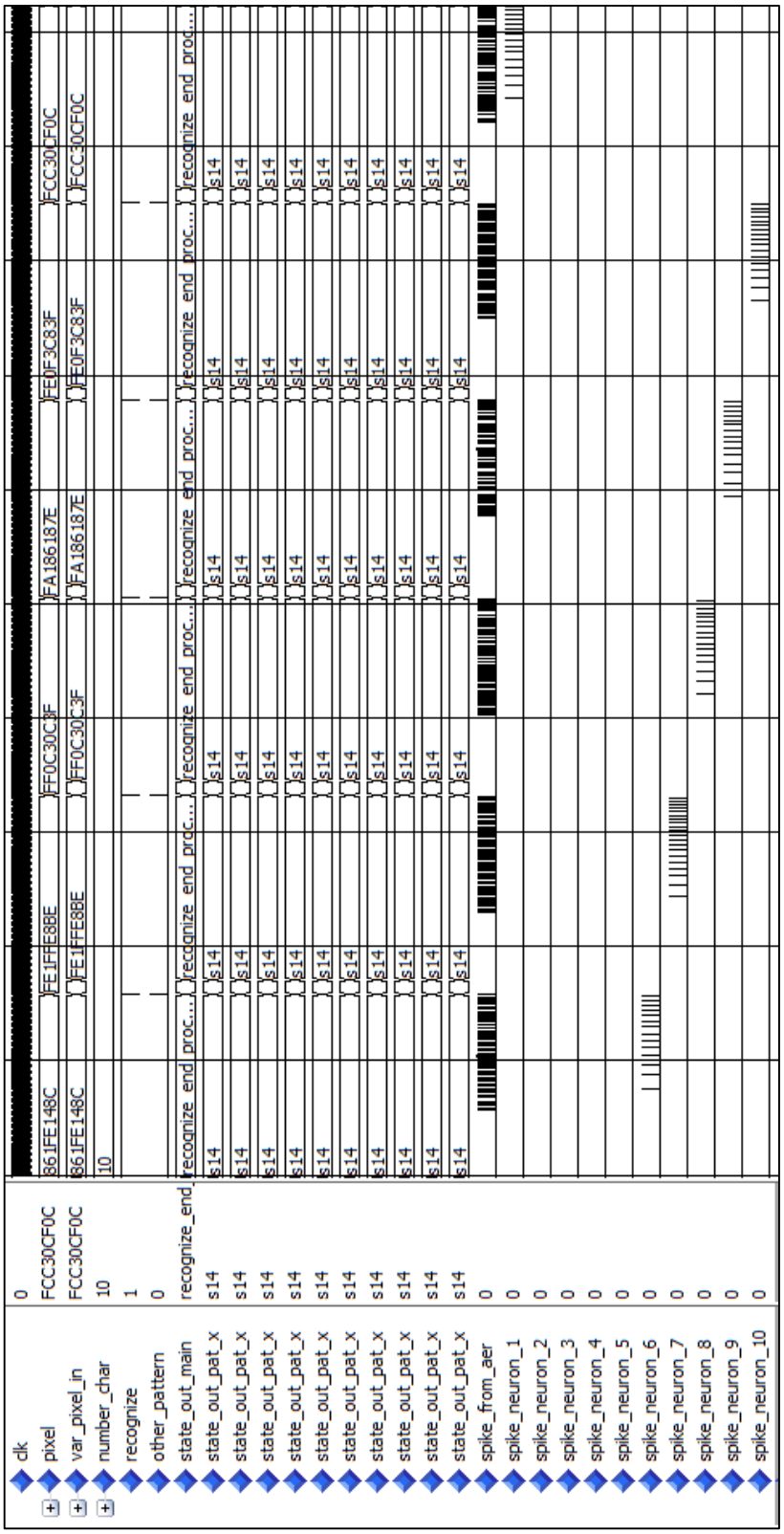

Fig. 4.40. Reconocimiento de caracteres en FPGA. Secuencia de reconocimiento: "A (861FE148C)";"B (FE1FFE8BE)";"C (FF0C30C3F)";"D (FA186187E)";"E (FE0F3C83F)" y "uno (FCC30CF0C)".

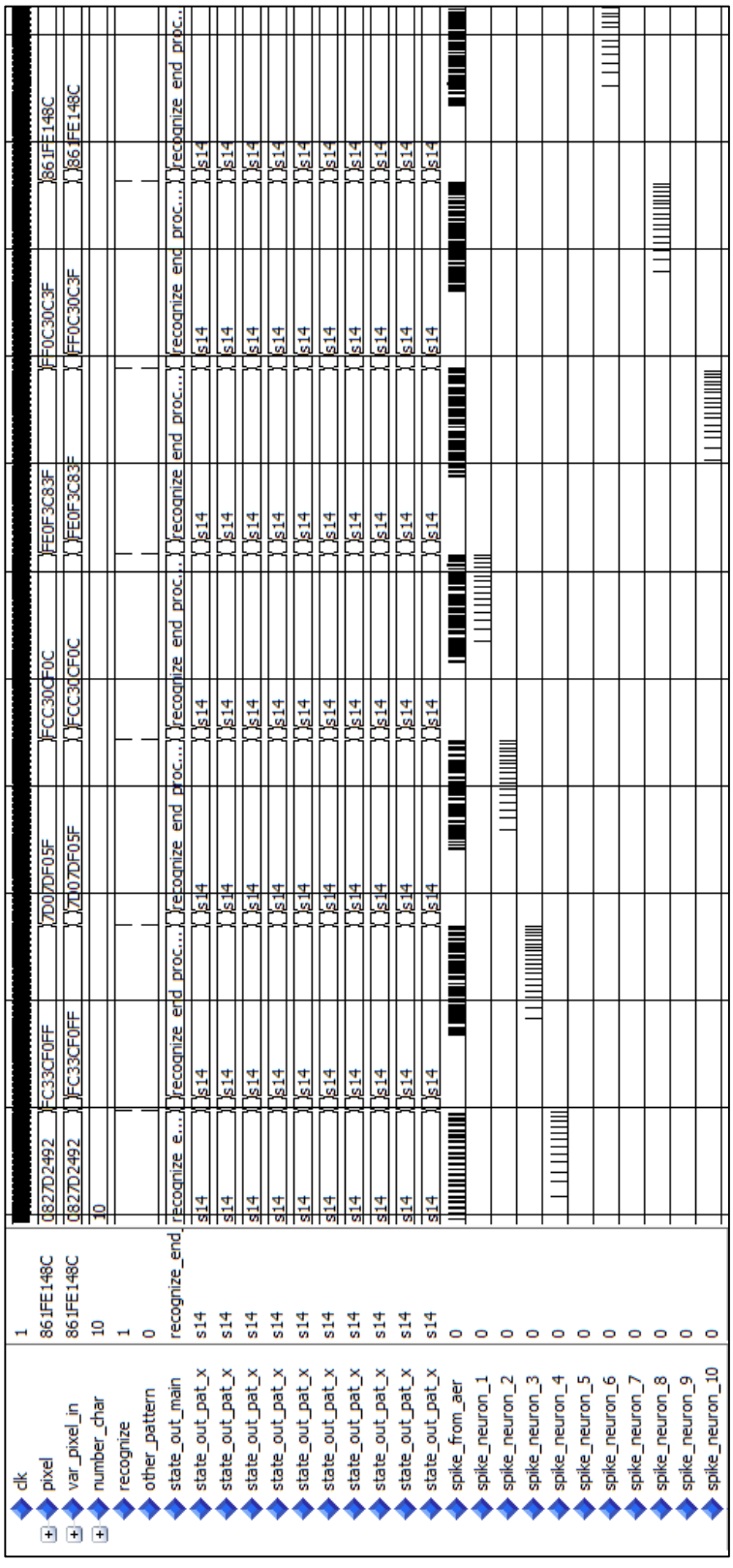

Fig. 4.41. Reconocimiento de caracteres en FPGA. Secuencia de reconocimiento: "cuatro (0827D2492)", "tres (FC33CF0FF)", "dos (7D07DF05F)", "uno (FCC30CF0C)", "E (FE0F3C83F)", "C (FF0C30C3F)"y "A (861FE148C)".

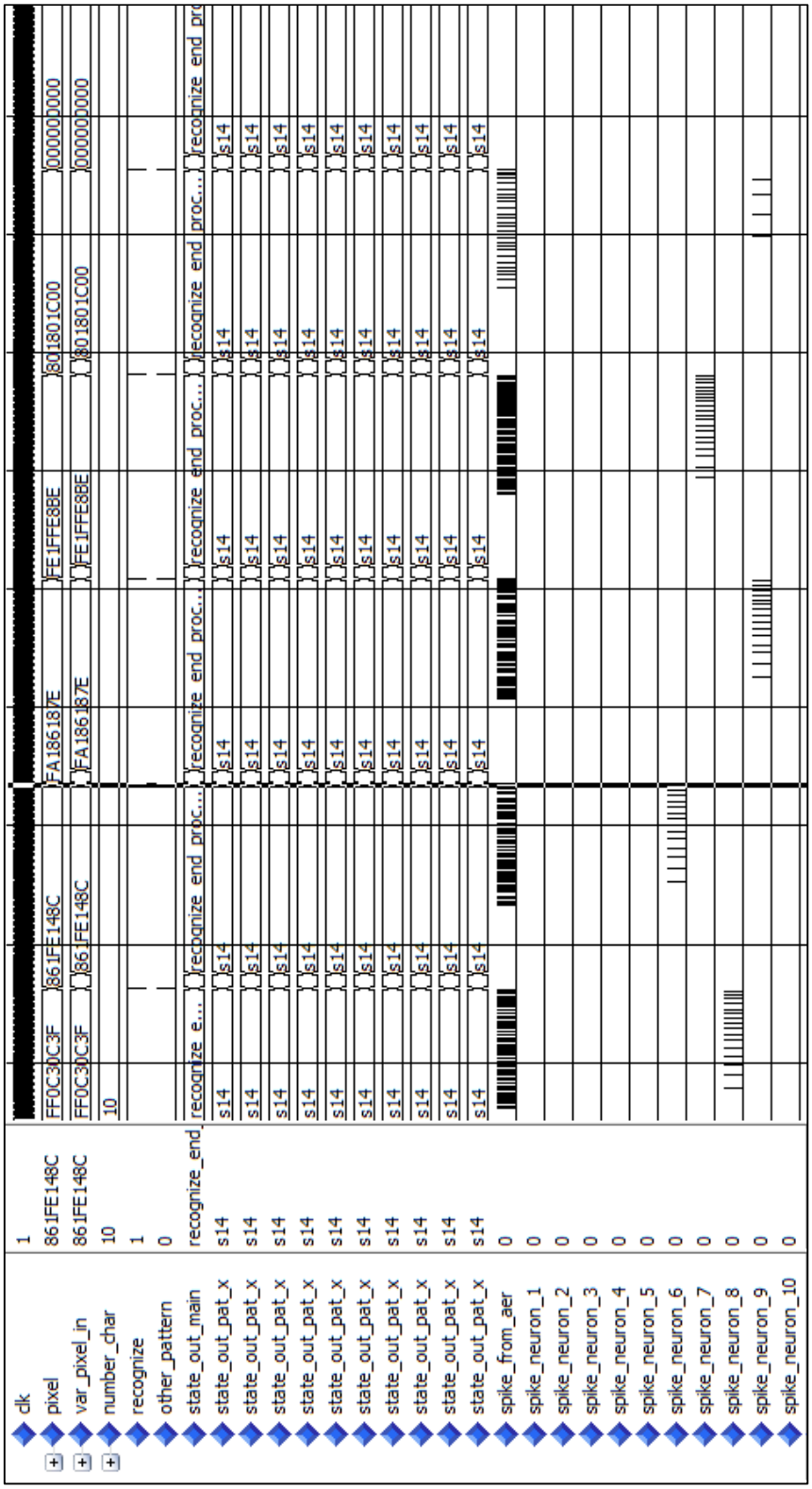

Fig. 4.42. Reconocimiento de caracteres en FPGA. "C (FF0C30C3F)", "A (861FE148C)", "D (FA186187E)", "B (FE1FFE8BE)" y caracter desconocido (801801C00).

4.2.5 Recursos en hardware

Los recursos necesarios para implementar el sistema de reconocimiento de caracteres con arquitectura neuromórfica, se muestra en la tabla (4.4); obsérvese, que la implementación de la regla de relajación, neurona y ventana de aprendizaje de STDP, son las que demandan más recursos de DSP´s.

Sistema completo de reconocimiento de caracteres				
Módulo	Logic Utilization	Used	Available	Utilization
Complete_SNN	Number of Slice Registers	29,593	126,800	23%
	Number of Slice LUTs	59,534	63,400	93%
	Number of IOBs	69	210	32%
	Number of DSP48E1s	218	240	90%
Modulos principales (Capa de entrada, salida y sistema de control)				
Módulo	Logic Utilization	Used	Available	Utilization
Entry_Process	Number of Slice Registers	2,654	126,800	2%
	Number of Slice LUTs	7,010	63,400	11%
	Number of DSP48E1s	108	240	45%
STDP_Process	Number of Slice Registers	2,587	126,800	2%
	Number of Slice LUTs	5,011	63,400	7%
	Number of DSP48E1s	8	240	3%
Main_Control	Number of Slice Registers	623	126,800	0%
	Number of Slice LUTs	569	63,400	0%
	Number of DSP48E1s	0	240	0%
AER, sinapsis y módulos que requieren DSP´s				
Módulo	Logic Utilization	Used	Available	Utilization
Modulo_STDP	Number of Slice Registers	623	126,800	0%
	Number of Slice LUTs	569	63,400	0%
	Number of DSP48E1s	4	240	1%
Relaxion_Rule	Number of Slice Registers	0	126,800	0%
	Number of Slice LUTs	72	63,400	0%
	Number of DSP48E1s	4	240	1%
Synapses	Number of Slice Registers	1,800	126,800	1%
	Number of Slice LUTs	3,025	63,400	4%
	Number of DSP48E1s	0	240	0%
AER_System	Number of Slice Registers	69	126,800	1%
	Number of Slice LUTs	200	63,400	0%
	Number of DSP48E1s	0	240	0%
Neurona_iz_24_b	Number of Slice Registers	51	126,800	1%
	Number of Slice LUTs	183	63,400	0%
	Number of DSP48E1s	3	240	1%

Tabla 4.4. Recursos en LUTS y DSP's del sistema de reconocimiento de caracteres en el FPGA "xc7a100t".

4.2.6 Estructura neuromórfica en VHDL

Como se mencionó en la sección (4.2.1), en la tabla (4.5) se muestran los archivos que conforman el sistema principal “SNN_complete”. En esta tabla se puede observar cada programa en VHDL que describe cada módulo en hardware (“archivo fuente”) y su correspondiente archivo de banco de pruebas (“Testbench”).

archivo fuente	Test bench	archivo fuente	Test bench	archivo fuente	Test bench
Neurona_iz	Neurona_iz_24_rst	Process_Data_Input	Process_Data_Input_tb	Entry_Process	Entry_Process_tb
Input_Pattern	Input_Pattern_tb				
Firing_Module	Firing module_tb				
Tri_State_Buffer_in	Tri_State_Buffer_In_tb				
Sender	N/A	AER_System	AER_System_tb		
Receiver	N/A				
Delay_AER_STDP	Delay_AER_STDP_tb	FIFO_all_in_one	N/A		
FIFO_presynap	FIFO_presynap_tb				
Memoria_RAM	Memoria_RAM_tb	N/A		STDP_Process	STDP_Process_tb
LFSR	LFSR_tb				
	LFSR_cases_tb				
STDP_Control	STDP_Control_tb	Modulo_STDP	Modulo_STDP_simple_tb		
STDP_Rectas	STDP_Rectas_tb	N/A			
Relaxion_Rule	relaxion_rule_tb				
Address_Low_Current	Address_Low_Current_tb				
Synapses	Synapses_tb				
Machine_State	Machine_State_tb				
Master_Control	N/A				
Main_Master_Control	N/A	N/A		Main_Control	Main_control_tb
Main_Machine_State	Main_Machine_State_tb				
Memoria_RAM_pattern	Memoria_RAM_pattern_tb				
Work_Window	Work_Window_tb				
General_Spike_control	General_Spike_control_tb				

Tabla 4.5. Programas que conforman al sistema neuromórfico etiquetado como “SNN_complete”

4.2.7 Conclusiones Hardware

Una arquitectura neuromórfica permite ejecutar de manera paralela los procesos aunque se esté trabajando de manera secuencial así como la operación en tiempo real, que es crucial en la inducción del aprendizaje STDP y reconocimiento de caracteres en la red.

En sistemas neuromórficos que intenten replicar el funcionamiento de STDP, siempre será necesario un nivel mínimo de corriente para que las neuronas post sinápticas comiencen a pulsar y que tenga lugar la comparación de pulsos pre y post sinápticos en entrenamiento y por lo tanto se dé el aprendizaje. Con base en esto, el nivel necesario de corriente, así como la velocidad y duración del proceso de acondicionamiento de la red, al que llamamos “integración de baja corriente”, varía dependiendo del modelo neuronal y sináptico usado, del valor de los pesos sinápticos iniciales y por ende de la cantidad de neuronas y relaciones sinápticas que tenga la red. En nuestro caso, por las características de la red, un incremento en la cantidad de neuronas, disminuiría el tiempo de pre procesamiento para que las neuronas post sinápticas comiencen a pulsar, ya que la corriente incrementa más rápido al generarse más pulsos desde la capa de entrada.

En sistemas de trabajo neuromórficos, los procesos biológicos de aprendizaje como el de STDP, pueden ser replicados desde un punto de vista completamente teórico o hasta llegar al experimental, porque todos los cambios de peso sináptico están en función del tiempo en el que tienen lugar y como ocurren realmente. En este sentido, el sistema de asociación de pulsos propuesto como “el primero en disparar” y el uso de la ventana de aprendizaje doble exponencial nos permite estudiar dicho comportamiento y en este contexto, validar que el método usado aquí es satisfactorio para inducir el aprendizaje en una red con STDP neuromórfico. De haber sido el caso contrario, simplemente no habría modificación sináptica positiva, ya que todos los pares obtenidos hubieran estado en la región LTD; por lo tanto, el sistema

de asociación de pulsos propuesto funciona correctamente con los parámetros de la ventana de aprendizaje seleccionada.

Una de las mayores ventajas de una red implementada en tiempo real, es que permite el procesamiento espacio-temporal con patrones de tiempo y espacio, es decir, cada imagen que pasa por un proceso de entrenamiento, configura sus propios valores en función de los tiempos de aparición de los pulsos pre sinápticos en reconocimiento o por aparición de pulsos asociados en aprendizaje, siguiendo pautas de orden y tiempo de aparición. Esto, por ejemplo, en un proceso de “integración de baja corriente” provoca que la duración sea diferente entre diversas neuronas, mientras que un proceso de aprendizaje de STDP, provoca que la modificación sináptica dependa del orden en el tiempo y la región de donde provienen los pulsos pre sinápticos. Un cambio, por sinapsis, terminaría afectando en tiempo la generación de pulsos post sinápticos generados, que el mismo protocolo STDP ocupa para generar cambios en el peso sináptico. En este contexto, la capacidad de inducción de aprendizaje en la red dependerá de la capacidad del protocolo de STDP para asociar pulsos pre y post sinápticos ya que no todos serán procesados de acuerdo a la base de reglas propuestas en la sección (3.5.3.1.2) o bien porque el control en ciertos casos, se encuentra realizando una tarea y ésta no puede llevarla a cabo. Hay que recordar que podemos procesar paralelamente los eventos porque todos los pulsos pre sinápticos no llegan al mismo tiempo. De la misma forma, en etapa de reconocimiento, el procesamiento espacio-temporal nos permite decodificar la información, ya que los tiempos de respuesta de las diversas neuronas post sinápticas son distintos a pesar de que éstas manejen la misma velocidad de operación.

Por otra parte, el módulo AER que es el responsable de realizar la función de integración espacio-temporal de los pulsos pre sinápticos y que hace posible el procesamiento por eventos del sistema, en etapas de reconocimiento es de suma importancia, porque produce lo que podríamos llamar “una huella digital” única para cada caracter insertado en la red. En efecto, el tren de pulsos

característico en tiempo y espacio de cada caracter, es el que permite determinar dicho proceso porque esa secuencia es única con ese caracter en particular y por lo tanto en proceso de reconocimiento; la neurona ganadora será aquella con los pesos que se adecuan mucho mejor a dicha secuencia, no como un simple valor numérico, ya que en proceso de sinapsis, solo se tomarán aquellos que formen a la secuencia generada en orden y tiempo de aparición.

En cuanto al proceso de aprendizaje dirigido de STDP, éste se encuentra íntimamente ligado con la cantidad de corriente post sináptica, que como hemos visto, siempre una corriente mayor favorece a la generación de pulsos post sinápticos. En este sentido, se incrementa la probabilidad de que se generen pares de pulsos potenciados, lo que nos lleva a la conclusión de que una mayor corriente posibilita el posicionamiento en tiempo de los pulsos post sinápticos posteriormente en relación a los pulsos pre sinápticos. En su defecto, se mantiene la relación, con lo que se logra que existan mayores incidencias en las relaciones de pulsos en la región de potenciación (LTP). En este sentido el esquema de STDP utilizado es un sistema que premia los pares de pulsos causales y penaliza a los a-causales.

Por otra parte, en cuanto a las desventajas del sistema propuesto en FPGA, el mayor problema es la escalabilidad, si se incrementan las dimensiones de la red, lo hace el sistema de control y, en general, la arquitectura basada en AER síncrona podría llegar a tener problemas de saturación. En este caso, el sistema no sería capaz de procesar secuencias de pulsos completos si el tiempo entre los eventos es demasiado corto, por lo que tendría que realizarse un estudio del rendimiento en cuanto a la velocidad y capacidad de procesamiento, para definir la cantidad de neuronas y de sinapsis que esta arquitectura en hardware puede soportar.

4.3 Conclusiones Generales

Las redes neuronales de tercera generación se basan en el comportamiento teórico y experimental de sus homólogas biológicas, por lo que para implementar cualquier red neuronal que sea biológicamente inspirada o que se aproxime a la plausibilidad biológica, solo se necesitarán de 5 elementos fundamentales: arquitectura, modelo neuronal pulsado, modelo sináptico, modelo de plasticidad sináptica y un método de codificación para la extracción de la información, pudiendo obtener redes con distintas capacidades de procesamiento solamente al variar los modelos usados.

Las redes de tercera generación tienen capacidad de procesamiento espacio-temporal. Esto quiere decir que los procesos de aprendizaje y de sinapsis ocurren de manera localizada y en un tiempo particular, por lo que la respuesta dinámica de la red se verá afectada por eventos presentes, pasados y futuros. En este contexto, las computadoras actuales tienen desventajas frente a los sistemas neuromórficos, que implementan paralelismo y capacidad de aprendizaje en tiempo real.

En este trabajo se abordó la implementación de sistemas equipados con aprendizaje dirigido de STDP no supervisado, para lo cual se realizó la implementación de sistemas de reconocimiento de caracteres usando la misma arquitectura neuronal, modelo neuronal, sináptico y de aprendizaje variando únicamente las dimensiones de la red y la plataforma a usar para llevar a cabo esta tarea. En general, se implementaron redes con capacidad de procesamiento de caracteres con dimensiones de 4x4, 5x5, y 6x6, siendo las primeras 2 en software y la de mayor dimensión en hardware, la cual presenta una arquitectura neuromórfica.

De los resultados obtenidos en las implementaciones en software podemos observar que el procesamiento espacio-temporal de la información solo es en el sentido numérico y dado que, las tareas se llevan a cabo de manera secuencial, es necesario trabajar con ventanas de tiempo para guardar la información y posteriormente procesarlas. En estas circunstancias, el

aprendizaje dirigido por STDP se ve afectado pudiendo converger o no para arquitecturas de mayores o menores dimensiones usando los mismos parámetros de la ventana de aprendizaje. Así, para lograr la convergencia del aprendizaje necesitaríamos buscar una ventana de tiempo que se ajuste a los parámetros de ventana de aprendizaje seleccionada o viceversa, aunque de manera teórica se asegure que solamente con que el área de la región de potenciación (LTP) sea menor que la región de depresión (LTD), lo que afecta, porque una red con las características provistas, si la ventana de tiempo disminuye o aumenta, provoca que la cantidad de ventanas de aprendizaje que ocupa el protocolo de STDP que entran en ellas varíe.

De los resultados obtenidos en la implementación en hardware, podemos observar que el procesamiento espacio-temporal de la información es numérico y en tiempo real, y por la plataforma neuromórfica usada, ésta puede realizarse porque la arquitectura basada en eventos hace posible el procesamiento paralelo. En estas circunstancias, el procesamiento de los pulsos no se lleva a cabo a través de una ventana de tiempo como se hace en software, sino que se realiza conforme transcurre el tiempo. Por esta razón, el aprendizaje dirigido por el protocolo de STDP depende propiamente de la dinámica neuronal (neuronas, sinapsis y la modificación sináptica) a lo largo del tiempo; podemos asegurar la inducción del aprendizaje a través de dos factores: el primero, proponiendo una ventana de aprendizaje donde el área de la región de potenciación sea mucho menor que la de la región de depresión (tal como se describe teóricamente), y el segundo, la selección de un sistema de asociación de pulsos basada en la teoría de la plasticidad neuronal (pares, triplet´s y cuadruplet´s) que en conjunto con la ventana de aprendizaje seleccionada provoque dicha inducción.

Por las razones antes expuestas, los sistemas en software y hardware propuestos no son sistemas equivalentes, a pesar de tener los mismos modelos neuronales, sinápticos y de aprendizaje, simplemente por el hecho de manejar sistemas de asociación de pulsos diferentes, los cuales, dependen

del modo de operación de la red en cada plataforma. En software el sistema no puede evolucionar más allá del tiempo establecido por la ventana de tiempo seleccionada y por su forma de asociación de pulsos y de operación, su comportamiento está lejos de aproximarse al comportamiento de una red biológica, sin embargo, una de las ventajas de la implementación de redes en software es que el sistema tiene tiempos fijos de operación pudiendo determinar la duración de las etapas de entrenamiento y reconocimiento. Por otra parte, en hardware, el sistema sí puede evolucionar a lo largo del tiempo ya que no se encuentra limitado por una ventana de tiempo, y por lo tanto es posible apegarse a la teoría provista por las neurociencias y servir como herramienta de validación, verificación y propuesta de nuevas teorías en el campo de la computación biológica. Sin embargo, dado que todos los valores obtenidos y procesos tienen una fuerte dependencia del tiempo, la duración de las etapas de entrenamiento y el reconocimiento pueden variar, provocando que en cierto momento, todas las variables que intervienen en la red, no se puedan estimar.

A partir de los experimentos, realizados podemos decir que el modelo neuronal de Izhikevich, el modelo de sinapsis simple y la arquitectura seleccionada, son factibles para el discernimiento de patrones, independientemente de si son caracteres o propiamente imágenes cuando se implementan con un algoritmo no supervisado por STDP para inducir el aprendizaje. Sin embargo, la mayor desventaja de éste, independientemente de si la implementación es en software o hardware, es que el reconocimiento es posible solamente para aquellos patrones usados en entrenamiento; si se ocupa algún otro patrón que no haya sido usado para entrenar la red, dado que estamos usando el método de decodificación de información basado en el primero en disparar, en hardware alguna neurona terminará por disparar, en este caso, aquella que se asemeje más con el patrón pulsado. En software se podría obtener una corriente post sináptica similar o igual a la esperada en la capa de salida que genera el mismo patrón de generación de pulsos; en este sentido, sería mucho más confiable implementar un algoritmo supervisado de STDP.

4.4 Trabajo Futuro

El trabajo futuro, dados los resultados de este trabajo, es enfocar todos los esfuerzos en desarrollar sistemas neuromórficos digitales sobre FPGA's, para lo cual es necesario abarcar desde el mejoramiento del sistema actual e investigar otros tópicos relacionados a este tipo de sistemas. Tales tareas se encuentran descritos a través de los siguientes puntos:

- Concluir la implementación del sistema presentado, proporcionando un módulo adicional para la interacción con el usuario y bajar finalmente la arquitectura a la placa Artix 4DDR, que por cuestiones del alcance del tema de tesis no se abarcó.
- Determinar las características de velocidad del módulo AER, para así determinar la capacidad máxima de integración espacio-temporal que puede manejar dicho módulo, ya que de éste depende la cantidad de neuronas al que un sistema neuromórfico puede llegar a utilizar.
- Con base en el punto anterior, estudiar la capacidad del procesamiento en tiempo real de los elementos, para determinar la cantidad máxima de neuronas y sinapsis que el sistema puede soportar con base en la arquitectura actual.
- Estudiar el proceso de aprendizaje en tiempo real, usando otros protocolos de asociación de pulsos, como el de triplet's y cuadruplet´s, para lo cual es necesario la implementación de 2 o más relojes en paralelo, para obtener la captura de las diferencias temporales entre pulsos pre y post sinápticos.
- Estudiar el desempeño del sistema usando otras ventanas de aprendizaje, que permitan la variación de sus parámetros, para lo cual es necesario el diseño de hardware reconfigurable para implementar otras funciones.
- Estudiar el desempeño del sistema usando modelos sinápticos mucho más plausibles, para lo cual es necesario, al igual que con el punto anterior, el diseño de hardware reconfigurable para implementarlos.

- Estudiar a fondo los métodos de codificación basados en pulsos en tiempo real, así como el diseño de la arquitectura asociada para realizar dicha tarea.
- Implementar sistemas donde las sinapsis inhibitorias se procesen de manera formal y no simplificada.
- Implementar propiamente sistemas neuromórficos en cuanto a la cantidad de neuronas que pueden manejar, en este caso, se optaría por la simplificación del modelo neuronal usado manteniendo las mismas características dinámicas que posee individualmente o bien el diseño de un procesador de bajos recursos que pueda computar los modelos sin algún tipo de simplificación y con ello incrementar la cantidad de neuronas del sistema.
- Estudiar sistemas de control que faciliten la escalabilidad de este tipo de sistemas, así como simplificar el sistema de control actual.

Como vemos, para realizar todo ello, es necesario diseñar arquitecturas lo suficientemente robustas para implementar tales cuestiones; esto implica diseñar sistemas altamente paralelizados, con sistemas de control que trabajen en tiempo real, escalables y de bajo consumo de recursos para que puedan ser embebidos en circuitos FPGA.

El estudio de sistemas neuromórficos digitales, analógicos o mixtos permitirá realizar el diseño de retinas y cócleas artificiales y, en un futuro, abordar problemas relacionados en el desarrollo de arquitecturas de cómputo basadas en el comportamiento de sus homólogos biológicos como lo que es “AER”, y que posibilite el tratamiento espacio-temporal de la información.

ANEXO

Anexo: RTL de los módulos principales sintetizados en FPGA

Como se describe en la organización de esta tesis, se colocan los “RTL” de los módulos principales del sistema de reconocimiento de caracteres en FPGA. Dicha estructura, corresponde con la tabla (4.5) del capítulo 4, sección (4.2.6) y que se explica a mayor detalle en el capítulo 3.

Módulo “SNN Complete”.

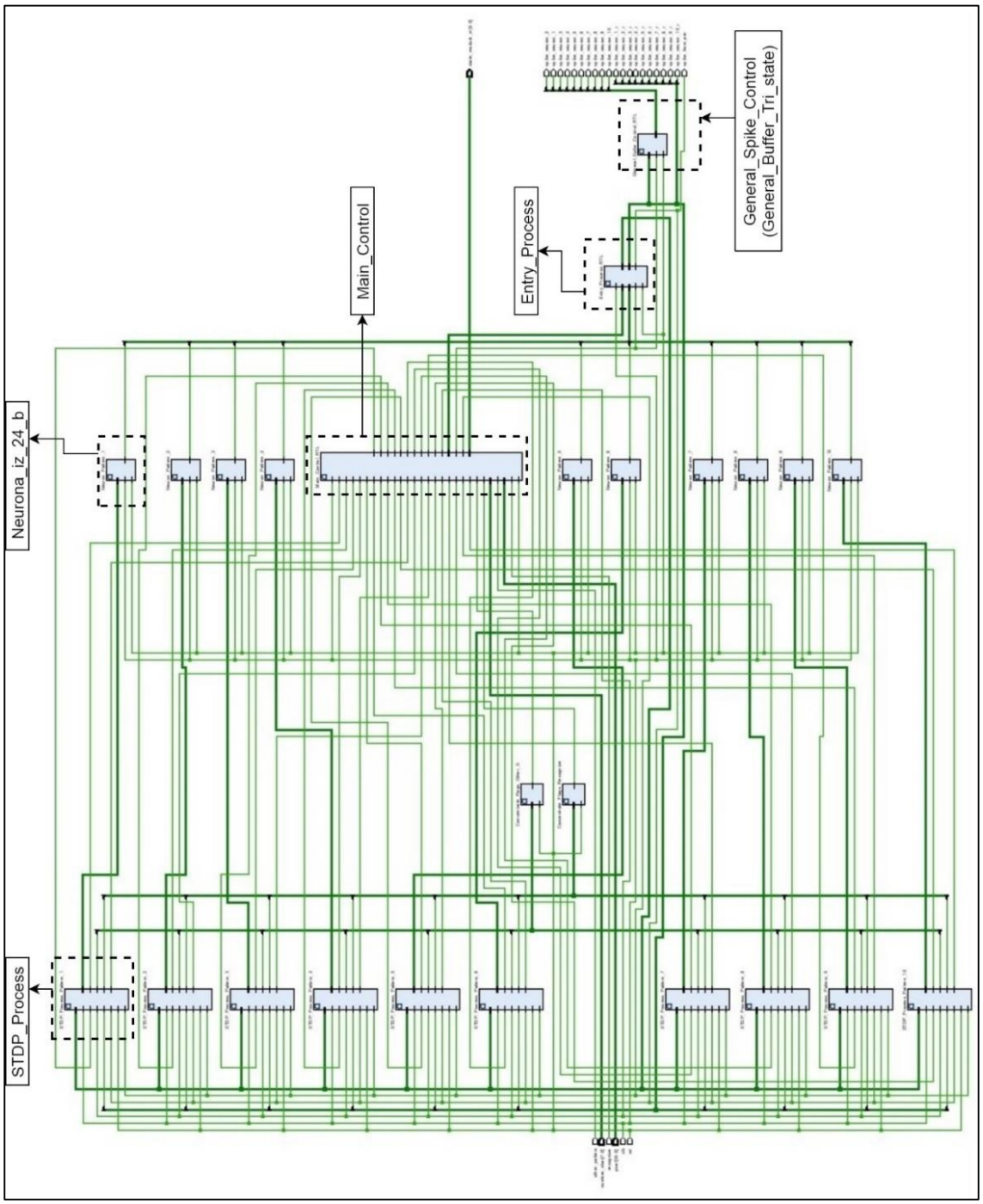

Fig. RTL del módulo “SNN_Complete”. Arriba se encuentran los puertos de salida y abajo se observan los de entrada. Se señalan los principales módulos que conforman el sistema de reconocimiento de caracteres

Módulo “STDP_Process”.

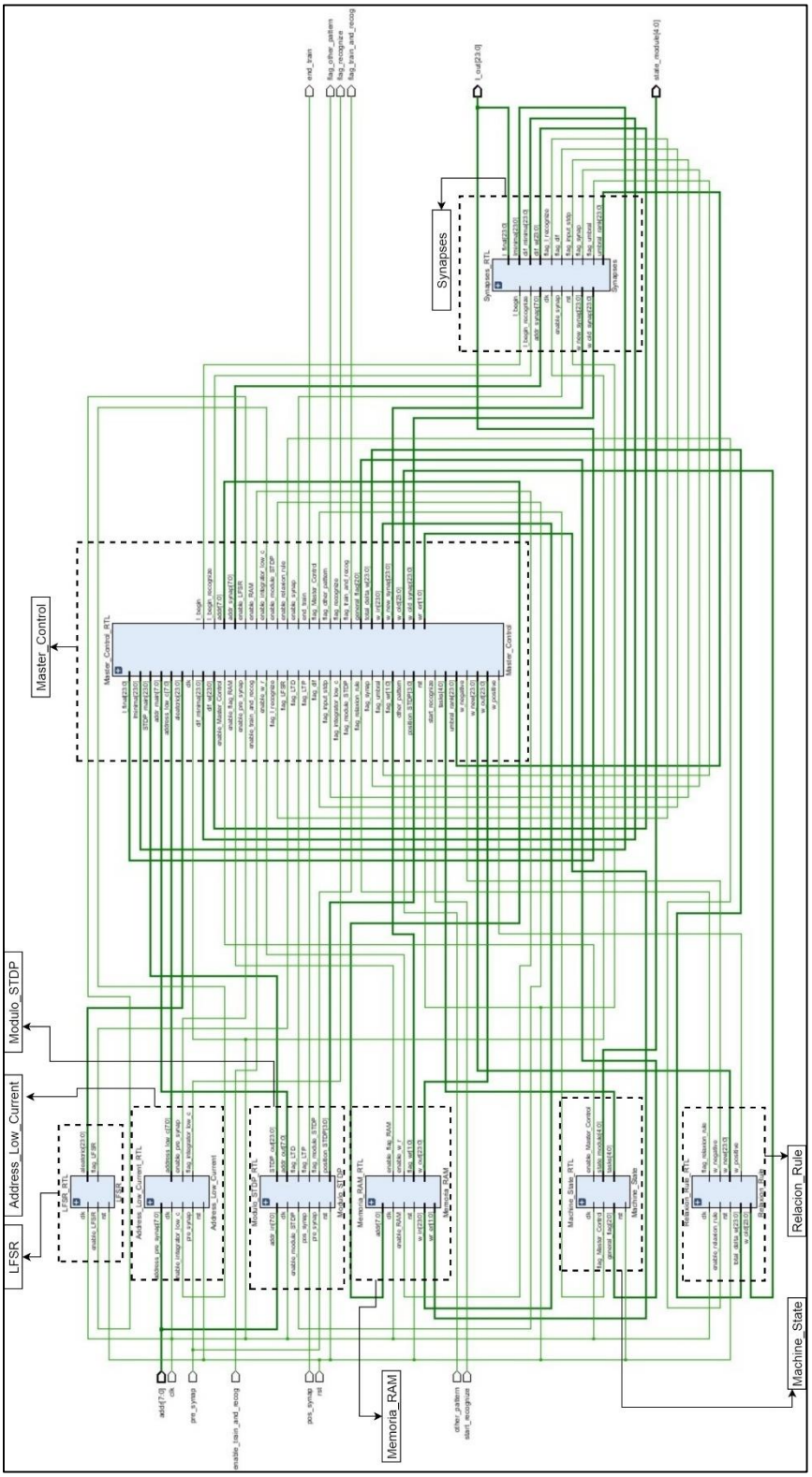

Fig. RTL del módulo “STDP_Process”. Arriba se encuentran los puertos de salida y abajo se observan los de entrada. Se señalan los principales módulos que lo conforman.

Módulo "LFSR".

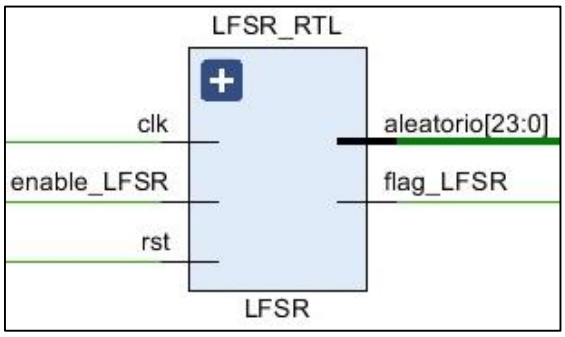

Fig. RTL del módulo "LFSR". A la derecha se encuentran sus puertos de salida y a la izquierda, los de entrada.

Módulo "Address_Low_Current".

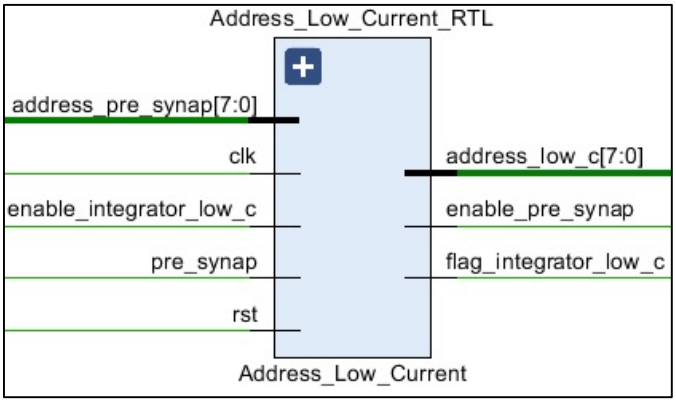

Fig. RTL del módulo "Address_Low_Current". A la derecha se encuentran sus puertos de salida y a la izquierda,los de entrada.

Módulo_STDP.

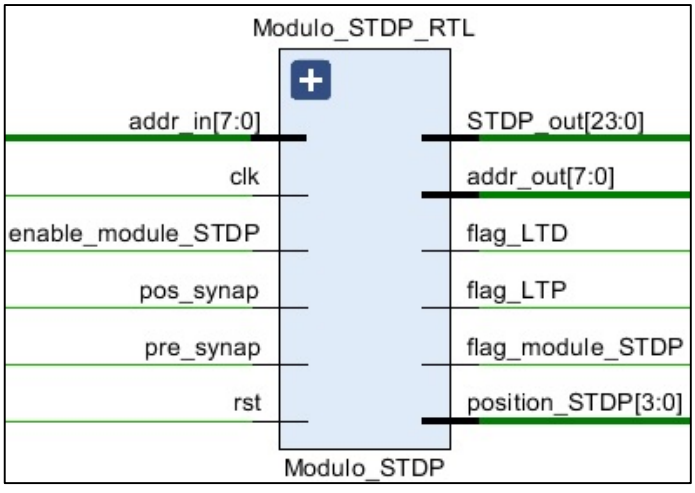

Fig. RTL del "Módulo_STDP". A la derecha se encuentran sus puertos de salida y a la izquierda, los de entrada.

Módulos que conforman a “Modulo_STDP”.

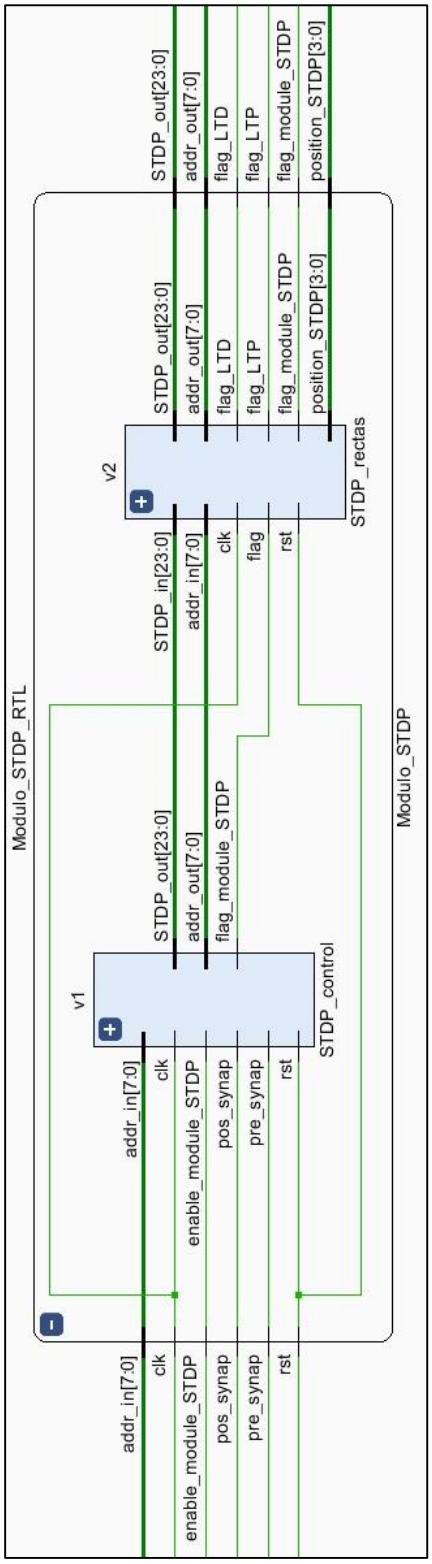

Fig. RTL de los módulos “STDP_Control” y “STDP_Rectas”. Arriba se encuentran los puertos de salida y abajo se observan los de entrada.

Módulo "Memoria_RAM".

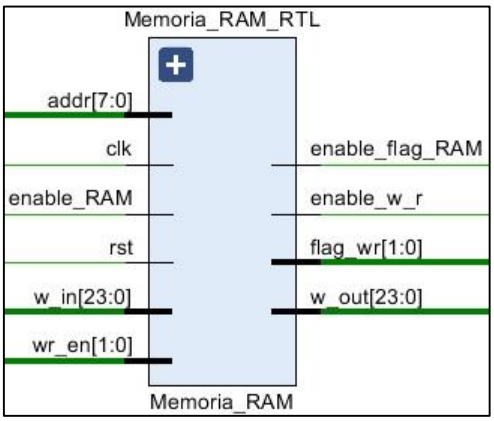

Fig. RTL del módulo "Memoria_RAM". A la derecha se encuentran sus puertos de salida y a la izquierda, los de entrada.

Módulo "Machine_State".

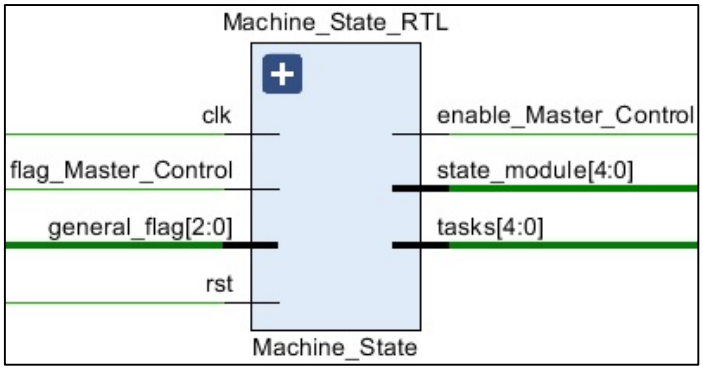

Fig. RTL del módulo "Machine_State". A la derecha se encuentran sus puertos de salida y a la izquierda, los de entrada.

Módulo "Relaxion_Rule".

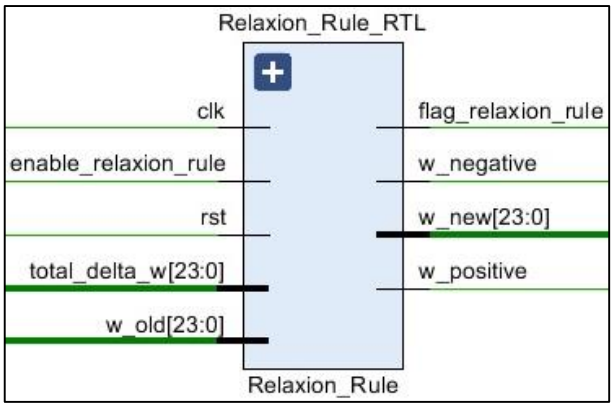

Fig. RTL del módulo "Relaxion_Rule". A la derecha se encuentran sus puertos de salida y a la izquierda, los de entrada.

Módulo "Master_Control".

Fig. RTL del módulo "Master_Control". A la derecha se encuentran sus puertos de salida y a la izquierda, los de entrada.

Módulo "Synapses".

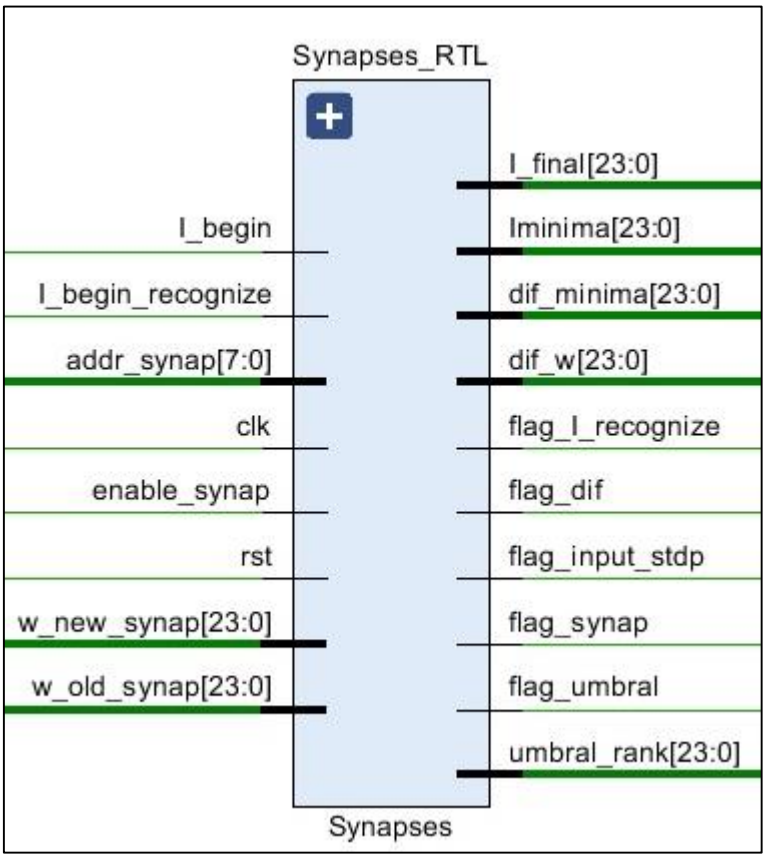

Fig. RTL del módulo "Synapses". A la derecha se encuentran sus puertos de salida y a la izquierda, los de entrada.

Módulo “Main_Control”

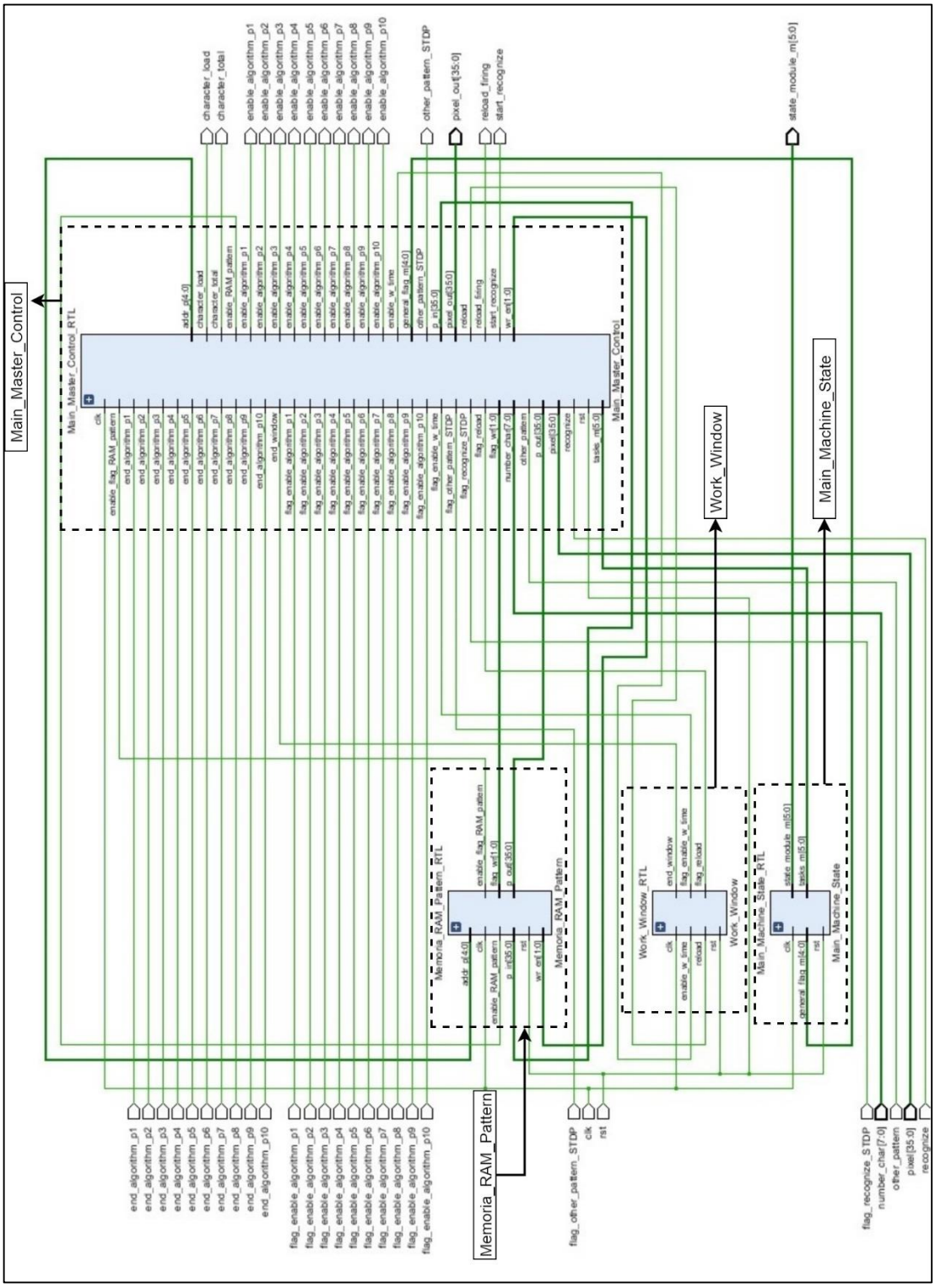

Fig. RTL del módulo “Main_Control”. Arriba se encuentran los puertos de salida y abajo se observan los de entrada. Se señalan los módulos que lo conforman.

Módulo "Main_Master_Control"

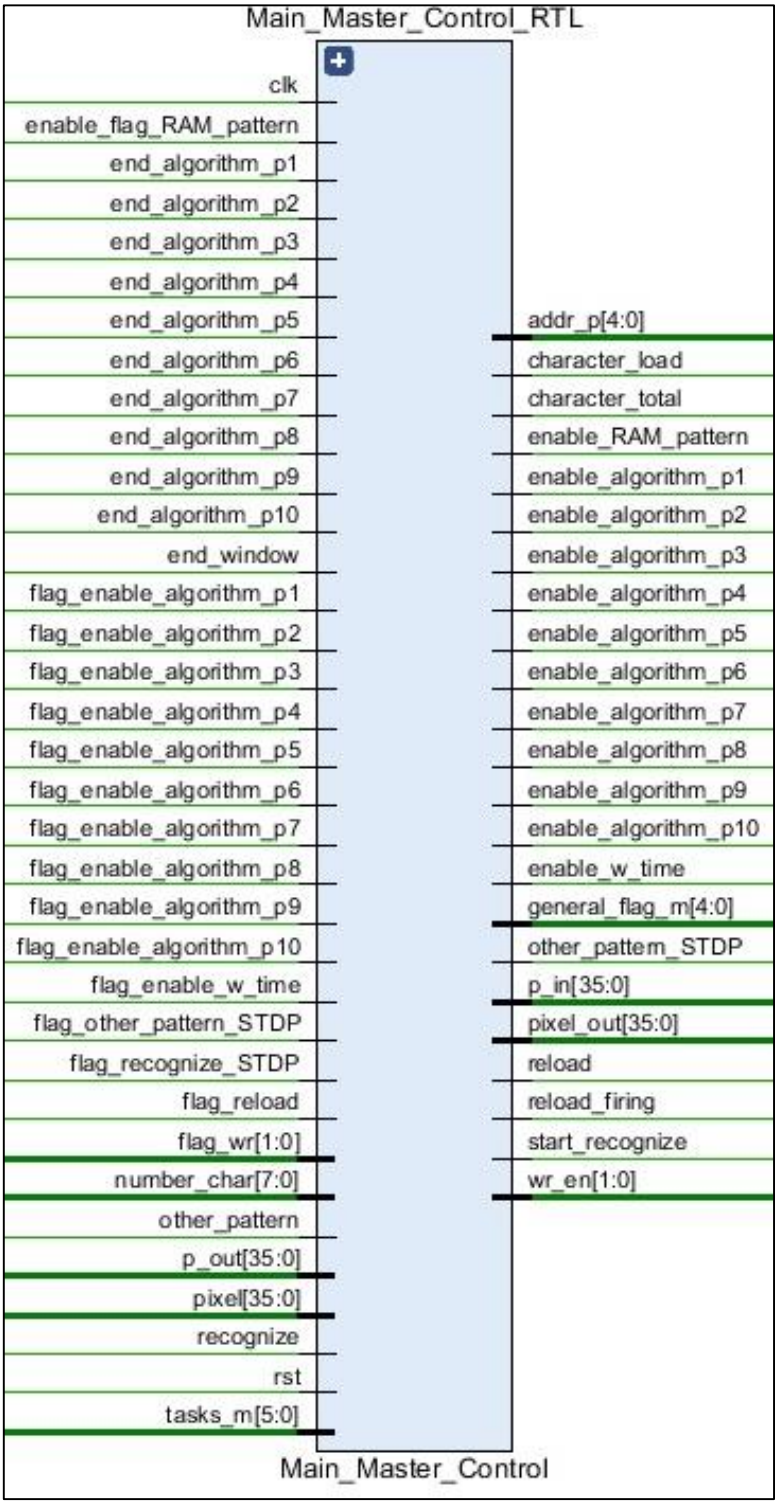

Fig. RTL del módulo "Main_Master_Control". A la derecha se encuentran los puertos de salida y a la izquierda, los de entrada.

Módulo “Work_Window”.

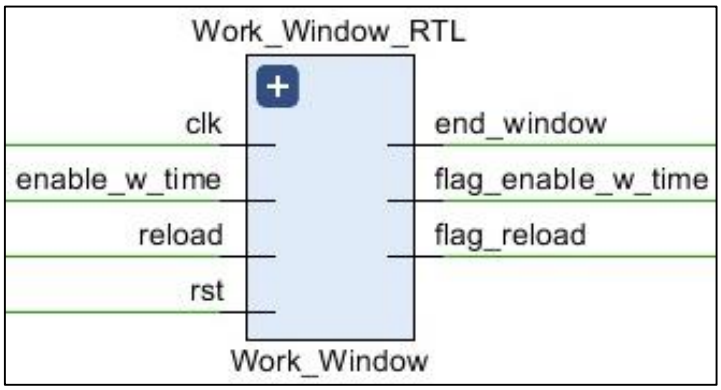

Fig. RTL del módulo “Work_Window”. A la derecha se encuentran los puertos de salida y a la izquierda, los de entrada.

Módulo “Memoria_RAM_Pattern”.

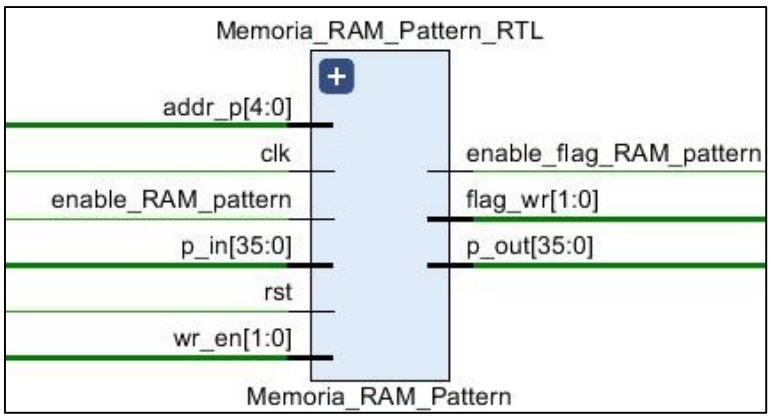

Fig. RTL del módulo “Memoria_RAM_Pattern”. A la derecha se encuentran los puertos de salida y a la izquierda, los de entrada.

Módulo “Main_Machine_State”.

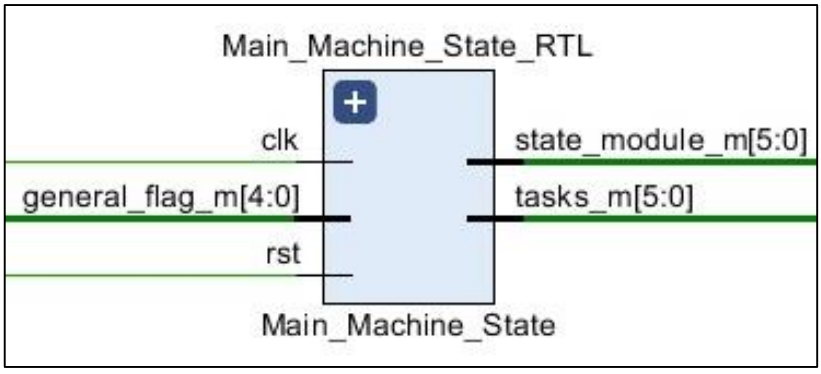

Fig. RTL del módulo “Main_Machine_State”. A la derecha se encuentran los puertos de salida y a la izquierda, los de entrada.

Módulo “Entry_Process”

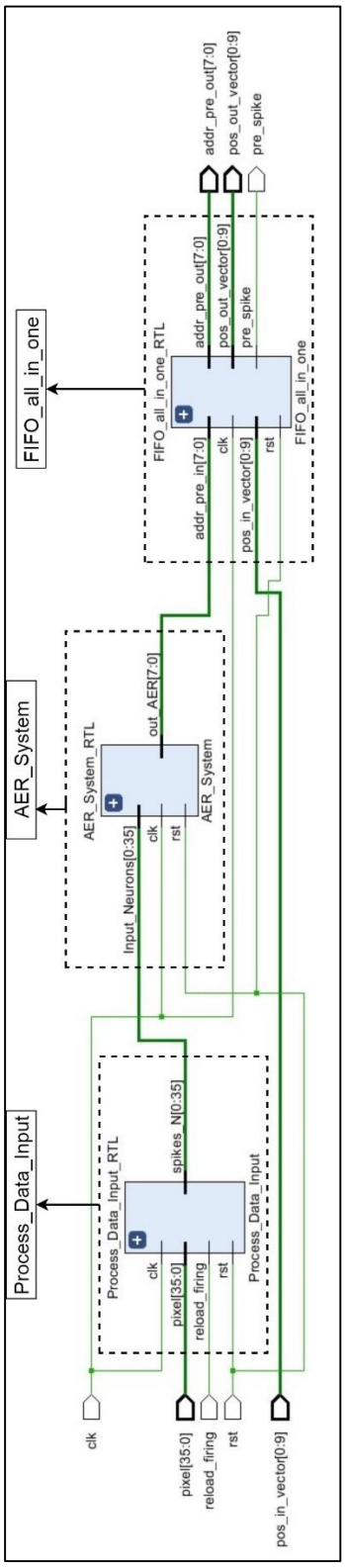

Fig. RTL del módulo “Entry_Process”. Arriba se encuentran los puertos de salida y abajo se observan los de entrada. Se señalan los módulos que lo conforman.

Módulo "Process_Data_Input".

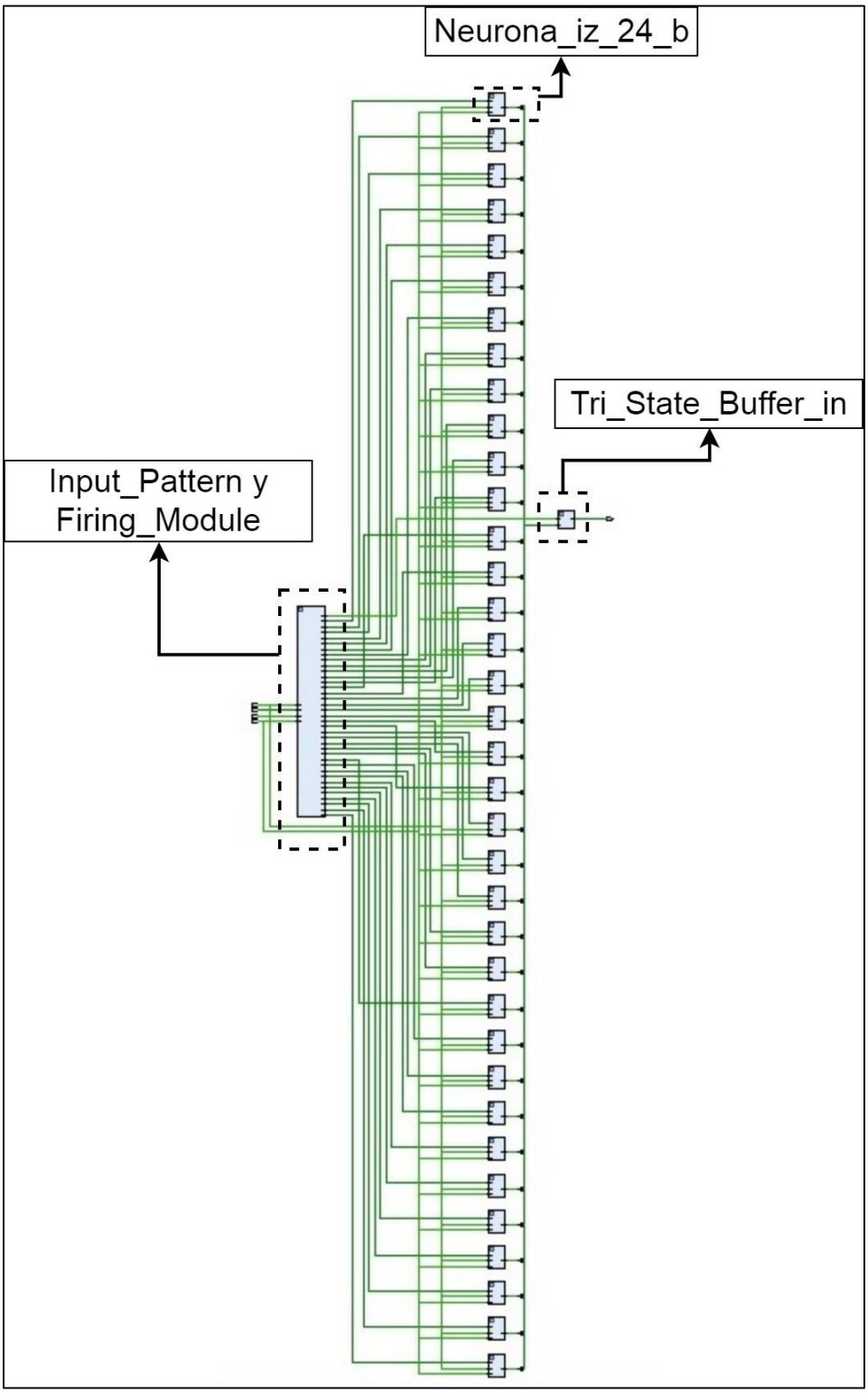

Fig. RTL del módulo "Process_Data_Input". Se señalan los módulos que lo conforman.

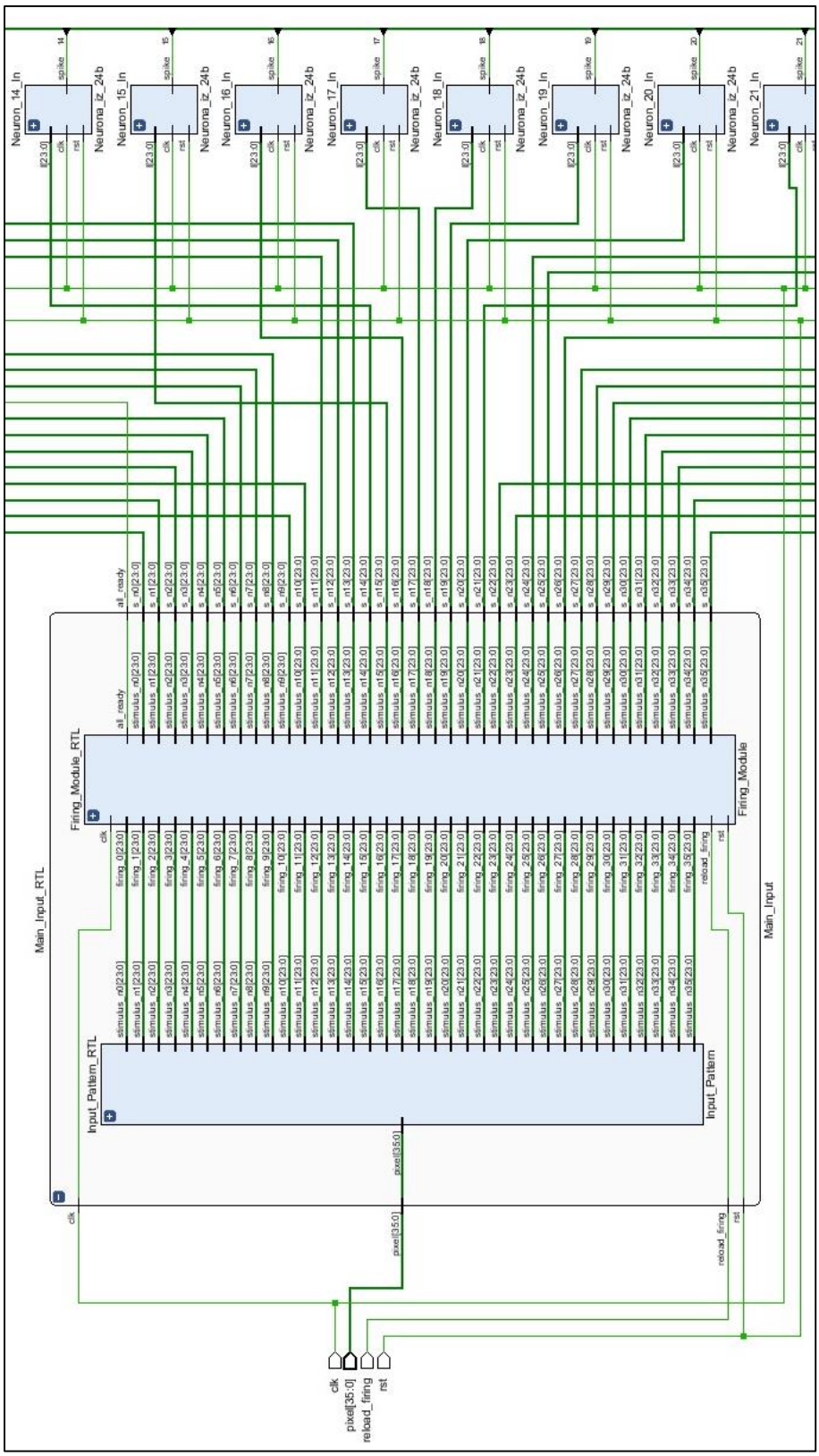

Fig. Acercamiento al módulo "Process_Data_Input", el cual contiene etiquetado a "Main_Input" al conjunto de los módulos "Input_Pattern" y "Firing_Module".

Módulo "AER_System"

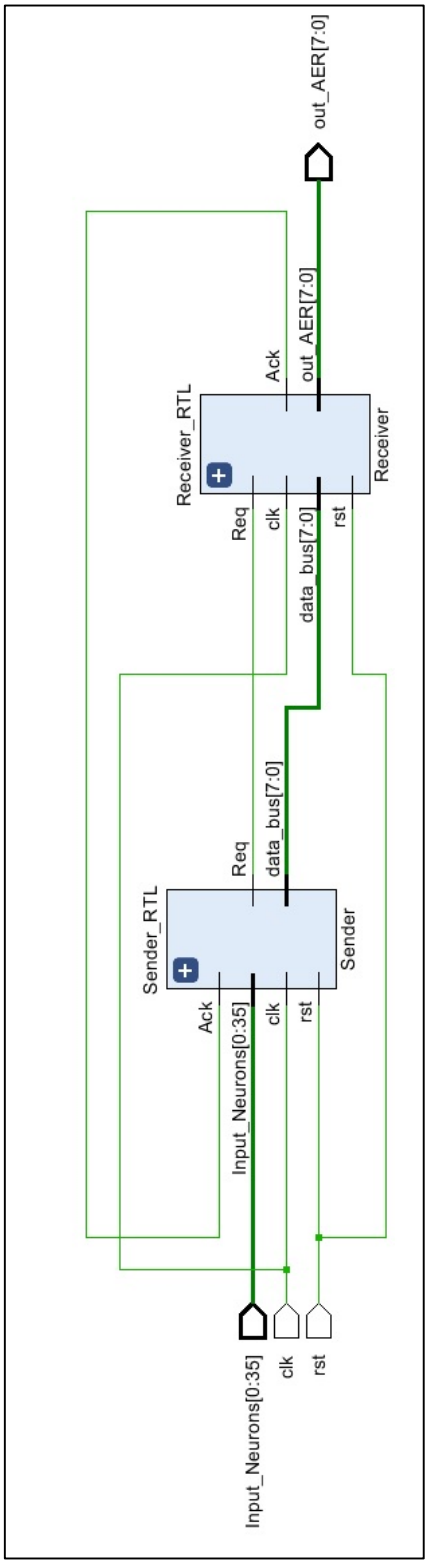

Fig. RTL del módulo "AER_System". Arriba se encuentran los puertos de salida y abajo se observan los de entrada.

Módulo "FIFO_all_in_one"

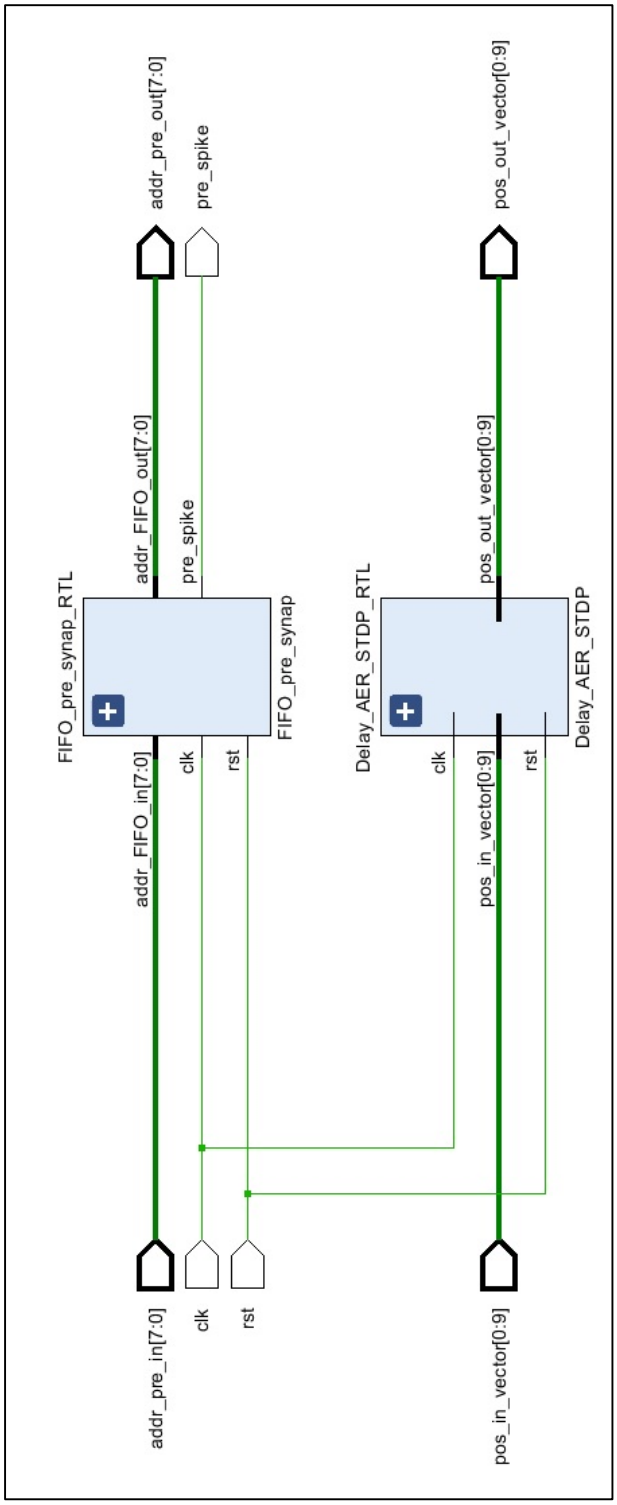

Fig. RTL del módulo "FIFO_all_in_one". Arriba se encuentran los puertos de salida y abajo se observan los de entrada.

Módulo “Neurona_iz_24_b”

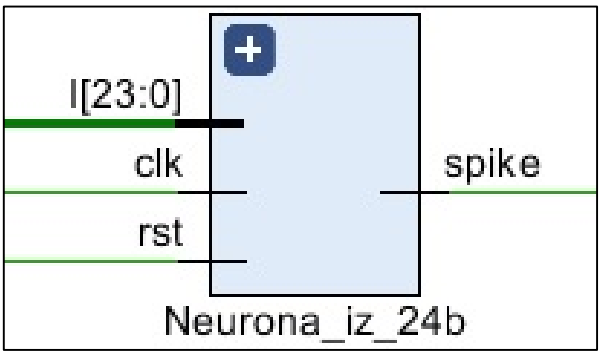

Fig. RTL del módulo “Neurona_iz_24_b”. A la derecha se encuentran los puertos de salida y a la izquierda, los de entrada.