Sistema basado en FPGA para la evaluación de redes neuronales orientadas al reconocimiento de imágenes

FPGA-based neural network evaluation system for image recognition

Ilber Adonayt Ruge Ruge
Ingeniera electrónica, magister en Ingeniería de Control Industrial. Docente de la Universidad de Cundinamarca. Fusagasugá, Colombia.
Contacto: *ilberruge@yahoo.es*

José David Alvarado
Ingeniero Electrónico, magister en Ingeniería de Control Industrial. Investigador de la Universidad del Tolima. Ibagué, Colombia.
Contacto: elejosedavid@gmail.com

Fecha de recepción: 10 de noviembre de 2011
Clasificación del artículo: Investigación
Fecha de aceptación: 27 de noviembre de 2012
Financiamiento: Universidad de Cundinamarca

Palabras clave: FPGA, inteligencia artificial, procesamiento digital de imágenes, sistemas de reconocimiento de caracteres.

Key words: FPGA, artificial intelligence, digital image processing, character recognition systems.

RESUMEN

En este artículo se presenta la utilización de Matrices Lógicas Programables por Campo (FPGA) para la implementación de un sistema de evaluación de redes neuronales artificiales (RNA), y la aplicación de reconocimiento de patrones de imágenes de caracteres alfabéticos, en el que se utilizan diferentes modelos de redes neuronales. Esta investigación se desarrolló para evaluar el desempeño de las redes neuronales implementadas en la FPGA y para explorar y analizar las características de funcionamiento de la FPGA Spartan-3 XCS200. El proyectó se realizó en cuatro fases, las cuales son: establecer los modelos de redes neuronales, desarrollo de una plataforma de entrenamiento de RNA utilizando Labview, diseño, implementación y verificación de los modelos de RNA, y por último, se establecieron los parámetros y se realizó la evaluación del desempeño de las RNA implementadas en la FPGA Spartan-3 XCS200.

ABSTRACT

This paper presents the use of Field Programmable Gate Arrays (FPGA) to evaluate an Artificial Neural Network (ANN) system, and also to apply image pattern recognition to alphabetic characters in contexts where different neural network models are utilized. This is performed in order to evaluate the performance of the neural networks implemented in the FPGA and also to analyze the Spartan-3 FPGA XCS200 performance. This study has four phases: neural network model selection; development of a training platform using Labview RNA; design, implementation and verification of RNA models; and parameter setup to evaluate RNA Spartan-3 FPGA XCS200 performance.

* * *

1. INTRODUCCIÓN

Una FPGA es un dispositivo lógico programable que se caracteriza por su funcionalidad (flexibilidad y versatilidad), y en la actualidad se encuentra presente en el desarrollo de aplicaciones en diferentes áreas de la ingeniería, por tal motivo se busca abordar este tipo de dispositivos para el desarrollo de una aplicación que nos permita explorar y evaluar esta herramienta tecnológica.

La aplicación seleccionada para el desarrollo del proyecto es el reconocimiento de patrones aplicados a caracteres alfabéticos utilizando redes neuronales artificiales (RNA), y se busca utilizar diferentes modelos de RNA que, dentro de sus aplicaciones, involucren esta aplicación, para posteriormente evaluar y analizar el rendimiento de los diferentes modelos implementados en la FPGA XCS200.

2. METODOLOGÍA

El sistema de evaluación de la Red Neuronal Artificial busca comparar el funcionamiento de diferentes modelos RNA (ver figura 1) implementados en una FPGA Spartan-3 XCS200 cuyas guías de identificación están referenciadas por [1] y [2], con el ánimo de establecer qué modelo de red tiene un mejor rendimiento con relación a la aplicación de reconocimiento de patrones.

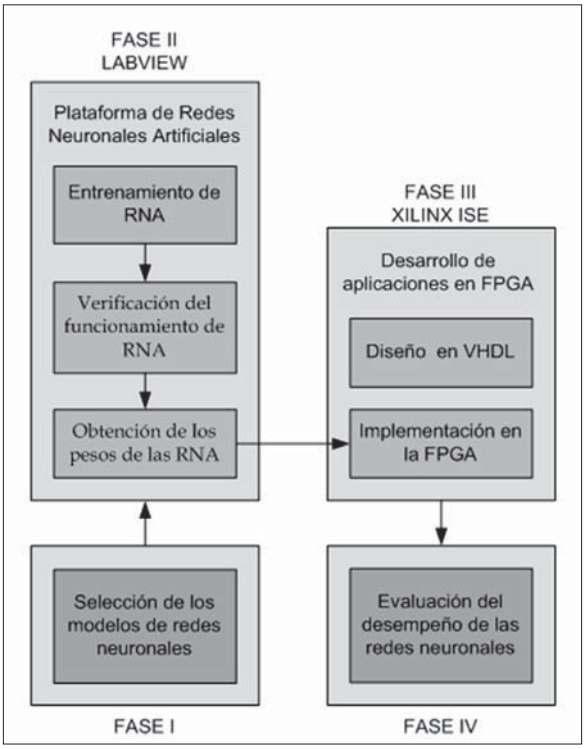

Figura 1. Sistema de evaluación de RNA

Fuente: elaboración propia.

En el proceso de selección de la RNA, se analizaron diferentes aspectos de la topología de las redes neuronales, para lo cual, Hilera José [3] y Freeman James [4] muestran algunas de estas características a tener en cuenta para realizar una buena clasificación, en términos del número de capas ocultas, así como el tipo de datos que va a procesar la red neuronal (redes discretas).

Después de analizar la información recopilada y de realizar un estudio de las diferentes topologías de RNA, teniendo en cuenta lo anterior, los modelos que mejor cumplen con estas características son: el modelo de resonancia adaptativa (ART) y el modelo Hopfield.

2.1 Plataforma de entrenamiento RNA

La plataforma es un software diseñado para realizar el entrenamiento de RNA (ver figura 2) para la aplicación de reconocimiento de patrones enfocado a caracteres alfabéticos, éste es desarrollado en Labview, que es una herramienta de instrumentación virtual que ofrece ventajas para el desarrollo de este tipo de aplicaciones; la plataforma cuenta con una interfaz de usuario que ofrece facilidades para realizar el entrenamiento y, posteriormente, efectuar la verificación del funcionamiento de RNA de una manera práctica, donde se encuentran los controles y las herramientas necesarias para este fin, así como la opción de exportar la información obtenida del entrenamiento de RNA.

Figura 2. Plataforma de entrenamiento de RNA

Fuente: elaboración propia.

La plataforma de entrenamiento cuenta con un módulo que permite realizar el entrenamiento y la verificación del funcionamiento del modelo de resonancia adaptativa, de igual manera, cuenta con un módulo para el modelo Hopfield que cumple la misma función, el módulo de comunicación con la FPGA permite presentar los patrones para ser transmitidos del PC a la FPGA y, por último, la opción de salir para terminar la ejecución de la aplicación.

2.2 Algoritmo de entrenamiento del modelo de resonancia adaptativa (ART)

El proceso de entrenamiento de las RNA, tal como lo anuncia Hilera José [3], consiste en el proceso de aprendizaje por el cual se modifican los pesos de la red en respuesta a una información de entrada, este tipo de mecanismo de aprendizaje se diferencia en cada modelo.

El modelo de resonancia Adaptativa ART es de tipo ON LINE y el número de neuronas y el de los pesos pueden variar durante el entrenamiento y el funcionamiento de la red [4]. La ecuación (1) muestra dicha relación.

$$V_{ij}(t=0)=1 \qquad W_{ij}(t=0)=\frac{1}{1+N} \qquad (1)$$

Para iniciar el proceso de adaptación de los pesos, la dimensión de red dependerá del número de neuronas de entrada y de patrones que se entrenarán inicialmente, cuando se presenta el patrón que se desea entrenar, la información se propaga a través de la conexiones hacia adelante y se almacena en los pesos V_{ij} y los pesos W_{ij} se actualizan a partir de la ecuación (2):

$$W_{ij}(t+1)=\frac{V_{ij}^{*}(t)e_i^k}{\gamma+\sum_{i=1}^{N}V_{ij}^{*}(t)e_i^k} \qquad (2)$$

Cuando la red entra en funcionamiento, la relación de semejanza es el parámetro encargado de

establecer si es necesario que un nuevo patrón sea aprendido por la red. El valor de relación de semejanza que se utilizará es de 90%, ya que los caracteres alfabéticos tienen bastante similitud, por tal razón hay que ser muy selectivos para poder realizar una buena clasificación. Dicho valor de relación de semejanza viene dado por la expresión mostrada en la ecuación (3):

$$R = \frac{\|V_k * X\|}{\|E_k\|} = \frac{\|E_k * V_j\|}{\|E_k\|} \qquad (3)$$

La adaptación de los pesos de la red, cuando se encuentra en funcionamiento, se realiza utilizando la ecuación (2) para las conexiones feedforward W_{ij}, y para las conexiones feedback V_{ij}, lo que se realiza es que el patrón de entrada se almacena en la matriz. La arquitectura de la red se modificará agregando nuevas neuronas en la capa de salida.

2.3 Algoritmo de entrenamiento del modelo Hopfield

La red neuronal tipo Hopfield tiene un mecanismo de aprendizaje *off line* [3] y [4], es decir, que inicialmente se realiza el proceso de aprendizaje, y cuando este procedimiento finaliza, comienza el proceso de funcionamiento de la red, a diferencia del aprendizaje *on line*, donde el proceso de aprendizaje y funcionamiento son paralelos o simultáneos.

El funcionamiento del modelo Hopfield se divide en dos partes: la primera es entrenamiento, en donde se presentan diferentes patrones que son almacenados o memorizados en la red neuronal; y la segunda parte es el funcionamiento, que consiste en que, si se le presenta a la entrada algunas de las informaciones almacenadas, la red se propaga hasta estabilizarse, ofreciendo entonces en la salida la información correspondiente a la información almacenada. Si, por el contrario, la información de entrada no coincide con ninguna de las almacenadas, por estar distorsionada o incompleta, la red se propaga generando en la salida lo más parecido posible a la información almacenada.

Para realizar a la adaptación de pesos a la matriz W_{ij} se realizará a partir de los patrones de entrada representados por los vectores E, la adaptación se realiza mediante la expresión dada en la ecuación (4):

$$W = \sum_{k=1}^{M} [E_k^T E_k - I] \qquad (4)$$

Donde la matriz E_k^Tes la transpuesta de la matriz E_k, e I es la matriz identidad de dimensiones NxN que anula los pesos de las conexiones autorrecurentes.

2.4 Implementación de las RNA en FPGA

Para la implementación de la Red Neuronal en la FPGA [5] se cuenta con la FPGA Spartan-3 XCS200 de Xilinx , y algunos aspectos importantes que se tuvieron en cuenta en la síntesis de la redes neuronales, basados en FPGA, son claramente expuestas por Sund Su Kim [6] y Coric [7]. De igual manera, un ítem importante tomado para el desarrollo del algoritmo de procesamiento de imagen en la FPGA es tomado de Leiner [8], aunque gran parte de los algoritmos desarrollados son de origen propio. El software utilizado es el Xilinx ISE 9,1, que incorpora herramientas para la síntesis, implementación, diseño y programación del dispositivo.

El software de desarrollo permite realizar el diseño utilizando lenguaje de descripción de hardware VHDL, además de permitir realizar programación modular basado en este mismo lenguaje de descripción. También se puede encontrar el lenguaje ABEL (Advanced Boolean Expression Language), VHDL (VHSIC Hardware Description Language) y Verilog. El lenguaje de descripción seleccionado para el desarrollo del proyecto es el VHDL, ya que este lenguaje es uno de los

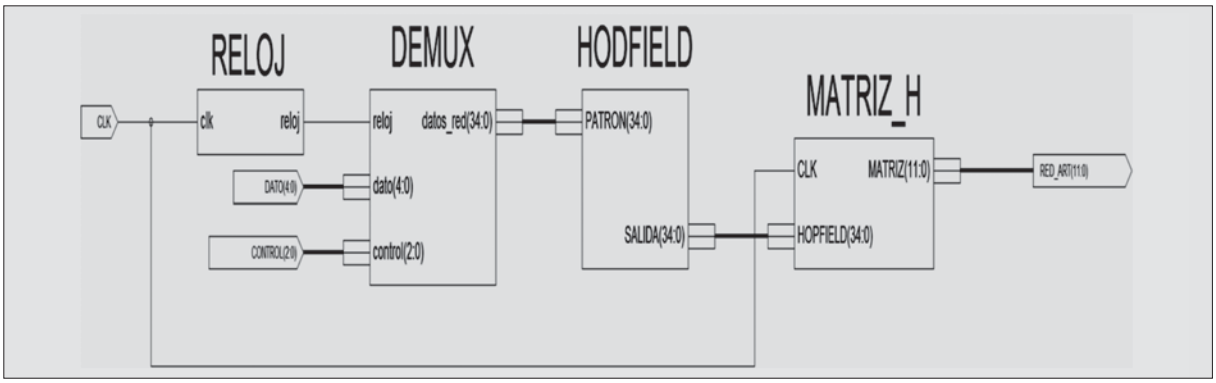

Figura 3. Diagrama completo del modelo ART

Fuente: elaboración propia.

más populares y la información para desarrollar aplicaciones basadas en este es amplia.

2.5 Diseño del modelo de resonancia adaptativa (ART)

El diseño del modelo de ART (ver figura 3) utilizando VHDL, se realizó incorporando el algoritmo de funcionamiento de modelo, en donde es necesario adicionar algunos componentes para el diseño del modelo como: reloj del sistema, demultiplexor y el sistema para la visualización.

El reloj del sistema es el encargado de la sincronización, el demultiplexor es utilizado para ajustar la información que proviene del PC (puerto paralelo) a 8 bits a los 35 bits que se requieren para representar la imagen, el bloque ART tiene incorporado el algoritmo de funcionamiento del modelo de la red neuronal; y por ultimo, se tiene el proceso de la visualización de la imagen que ajusta la señal d salida del modelo ART para representar el patrón en una matriz de 7x5.

3. DISEÑO DEL MODELO HOPFIELD UTILIZANDO VHDL

Para realizar la implementación del modelo Hopfield (ver figura 4) se desarrolla el algoritmo de funcionamiento en VHDL, donde los bloques que

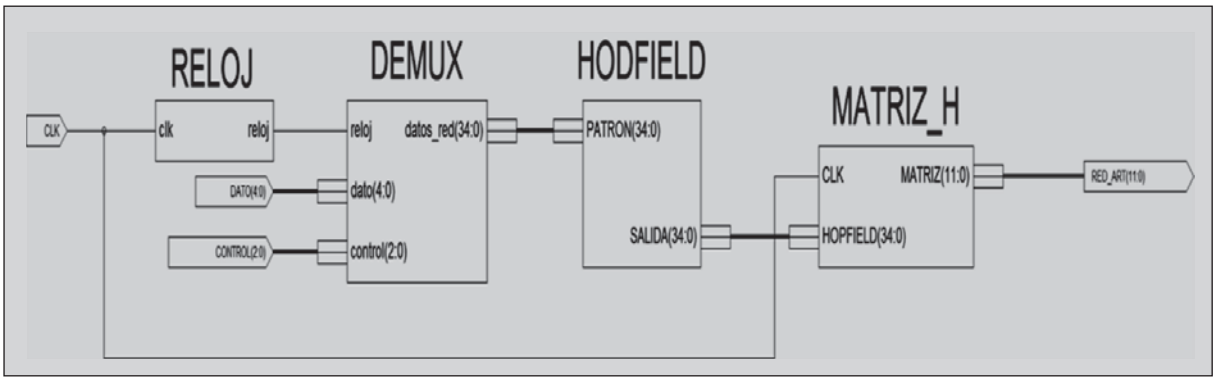

Figura 4. Diagrama completo del modelo Hopfield

Fuente: elaboración propia.

hacen referencia al reloj del sistema y el de multiplexor se utilizan con los descritos anteriormente en el modelo ART (ver figura 3).

En el bloque Hopfield se desarrolló el algoritmo de funcionamiento del modelo de red neuronal utilizando VHDL, en el cual, el módulo de visualización realiza el ajuste de la señal de salida del módulo Hopfield para representarla en un matriz de 7x5.

4. ADQUISICIÓN Y PROCESAMIENTO DE LA IMAGEN

El proceso de adquisición de la imagen se realiza utilizando una herramienta, la cual permite plasmar una imagen de tamaño 350x250 píxeles, posteriormente, se ejecuta un procesamiento de la imagen realizando una caracterización de una imagen obteniendo así un patrón representativo de la original.

4.1 Implementación de las RNA en FPGA

Una vez concluida la etapa de diseño del código en VHDL que describe cada uno de los módulos

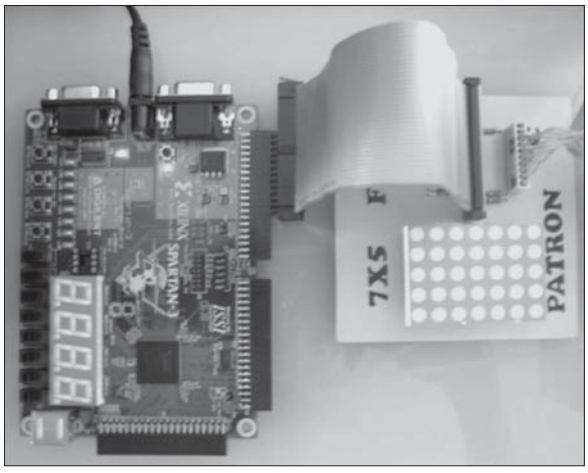

Figura 5. Sistema de evaluación de RNA
Fuente: elaboración propia.

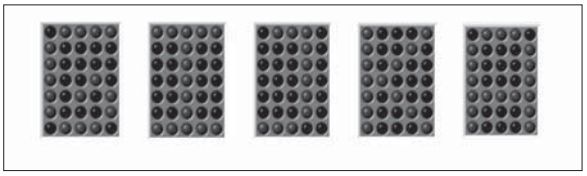

Figura 6. Características de los patrones entrenados en el modelo ART y Hopfield

Fuente: elaboración propia.

necesarios para la implementación de los modelos de redes seleccionados, el software Xilinx ISE utilizado para el desarrollo de aplicaciones basadas en FPGA permite realizar la metodología de diseño necesaria para la implementación, los pasos necesarios para realizar la programación (reconfiguración de hardware) del dispositivo [3].

4.2 Implementación del modelo ART

El modelo ART es entrenado para clasificar diez patrones diferentes, compuestos por pixeles que representan los primeros caracteres del alfabeto, obteniendo una arquitectura del modelo red neuronal de 35 neuronas en la capa de entrada y de 10 neuronas en la capa de salida alcanzando 350 conexiones e igual número de pesos entre las neuronas figura 6.

4.3 Implementación del modelo Hopfield

El modelo Hopfield es entrenado para clasificar cinco patrones diferentes, el número de patrones utilizado se debe a que el autor establece una limitación de la capacidad del número de patrones que se pueden entrenar para obtener una buena clasificación, debe ser menor que el 13.8% del número de neuronas de la capa de entrada, otra limitación que presenta el modelo es que los patrones entrenados deben tener una diferencia igual o mayor a 30 %, teniendo en cuenta lo anterior, los patrones entrenados son: G, I, J, K, I, en la figura 6 se encuentran las características de los patrones.

La arquitectura de red obtenida al realizar el entrenamiento es de 35 neuronas en la capa de entrada y 35 neuronas en la capa de salida, para un total de 1190 conexiones entre la neuronas de la red, vale la pena resaltar que el número de pesos obtenido en el entrenamiento no se modificará si se modifica el número de patrones entrenados en la red.

5. RESULTADOS

La evaluación de RNA se realizó en dos etapas: lo primero, es efectuar la verificación del funcionamiento de los modelos de redes implementadas en la FPGA para, posteriormente, evaluar el desempeño de las redes implementadas con respeto a la aplicación de reconocimiento de patrones.

5.1 Verificación modelo ART

Utilizando el módulo de comunicación para presentarle diferentes patrones a la RNA implementada en la FPGA, los resultados que se obtienen se pueden ver en tabla 1.

El funcionamiento de red neuronal implementada en la FPGA tiene un comportamiento óptimo, comparado con los resultados obtenidos en la simulación. La plataforma sólo tiene problemas en la clasificación de los patrones correspondientes a los caracteres E y F, y los patrones C y G, debido a la similitud que presentan estos. Los resultados mostrados en la tabla 1 corresponden a los obtenidos en la simulación, y se puede considerar que son satisfactorios para los propósitos de la investigación.

5.2 Verificación del Modelo Hopfield

El proceso de verificación del funcionamiento no se logró realizar debido a que la capacidad de la FPGA utilizada para el desarrollo del proyecto no

Tabla 1. Funcionamiento para el modelo de art utilizando patrones con ruido

PATRÓN DE ENTRADA	PATRÓN DE SALIDA FPGA	PATRÓN DE SALIDA SIMULACIÓN
		
		

Fuente: elaboración propia.

cuenta con los recursos lógicos necesarios para la implementación de la red neuronal, según el Report HTML entregado por el software de desarrollo ISE 9,1.

Debido a que no se pudo realizar la implementación del sistema completo, se utilizó un número menor de neuronas para verificar el funcionamiento del modelo Hopfield, los resultados obtenidos al comparar el funcionamiento de cinco neuronas implementadas en la FPGA, con los resultados de la simulación, es que tenían un correcto funcionamiento; debido a esto, se puede decir que el código VHDL diseñado con el fin de describir el funcionamiento del modelo Hopfield es correcto.

Después de realizar un breve análisis al sistema de evaluación, el parámetro más relevante que se estableció es la cantidad de recursos con lo que dispone la FPGA Spartan-3 utilizada para el desarrollo del proyecto. Esto se estableció teniendo en cuenta las características y la arquitectura de las RNA para la aplicación de reconocimiento de patrones, como se muestra en la tabla 2, debido a la cantidad de recursos que necesita cada modelo para ser implementadas en la FPGA.

Tabla 2. Componentes y capacidad de la FPGA Spartan-3

COMPONENTES	DISPONIBLES
Slice Flip Flops	3840
LUTs	3840
Slices	1920
IOBs	173
GCLK	8

Fuente: elaboración propia.

Los recursos utilizados por el modelo ART y el modelo Hopfield se muestran en la tabla 3, en donde se encuentran los recursos usados y el porcentaje de utilización.

Los recursos utilizados por el modelo ART son relativamente bajos, con respecto a la arquitectura de la red neuronal, y se puede establecer que cada neurona utiliza algo más del 2 % de los recursos de la FPGA XC200, en base a esto, es posible entrenar de 30 a 35 patrones adicionales ya que se dispone del 70 % de los recursos de la FPGA XC200.

Para el modelo Hopfield no se puede realizar la implementación del sistema propuesto, debido a que los recursos de la FPGA no son suficientes como se observa en la tabla 4, esto se debe, en parte, a las características de la red como la representación de la información de entrada, que en este caso utilizan valores binarios de 1 y -1, y debido a ésto el valor de alguno de los pesos de la red tiene valores negativos lo que implica un gasto computacional elevado, otro aspecto es la arquitectura de la red que en este caso es de 35 neuronas en la capa de entrada y 35 neuronas en la capa de salida que generan 1190 conexiones entre las neuronas de la red.

Debido a que no se pudo realizar la implementación del modelo Hopfield en la FPGA XC200, se buscaron alternativas de diseño para optimizar el código VHDL y a partir de esto reducir la cantidad de recursos requeridos. La alternativa que se utilizó para realizar la optimización del código VHDL es la herramienta de Xilinx System Generator, el cual permite diseñar una aplicación para FPGA utilizando el simulink de Matlab, y cuyos resultados de requerimientos de implementación se muestran en la tabla 4.

Si se comparan los resultados obtenidos entre la opción A y la opción B, el System Generator realmente permite optimizar los recursos utilizados por la FPGA XCS200 Spartan-3, debido a que se presenta una disminución del 40 % de la opción B con respecto a la opción A, pero la implementación del modelo Hopfield no se puede realizar por que la reducción de los recursos utilizados por la aplicación superan la capacidad de FPGA con la que se cuenta para el desarrollo del proyecto.

Tabla 3. Recursos usados por el modelo ART y Hopfield

COMPONENTES	ART		HOPFIELD	
Slice Flip Flops	490	25 %	5495	286 %
LUTs	106	2 %	106	2 %
Slices	869	22 %	10047	261 %
IOBs	56	32 %	56	32 %
GCLK	2	25 %	2	25 %

Fuente: elaboración propia.

Tabla 4. Recursos utilizados por la opción a y la opción b del modelo Hopfield

Componente	Hopfield A		Hopfield B	
Slice Flip Flops	5495	286 %	4644	241 %
LUTs	106	2 %	109	2 %
Slices	10047	261 %	8376	218 %
IOBs	56	32 %	68	39 %
GCLK	2	25 %	2	25 %

Fuente: elaboración propia.

Debido a que la FPGA utilizada para el desarrollo del proyecto no posee los recursos necesarios lógicos para realizar la implementación del modelo Hopfield, a través de software Xilinx ISE se realizaron simulaciones con el ánimo de establecer que FPGA tiene los recursos necesarios para la implementación, después de realizar diferentes simulaciones, se obtuvo que la FPGA Spartan-3 XCS1000 que dispone de 1000k compuertas, cuenta con los recursos necesarios para realizar la implementación del modelo Hopfield.

6. CONCLUSIONES

La utilización de una FPGA para la implementación de RNA en un hardware, ofrece ventajas para el diseño y desarrollo de este tipo de aplicación, como la disminución del tiempo de desarrollo, la reducción de los costos de fabricación, el buen rendimiento del sistema, una alta escala de integración y de capacidad, y, la más determinante, es que permite la reconfiguración de hardware sin realizar procesos que impliquen grandes cambios de diseño comparados con otras tecnologías como DSP, VLSI ,PCB y ASIC.

El modelo de red neuronal que mejor desempeño demostró es el modelo ART, esto se debe principalmente a que la FPGA Spartan-3 XCS200 contaba con los recursos suficientes para la implementación del modelo de red neuronal y, al realizar la verificación entre la FPGA con respecto a la simulación, el resultado que se obtuvo fue que el sistema implementado presentó un correcto funcionamiento. Por otra parte, el número de patrones que puede clasificar el modelo ART es ampliamente mayor al modelo Hopfield en este caso, aunque este margen se podrá reducir al aumentar las dimensiones del patrón de entrada.

REFERENCIAS

[1] Xilinx, *Spartan-3 Starter Kit Board User Guide, UG130 (v1.1),* Xilinx, 2005.

[2] K. Parnell and N. Metha, Nick, *Programmable logic Design Quick Start Hand Book*, Fourth Edition ISE 5,1i, Xilinx, 2003.

[3] J. Hilera y V.Martínez, *Redes Neuronales Artificiales fundamentos modelo y aplicaciones*, RAMA, 1995.

[4] J. Freeman y D. Skapura, *Redes Neuronales Algoritmos, aplicaciones y técnicas de programación*, Addison Wesley, 1993.

[5] F. Pardo y J.Baluda, *VHDL lenguaje para síntesis y modelado de circuitos.* Alfaomega, 2004.

[6] S. Sund and J. Seul, "Hardware implementation of real time neural network controller with a DSP and FPGA", *IEEE International Conference on Robotics and Automation,* Vol. 5, pp. 3161-3165, April, 2004.

[7] S. Coric, "A neural network FPGA implementation, Neural Network Applications", in *Electrical Engineering 2000. NEUREL 2000. Proceeding of the 5th seminar on Northeastern University*, Boston, 2000.

[8] B. Leiner y V. Lorena, "Hardware architecture for FPGA implemetation of neural network and its application in images processing programmable logic", in *4th Southern Conference on Digital*, pp. 209-212, 2008.