SISTEMA DE DESARROLLO DE REDES NEURALES SOBRE FPGA

Mauricio Francisco Muñoz Quevedo mfmunoz@venus.javeriana.edu.co
Germán Giovanni Rodríguez Ortega grodrigu@venus.javeriana.edu.co
Jorge Francisco García Pabón jgarcia@venus.javeriana.edu.co

Departamento de Ingeniería Electrónica
Pontificia Universidad Javeriana
Santafé de Bogotá, D.C.
Colombia

Abstract

Este trabajo ofrece las ventajas de la lógica reconfigurable, en cuanto a metodología de diseño y a características de arquitectura, para la implementación de un circuito específico que requiere alto nivel de paralelismo. Para esto se usaron FPGAs y se construyó un sistema de desarrollo que permite crear, modificar y entrenar redes neurales digitales con conexiones sólo hacia adelante y perceptrones binarios. La interfaz con el usuario se realiza por medio de un computador personal. Fue probada la escalabilidad del sistema y se diseñó sobre éste una aplicación para el reconocimiento de dígitos.

Palabras claves

FPGA, Redes neurales, sistemas digitales, VHDL, Inteligencia artificial.

1. Introducción

Las redes neurales, modelos matemáticos que buscan simular el funcionamiento del cerebro humano[3, 6, 7, 9, 14] poseen algoritmos en los que se puede apreciar su carácter altamente paralelo. Implementaciones en software utilizando microprocesadores, microcontroladores o DSP, no ofrecen el paralelismo exigido, incluso trabajando con varios de estos dispositivos simultáneamente; por este motivo es necesario pensar en una implementación sobre hardware. En este trabajo se realizaron: descripciones de hardware en VHDL [2, 10] de circuitos de redes neurales digitales con conexiones sólo hacia adelante y perceptrones binarios; y las rutinas en lenguaje de alto nivel para Windows 95/98, que permiten crear, modificar y entrenar dichas redes. La escalabilidad de las redes fue probada configurando varias capas de neuronas, cada una en un FPGA Altera x5200[1, 4, 12], logrando que el tiempo de respuesta se incrementa linealmente con respecto al número de capas. Para validar el funcionamiento se utilizaron varias aplicaciones de funciones digitales sencillas [5] y una aplicación para el reconocimiento de dígitos escritos en una matriz de 4x7 píxeles. Nuestro sistema de desarrollo reducirá el tiempo de diseño y los costos para el montaje de aplicaciones de redes neurales.

2. Planteamiento del problema

Hasta el momento, la mayoría de las simulaciones que se hacen en redes neurales son basadas en software, ya sea en computadores o estaciones de trabajo, o en bajo nivel sobre procesadores, microcontroladores o DSPs, los cuales aunque crean un ambiente adecuado y parecido al de las RNs, estas simulaciones no pueden emular realmente todas las condiciones de trabajo, debido a que su arquitectura de tipo secuencial contrasta con el alto nivel de paralelismo innato de las RNs. Con el objetivo de superar estas limitaciones se propone una solución novedosa, cuya principal característica es el uso de un sistema basado en hardware reconfigurble.

En cuanto a trabajos realizados en otras universidades, se han hecho pocas implementaciones de RNs en hardware utilizando las posibilidades ya mencionadas. Entre éstos se encuentra el realizado en la universidad Virginia Tech, por Keller [8], en el que se implementa una RN sobre un arreglo de 4 FPGAs, para una aplicación de reconocimiento de caracteres. Las diferencias entre el trabajo propuesto en este trabajo y el expuesto en el artículo de Keller residen en dos puntos básicos. En primer lugar, el enfoque de este trabajo de grado es hacia la construcción de un sistema de desarrollo más que hacia la de una aplicación específica. La otra diferencia es su arquitectura, debido a que la topología de la red usada por Keller es fija, el número de circuitos integrados es mayor y, en general, la cantidad, la organización y la clase de componentes internos implementados son distintos.

En el tema de implementación de RNs sobre circuitos de lógica reconfigurable se encuentra el trabajo desarrollado por Marcelo Tossini [11], que se divide básicamente en dos partes una esencialmente de software donde se definen ciertos comandos que permiten definir completamente una amplia gama de redes neurales y en cuanto a la parte de hardware, se implementa una arquitectura ortogonal y unos caminos de datos digitales optimizados de manera que se pueden reutilizar los circuitos y de esta forma generar redes neurales sin limitar la topología. Sin embargo, se sacrifica bastante el paralelismo de las redes neurales por el mismo hecho de tener que reutilizar ciertos componentes para la realización de los cálculos aritméticos.

3. Descripción General

La figura 1 muestra el diagrama en bloques del sistema de desarrollo propuesto. Este sistema de desarrollo cuenta con dos componentes principales: una parte de hardware, en la cual se encuentran los circuitos de operación, la comunicación de éstos con el PC y los circuitos de entrada y de salida; y una parte de software que es la encargada de realizar las funciones relativas a la comunicación con el usuario, a la ejecución de los algoritmos de aprendizaje y a la configuración de los FPGAs. En los párrafos siguientes se explica en detalle cada uno de los bloques.

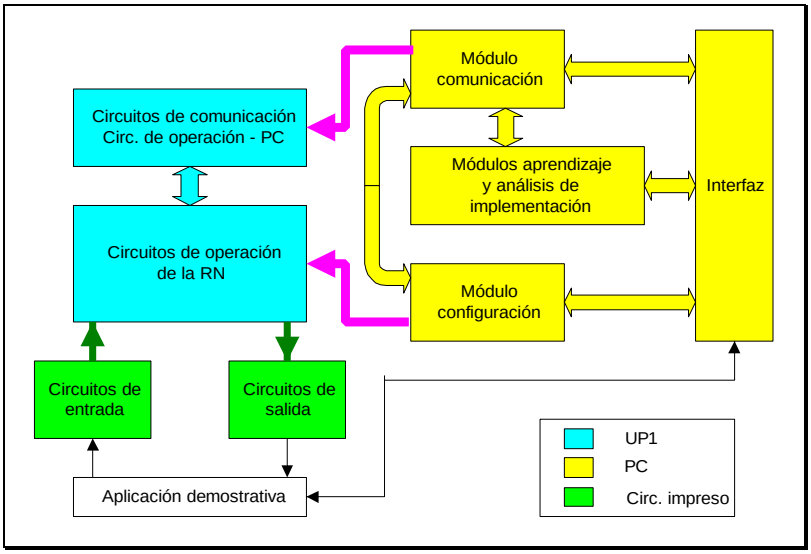

Figura 1 Diagrama en bloques del sistema

Circuitos de operación de la RN. En éstos se encuentran las neuronas, los pesos sinápticos, las conexiones entre neuronas y los circuitos de control para estos elementos. Su diseño se hace usando el lenguaje estándar de descripción de hardware VHDL y siguiendo el flujo de diseño digital con FPGAs[1]. Los circuitos de operación de la RN son implementados en dispositivos EPF10K20RC240-4 de Altera haciendo uso de la característica ISP (*In System Programmability*) que permite reconfigurar el FPGA sin ser desconectado del circuito [1, 13].

La metodología utilizada incluye:

- Diseño modular.
- Uso de las descripciones funcionales y estructurales.
- Estudio de los factores implicados en el resultado del diseño.
  Tiempo de diseño.
  Velocidad – Área del CI.
  Costo.

Circuitos de comunicación entre los C*irc. de operación y el PC*. Reciben las tramas de comunicación provenientes del puerto serial asincrónico de un PC, con información referente a escritura - lectura de pesos, e iniciación - detención del funcionamiento. Estas tramas son procesadas para enviar las señales adecuadas a los circuitos de operación. La metodología para el diseño de estos circuitos de comunicación es la misma que para los circuitos de operación, teniendo en cuenta que son implementados sobre una arquitectura diferente de FPGAs (MAX7128). Incluyen además un circuito basado en el componente MC1489AP, que hace la conversión de niveles de voltaje RS-232 manejados por el PC (-12V y 12V para 'uno' y 'cero' respectivamente) a niveles de voltaje TTL manejados por el MAX7128 (5V y 0V para 'uno' y 'cero' respectivamente).

Circuitos de entrada y de salida. Para la demostración del funcionamiento del sistema de desarrollo se implementaron circuitos sencillos que permiten colocar las entradas a la RN y observar sus salidas de

una forma didáctica. Sin embargo el uso de estos circuitos es opcional ya que pueden ser reemplazados por aquellos que el usuario estime convenientes para su aplicación.

Módulo aprendizaje. Son rutinas independientes de software implementadas en el PC, que corresponden a los siguientes algoritmos de aprendizaje de RNs con conexiones hacia adelante: Perceptrón; Regla delta, Adaline/Madaline o LMS; y Regla delta generalizada o Backpropagation.

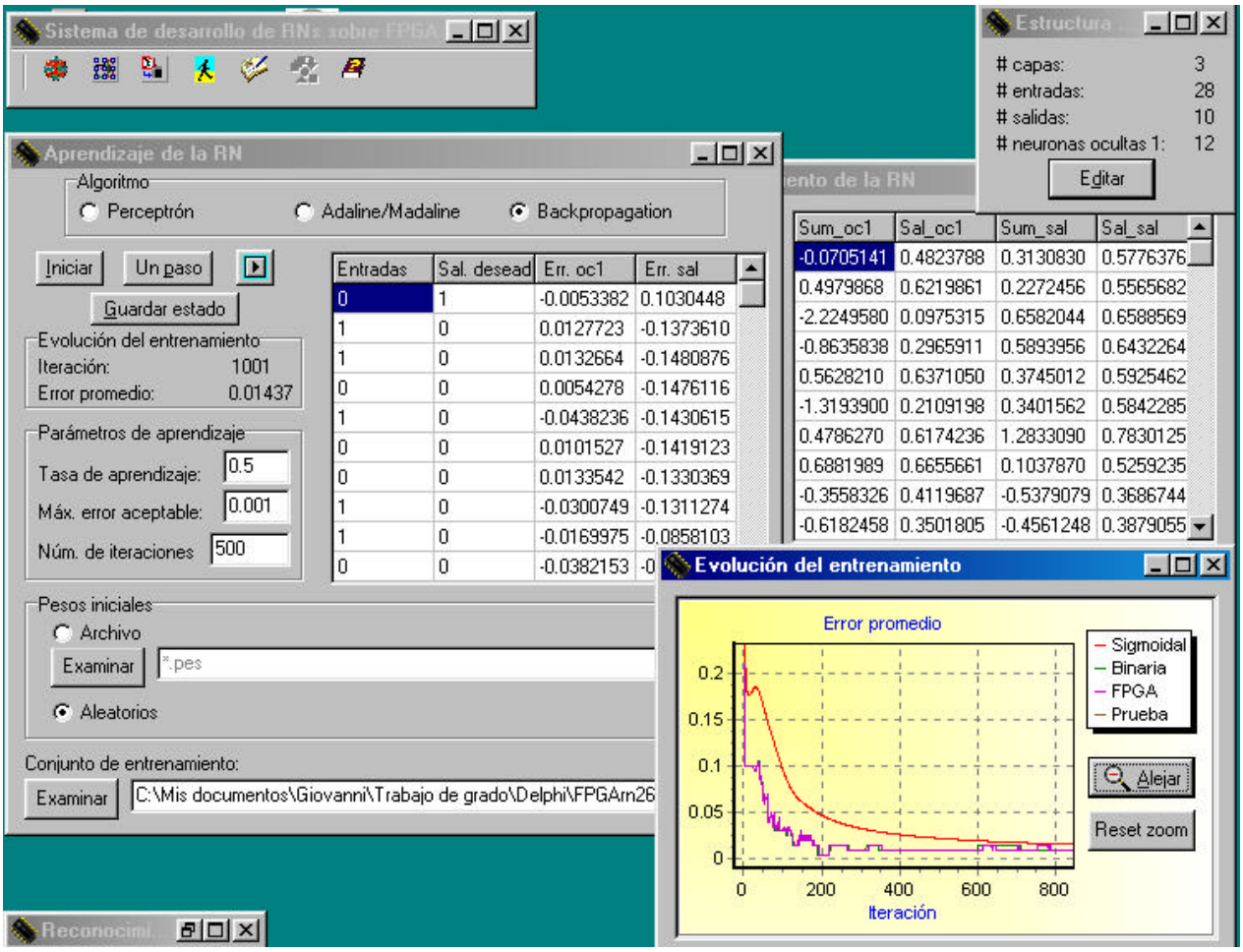

Figura 2 Interfaz del módulo de aprendizaje

En la figura 2 se muestra la interfaz de este módulo. Dependiendo de la estructura escogida (número de capas, entradas, salidas, etc..) el usuario selecciona el algoritmo de aprendizaje, los parámetros (tasa de aprendizaje, máximo error aceptable, número de iteraciones), los pesos iniciales (aleatorios o desde un archivo) y el archivo del conjunto de entrenamiento.

Como resultado, al usuario se le da información que consta de la variable *net* (la suma de los productos de los pesos por las entradas) y la salida de cada una de las neuronas. También es generada una gráfica del error contra el número de iteraciones tanto para los pesos análogos como para los pesos digítales. Esta gráfica es muy útil para saber en qué momento se debe detener el proceso de aprendizaje dado que muestra la forma en que converge el algoritmo.

Módulo análisis de implementación. Toma los resultados arrojados por el proceso de entrenamiento y los transforma para ser implementados en los FPGAs, indicando las acciones realizadas y analizando posibles consecuencias de las transformaciones.

Módulo configuración. La configuración de los FPGAs es realizada por el PC mediante la herramienta programador, que envía las señales IEEE 1149.1 hacia el puerto paralelo de un PC, para que éstas viajen por un cable propietario (Byteblaster) hacia el conector JTAG (Joint Test Action Group, que corresponde al mismo protocolo IEEE 1149.1) de la tarjeta UP1, configurada en ese momento para programar sólo al FLEX10K20 [1]. El módulo configuración se encarga de seleccionar el archivo de configuración apropiado de acuerdo con la estructura de RN especificada por el usuario y de llamar luego al programador para construir el circuito adecuado. En la parte derecha de la figura 3 se puede ver la interfaz del módulo de configuración. En este módulo se pueden configurar el número de capas, de neuronas en cada una de estas capas, así como el número de entradas y salidas del circuito. Está también el campo que muestra el resultado del proceso de configuración.

Módulo de comunicación. Su función es enviar las tramas de comunicación hacia los circuitos de operación. Los tipos de trama que pueden ser enviados son de lectura y escritura de pesos y de iniciación y detención de funcionamiento. Para cumplir con su función, el módulo de comunicación intercambia datos con los módulos de análisis de implementación y de configuración, y se conecta con el puerto serial asincrónico del PC. En la figura 3 se puede observar la interfaz del módulo de comunicaciones (Izquierda). Por medio de éste es posible escribir los pesos sinápticos ya sea en forma individual o desde un archivo. Se puede leer el valor de un peso sináptico de cualquiera de las neuronas de la red y este dato es desplegado en un visualizador siete segmentos de dos dígitos en formato hexadecimal.

A través de esta interfaz se puede enviar la orden de inicio o detención a la red neural.

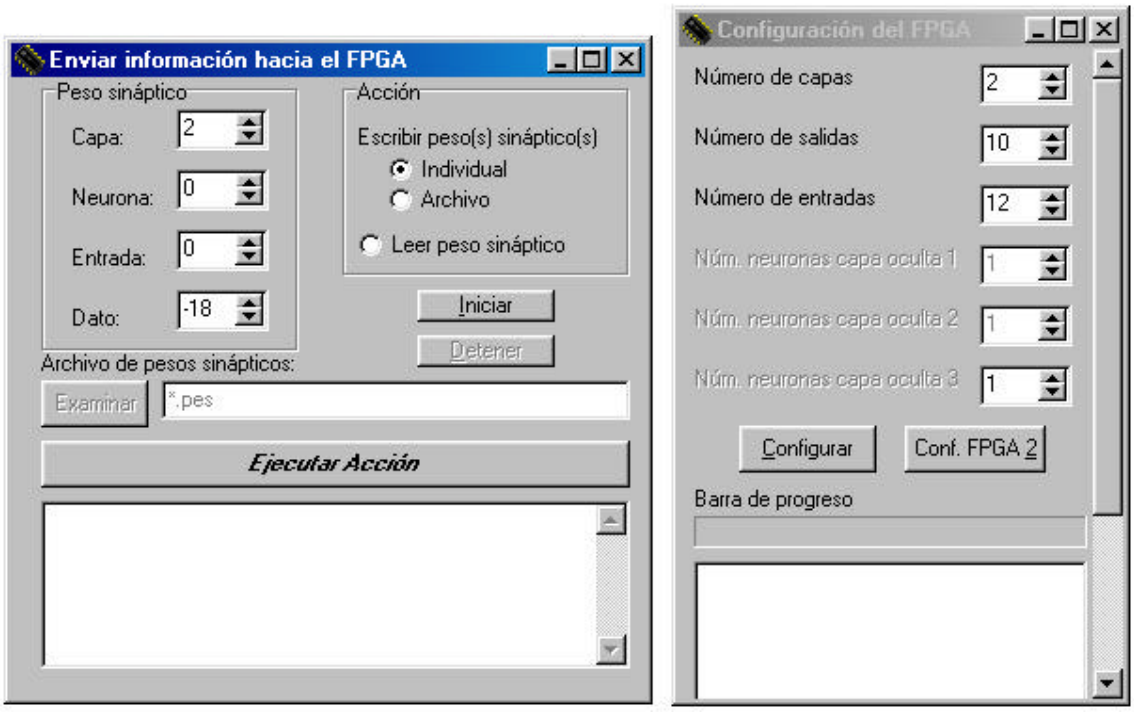

Figura 3 Interfaz de los módulos configuración y comunicación

Módulo interfaz. El enlace entre el sistema de desarrollo y el usuario se lleva a cabo a través de un computador personal, en el que por medio de ventanas con las características del sistema operativo Windows 95/98, se ofrece un programa con presentación gráfica que permite tener acceso a todos los módulos de software del sistema, de una forma amigable. Además se incluye el manual del usuario en la forma de archivo de ayuda, al que se puede acceder desde la ventana principal de la aplicación.

La interfaz diseñada permite al usuario crear, modificar, poner en funcionamiento y obtener información de la RN implementada en hardware. El manual del usuario, que hace parte de los módulos de software como archivo de ayuda, brinda un soporte permanente al usuario, con utilidades de búsqueda y ambiente gráfico amigable.

3.1. Módulos de los Circuitos de Operación de la RN

Son circuitos sincrónicos que usan el mismo reloj de todo el sistema de desarrollo. Están divididos en dos partes, una encargada del procesamiento aritmético a partir de las entradas y de los pesos sinápticos, y otra de controlar los procesos que tienen lugar en la red neural.

En la figura 4 se muestra el diagrama en bloques del módulo de procesamiento aritmético, conformado por una memoria y un sumador de complemento a dos.

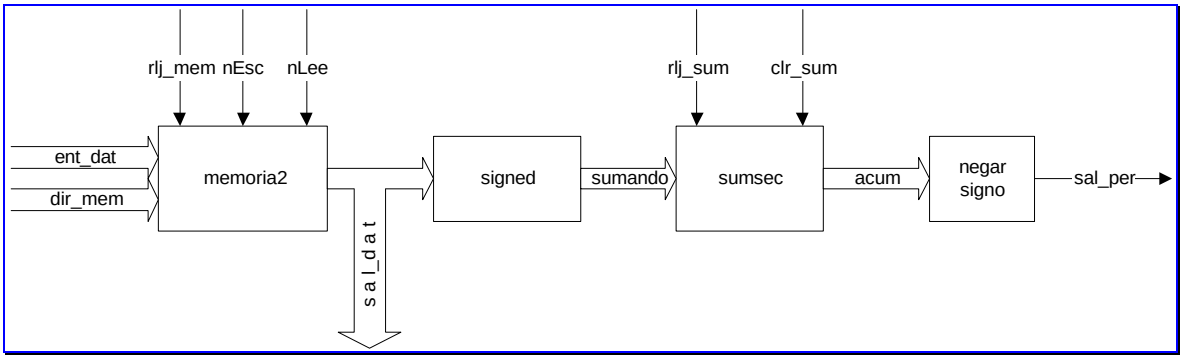

Figura 4 Diagrama en bloques del módulo de procesamiento aritmético

En la figura 5 está el esquema del módulo de control que genera las señales para que los módulos de procesamiento aritmético funcionen de forma simultánea cuando la RN se encuentra en funcionamiento, y que controla la escritura y lectura de pesos sinápticos.

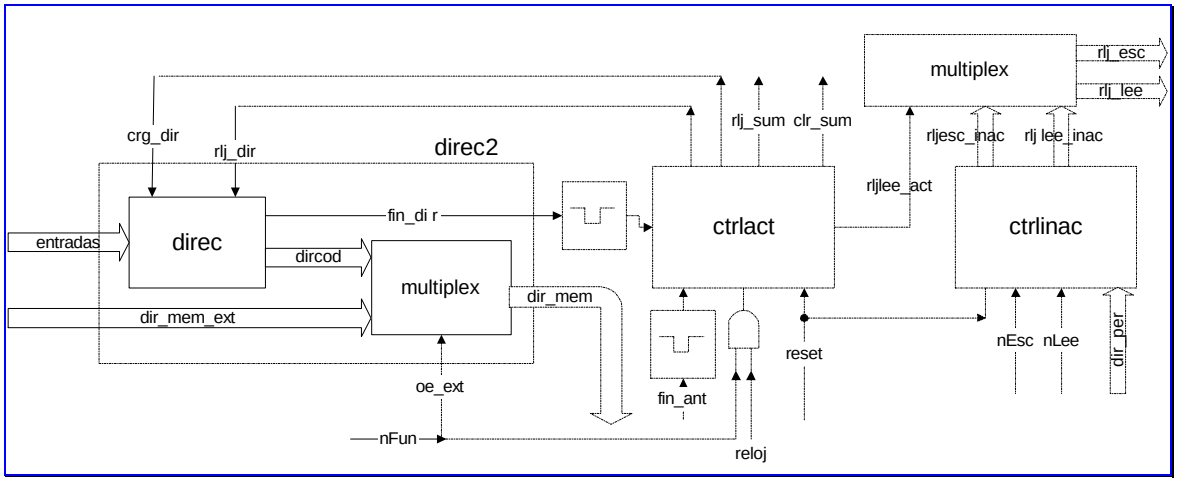

Figura 5 Diagrama en bloques del módulo de control

3.2. Modelos de Perceptrones

Los bloques de memoria dedicados EABs [1] son utilizados como memoria de pesos sinápticos y de acuerdo con ello se consideraron los siguientes modelos:

- Unidades aritméticas combinatorias de una RN. El análisis de las implementaciones hechas demostró la imposibilidad de realizar configuraciones útiles de RNs utilizando este modelo.
- Capa de perceptrones secuenciales que no comparten EABs. Las configuraciones logradas usando este diseño tienen características que permitieron su implementación.

- Perceptrones secuenciales que comparten EABs. Para un mejor aprovechamiento del área provista por el FPGA, se diseñaron estructuras de tipo secuencial que comparten un mismo bloque de memoria, tanto para perceptrones en una capa como para aquellos de capas distintas.
- Implementación de memorias fuera de los EABs. También para hacer un uso eficiente del área se ubicaron memorias usando LABs en lugar de EABs.

3.3. Interconexión de FPGAs

En virtud de la forma como fueron diseñadas las señales de control de entrada/salida para la red neural, se logró conectar varias de éstas para construir una más grande sin recurrir a la utilización de hardware adicional.

Cada FPGA puede contener una o más capas. No es permitida la implementación de un nivel de perceptrones repartido en más de un FPGA.

3.4. Aprendizaje de la RN

El proceso de entrenamiento de la red neural es realizado por el sistema de desarrollo, teniendo en cuenta las características de las RNs implementadas en el FPGA y que los pesos sinápticos son valores enteros de ocho bits en complemento a dos (entre -128 y 127) [8]. Se brinda permanente información gráfica al usuario sobre la evolución del proceso.

4. Intervalos de Operación

El la tabla 1 se muestra las especificaciones máximas de las configuraciones de RN que puede manejar el sistema de desarrollo.

Parámetro	*Valor máximo*
Entradas a una capa	31
Neuronas por capa	24
Capas de neuronas	4

Tabla 1 – Intervalos de Operación

5. Reconocimiento de Dígitos

Utilizando el sistema de desarrollo, fue diseñada una aplicación para el reconocimiento de dígitos escritos en una matriz de 4x7 píxeles. A partir del conjunto de entrenamiento construido, se obtuvieron los pesos sinápticos para una red de 28 entradas, 24 neuronas ocultas y 10 de salida. El programa aplicativo hecho, cuya interfaz se muestra en la figura 6, simula exactamente lo que ocurre dentro de los circuitos de operación implementados en FPGAs.

Figura 6 Reconocimiento de dígitos utilizando el sistema de desarrollo de redes neurales sobre FPGA

Para poder comprobar el correcto funcionamiento del circuito se realizó en paralelo un programa el cual utiliza la misma red neural con el mismo algoritmo de aprendizaje. El hardware consistía en dos tarjetas de desarrollo, la primera tarjeta de desarrollo contiene la primera capa oculta y la segunda en otra. Se tiene una serie de interruptores para simular cada uno de las casillas de la matriz y una serie de diodos para mostrar la salida del circuito. Los resultados obtenidos en el circuito son los mismos que en el programa.

6. Conclusiones y Futuros Desarrollos

El sistema de desarrollo de redes neurales sobre FPGA permite crear y depurar aplicaciones de RNs artificiales, implementándolas sobre un circuito integrado de lógica reconfigurable.

El uso del sistema de desarrollo reduce el tiempo de diseño y de implementación, para sistemas digitales de alto desempeño (como lo son las configuraciones de redes neurales desarrolladas), esto provoca una disminución en los costos.

Las implementaciones de RNs, digitales con perceptrones binarios y conexiones hacia adelante, demuestran funcionar de forma correcta sobre FPGAs, incluso con los compromisos entre velocidad, área y tiempo de diseño; sin embargo, las configuraciones desarrolladas en este trabajo constituyen tan solo un modelo que como tal es susceptible de mejoras.

Las redes neurales construidas sobre hardware incrementan su tiempo de respuesta linealmente con respecto al número de capas y dentro de cada una de ellas el procesamiento es concurrente para todos los perceptrones, ventaja que no es posible alcanzar con una implementación en software que no utilice procesamiento paralelo.

La ISP permite al usuario modificar la estructura de la RN implementada con el fin de evaluar distintas configuraciones de RNs sobre hardware, ya que la operación del FPGA es suspendida durante la programación de nuevos circuitos, sin desconectarlo.

El diseño de aplicaciones específicas como reconocimiento de patrones y sistemas de control, y de prácticas de laboratorio utilizando el sistema de desarrollo de RNs sobre FPGA, permitirá poner de manifiesto las virtudes del sistema y alcanzar un conocimiento profundo de sus características.

Con el uso del sistema de desarrollo, es posible enfocar los esfuerzos al diseño de una aplicación completa para ser simulada en software e implementada en hardware que adicionalmente, si se cuenta con un intercambio de datos con sistemas digitales de algún tipo (microcontrolados, reconfigurables, etc.), se tendría un prototipo completo, con procesamiento neural en hardware.

En cuanto a futuros desarrollos, debido a la modularidad del sistema, se pueden implementar nuevos algoritmos de entrenamiento e incluso acomodar una nueva interfaz a los requerimientos del usuario, sin dejar de usar los módulos de hardware descritos en VHDL.

Al contar con un mayor espacio lógico, es decir si se tienen una mayor cantidad de FPGA o, con el avance de la tecnología, un FPGA con densidades mayores, se puede pensar en cambiar el enfoque netamente binario de las redes neurales para pasar a unas que aunque sigan siendo de tipo digital, presenten comportamiento similar al análogo, haciendo cuantificaciones de los valores de activación y usando funciones de activación distintas a la función paso y realizar operaciones de punto flotante en lugar de las operaciones enteras.

7. Referencias

[1] ALTERA. FLEX 10K Embbeded programmable logic family. Junio de 1996

[2] ARMSTRONG, James R. y GRAY, F. Gail. Structured Logic Design with VHDL. New Jersey : Prentice Hall, 1993. 482 p.

[3] BISHOP, Christopher M. Neural Networks for Pattern Recognition. New York : Oxford University Press, 1995. 482 p.

[4] DORF, Richard C. Y OLDFIELD, John V. Field Programmable Gate Arrays. New York : John Willey & Sons, 1995. 327 p.

[5] EJEMPLO DE simulador. En : http://www.gc.ssr.upm.es/inves/neural/ann2/example/example.htm.

[6] HAYKIN, Simon. Neural networks : a comprehensive foundation. New Jersey : Prentice Hall, 1994. 696 p.

[7] HILERA, José Ramón y MARTÍNEZ, Víctor José. Redes neuronales artificiales: fundamentos, modelos y aplicaciones. s.l. : Addison-Wesley Iberoamericana, 1995. 390 p.

[8] KELLER, Eric Robert. Reconfigurable computing : character recognition with a neural network feed forward stage. En :
http://www.vt.edu:10021/E/ekeller/ann_paper.html. (dic. 1998)

[9] LOONEY, Carl G. Pattern recognition using neural networks: Theory and algorithms for engineers and scientists.

[10] SKAHILL, Kevin. VHDL for Programmable Logic. Menlo Park, CA : Addison-Wesley Publishing Company, 1996. 493 p.

[11] TOSINI, Marcelo A. Implementación en hardware de una ruta de datos segmentada para redes neuronales. Sexto Workshop IBERCHIP. Marzo de 2000.

[12] VILLASENOR, John y HUTCHINGS, Brad. The flexibility of configurable computing. En : IEEE Signal Processing. (sep. 1998)

[13] XILINX. XC4000E and XC4000X Series Field Programmable Gate Arrays. Enero de 1999 (Versión1.5)

[14] ZURADA, Jacket M. Introduction to Artificial Neural Systems. St. Paul, MN : West Publishing Company. 1992.