DESARROLLO DE REDES NEURONALES EN FPGA

Leonardo Navarría, José A. Rapallini, Antonio A. Quijano

Centro de Técnicas Analógico Digitales (CeTAD) - Codiseño Harware/Software (CoHS)
Facultad de Ingeniería. Universidad Nacional de La Plata
Calle 48 y 116, La Plata 1900, Argentina
leonardonavarria@gmail.com josrap@ing.unlp.edu.ar quijano@ing.unlp.edu.ar

RESUMEN

Uno de los motivos más importantes del resurgir de las redes neuronales en la década de los ochenta fue el desarrollo de la tecnología microelectrónica de alta escala de integración o VLSI (Very Large Scale Integration), debido a dos circunstancias. Por una parte, posibilitó el desarrollo de computadores potentes y baratos, lo que facilitó la simulación de modelos de redes neuronales artificiales de un relativamente alto nivel de complejidad, permitiendo su aplicación a numerosos problemas prácticos en los que demostraron un excelente comportamiento. Por otra parte, la integración VLSI posibilitó la realización hardware directa de red neuronal, como dispositivos de cálculo paralelo aplicables a problemas computacionalmente costosos, como visión o reconocimiento de patrones.

1. INTRODUCCIÓN

Durante los últimos 50 años las últimas generaciones han visto como las computadoras han formado parte de la vida cotidiana. Muchas aplicaciones son visibles como ser cajeros automáticos, calculadoras, celulares, en otros casos se incluyen aplicaciones en cuanto instrumental médico, supervisión y seguridad, comunicaciones.

El desarrollo de las ciencias de la computación ha seguido la línea arquitectónica del matemático John Von Neumannn, que en 1947 diseñó una estructura basada en un procesamiento secuencial de datos e instrucciones. Esta estructura sigue rigurosamente un programa secuencial almacenado en la memoria. La arquitectura de Von Neumann se basa en la lógica de procedimientos que comúnmente utilizamos, hallando soluciones parciales a un problema, que luego son utilizadas para lograr la solución final.

Por lo contrario, una visión del cerebro a distancias microscópicas no concuerda con estructuras tipo Von Neumann ni con programas almacenados, lo que sí existe es una computación masiva de unidades concurrentes y redundancias de tareas y conexiones proporcionales a la robustez. El cerebro está especialmente orientado al procesamiento de información sensorial compleja y ruidosa.

Los avances en la arquitectura de los dispositivos FPGAs así como en sus herramientas de diseño han llevado a mejoras significativas en el diseño de sistemas digitales basados en esta tecnología. Esto se debe a que las recientes arquitecturas cuentan con la habilidad para reconfigurar una porción del dispositivo mientras el resto permanece aun operando. Las metodologías y herramientas actuales de diseño están en su mayoría enfocadas a diseños estáticos, lo que limita la adopción de la tecnología de reconfiguración dinámica en el desarrollo e implementación de sistemas. Como una solución a la escasa utilización de diseños reconfigurables dinámicamente, en años recientes han surgido nuevas metodologías y herramientas de diseño a nivel de investigación, que soportan la característica de reconfiguración dinámica en sus diseños. Sin embargo el diseño de sistemas con estas herramientas sigue siendo complejo debido a que presentan algunas desventajas como: requieren alto conocimiento de la arquitectura del dispositivo, la especificación de regiones reconfigurables dinámicamente se lleva a cabo en la última etapa del proceso de diseño (obtención del archivo de configuración), se ejecutan a través de una línea de comandos, el manejo de recursos se realiza a bajo nivel, solo soportan determinadas arquitecturas y no cuentan con un procedimiento de diseño específico.

2. NEUROCOMPUTADORES Y CHIPS NEURONALES

Para realizar redes neuronales artificiales se hace uso de las tecnologías microelectrónicas, desde los emuladores (aceleradores) hardware, especialmente concebidos para la emulación de las redes neuronales, hasta los chips neuronales, más próximos a la realización fiel de la arquitectura de la red. La implementación hardware de las redes neuronales no resulta una cuestión

anecdótica, pues el impacto tecnológico (y económico) de las redes neuronales está siendo, y en muchas aplicaciones, debido a los grandes recursos computacionales necesarios, hay que recurrir a realizaciones electrónicas.

Por neuroprocesador se entiende un dispositivo con capacidad de cálculo paralelo, diseñado para la implementación de redes neuronales artificiales. Puede ser de propósito general o específico, y puede realizarse como un chip neuronal o como una placa aceleradora dependiente de un computador host (Figura 1).

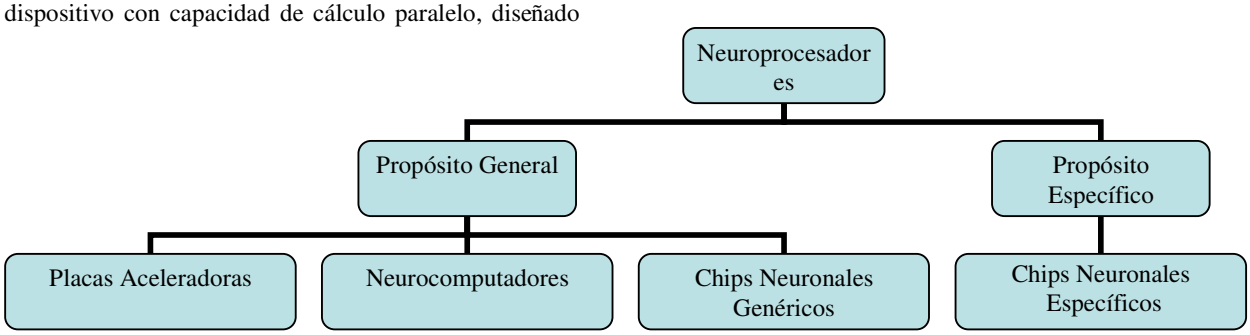

Figura 1

3. MODELO GENERAL DE NEURONA ARTIFICIAL

Se denomina procesador elemental o neurona a un dispositivo simple de cálculo que, a partir de un vector de entrada procedente del exterior o de otras neuronas, proporciona una única respuesta o salida. Una neurona artificial forma la unidad básica del una red neuronal artificial. Los elementos básicos de una red artificial son las entradas **x** de un nodo, indexadas y reciben la señal correspondiente, expresados en forma vectorial como: x= (x1, x2, …. xn). Las variables de entrada y salida pueden ser binarias (digitales) o continuas (analógicas), dependiendo del modelo y aplicación.

Cada señal de entrada pasa a través de una ganancia o peso, llamado peso sináptico o fortaleza de la conexión cuya función es análoga a la de la función sináptica de la neurona biológica. Los pesos pueden ser positivos (excitatorios), o negativos (inhibitorios) y se denotan por w= (w1, w2, ….. wn). El peso sináptico w, define en este caso la intensidad de interacción entre la neurona presináptica y la postsináptica. Dada una durada positiva (procedente de un sensor o simplemente la salida de oirá neurona), si el peso es positivo tenderá a excitar a la neurona postsináptica, si el peso es negativo tendería a inhibirla. Así se habla de sinapsis excitadoras (de peso positivo) e inhibidoras (de peso negativo).

La entrada neta a cada unidad puede escribirse de la siguiente manera:

$$neta_j = \sum_{i=1}^{n} x_i w_i = X.W \qquad (1.1)$$

Como tercer componente tenemos la función umbral o función de transferencia que se encarga de pasar a la salida las señales de entrada acumuladas y sumadas en el nodo sumatorio. La función de activación o de transferencia proporciona el estado de activación actual a partir del potencial postsináptico y del propio estado de activador anterior. Sin embargo, en muchos modelos de redes se considera que el estado actual de la neurona no depende de su estado anterior, sino únicamente del actual.

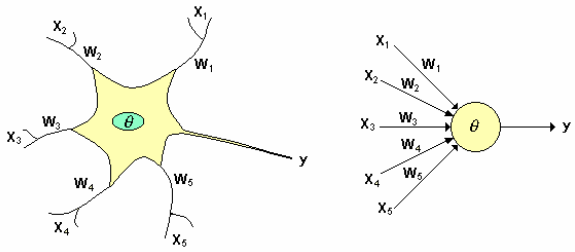

Figura 2: Similitudes entre una neurona biológica y una artificial.

La función de activación se suele considerar determinista, y en la mayor parte de los modelos es monótona creciente y continua, como se observa habitualmente en las neuronas biológicas.

4. ARQUITECTURA DE REDES NEURONALES

Una definición de arquitectura es la topología, estructura o patrón de conexionado de una red neuronal. En una red neuronal los nodos se conectan por medio de sinopsis, la cual determina el comportamiento de la red. Las conexiones sinópticas son direccionales, es decir, la información solamente puede propagarse en un único sentido.

En general, las neuronas se suelen agrupar en unidades estructurales que denominaremos capas. Las neuronas de una capa pueden agruparse, a su vez, formando grupos neuronales (clusters). Dentro de un grupo, o de una capa si no existe este tipo de agrupación, las neuronas suelen ser del mismo tipo. Finalmente, el conjunto de una o más capas constituye la red neuronal.

Se distinguen tres tipos de capas: de entrada, de salida y ocultas. Una capa de entrada o sensorial está compuesta por neuronas que reciben datos o señales procedentes del entorno (por ejemplo, proporcionados por sensores). Una capa de salida es aquélla cuyas neuronas proporcionan la respuesta de la red neuronal (sus neuronas pueden estar conectadas a efectores). Una capa oculta es aquella que no tiene una conexión directa con el entorno, es decir, que no se conecta directamente ni a órganos sensores ni a efectores. Este tipo de capa proporciona a la red neuronal grados de libertad adicionales, gracias a los cuales puede encontrar representaciones internas correspondientes a determinados rasgos del entorno, proporcionando una; \mayor riqueza computacional.

Las conexiones entre las neuronas pueden ser excitatorias o inhibitorias: u peso sináptico negativo define una conexión inhibitoria, mientras que uno positivo determina una conexión excitatoria. Habitualmente, no se suele definir una conexión como de un tipo o de otro, sino que por medio del aprendizaje se obtiene un valor el peso, que incluye signo y magnitud.

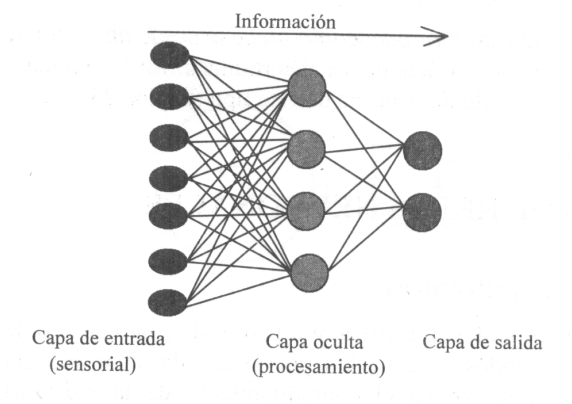

Figura 3: Arquitectura Unidireccional de tres capas: de entrada, oculta y de salida

5. RECUERDO Y APRENDIZAJE

Se distinguen dos modos de operación en los sistemas neuronal al modo recuerdo o ejecución, y el modo aprendizaje o entrenamiento. Este último es de particular interés, pues una característica fundamental de las Redes Neuronales es que se trata sistemas entrenables, capaces de realizar un determinado tipo de procesamiento cómputo aprendiéndolo a partir de un conjunto de patrones de aprendizaje o ejemplos.

Se puede definir el aprendizaje para las redes neuronales como el proceso por el que se produce el ajuste de los parámetros libres de la red a partir de un proceso de estimulación por el entorno que rodea la red. El tipo de aprendizaje vendrá determinado por la forma en la que dichos parámetros son adaptados. En la mayor parte de las ocasiones el aprendizaje consiste simplemente en determinar un conjunto de pesos sinápticos que permita a la red realizar correctamente el tipo de procesamiento deseado.

Al construir un sistema neuronal, se parte de un cierto modelo de neurona y de una determinada arquitectura de red, estableciéndose los pesos sinápticos iniciales como nulos o aleatorios. Para que la red resulte operativa es necesario entrenarla, lo que constituye el modo aprendizaje. El entrenamiento o aprendizaje se puede llevar a cabo a dos niveles. El más convencional es el de modelado de las sinapsis, que consiste en modificar los pesos sinápticos siguiendo una cierta regla de aprendizaje, construida normalmente a partir de la optimización de una función de error o coste, que mide la eficacia actual de la operación la red. Si denominamos $\underline{Wij}(t)$ al peso que conecta la neurona presináptica y con la postsináptica j en la iteración t, el algoritmo de aprendizaje, en función de las señales que en el instante t llegan procedentes del entorno, proporcionará el valor $\Delta\underline{Wij}(t)$ que da la modificación que se debe incorporar en dicho peso, el cual quedará actualizado de la forma

$$\Delta w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t)$$

El proceso de aprendizaje es iterativo, actualizándose los pesos de la manera anterior, una y otra vez, hasta que la red neuronal alcanza el rendimiento deseado.

Existen cuatro tipos de aprendizaje:

1. Aprendizaje supervisado
2. Aprendizaje no supervisado o autoorganizado.
3. Aprendizaje híbrido
4. Aprendizaje reforzado

En la mayoría de los algoritmos de aprendizaje se basan en métodos numéricos iterativos que tratan de minimizar una función coste, lo que puede dar lugar en ocasiones a problemas en la convergencia del algoritmo.

6. IMPLEMENTACION DE UN MODULO DE RED NEURONAL PROGRAMABLE EN UNA FPGA (FPNN)

El primer tema del concepto del FPNA field programmable neural array es el desarrollo de una estructura neuronal que es fácil de mapear dentro del hardware digital, gracias a la topología simple y flexible. La estructura de un FPNA se basa en el principio de los FPGAs: funciones complejas implementadas por un set de recursos programables. La naturaleza y las relaciones de estos recursos de FPNA son derivados desde el proceso matemático que las FPNAs deben obtener.

Un Modulo FPNN (filed programmable neural network) es un FPNA configurado, es decir un FPNA

cuyos recursos han sido configurados de cierta manera, además algunos parámetros deben ser dados para cada recurso de procesamiento. En otras palabras, la capa bajo el FPNN es un FPNA con una topología especificada en los recursos de la red neuronal. Esta FPNN es dada en un camino para especificar cuales de estos recursos interactuaran para definir un comportamiento funcional.

Para la implementación de un FPNN se requiere un bloque predefinido que es simplemente ensamblando sobre una capa de una arquitectura de FPNA (el mapeo se hace de manera directa y paralela sobre la estructura neuronal en sobre la FPGA).

implementación muestra como el tiempo debe ser inherente en los cómputos de FPNN.

En la figura 4 se muestra una implementación de un link (p,n) y todas las señales de para los flip-flops están activas en estado alto.

El bloque select recibe todas las señales de requerimiento que deben ser enviadas por los predecesores del link (p,b) acorde a la topología del FPNA. También recibe una señal "free" que es setenada alta cuando el link (p,n) no produce el llamado al intercambio (handling).

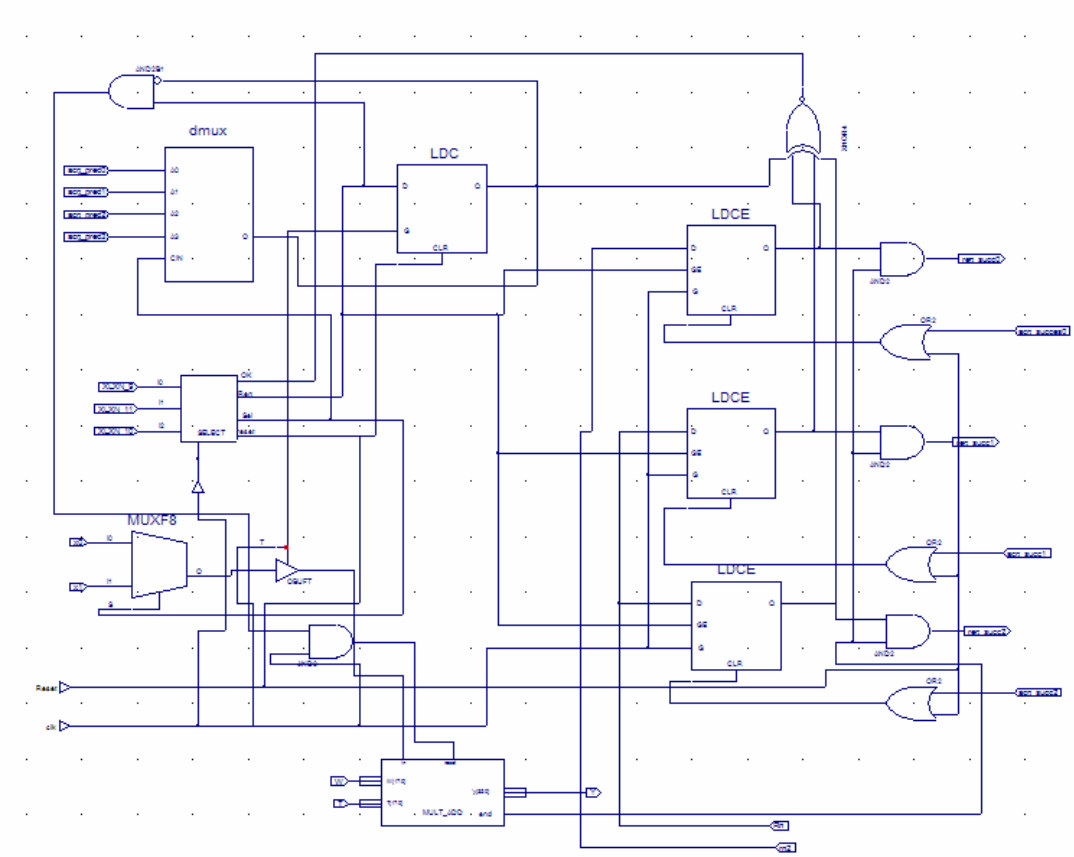

Figura 4

Los recursos para poder implementar una FPNA son el conexionado y la activación. Cada uno de estos recursos corresponde a un bloque predefinido que cumple la función de recurso de procesamiento y protocolo de comunicación asincrónico y todos los bloques son ensamblado de manera acorde al un gráfico de FPNA. Esta implementación debe ser usada para cualquier FPNN derivado del FPNA: algunos elementos deben ser correctamente configurado dentro de cada bloque, tales como multiplexers o registradores. Tales FPNNs deben computar funciones complejas a pesar que su FPNA es simple. Además que este método de implementación es flexible y compatible con las exigencias del hardware. En una manera mas general esta

La señal start es una salida que es setada alto durante un ciclo de reloj cuando el proceso de request comienza. La señal acq es una señal de reconocimiento que es ruteada hacia el recurso que es enviada a la señal de solicitud actual. La señal sel codifica el numero que la señal de solicitud. Controla la entra al mux y el reconocimiento de dmux.

Este bloque MULT_ADD recibe el valor almacenado en el registro X (valor de la solicitud). También recibe Wn(p) y Tn(p) y procesa la transformación Wn(p)X+Tn(p) del link.

La salida de señal Ready es puesta en alta cuando la señal y(1…nb) contiene el valor de salida del link (p,n).

Cuando el bloque OBUF tiene el valor de la entrada, un set de slip- flops almacena la señal de solicitud que sera enviada al link sucesor (p,b). Estos slip-flops son reseteado cuando el correspondiente reconocimiento ha sido recibido. La señal free es reseteada cuando todos los reconocimientos han sido recibidos. La salida efectiva de la señal se mantiene baja hasta que la señal Reddy no este activa.

El gráfico de señales de la Figura muestra el ejemplo de sucesivas solicitudes procesada por el link (p,n) con 4 recursos predecesores y 4 sucesores y con un rn(p)=1, Rn(p,s0) = Rn(p,s2)=1 and Rn(p,s1)= Rn(p,s3)=0. El bloque MULT_ADD es asumido para procesar el resultado con 4 ciclos de reloj, corresponde el uso de un multiplicador semi paralelo donde cada operando es partido en 2 para que tal multiplicador ejecute 4 productos de b/2 bits operando de manera secuencia y siendo procesados por un multiplicador paralelo de b/2 bits, y luego sumado de manera correcta para obtener el valor de productos de un valor de b bits).

Cuando todos los sucesores están libres (por ejemplo están habilitados para enviar de manera inmediata los reconocimientos), una petición de proceso requiere 5 ciclos de reloj, lo cual significa que el protocolo involucrado en el computo de FPNN que solo cuesta un ciclo de reloj. Los requerimientos del bloque necesarios para establecer la comunicación son solo 6 CLBs en la Spartan 3E de Xilinx.

El cambio principal con respecto a las conexiones está limitado a los operadores aritméticos, entonces in y fn esta procesados en lugar de transformarse en un enlace de comunicación. Los operadores ITER y OUTPUT son usados en lugar de MULT_ADD. Estos bloques ejecutan el procesamiento iterativo del procesamiento de la salida. ITER es la función in de entrada que es aplicada a la entrada y al la variable del intervalo para actualizarse posteriormente. OUTPUT es la función fn que es aplicada al final del estado de una variable interna para obtener la salida. El cómputo de OUTPUT solo ocurre cuando los valores han sido procesados por ITER.

Un registro de desplazamiento hacia atrás es requerido para el establecimiento de la comunicación, así el protocolo asincrónico requiere 6 CLBs para la activación (si in ≤ 15) en lugar de los 5 que requería para lograr una comunicación. Una ves mas el costo de un protocolo de un FPNN asincrónico con respecto a los operadores aritméticos requeridos por la activación es insignificante.

8. CONCLUSIONES

Los FPNA han sido redefinidos para poder armar las topologías de de hardware simples con un complejo sistema de arquitectura de una red neuronal compleja gracias a los esquemas de procesamiento que crean numerosas conexiones virtuales con el uso de pocos enlaces del dispositivo, la aritmética y la estructura de la red neuronal. Este paradigma define el modelo neuronal cuyo poder de cómputo es similar a la red neuronal a pesar de la simplificada estructura que se encuentra bien emplazada por la implementación del hardware.
Tanto FPNA como FPNNs son ámbitos de trabajo para procesamiento de redes neuronales y una eficiente manera de adaptar una red neuronal a un hardware digital. La implementación de una red neuronal FPNA en un FPGA ha provisto de una alta eficiencia a las implementaciones basadas en un FPGA.
El inconveniente mas importante aparece en el aprendizaje de un FPNN debido a que los recursos de neuronas se ven decrementados por las conexiones de los pesos.

9. REFERENCIAS

[1] L. J. Navarría, J.A. Rapallini, A.A. Quijano, "Diseño reconfigurables en filtros de baja frecuencia" *XIV Workshop IBERCHIP IWS'2008*, Puebla, México, Febrero de 2008

[2] L. J. Navarría, J.A. Rapallini, A.A. Quijano, "Sistemas Reconfigurables" Agosto 2007, Trabajo Final, Facultad de Ingeniería UNLP.

[3] David. M. Skapura, Building Neural Networks, New York: Addison Wesley Professional, 1996.

[4] Redes Neuronales y Sistemas difusos: Martin del Brio, Bonifacio, Sanz Molina, Alfredo, 2da. Edicion Alfaomega Grupo Editor, 2002