VHDL Model of configurable neural networks applied to decoding in cognitive radio

Cecilia E. Sandoval-Ruiz*

Dirección de Postgrado, Facultad de Ingeniería, Universidad de Carabobo, Valencia, Venezuela.

Abstract.-

In the present investigation, the components of the coding stage in digital communication systems and their implementation with neural networks are studied, under the cognitive radio approach. The design method consisted of identifying an alternative configuration of the RNA based on the architectures of the encoders. Next, the modeling of the components was done in VHDL hardware descriptor language, for the types of dynamic linear, multilayer auto-regressive, multilayer parallel networks. Finally, the Reed Solomon decoder (7,3) was trained, applying a multilayer network, which validated the correction of errors, these results can be extrapolated for hybrid encoders. Efficient implementation models were established, considering the parallel processing. The main contribution consisted in the generalized neuro-models in VHDL, for the treatment of the codes, which can be reconfigured in circuit and adjust their parameters in an adaptive way, according to the requirements of the application.

Keywords: reconfigurable neural networks; signal processing; adaptive training; Reed Solomon decoder; neuro-cognitive; VHDL; FPGA.

Modelo en VHDL de redes neuronales configurables aplicadas a decodificación en radio cognitivo

Resumen.-

En la presente investigación se estudian los componentes de la etapa de codificación en sistemas de comunicación digital y su implementación con redes neuronales, bajo el enfoque de radio cognitivo. El método de diseño consistió en identificar una alternativa de configuración de las RNA basada en las arquitecturas de los codificadores. Seguidamente, se realizó el modelado de los componentes en lenguaje descriptor de hardware VHDL, para los tipos de redes lineal dinámica, multicapa auto-regresiva, multicapa paralela. Finalmente, se entrenó el decodificador Reed Solomon (7,3), aplicando una red multicapa, con lo que se validó la corrección de errores, a través de simulación, estos resultados pueden ser extrapolados para codificadores híbridos. Se lograron establecer modelos eficientes de implementación, considerando el procesamiento paralelo. El principal aporte consistió en los neuro-modelos generalizados en VHDL, para el tratamiento de los códigos, que pueden ser reconfigurados en circuito y ajustar sus parámetros de manera adaptativa, de acuerdo a los requerimientos de la aplicación.

Palabras clave: redes neuronales reconfigurables; procesamiento de señales; entrenamiento adaptativo; decodificador Reed Solomon; sistemas neuro-cognitivo; VHDL; FPGA.

Recibido: agosto 2017
Aceptado: noviembre 2017

1. Introducción

Actualmente, se presentan diversos diseños que integran módulos de procesamiento inteligente, aplicados a las etapas de los sistemas de comunicación, considerando la complejidad asociada a estos módulos, se han planteado trabajos que utilizan

*Autor para correspondencia
Correo-e: cecisandova@yahoo.com (Cecilia E. Sandoval-Ruiz)

herramientas alternativas para el procesamiento de los datos recibidos. Uno de los conceptos que están asociados al manejo eficiente de los canales de comunicación corresponde a los sistemas de radio cognitivo, en los que se estudian los algoritmos y herramientas de aprendizaje que pueden realizar la asignación óptima del espectro radio eléctrico, considerando las redes neuronales como una de éstas [1].

De esta manera, los sistemas inteligentes, que abordan la capacidad de adaptación de sus módulos modificando parámetros para el aprovechamiento de las condiciones del entorno, en este caso el canal de comunicación, requieren de una etapa de percepción, análisis, reconocimiento, decisión y procesamiento, todas estas funciones están relacionadas con las redes neuronales, sin embargo, en esta investigación se tratará el procesamiento neuronal de las señales. Entre las investigaciones encontradas, que proponen soluciones a partir del uso de redes neuronales, algoritmos genéticos y aprendizaje profundo [2], se encuentran diseños de diversos módulos de decodificación de canal, tales como el código Vitervi [3], códigos de bloques [4] y Turbo Códigos [5], siendo este método de decodificación una alternativa eficiente, que se puede combinar para el abordaje de códigos híbridos, basados en neuro-decodificadores [6].

En tal sentido, se propone la presente investigación a fin de abordar un método de diseño con redes neuronales en VHDL (*VHSIC – Very High Speed Integrated Circuit – Hardware Description Language*), como alternativa a módulos de comunicación complejos. Este temática resulta inédita con respecto al tratamiento de redes neuronales reconfigurables en hardware, que permita proponer una plataforma de comunicación inteligente. En este trabajo se exploran los conceptos de sistemas inteligentes de comunicación, tratando aspectos tales como esquemas de codificación y modulación reconfigurables, con arquitectura selectiva y adaptativa, a través del proceso de aprendizaje del módulo usando datos de entrenamiento en el encabezado de la información a transmitir, basado en la técnica de codificación con símbolos pilotos para estimar el modelo de referencia PSAC (*Pilot Symbol Assisted Coding*), el cual permite conocer las características del canal y tomar decisiones [7].

2. Antecedentes

En trabajos previos se han desarrollado modelos neuro-adaptativos en lenguaje descriptor de hardware aplicados a sistemas sostenibles [8], en esta investigación se considera el estudio de los codificadores 2D-RS modelados para tecnología FPGA (*Field Programmable Gate Array*) [9]. Siendo la etapa de decodificación altamente compleja, se ha planteado abordarla con redes neuronales artificiales, bajo el criterio de descripción de hardware en VHDL. En este orden de ideas, se parte por la interpretación del codificador Reed Solomon – *RS (7,3)* – como modelo de análisis, basando la configuración de la red en la estructura del codificador. En primer lugar, se mantiene su arquitectura LFSR (*Linear Feedback Shift Register*) [10], para una red neuronal multicapa dinámica, donde el módulo de multiplicación por los coeficientes del polinomio generador de código, puede ser tratado como una sub-red neuronal para la operación del multiplicador en campos finitos de Galois GF(2^m) estudiado en [11] y finalmente, reconociendo una estructura circuital fractal [12] en la red neuronal para el codificador RS.

Para el diseño se requiere el estudio de codificadores de bloque como el código Reed Solomon y códigos compuestos de estos [9], el análisis de los criterios aplicados en moduladores configurables [13], como elementos de partida para el desarrollo de los módulos complementarios, desde un entrenamiento supervisado. La investigación parte de la descripción en VHDL de diversos codificadores [9, 14, 15], el estudio de arreglos circuitales basados en sistemas de funciones iteradas para hardware [12], la relación entre los generadores de códigos [10] y la aplicación de conceptos de modelado neuro-adaptativo. En este punto, se han estudiado la arquitectura de las redes neuronales, de acuerdo a las diversas configuraciones y aplicaciones. Partiendo del principio de correspondencia, se ha re-interpretado el modelado de los codificadores para ser configurados a través de una red neuronal.

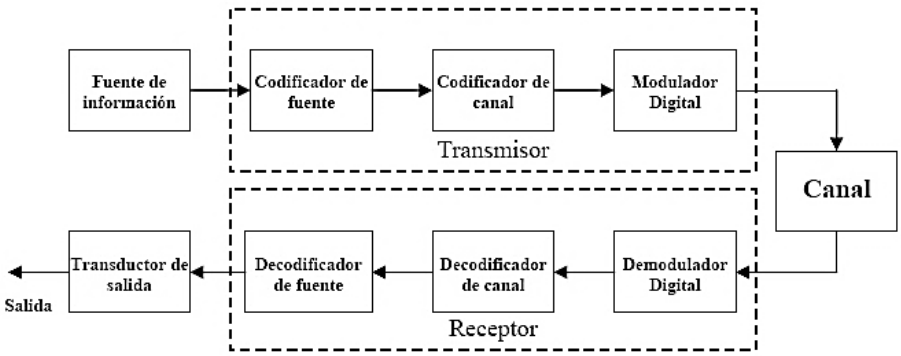

Figura 1: Diagrama de Bloques de un Sistema de Comunicación Digital.

3. Fundamento del procesamiento de señales con redes neuronales para radio cognitivo

En [1] indica que, de acuerdo con la Administración Nacional de la Información y las Comunicaciones (NTIA), de Estados Unidos, la *Radio Cognitiva*, es un sistema que detecta su entorno electromagnético de operación y puede ajustar de forma dinámica y autónoma sus parámetros de operación de radio para modificar la operación del sistema, maximizar el rendimiento, reducir la interferencia y facilitar la interoperabilidad. En estos sistemas se puede aplicar el aprendizaje supervisado de las redes neuronales para la definición de sus funciones, éste se usa cuando en el entrenamiento se conoce los datos de salida del sistema.

El estudio parte desde los componentes de un sistema de comunicación digital (Figura 1), en trabajos previos [16] se plantea la configuración de estos módulos, con la tecnología FPGA para su configuración, a través de lenguaje descriptor de hardware VHDL, destacando el nivel de complejidad propio de la etapa del receptor, por lo que en esta investigación se plantea una alternativa aplicando redes neuronales.

Entre los módulos de los sistemas de comunicaciones se han considerado el codificador de fuente, con código de longitud ajustable [14], esto representa para la red neuronal una estructura reconfigurable, con un número de neuronas de salida variable de acuerdo a la palabra de código, y en el caso del decodificador, las neuronas de la capa de entrada puedan interpretar el dato recibido.

Seguido de un módulo de codificación de canal, en esta etapa, se puede presentar la concatenación entre códigos, como una técnica muy práctica para obtener un código de longitud suficientemente alta y una capacidad correctora extremadamente elevada, eso se logra utilizando múltiples niveles de codificación. Para el procesamiento de los datos se puede seleccionar la configuración generalmente sobre dos niveles, porque con este tipo de concatenación se logran buenos resultados, en esta oportunidad se han considerado los códigos Reed Solomon & Viterbi. En tal sentido, se analiza la configuración de los códigos de bloque considerando la importancia de los códigos Reed Solomon y sus componentes, así como la concatenación de estos códigos en modelos más complejos, como los códigos 2D-RS [9] y con mayores requerimientos en la etapa de decodificación.

En el caso del modulador se ha tomado el concepto de modulación digital en VHDL para radio cognitivo [13], donde a partir de las condiciones del canal se realiza un análisis de datos y se selecciona el esquema de modulación óptimo, en esta etapa una red neuronal puede realizar el tratamiento de la señal con dos posibles modos de implementación: (1) demodulador reconfigurable compuesto por un conjunto de redes preentrenadas ASK, π/4PSK, FSK, OFDM, etc. y una etapa de reconocimiento, la cual pueda conmutar entre los esquemas de modulación de acuerdo a la señal recibida, (2) demodulador adaptativo, en el que se selecciona una red dinámica recurrente adaptativa, ajustando los parámetros de la red a

las características de la señal recibida, estos son re-entrenados en circuito a partir de símbolos conocidos, en un encabezado de trama para entrenamiento con el uso del principio PSAC.

4. Metodología

El método de diseño consistió en identificar estructuras de similitud entre la arquitectura de estos y las topologías de redes neuronales [17], se seleccionaron dos tipos de códigos por su amplia utilización e importancia, estableciendo la relación con los modelos neuronales con: (*a*) el procesamiento secuencial del código convolucional (secuencial), (*b*) el procesamiento paralelo de los códigos de bloques Reed Solomon, que se puede adaptar a las características de la red neuronal a aplicar. Seguidamente, se realizó la descripción de comportamiento en VHDL de los tipos de redes estudiadas para la implementación, analizando el algoritmo de entrenamiento para VHDL. De esta manera, se entrenó el decodificador Reed Solomon (7,3), aplicando una red multicapa MPL, para validar la corrección de errores, a través de simulación.

Codificador-decodificador convolucional con RNA

Se analizan los códigos convolucionales [18] , definidos por una palabra *C(x)*, la cual se genera a través de la suma módulo 2 de los bits, de los últimos K mensajes, teniendo en su arquitectura líneas de retardos – TDL (*Time Delays Line*), que almacenan los k bits de cada mensaje. El código recibe el nombre de convolucional porque en el caso de *k=1*, la palabra de código se puede expresar de la forma dada por: $C(x) = G(x) * D(x)$, siendo *D(x)* el mensaje o datos a codificar y *G(x)* la función de transferencia asociado al código, es decir la secuencia de generación del código, donde se aplica la suma de convolución discreta entre *G(x)* y *D(x)*. Estos pueden ser configurados con una red neuronal por presentar una arquitectura similar a las redes lineal dinámica, con una sola neurona, que incluye una línea de retardo TDL para el manejo de los bits a codificar. En el caso del decodificador Viterbi, se establece la secuencia de estados de acuerdo a las entradas, analizando la relación de correcciones en función de los datos recibidos *R(t)* y los datos retrasados *R(t-1)*, a partir del diagrama de estados de la Figura 2.

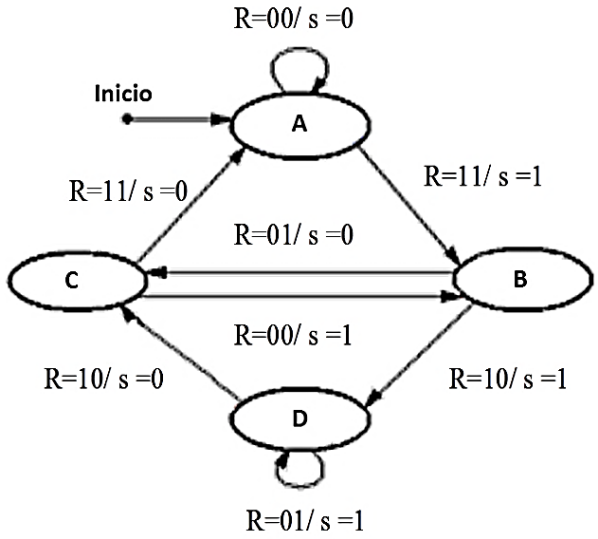

Figura 2: Diagrama de estados del decodificador Viterbi.

Para entrenar la red para el decodificador Viterbi, se deben establecer las entradas y estados presentes, la salida (Target) y estados futuros. Si el código presenta errores, la red puede solucionarlos conociendo las entradas futuras, en ese caso *R1(k)* y *R0(k)* serán las entradas próximas, en tanto que *R1(k-1)* y *R0(k-1)* las entradas actuales recibidas, es decir que la secuencia de entrada y secuencia de estados manejará con *k-1* estados presentes y *k* estados futuros. *Start* permite definir el estado inicial de ABCD, que corresponde a los valores que se inicializan en los TDL, en este caso Ai= 1 0 0 0. La ecuación (1), define la relación para la red diseñada para el decodificador.

$$\begin{aligned} s(k) =& R1(k)w10 + R0(k)w00 + \\ & R1(k-1)w11 + R0(k-1)w01 + \\ & + A(k-1)wA1 + B(k-1)wB1 + \\ & C(k-1)wC1 + D(k-1)wD1 + b \end{aligned} \quad (1)$$

Donde *R1* y *R0*, corresponden a las entradas de la red, datos recibidos, en la muestra actual *k*, en tanto que *A, B, C, D* son los estados realimentados, la matriz *w*, corresponde a los pesos sinápticos de la red y *b*, la polarización de la red. El análisis de esta secuencia permite establecer la data

de entrenamiento para el decodificador Viterbi (Tabla 1).

Tabla 1: Datos de entrenamiento del decodificador Viterbi.

R1'	R0'	R1	R0	A	B	C	D	A	B	C	D'	S
x	x	0	0	1	0	0	0	1	0	0	0	0
x	x	1	1	1	0	0	0	0	1	0	0	1
x	x	0	1	0	1	0	0	0	0	1	0	0
x	x	1	0	0	1	0	0	0	0	0	1	1
x	x	0	0	0	0	1	0	0	1	0	0	1
x	x	1	1	0	0	1	0	1	0	0	0	0
x	x	0	1	0	0	0	1	0	0	0	1	1
x	x	1	0	0	0	0	1	0	0	1	0	0
0	1	0	1	1	0	0	0	0	1	0	0	1
1	0	0	1	1	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	0	1	0	0	0	0
1	1	1	0	1	0	0	0	1	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0	1	1
1	0	0	0	0	1	0	0	0	0	0	1	1
0	0	1	1	0	1	0	0	0	0	1	0	0
1	1	1	1	0	1	0	0	0	0	1	0	0
0	0	0	1	0	0	1	0	1	0	0	0	0
1	1	0	1	0	0	1	0	1	0	0	0	0
0	1	1	0	0	0	1	0	0	1	0	0	1
1	0	1	0	0	0	1	0	0	1	0	0	1
0	0	0	0	0	0	0	1	0	0	1	0	0
1	1	0	0	0	0	0	1	0	0	1	0	0
0	1	1	1	0	0	0	1	0	0	0	1	1
1	0	1	1	0	0	0	1	0	0	0	1	1

Éste puede ser tratado como una máquina de estado con realimentación de los estados de salida, definiendo una arquitectura de red neuronal recurrente, como se muestra en la Figura 3.

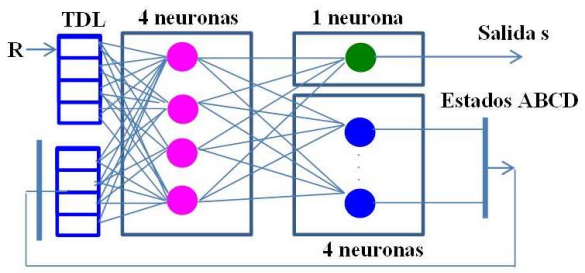

Figura 3: Modelo Neuronal del decodificador Viterbi.

Codificador-Decodificador Reed Solomon con RNA

Veamos el modelo del codificador Reed Solomon como una red dinámica de *n-k* capas (Figura 4), cada una de estas capas presentan un procesamiento asociado al multiplicador en campos finitos (particular), y un sumador módulo-2, con *delay* de las entradas y unos pesos sinápticos dados por uno para la entrada de la capa anterior (al sumador) y el coeficiente correspondiente de la capa a procesar, que se opera con la entrada común de la red neuronal.

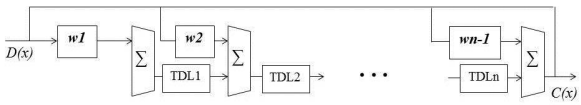

Figura 4: Modelo neuronal del codificador Reed Solomon.

En el diseño del multiplicador en campos finitos de Galois GF(2^m), se consideró la multiplicación para los primeros 16 elementos, partiendo de un conjunto de coeficientes pre-definidos, esto por su utilidad para la codificación Reed Solomon, siendo la Tabla 2 la representación de los productos a entrenar en la red neuronal.

Esto se puede interpretar como una red anidada, es decir que el bloque definido como la matriz de pesos sinápticos, no corresponde a un producto convencional sino a una sub-red neuronal, encargada del procesamiento de los datos de entrada con los coeficientes del polinomio generador *G(x)*, dando como resultado una red fractal [12], con lo que se presenta un enfoque teórico versátil para diversas aplicaciones como los códigos Reed Solomon. De esta manera se pueden definir dos propuestas de configuración del codificador, a través de una Red Dinámica con TDL y multiplicador neuronal para implementación fractal, asociado al modelo del codificador, que permite simplificar el proceso de diseño y entrenamiento de la red, o bien, una red multicapa MPL para implementación paralela, donde la respuesta está directamente asociada al entrenamiento de la red, contrastando la eficiencia hardware y la velocidad de procesamiento de los datos. Estos modelos son importantes para el análisis de la etapa de decodificación. Es así como el modelo neuronal se perfila como una alternativa apropiada, partiendo de la tabla de entrenamiento del decodificador (Tabla 3), para la generalización y corrección de errores por parte de la red diseñada.

Tabla 2: Data de entrenamiento del multiplicador en campos finitos GF.

#	Data															
1	59	13	104	189	68	209	30	8	163	65	41	229	98	50	36	59
2	118	26	208	103	136	191	60	16	91	130	82	215	196	100	72	118
3	77	23	184	218	204	110	34	24	248	195	123	50	166	86	108	77
4	236	52	189	206	13	99	120	32	182	25	164	179	149	200	144	236
5	215	57	213	115	73	178	102	40	21	88	141	86	247	250	180	215
6	154	46	109	169	133	220	68	48	237	155	246	100	81	172	216	154
7	161	35	5	20	193	13	90	56	78	218	223	129	51	158	252	161
8	197	104	103	129	26	198	240	64	113	50	85	123	55	141	61	197
9	254	101	15	60	94	23	238	72	210	115	124	158	85	191	25	254
10	179	114	183	230	146	121	204	80	42	176	7	172	243	233	117	179
11	136	127	223	91	214	168	210	88	137	241	46	73	145	219	81	136
12	41	92	218	79	23	165	136	96	199	43	241	200	162	69	173	41
13	18	81	178	242	83	116	150	104	100	106	216	45	192	119	137	18
14	95	70	10	40	159	26	180	112	156	169	163	31	102	33	229	95
15	100	75	98	149	219	203	170	120	63	232	138	250	4	19	193	100
16	151	208	206	31	52	145	253	128	226	100	170	246	110	7	122	151

Tabla 3: Salida del Decodificador RS.

Palabra Recibida	Palabra Decodificada
1 3 7 0 1 1 5	1 3 7
1 2 3 7 6 4 5	1 2 3
2 3 4 1 6 7 5	2 3 4
5 6 7 6 5 1 6	5 6 7
1 1 1 7 2 6 2	1 1 1
2 4 6 5 7 3 1	2 4 6
2 1 3 2 2 7 7	2 1 3
2 4 1 4 4 0 7	2 4 1
2 5 2 2 0 6 1	2 5 2
2 6 3 5 6 6 2	2 6 3
2 7 4 5 3 1 6	2 7 4
3 1 2 4 6 5 7	3 1 2
3 2 3 3 0 5 4	3 2 3
3 3 4 3 5 2 0	3 3 4
3 4 5 0 6 4 0	3 4 5
3 5 6 6 2 2 6	3 5 6
3 6 7 1 4 2 5	3 6 7
4 1 2 1 4 3 1	4 1 2
4 2 3 6 2 3 2	4 2 3
4 3 4 6 7 4 6	4 3 4

Descripción de las Redes Neuronales en VHDL

El método empleado para el diseño de los módulos en VHDL, correspondientes a los componentes de las redes neuronales, consistió en la descripción generalizada de los tipos de redes, considerando su arquitectura, características de las capas, función de salida de las neuronas en cada capa y modo de entrenamiento. En el caso de describir las redes en hardware, la clasificación por tipo de función, pasa a ser una tabla de búsqueda y las capas reutilizan componentes definidos. Es así como se realizó una configuración generalizada, en la que se pueden definir el número de neuronas por capas, retardos, interacción entre capas, tipo de función de salida de las neuronas, entre otras. De acuerdo con esto, se describió una red neuronal de tres capas, usando la plataforma de desarrollo ISE 11 de Xilinx (Figura 5).

Las operaciones se definen en la descripción del neuro-decodificador, éste sería el archivo principal que establece la correspondencia entre los puertos de la red, asignando las señales a los puertos de cada componente, allí se realizan las operaciones y devuelve la salida de cada neurona, para el procesamiento de esas señales en los componentes de la siguiente capa. Por otra parte, la expresión *port map*, se encarga de hacer la asignación de cada señal externa del módulo principal (en este caso la red neuronal) a los puertos definidos en el componente, siendo las salidas de estos, definidas como entradas-salida para ser reasignadas en el módulo principal.

Lo que se trata de destacar es la posibilidad de manejo de las diversas redes en hardware:

1. En la multicapa tenemos una red definida de modo paralelo, las señales de entrada serán procesadas de forma concurrente en los componentes.
2. Las redes dinámicas, tendrán asociadas componentes secuenciales (registros), que son manejados por la señal de reloj *clk*, para sincronización del sistema.
3. En el caso de las redes adaptativas, el cálculo de los pesos se realizará sobre el hardware, siendo un entrenamiento secuencial, dentro de una red que puede ser paralela en su arquitectura.

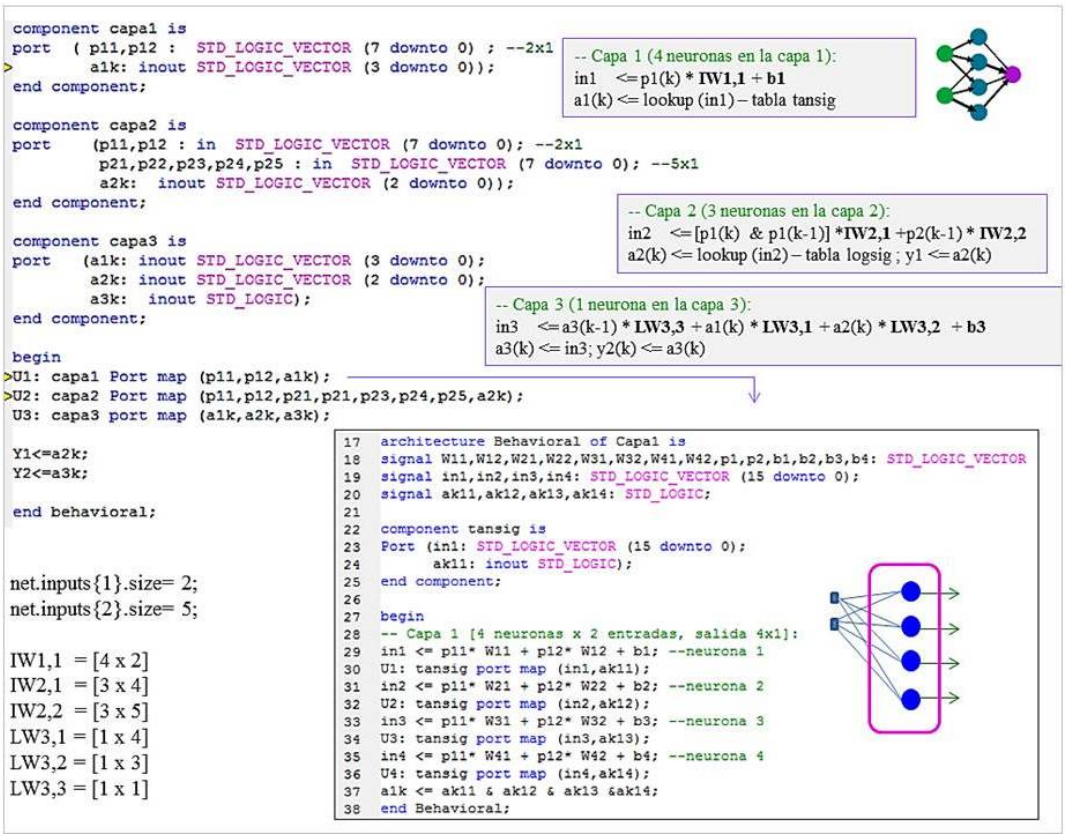

Figura 5: Descripción VHDL de la red neuronal multicapa.

4. En el caso de las redes híbridas se tendría que definir una función particular para la capa no supervisada. Todas estas configuraciones pueden ser definidas para las aplicaciones acá estudiadas.

Finalmente, la red adaptativa, que ajusta los parámetros en el circuito (en línea como se menciona para el caso de redes implementadas en hardware), se debe crear el código para el entrenamiento, y para cada patrón de entrada ajustar los pesos en ese módulo, estos nuevos pesos calculados se le reasignan al componente (las neuronas modificadas). En ese caso, el código en VHDL para definir el algoritmo de ajuste de pesos por corrección de error (en redes supervisadas), estaría ejecutándose para cada nuevo patrón de entrada a la red. Los parámetros de pesos y *bias* pueden modificarse como señales en el componente. Ahora, si se requiere una reconfiguración de la arquitectura se puede modificar de forma dinámica el mapa de bits que define el circuito (programado) de la FPGA. Dando como resultado el diagrama esquemático de la red, presentado en la Figura 6.

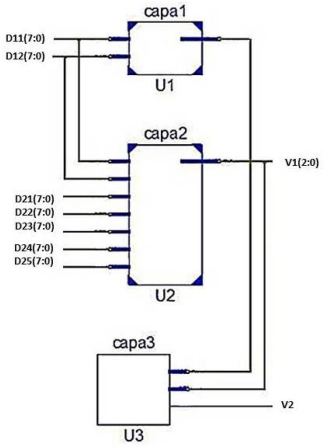

Figura 6: Esquema de configuración de una red neuronal multicapa.

Algoritmos de Entrenamiento de la Red Neuronal en VHDL

En el entrenamiento supervisado (con *targets* conocidos), para el caso de redes lineales de una

sola capa, se aplica la regla de actualización del algoritmo LMS (Least Mean Square), basado en el método de descenso de gradiente SGD (*Stochastic Gradient Descent*), lo que permite calcular los parámetros de la red neuronal (*pesos y bias*), al momento de entrenar la red, con el objetivo de minimizar el error de la salida con respecto al *target*. Este proceso puede tratarse como un entrenamiento por lotes (para la data de entrenamiento, en tiempo de configuración) o un entrenamiento incremental (presentando cada patrón, en tiempo de operación de la red). Este algoritmo permite la configuración de los parámetros de la red en hardware, ya que pueden ser descritos como una rutina de operaciones en VHDL, a partir de su diagrama de flujo (ver Figura 7).

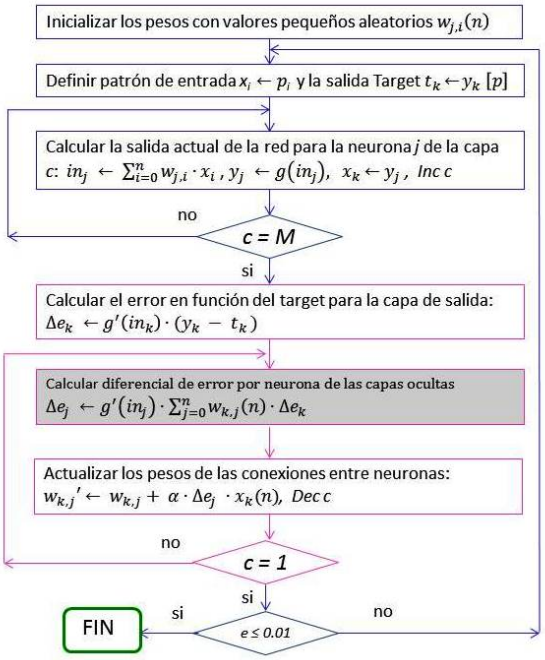

Figura 7: Algoritmo de Entrenamiento para redes multicapa.

Este algoritmo de entrenamiento puede ser configurado en VHDL para la ejecución secuencial del cálculo de los parámetros de la red multicapa, con retro-propagación del error, donde la actualización de los pesos sinápticos y polarización de la capa de salida es calculada directamente del error de la red, en tanto que para las neuronas de la capa oculta se calcula un error diferencial en función de la derivada de la función de salida de la neurona, con el cual se ajustan los pesos de estas neuronas ocultas, como se presenta en la Figura 8.

```
ENTRENAMIENTO ADAPTATIVO:
–Rutina de entrenamiento de los q patrones

x(n) <= leer puerto;
– patrón a entrenar (vector de R entradas)

q <= q + 1;
– incrementa el número de patrones

y(n) <= w(n) * x(n) + b(n);
– calcular la salida de la red

e(n)<= t(n) - y(n);
– error de la neurona para el patrón n

ea <= e(n) + ea;
– error acumulado

– cálculo del error para el patrón x(n) , t(n)
– con la matriz de pesos w(n) actual
For i=1 to R
– para cada entrada i del patrón n

– F[e(n)], vendrá dado por la derivada de la función
–de salida para las neuronas de la capa oculta

wi(n+1) <= wi(n) + u. xi(n). F[e(n)];
– w(n) es el vector de 1 a R elementos

b(n+1) <= b(n) + u. e(n);

Next;
```

Figura 8: Algoritmo de Entrenamiento de la Red Neuronal.

5. Resultados del Decodificador Reed Solomon (7,3)

Al momento de describir la arquitectura de la red neuronal para hardware, podemos establecer este tipo de clasificaciones de acuerdo a las características y comportamiento. A la vez que se está independizando el manejo de las redes de una herramienta específica, en el caso de estos modelos circuitales, como se muestra en la Tabla 4.

Luego del diseño, se realizó la configuración de la red, en la que se definieron las siete entradas compuestas por tres símbolos de datos y cuatro símbolos de redundancia, y tres neuronas de salida correspondiente a los datos decodificados, teniendo así tres neuronas en la capa de salida (función *lineal*) y se seleccionó cuatro neuronas para la capa oculta (función *sigmoide*), obteniendo como resultados del entrenamiento los parámetros de la red neuronal (Tabla 5).

Con esta red se logra la decodificación y una buena generalización, es decir que decodifica

Tabla 4: Esquemas de Configuración de las RNA en VHDL.

Aplicación	Tipo de Red	Código VHDL
Modelado Dinámico del Código	Multicapa Dinámica	yp: in std_logic_vector (7 downto 0): ... -- d1: TDL port map (in,out); c1: capa_sigma port map (*yp,u,a1*); n1: neurona port map (pi,wi,bi,y1); ... nS: neurona port map (pi,wi,bi,yS); c2: capa_lineal port map (*a1, ym*);
Codificación Convolucional	Lineal Dinámica adaptativa	yp: in std_logic_vector (7 downto 0): ... d1: TDL port map (in,out); -- wij: alg port map (wd,xi,yi); -- wn+1 <= wn+coef∗xi∗en ... nS: neurona port map (pi,wi,bi,yS); c2: capa_lineal port map (*a1, ym*);
Modelado Paralelo del Código	Multicapa MPL	yp: in std_logic_vector (7 downto 0): ... c1: capa_sigma port map (*yp,u,a1*); n1: neurona port map (pi,wi,bi,y1); ... nS: neurona port map (pi,wi,bi,yS); c2: capa_lineal port map (*a1, ym*);

Tabla 5: Parámetros de la RNA(4,3) del Decodificador RS(7,3).

Pesos de la Capa Oculta (7 Entradas / 4 Neuronas)						
0.28	-0.63	1.00	-0.78	0.25	-0.58	0.79
0.35	-0.38	-0.51	0.79	0.48	0.73	-0.45
-0.33	0.89	0.90	-0.59	-0.16	-0.28	1.09
-0.46	-0.04	1.32	-0.40	0.30	-0.36	-0.10
Pesos de la Capa de Salida (4,3)						
0.51	3.47	0.20	0.23			
0.25	0.04	0.98	0.24			
2.44	0.11	1.84	1.82			
Polarización de la Capa Oculta (4 Neuronas)						
-0.62	0.08	-0.17	0.34			
Polarización de la Capa de Salida (3 Neuronas)						
1.08	1.05	0.87				

palabras de código que no han sido entrenadas, sin embargo su respuesta cuando se simula el ruido en el canal, sobre algún símbolo, solo da la respuesta correcta si éste se presenta en la trama de redundancia. Por lo que se requirió re-diseñar la red para lograr la corrección de datos, agregando neuronas en la capa oculta, obteniendo la salida corregida, con veinte neuronas, como se observa en la Figura 9.

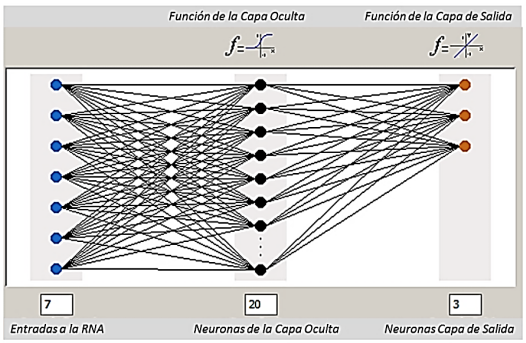

Figura 9: Configuración de la red neuronal MPL.

La Tabla 6 presenta los parámetros de la red neuronal entrenada como decodificador RS(7,3).

Tabla 6: Parámetros de la RNA(20,3) del Decodificador RS(7,3).

Pesos de la Capa Oculta (7 Entradas / 20 Neuronas)						
0.31	-0.65	-1.45	-0.73	1.06	-0.09	-0.36
-0.80	-0.33	-0.50	0.47	-0.75	-0.31	-0.64
1.12	0.15	-0.22	0.98	-0.75	-0.20	0.67
-0.17	-0.64	-0.01	-0.54	1.13	-0.77	2.40
-0.93	0.67	-0.40	0.52	0.66	-1.55	0.35
0.05	0.29	-0.27	1.83	-0.62	-1.15	-1.15
2.11	-0.42	0.55	-0.86	-0.28	2.23	-0.72
-0.61	-0.33	0.25	0.71	2.06	0.64	-0.91
0.05	0.21	0.15	-0.84	-0.98	0.08	1.60
1.04	0.31	0.26	0.14	1.20	-0.50	-1.80
2.11	-0.35	-0.31	0.44	-0.51	-1.44	1.49
1.49	-0.30	0.02	-0.44	-0.74	-1.03	-1.44
0.38	1.79	-0.96	0.86	-1.21	0.31	-0.63
0.95	-0.72	-1.20	-0.03	0.99	-0.23	0.74
-0.27	-0.48	0.51	-1.15	-0.47	-0.79	0.65
-0.20	0.08	2.65	-0.95	-1.13	-0.31	1.28
0.68	-0.35	1.48	0.03	-1.66	-1.66	0.76
-1.17	2.11	1.94	0.12	-0.72	-0.58	-1.72
1.35	0.63	-2.16	-0.74	-0.74	0.70	0.45
-0.10	-0.13	0.03	2.00	0.37	-0.76	-0.70
Pesos de la Capa de Salida (20,3)						
1.82	-0.12	-1.90	-1.13	0.20	-0.41	1.94
-0.25	1.14	1.45	1.28	-0.98	0.44	0.69
-1.09	1.07	0.95	-0.28	0.34	1.34	
-0.76	0.27	1.69	0.66	0.01	-2.11	-1.75
1.04	0.58	1.52	2.00	0.89	1.46	1.08
-1.16	0.61	2.04	1.93	-1.07	-0.11	
-0.74	0.54	0.08	2.13	1.34	0.67	2.68
-1.53	1.46	0.45	-0.24	0.85	0.28	0.80
0.22	1.47	1.47	1.30	-0.96	-1.65	
Polarización de la Caja Oculta (20 Neuronas)						
0.99	-0.21	-0.40	-1.16	0.31		
0.38	-0.04	0.91	0.65	-0.21		
0.38	0.45	0.05	0.28	-0.16		
-0.76	-0.77	-0.86	0.31	-0.12		
Polarización de la Capa de Salida (3 Neuronas)						
-1.15	0.08	0.01				

Los parámetros se encuentran representados por las matrices de pesos sinápticos y polarizaciones de las neuronas, en cada capa. Para esta configuración de la MPL con 20 neuronas en la capa oculta y 3 neuronas en la capa de salida, definida como una RNA (20,3).

Se realizó un conjunto de pruebas, introducien-

do símbolos con ruido en alguna de las posiciones de la palabra de código (una entrada con uno de los símbolos de datos errado), en todos los casos se obtuvo la salida correcta, de acuerdo al target. Es decir, se realiza la corrección de los errores, por parte de la red neuronal MPL diseñada como decodificador Reed Solomon (7,3).

Una de las pruebas consistió en introducir un error en el tercer símbolo de datos, para la trama de datos [3.0 2.0 3.0], se colocó como entrada de la red neuronal la trama de datos [3.0 2.0 1.0], con los símbolos de redundancia correspondientes en los patrones de entrada, como se presenta en la Figura 10.

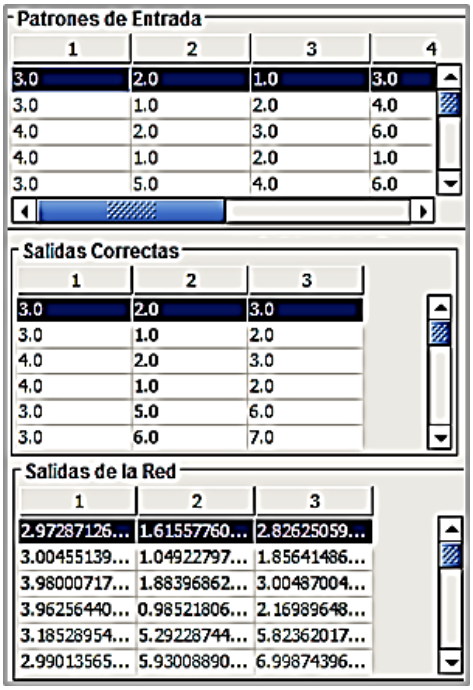

Figura 10: Resultados de la RNA (20,3) del Decodificador Reed Solomon.

Se puede observar en la Figura 10 la corrección de los datos, obteniendo en las salidas de la red la trama de datos [2.97287126 1.61557760 2.82625059], donde se corrige el tercer símbolo que corresponde con la salida correcta de la trama [3.0 2.0 3.0].

6. Conclusiones

En esta investigación se consideró el diseño de las etapas de procesamiento de códigos de canal, se partió del diseño de los codificadores en base a modelos circuitales, usando redes neuronales dinámicas recurrentes multicapa, obteniendo resultados válidos a partir de nueve neuronas en la capa oculta, destacando que el algoritmo de entrenamiento aumenta su complejidad para la implementación en hardware. En el caso del código convolucional, se realizó el análisis del diagrama de estados para definir la tabla de entrenamiento, de la máquina secuencial. Este modelado en VHDL de las etapas del sistema de comunicación, permitió establecer una correspondencia entre la topología de las redes neuronales con la aplicación a implementar.

En el caso del codificador Reed Solomon aplicando multiplicadores en campos finitos [11], modelados a través de una red neuronal interna para el procesamiento de la señal de entrada a la red neuronal codificadora, se evidencia un modelo fractal [12], con una red neuronal compuesta, que permite una fácil descripción y modelado, disminuyendo las exigencias del entrenamiento. En segundo lugar se realizó el diseño y entrenamiento de una red multicapa sin retardos ni realimentación, para establecer el modelo paralelo del circuito decodificador Reed Solomon, lo que permitió un avance en la eficiencia lograda y una valiosa mejora en los tiempos de respuesta, con relación al procesamiento secuencial.

El diseño de estos decodificadores permitió observar las ventajas de su implementación a través de redes neuronales en hardware, siendo ésta una solución innovadora, eficiente y en hardware libre. La descripción generalizada de los modelos en VHDL que se han desarrollado, permite el diseño de redes neuronales reconfigurables, que pueden modificar su estructura a partir de las condiciones detectadas en el entorno de operación. Siendo éste el objetivo de los sistemas de radio cognitivo, donde se busca la percepción de las características del canal de comunicación y la adaptación inteligente de los módulos de procesamiento de datos para el tratamiento de las señales, ajustando su esquema de codificación/modulación y parámetros asociados, como se logra definir en la descripción VHDL de las redes neuronales.

El aporte principal del trabajo viene dado por

la configuración de estos componentes, a través de redes neuronales, por ser éstas estructuras flexibles con parámetros adaptativos como son los pesos sinápticos y *bias*. En tanto, que la descripción de los algoritmos de entrenamiento en hardware, permite el soporte de redes adaptativas, cuyos parámetros pueden ser calculados en el circuito, a partir de un entrenamiento en tiempo real, para lo que se propone tramas de encabezados conocidos, a fin de ajustar los parámetros de la red.

Los modelos neuronales descritos pueden ser utilizados para decodificación/demodulación adaptativa, basada en las diversas redes neuronales para reconocimiento de señal, ajustando la configuración de la red con los datos establecidos como señal piloto, es decir que se pueden realizar entrenamientos dinámicos, en los circuitos desarrollados para la tecnología de arreglos de compuerta programable – FPGA. De esta manera, se puede definir un nuevo concepto de sistemas de comunicaciones digitales neuro-cognitivos, los cuales puedan aprender y optimizar su desempeño a partir de las condiciones detectadas. Lo que crea un área de investigación para la producción de conocimiento y desarrollo de tecnología, que optimiza la eficiencia de los diseños, con una disminución del consumo de energía, basados en modelos neuronales reconfigurables.

Referencias

[1] Cesar Hernández, Luis Fernando Pedraza Martínez y Fredy Hernán Martínez Sarmiento. Algoritmos para asignación de espectro en redes de radio cognitiva. *Tecnura*, 20(48):69–88, 2016.

[2] Tobias Gruber, Sebastian Cammerer, Jakob Hoydis, and Stephan ten Brink. On deep learning-based channel decoding. In *Information Sciences and Systems (CISS), 2017 51st Annual Conference on*, pages 1–6. IEEE, 2017.

[3] X-A Wang and S. B. Wicker. An artificial neural net Viterbi decoder. *IEEE Transactions on Communication*, 44(2):165–171, 1996.

[4] Ja-Ling Wu, Yuen-Hsien Tseng, and Yuh-Ming Huang. Neural network decoders for linear block codes. *International Journal of Computational Engineering Science*, 3(03):235–255, 2002.

[5] Michael Eoin Buckley and Stephen B. Wicker. The design and performance of a neural network for predicting turbo decoding error with application to hybrid arq protocols. *IEEE Transactions on Communications*, 48(4):566–576, 2000.

[6] Cecilia Sandoval Ruiz. FPGA prototyping of neuro-adaptive decoder. In *Proceedings of the 9th WSEAS International Conference on Computational Intelligence, Man-machine Systems and Cybernetics*, CIMMACS '10, pages 99–104, Stevens Point, Wisconsin, USA, 2010. World Scientific and Engineering Academy and Society (WSEAS).

[7] Nicholas Bonello, Sheng Chen, and Lajos Hanzo. On the design of pilot symbol assisted codes. In *Vehicular Technology Conference Fall (VTC 2009-Fall), 2009 IEEE 70th*, pages 1–5. IEEE, 2009.

[8] Cecilia Sandoval. Modelo neuro-adaptativo en VHDL, basado en circuitos NLFSR, para control de un sistema inteligente de tecnología sostenible. *Universidad Ciencia y Tecnología*, 21(85):140–149, 2017.

[9] Cecilia E. Sandoval-Ruiz. Logical-mathematical model of encoder 2D-RS for hardware description in VHDL. *Revista Ingeniería UC*, 24(1):28–39, 2017.

[10] C. Sandoval-Ruiz. Modelo de estructuras reconfigurables con registro desplazamiento, para lenguaje descriptor de hardware VHDL. *Revista de la Facultad de Ingeniería U.C.V.*, 31(3):63–72, 2016.

[11] Cecilia Esperanza Sandoval-Ruiz. VHDL optimized model of a multiplier in finite fields. *Ingeniería y Universidad*, 21(2):195–211, 2017.

[12] Cecilia Sandoval-Ruiz. Análisis de circuitos fractales y modelado a través de sistema de funciones iteradas para VHDL caso de estudio: codificador Reed Solomon. *Ciencia e Ingeniería*, 38(1):3–16, 2017.

[13] V. Fernandes, M. Díaz y C. Sandoval-Ruiz. Sistema eficiente de modulación digital basado en radio congnitivo. In 1^{er} *Congreso Venezolano de Ciencia, Tecnología e Innovación*, volume 1, page 285, Venezuela, 2012. Ministerio del Poder Popular de Ciencia, Tecnología e Innovación.

[14] Cecilia Sandoval and Antonio Fedón. Codificador de fuente programado en VHDL para dispositivos de hardware reconfigurable. *Ciencia e Ingeniería*, 28(1):37–40, 2007.

[15] Cecilia Esperanza Sandoval-Ruiz and Antonio Fedón-Rovira. Codificador RS (255, k) en hardware reconfigurable orientado a radio cognitivo. *Ingeniería y Universidad*, 17(1):77–91, 2013.

[16] C. Sandoval Ruiz. Diseño modular de un sistema para procesamiento y comunicación digital en banda base usando programación en VHDL. Tesis de Maestría, Dirección de Postgrado, Facultad de Ingeniería, Universidad de Carabobo, Venezuela, 2012.

[17] Howard B. Demuth, Mark H. Beale, Orlando De Jess, and Martin T. Hagan. *Neural Network Design*. Martin Hagan, USA, 2^{nd} edition, 2014.

[18] Ken Alberto Harima Sakaguchi. Decodificación de salida suave para códigos Reed-Solomon y su aplicación a códigos concatenados. Tesis de Maestría,

Decanato de Estudios de Postgrado, Universidad Simón Bolívar, Caracas, Venezuela, 2006.