Ingeniería Investigación y Tecnología
volumen XX (número 3), julio-septiembre 2019 1-10
ISSN 2594-0732 FI-UNAM artículo arbitrado
Información del artículo: Recibido: 9 de abril de 2018, reevaluado: 21 de marzo de 2019, aceptado: 3 de mayo de 2019

http://dx.doi.org/10.22201/fi.25940732e.2019.20n3.032

Métodos numéricos en diferencias finitas para la estimación de recursos de Hardware FPGA en arquitecturas LFSR(*n,k*) fractales

Numerical methods in finite differences for the estimation of FPGA hardware resources in LFSR (*n,k*) fractal architectures

Sandoval-Ruiz Cecilia E.
Universidad de Carabobo, Venezuela
Instituto de Matemática y Cálculo Aplicado, IMYCA
Correo: cesandova@gmail.com
http://orcid.org/0000-0001-5980-292X

Resumen

En este trabajo se estudia el método numérico en diferencias finitas para la estimación de recursos en hardware, específicamente sobre tecnología FPGA (arreglo de compuertas programables por campo), aplicado a un modelo fractal de componentes de arquitectura LFSR (registros desplazamiento con realimentación lineal) paralelizada, como elemento básico de sistemas de multiplicadores en campos finitos, código Reed Solomon y Redes Neuronales Artificiales. El método abordado consistió en la discretización de las variables arrojadas en el reporte de síntesis sobre hardware de los casos de estudio, por medio del modelado matemático se obtienen las ecuaciones descriptivas, lo que permite validar las estrategias de optimización de los diseños usando la base de un operador matemático con estructura concurrente de realimentación lineal LFCS (*n,k*), definido por funciones compuestas con auto similitud. Se obtiene como resultado un conjunto de ecuaciones que describen el comportamiento del parámetro estimado, facilitando la evaluación del diseño en etapas previas, así como la aplicación de un elemento recursivo, que permita obtener el consumo de recursos en función a este operador lógico-matemático. Estos métodos pueden ser extendidos en el análisis y estimación de eficiencia energética de los diseños, con lo que se aportan soluciones en materia de consumo de energía de los modelos electrónicos y rendimiento de sistemas.
Descriptores: Método numérico en diferencias finitas, estimación de recursos, LFCS, modelado fractal en VHDL, FPGA.

Abstract

In this paper we study the numerical method in finite differences for the estimation of resources in hardware, specifically on FPGA (Field Programmable Gate Arrays) technology, applied to a fractal model of LFSR architecture components (Linear Feedback Shift Registers) parallelized, as a basic element of multiplier systems in finite fields, Reed Solomon code and Artificial Neural Networks. The method addressed consisted of the discretization of the variables in the hardware synthesis report of the case studies, by means of mathematical modeling the descriptive equations are obtained, which allows to validate the strategies of optimization of the designs using the base of a mathematical operator with Linear Feedback Concurrent Structure LFCS (*n,k*), defined by composite functions with self-similarity. We obtain as a result a set of equations that describe the behavior of the estimated parameter, facilitating the evaluation of the design in previous stages, as well as the application of a recursive element, which allows obtaining the consumption of resources according to this logical-mathematical operator. These methods can be extended in the analysis and estimation of energy efficiency of the designs, which provide solutions in terms of energy consumption of electronic models and system performance.
Keywords: Numerical method in finite differences, resource estimation, LFCS, fractal modeling in VHDL, FPGA.

Introducción

Existe dos grandes familias de métodos numéricos: los métodos de elementos finitos y los métodos en diferencias finitas (Quintana, 2016). Este último ha sido considerado en esta investigación, como una solución alternativa para la estimación de recursos de modelos en VHDL (*Very High Speed Integrates Circuit Hardware Description. Language*), al momento de seleccionar el objeto de estudio se ha identificado un operador de interés como es el caso de la estructura LFCS (Sandoval, 2013), donde se ha reconocido una estructura auto-similares, que pueden ser implementadas en hardware con circuitos fractales y estructuras como elementos discretos (Magaña *et al*., 2011).

En estudios recientes se han desarrollado los códigos del elemento circuital LFCS para diversas aplicaciones en el campo de la ingeniería dada su relevancia y vigencia como tema de estudio, en tratamiento de señales (Sandoval, 2016), estructuras fractales (2017a), multiplicadores en campos finitos (Sandoval, 2017b), códigos RS(n,k) - Reed Solomon (Sandoval y Fedón, 2007), códigos 2D-RS (Sandoval, 2017c), radio definidos por software (Sandoval, 2017d), decodificadores neuronales (Sandoval, 2017e) y sistemas de control adaptativo (Sandoval, 2017f). Igualmente se ha encontrado su aplicación en redes distribuidas de energías renovables (Sandoval, 2018a, 2018b), entre otros, que están asociadas a conceptos de, lo que permite identificar la vigencia de desarrollar un modelo optimizado del circuito.

Actualmente, la eficiencia en los diseños electrónicos de hardware reconfigurable se ha convertido en un eje de investigación, lo que nos lleva a establecer elementos de medición del consumo de recursos y consumo de potencia. Para la investigación se han seleccionado como caso de estudio aplicaciones dentro del área de los códigos correctores de error (Marchesan, 2007), estos son elementos de interés por corregir los símbolos alterados, que se presentan por efecto del ruido comunicación, logrando garantizar la calidad en la transmisión con menor potencia (Sandoval, 2013). Dada la complejidad matemática y circuital de estos esquemas de codificación, se han diseñado soportados sobre tecnología FPGA-*Field Programmable Gate Arrays* (Rodríguez *et al*., 2017), que pueden definir su comportamiento algorítmico (Sandoval, 2008), o bien, de manera adaptativa basados en el entrenamiento de una red neuronal. Lo que se traduce en la posibilidad de autoconfiguración del sistema a partir de inteligencia computacional, permitiendo la optimización del sistema, en función de la capacidad de generalización de las redes neuronales.

Las funciones iteradas (Rivera y López, 2012) han permitido identificar estructuras y modelos para el análisis de procesos dinámicos, sistemas complejos, incluyendo circuitos eléctricos. Al momento de diseñar una aplicación del sistema de códigos Reed Solomon, con funciones específicas, como es el caso del multiplicador en campos Finitos de Galois, el cual puede ser entrenado en una subred, se encuentra el modelo de una red con características de auto-similitud entre sus componentes, donde este es un modelo fractal de red neuronal artificial.

A partir de estos sistemas con procesamiento caracterizado por la operación conocida como convolución, se encuentra que esta presenta alta coincidencia con las operaciones neuronales, que definen el procesamiento de la suma ponderada en las neuronas. Esta característica puede ser aprovechada para una técnica de diseño novedoso en esta área de investigación, en la que se defina un operador lógico para la descripción en VHDL de funciones donde se utiliza este tipo de procesamiento.

Estructuras fractales en aplicaciones computacionales

En esta etapa de estudio se encuentran aplicaciones como sistemas de códigos compuestos en campos finitos, redes neuronales fractales FNN, sistemas MIMO, entre otros. En los cuales se presentan estructuras autosimilares de operadores concatenados, este tema se ha estudiado detalladamente en el modelado matemático de operadores LFC-*Linear FeedBack Concatenated*.

Elementos básicos LFCS en campos finitos - GF(2m)

Los Campos Finitos de de Galois (GF) constituyen un área específica de la matemática desarrollada por E. Galois. Donde el campo se especifica a través de un elemento primo p, base del campo; y un entero positivo m, longitud del elemento del campo. Se cumple que p^m corresponde al número de elementos del campo, y las operaciones aritméticas sobre el campo finito da como resultado un elemento que pertenece al mismo (Saqib Nazar, 2004). Estas propiedades se utilizan en aplicaciones de codificación con LFCS (Sandoval, 2012; Xilinx, 2011) y criptografía. Una presentación ampliamente utilizada es la forma polinomial, en la cual se define un polinomio generador del campo, conocido como polinomio irreducible $p(x)$, que operará como módulo con los resultados de las operaciones para llevar a cabo el resultado a la longitud fija definida para el campo.

Este análisis parte del estudio del modelo matemático de la representación polinomial, en el que se enun-

http://dx.doi.org/10.22201/fi.25940732e.2019.20n3.032

cia: Si *p(x)* es el polinomio irreducible, entonces la multiplicación de dos elementos del campo, representados como los polinomios *A(x)* y *B(x)* es el producto algebraico de los dos polinomios, y la operación módulo del polinomio *p(x)*, también conocido como reducción modular, es el mostrado en la ecuación 1.

$$C(x) = A(x) \otimes B(x) \leftrightarrow C(x) = A(x)\ B(x) (mod\ p(x)) \leftrightarrow C(x) = B(x)A(x) (mod\ p(x)) \quad (1)$$

La multiplicación de polinomios es asociativa, conmutativa y distributiva, respecto a la adicción por lo cual se obtienen la ecuación 2.

$$C(x)=B(x)(\sum_{i=0}^{m-1} A_i x^i)\ mod\ p(x) \rightarrow C(x)= \sum_{i=0}^{m-1} B_i (A(x)\ x^i mod\ p(x)) \quad (2)$$

Donde *A(x)* y *B(x)* corresponden a la representación polinomial de los operandos. En el caso del codificador RS (Reed Solomon) el multiplicando *A(x)* corresponde a un coeficiente del polinomio generador del código y el multiplicador *B(x)* a la entrada de datos, y *p(x)* es el polinomio irreducible del campo de *Galois*.

Redes Neuronales Fractales-FNN

Un tópico de estudio de interés corresponde a los circuitos electrónicos con estructura fractal, cuyo modelado en lenguaje descriptor de hardware puede realizarse a través de ecuaciones con características de funciones iteradas (Sandoval y Fedón, 2007), en el cual las redes neuronales pueden coincidir y se pueden modelar bajo este criterio.

La arquitectura de una Red Neuronal Artificial está dada por un conjunto de neuronas, que intercambian información a través de conexiones, cada una de estas tendrá una ponderación relacionada con el proceso de aprendizaje. La información procede de las señales de entrada a la red *x*, tendrá una estructura de capa oculta, capa de salida que procesan la información, a fin de obtener las señales de salida *a* de la red, esta arquitectura ha sido re-organizada como se presenta en la Figura 1.

Donde los bloques señalados, *wfi* corresponden a la ganancia del LFCS, es decir, el peso del elemento fractal, que ha sido desglosada para el caso de la entrada al primer módulo, corresponde a una ganancia sobre la operación compuesta representada por el arreglo de la estructura que anida la sumatoria de convolución sobre la ganancia *wf1* (como corresponde a la multiplicación en campos finitos). Los elementos internos, *wik* corresponden a las ganancias de la rama en la posición *k* para la señal de entrada *xi*, para el caso representado de la Figura1 la señal *x1* ilustra al peso de una neurona en dicha posición, los bloques z^{-k} corresponden a los elementos de memoria en la posición k, que introducen retardos entre las ramas del arreglo LFCS.

El esquema permite observar, en primer lugar los módulos de multiplicación que pueden ser esquematizados como la suma ponderada de los bits de entrada de la señal x_k, el componente producto presenta una arquitectura similar a la arquitectura de la neurona, presentando similitud con el modelo de la red, se puede concluir que entre la estructura de la red y los componentes se presenta una estructura circuital auto-similar. Las ecuaciones de la red neuronal fractal se pueden expresar a partir de la base de una neurona. Esta operación que corresponde a la convolución, donde la operación base sustituye la operación producto, como se presenta en la ecuación 3.

$$a(n) = \sum_{k=0}^{Rc} w_{j,k} (\sum_{i=0}^{Rf} w_{i,j} \cdot x_i) \quad (3)$$

Donde *a(n)* corresponde a la salida de la red, $w_{j,k}$ los pesos sinápticos de la red externa, $w_{i,j}$ los pesos sinápticos de la subred componente y x_i las entradas de la red neuronal, se tiene que la operación entre los componentes de la red están siendo operados bajo el producto de convolución (Sandoval, 2016), suma de productos, que se definirá como operación de *sinapsis neuronal*.

Método de Diferencias Finitas-MDF

La generalidad del método DF es especialmente importante en modelos que se extienden más allá de los coeficientes constantes, ya que puede manejar procesos con coeficientes variables en el tiempo, modelos de tasa de interés simples o multifactoriales. Además ofrece una considerable flexibilidad en las opciones de mallas en las dimensiones de tiempo y espacio, lo que resulta útil

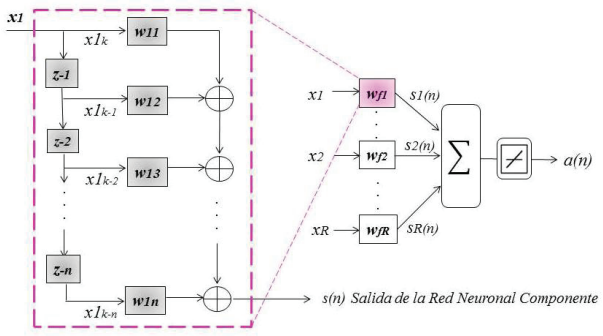

Figura 1. Esquema general de la red neuronal fractal

al tratar con dividendos discretos, barreras y otros escenarios comunes, mientras que a través de aproximaciones de mayor orden es posible mejorar la convergencia (Otero *et al*., 2008).

El método de diferencias finitas permite discretizar ecuaciones diferenciales y permitir un tratamiento más simple del problema diferencial parcial, el desarrollo de la matemática de soporte es detallada en Skiba (2005) y Zozaya (2004), el principio del método comprende la definición de las derivadas parciales de la función *f(x)*, como se presenta en la ecuación 4.

$$\frac{\partial F}{\partial x} \approx \frac{F(x+\Delta x, y) - F(x,y)}{\Delta x} \tag{4}$$

De la misma forma, la segunda derivada viene dada por la ecuación 5.

$$\frac{\partial^2 F}{\partial x^2} \approx \frac{F(x+\Delta x, y) - 2F(x,y) + F(x-\Delta x, y)}{\Delta x^2} \tag{5}$$

Estas derivadas parciales pueden expresarse en el dominio discreto como diferencias, considerando que los datos están uniformemente espaciados si $x_{i+1} - x_i = \Delta x$ es constante para i = 1, 2, 3,… Para el caso particular de datos uniformemente espaciados, es posible encontrar una forma más sencilla del polinomio de Newton, donde se definen las diferencias según el orden, para orden cero: $\Delta^0 f_i = f_i$, para orden uno: $\Delta^1 f_i = f_{i+1} - f_i$, para orden dos: $\Delta^2 f_i = f_{i+2} - 2f_{i+1} + f_i$, para orden tres: $\Delta^3 f_i = f_{i+3} - 3f_{i+2} + 3f_{i-1} - f_i$, para orden k: $\Delta^k f_i = f_{i+k} - k f_{i+k-1} + k(k-1)/2!\, f_{i+k-2} - k(k-1)(k-2)/3!\, f_{i+k-3} + \ldots$ Estas serán reescritas en función de la longitud del intervalo para su aplicación.

La solución se obtiene de aproximar estas a los valores discretos conocidos de la función. El cálculo de las diferencias finitas proporciona una herramienta poderosa para el tratamiento de las variables medidas; una vez que se hayan determinado matemáticamente los valores necesarios se puede proceder a aplicarlos a las mediciones sobre el modelo. Consideremos una función conocida *y=f(x)*, que puede expresarse partiendo de un desarrollo de Taylor alrededor del valor $x = a$, como se presenta en la ecuación 6.

$$f(x) = f(a) + (x-a)\left(\frac{df}{dx}\right)_a + \frac{(x-a)^2}{2!}\left(\frac{d^2 f}{dx^2}\right)_a + \ldots \tag{6}$$

Una función tal se define sobre un intervalo continuo de valores en la escala *x,y*, para que la teoría sea aplicable a las variables medidas se debe transformar la formulación de modo que se refiera a valores discretos de *x*. Donde estos valores son equidistantes a intervalos de longitud *h* a partir de *a*. Dados por: *x=a, x=a+h, x=a+2h, x=a+3h,* donde los valores de *y* para estos valores discretos de x serán: *f(a), f(a+h), f(a+2h), f(a+3h)* y en forma correspondiente se tiene: $\Delta f(a+h) = f(a+2h) - f(a+h)$. Magnitudes que estarán relacionadas con las primeras derivadas de la función en los diferentes valores de *x*, de manera semejante, definimos la segunda diferencia como: $\Delta f^2(a) = \Delta f(a+h) - \Delta f(a)$ y así sucesivamente para las terceras diferencias y de orden superior. La forma original de la ecuación de Newton-Gregory se convierte en:

$$f(x) = f(a) + \frac{1}{h}(x-a)\Delta + \frac{1}{2!}\frac{1}{h^2}(x-a)(x-a-1)\Delta^2$$

$$+ \frac{1}{3!}\frac{1}{h^3}(x-a)(x-a-1)(x-a-2)\Delta^3 + \ldots$$

Este método puede ser aplicado sobre los valores observados para el consumo de recursos y potencia de los codificadores Reed Solomon con variable discreta *b* (número de bits), a fin de obtener un modelo de estimación de estos parámetros para valores de *b>8*.

Modelado matemático del consumo de recursos

Una vez definido el modelo teórico en función de los componentes relacionados en la arquitectura fractal, se establecen las ecuaciones que permiten la estimación de recursos hardware. El método consiste en la combinación de un método numérico de diferencias finitas, partiendo de los reportes generados por la herramienta de síntesis en hardware de los modelos en VHDL, a fin de obtener una ecuación que permita estimar el consumo de recursos asociado a los parámetros de diseño, estos resultados pueden ser interpolados para valores no simulados, con base en los polinomios de *Lagrange* (ec. 7)

$$f(x) = \sum_{i=1}^{n+1} f_i \prod_{\substack{i=1 \\ j\neq 1}}^{n+1} \frac{x - x_j}{x_i - x_j} \tag{7}$$

Obteniendo una ecuación que se puede expresar de la siguiente forma

$$f(x) = \begin{cases} f_i & \forall x_i \in [a,b) \\ f_j & \forall x_j \in (b,c] \end{cases} \tag{8}$$

Así como la aplicación de las ecuaciones *Newton-Gregory* para diferencias finitas, discretizando los resultados en consumo de recursos por número de bits del

circuito, así como relación de ramas del esquema (*n*,*k*), según la ecuación (9).

$$f(x)=\sum_{i=1}^{n+1}\frac{\Delta^i}{i!}\prod_{j=1}^{i-1}(x-x_j) \quad (9)$$

Estos resultados pueden ser extrapolados, partiendo del análisis del circuito compuesto. Se analiza el consumo de compuertas XOR, a partir de los modelos desarrollados en (Sandoval, 2013). La ecuación 10 presenta el aporte de cada elemento del elemento multiplicador.

$$\#XOR_{mult}=(p.m-p-2.m+2)_{LFSR}+(m^2-2.m+1)_{PROD}$$
$$\#XOR_{mult}=m^2+p.m-4.m-p+3 \quad (10)$$

Donde m corresponde al número de bits de cada palabra del campo y p corresponde al número de bits no nulos del polinomio irreducible *p(x).* En este caso, se analizará un polinomio general con *p=m+1*, la ecuación se puede reescribir:

$$\#XOR_{LFSR}=m(m+1)-(m+1)-2\cdot m+2 \quad (11)$$
$$f(m)=m^2-2\cdot m+1$$

Lo que permite obtener una estimación a nivel de compuertas de forma analítica

$$\#XOR_{mult}=m^2+m(m+1)-4.m-(m+1)+3 \quad (12)$$
$$f(m)=2\cdot m^2-4\cdot m+4$$

Para el cálculo del consumo de recursos del multiplicador paralelo, en primer lugar, se estimó el número de compuertas de acuerdo con la descripción del LFCS, esto contabilizando el número de operadores *AND* y *XOR* utilizados y asociados con el número de bits del multiplicador *m*. Igualmente, se obtuvieron los reportes de síntesis del componente LFCS, a fin de evaluar el consumo de recursos de hardware del diseño. Para ello, se tomó como indicador el número de Slices, los cuales están basados en tablas de búsqueda LUTs (LookUp Tables) de 4 entradas para el dispositivo seleccionado. Estos datos se presentan en la Tabla 1.

En este caso, la optimización corresponde a la simplificación de operaciones, en función del número de coeficientes *p(x)* no nulos. La comparación entre el modelo LFCS general y simplificado se presenta en la Tabla 2.

Los resultados obtenidos se calcularon a través de la suma de compuertas de cada uno de los tres módulos LFCS, ANDs y XORs de la arquitectura del multiplicador GF, otras investigaciones obtienen el cálculo del número de compuertas *AND* y *XOR* aplicando algoritmos de segmentación para disminuir el número de compuertas requeridas (Saqib, 2004; Kindap, 2004). Por otra parte, al implementar el multiplicador la herramienta de desarrollo reporta el consumo de recursos del diseño, donde se utilizaron 44 LUTs para la implementación, como se presenta en la Tabla 3.

Es importante destacar, que estos cálculos están basados en el procesamiento concurrente de los elementos del campo, lo cual puede llevar a pensar en un aumento de recursos por el procesado paralelo, en la Tabla 3 se puede observar que el presente diseño ha alcanzado una reducción del número de LUTs del multiplicador respecto de sus componentes, a través de la simplificación propuesta. Por otra parte, estos resultados se compararon con la complejidad teórica reportada en trabajos previos (Imaña *et al*., 2002), que estaba en el orden de m^2*-1* compuertas *xor*, al sustituir *m=8* se obtiene una optimización de un 54.05%, a nivel de utilización de recursos hardware.

Tabla 1. Reporte de utilización para el LFCS con *p(x)* ajustable

Recursos del LFSR	AND	XOR	Slices	LUTs
Modelo para *m* genérico	$(m-1)^2+m$	$(m-1)^2$	-	-
LFCS con parámetro *m=8*	56	49	24	48

Tabla 2. Compuertas utilizadas para el LFCS con P(x) = 100011101

Compuertas del Multiplicador	AND2	XOR2	XOR8
LFCS general	$(m^2-2m+1) * m^2$	m^2-2m+1	m–1
LFCS simplificado con p(x)	m^2	p.m–p-2m+2	m–1

Tabla 3. Reporte de utilización del multiplicador GF con p(x) = 100011101

Recursos del Multiplicador GF	AND2	*and*	XOR2	*xor*	*Slices*	*LUTs*
R(x) = A(x) mod P(x) con m=8 y p=5	0	0	*p.m–p–2m+2*	21	5	8
C(x) = R(x) AND B(x)	m^2	64	0	0	-	64
D(x) = XOR C(x)	0	0	m^2-m	56	-	56
Multiplicador basado en LFCS	m^2	64	m^2 + p.*m–p–3m*+2	77	17	44

Para la evaluación del consumo de potencia del modelo propuesto, se consideraron los cálculos desarrollados en Allen (2008),Paar (1996), Biard y Noguet (2008), estimando el consumo de recursos y potencia a partir de la complejidad computacional dada en compuertas, realizando la comparación con el modelo diseñado, destacando la extrapolación del diseño fractal con el factor de optimización presentado en la Tabla 4.

Análisis de resultados en el modelo LFCS en composición generalizada fractal

Una vez realizado el estudio de un operador LFC(*n,k*) para su utilización en funciones matemáticas, compuestas por productos en campos finitos y su arreglo fractal para sumas ponderadas de la misma estructura circuital aquí estudiada, se cuenta con una herramienta de soporte para la implementación de una amplia gama de aplicaciones. Al mismo tiempo, se considera su versatilidad e importancia del operador como componente fractal en los circuitos de hardware reconfigurables. El siguiente paso consistió en analizar la implementación circuital del diseño, esto se realizó a través de los reportes de la herramienta de desarrollo del ISE11, donde se pueden observar los recursos consumidos por el diseño. Lo importante en este modelo de operador es que amplía la aplicabilidad de estos elementos matemáticos, basado en la actualización de los esquemas reconfigurables, simplificando la matriz hardware de componentes de diseño y configuración.

Esto se traduce en un aporte significativo sobre los esquemas de hardware reconfigurables compuestos, que serán más generalizados, aplicables en soluciones escalables, através de la reformulación de los modelos a partir del circuito base LFC, es decir, que un modelo matemático puede ser re-escrito en funciones LFC fractales y se puede implementar con estos elementos en una matriz reconfigurable. Así se logra una simplificación de la complejidad del modelo y una independencia tecnológica, por su capacidad de adaptación. En el análisis de los resultados de los diseños se consideran los reportes de la herramienta de desarrollo correspondientes al análisis de recursos en el modelo del RS (*255,k*) presentados en la Tabla 5.

Estos resultados pueden analizarse, bien en función del consumo de compuertas lógicas o de componentes de la estructura FPGA, en este caso, estudiando el reporte de consumo de LUTs, como se muestra en la Tabla 6.

Los resultados pueden tratarse por métodos de diferencias finitas para obtener una ecuación en función del parámetro *k*, definida como *f(k)*. Sin embargo, la solución acá planteada corresponde a la extrapolación del modelo auto-similar de componentes LFC, como una función multi-variable dada por *f(n,k,m)*.

Para este método de modelado fractal, se realiza el análisis de consumo de recursos para la unidad básica estableciendo los modelos en función de LFCS. Al mismo tiempo, establecer las ecuaciones en función de las LUTs y SLICEs con método de diferencias finitas. En primer lugar, se obtuvieron los reportes de consumo de recursos, en esta oportunidad nos concentramos en las tablas de búsqueda de 4 entradas LUTs, como se muestra en la Tabla 7.

De esta manera se obtuvo una primera estimación, en la cual se calcularon las diferencias parciales entre los recursos consumidos en función del número de bits del multiplicador en campos finitos, resumidos en la Tabla 8.

Se realizó una aproximación promediada del consumo de LUTs, con el propósito de obtener un modelo más generalizado por excedente de consumo de recursos, presentados en la Tabla 9, considerando el error de corrección para simplificar el cálculo, que luego será compensado con un factor adaptivo en el modelo de estimación final.

Tabla 4. Complejidad computacional y consumo de potencia

Operadores	Complejidad Computacional	Consumo de Potencia	(pJ)
GFmult (Biard & Noguet, 2008)	m^2 AND + $3(m-1)^2/2$ XOR	$0.4\ (m^2 + 3(m-1)^2/2)$	55
GF adder (Biard & Noguet, 2008)	m XOR	0.4 m	3.2
GF inv (Biard & Noguet, 2008)	m ROM read	8m	64
GF reg (Biard & Noguet, 2008)	m REG write	2m	16
GF mem	m RAM read and write	10m	80
*LFSR	p.m-p-2.m+2 XOR	0.4 (p.m-p-2.m+2)	8.4
*GFmult	m^2 AND + (p.m-p-2.m+2) (m-1) XOR	$0.4(m^2 + (p.m-p-2.m+2)(m-1))$	39.2
*LFCS (n,k)	$(n-k)\cdot f_{opt}\ [m^2 + (p.m-p-2.m+2)\ (m-1)]$	$(n-k)\cdot f_{opt}\ [\ m^2 + (p.m-p-2.m+2)(m-1)]$	-

*Resultados de la presente investigación, 2018

Tabla 5a. Reporte de consumo de recursos LFC del Reed Solomon (255, 247)

Device Utilization Summary (estimated values)			
Module Name: Codificador_RS		Target Device: xc4vlx15-12sf363	
Logic Utilization	Used	Available	Utilization
Number of Slice	91	6144	1%
Number of slice FF	64	12288	0%
Number of LUTs4	175	12288	1%
Number of IOBs	26	240	10%
Number of Gclks	1	32	3%

Tabla 5b. Reporte de consumo de recursos LFC del Reed Solomon (255, 239)

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slice	112	3072	3%
Number of slice FF	128	6144	2%
Number of LUTs4	209	6144	3%
Number of IOBs	25	180	13%
Number of Gclks	1	4	25%

Tabla 5c. Reporte de consumo de recursos LFC del Reed Solomon (255, 223)

Device Utilization Summary (estimated values)			
Module Name: Codificador_RS		Target Device: xc4vlx15-12sf363	
Logic Utilization	Used	Available	Utilization
Number of Slice	271	6144	4%
Number of slice FF	257	12288	2%
Number of LUTs4	515	12288	4%
Number of IOBs	18	240	7%
Number of Gclks	1	32	3%

Tabla 6. Análisis de recursos de implementación de códigos RS

Simb.	RS(255,247)	RS(255,239)	RS(255,223)
XOR	# Registers : 8	# Registers : 16	# Registers : 32
	8-bit register : 8	8-bit register : 16	# Xors : 686
	# Xors : 184	# Xors : 368	1-bit xor2 : 624
	1-bit xor2 : 168	1-bit xor2 : 336	8-bit xor2 : 37
	8-bit xor2 : 8	8-bit xor2 : 16	8-bit xor3 : 7
	8-bit xor8 : 8	8-bit xor8 : 16	8-bit xor4 : 7
	-		8-bit xor5 : 8
	-		8-bit xor6 : 1
	-		8-bit xor7 : 2
LUT4	# LUT2 : 14	# LUT2 : 9	# LUT2 : 25
	# LUT2_L : 5	# LUT3 : 62	# LUT2_D : 1
	# LUT3 : 29	# LUT3_D : 6	# LUT2_L : 13
	# LUT3_D : 2	# LUT3_L : 18	# LUT3 : 81
	# LUT3_L : 5	# LUT4 : 162	# LUT3_D : 11
	# LUT4 : 87	# LUT4_D : 11	# LUT3_L : 15
	# LUT4_D : 16	# LUT4_L : 44	# LUT4 : 286
	# LUT4_L : 17		# LUT4_D : 24
	-		# LUT4_L : 59
LUTs	175	312	515

Tabla 7. Reporte de consumo de recursos LFC del multiplicador de 8 bits

Device Utilization Summary (estimated values)			
Module Name: multiplicadorDF8		Target Device: xc4vlx15-12sf363	
Logic Utilization	Used	Available	Utilization
Number of Slice	35	6144	0%
Number of LUTs4	62	12288	0%
Number of IOBs	24	240	10%

Tabla 8. Recursos reportados con el dispositivo 4vfx140ff1517-11

bits	3	4	5
XOR	# Xors : 3	# Xors : 4	# Xors : 9
	1-bit xor2 : 2	1-bit xor2 : 3	1-bit xor2 : 8
	3-bit xor3 : 1	4-bit xor4 : 1	5-bit xor5 : 1
LUT4	# LUT2 : 1	# LUT2 : 2	# LUT2 : 2
	# LUT3 : 3	# LUT3 : 2	# LUT3 : 3
	# LUT4 : 4	# LUT4 : 8	# LUT4 : 16
	8	12	21

bits	6	7	8
XOR	# Xors : 16	# Xors : 19	# Xors : 22
	1-bit xor2 : 15	1-bit xor2 : 18	1-bit xor2 : 21
	6-bit xor6 : 1	7-bit xor7 : 1	8-bit xor8 : 1
LUT4	# LUT2 : 4	# LUT2 : 7	# LUT2 : 4
	# LUT3 : 6	# LUT3 : 8	# LUT3 : 11
	# LUT4 : 26	# LUT4 : 36	# LUT4 : 47
	36	51	62
LUT6		# LUT2 : 6	# LUT2 : 5
		# LUT3 : 4 # LUT3 : 5	
		# LUT4 : 6 # LUT4 : 7	
		# LUT5 : 4 # LUT5 : 4	
		# LUT6 : 16 # LUT6 : 23	
	33	45	58

Tabla 9. Resultados de consumo de LUTs

bits	LUTs	Δ	Δ^2	Δ^3	Δ^4	Δ^5
0						
1						
2						
3	8					
4	12	4				
5	21	9	5			
6	36	15	6	1		
7	48	12	-3	-9	-10	
8	62	14	2	5	14	24

(a) Reporte síntesis

bits	LUTs	Δ	Δ^2	Δ^3	Δ^4	Δ^5
0						
1						
2						
3	11					
4	17	6				
5	25	8	2			
6	35	10	2	0		
7	47	12	2	0	0	
8	61	14	2	0	0	0

(b) Considerando error de aproximación

A partir de la aplicación del método DF se obtuvo el modelo que describe el consumo de recursos, partiendo de la ecuación (13).

$$f(x)=\frac{\Delta}{1!}(x-a)+\frac{\Delta^2}{2!}(x-a)(x-a-1)+\ldots$$
$$\ldots+\frac{\Delta^n}{n!}(x-a)(\mathrm{x}-\mathrm{a}-1)\ldots(x-n-1)+f(a) \quad (13)$$

Para la sustitución de los elementos de la Tabla 9, se identifican los componentes de la ecuación de interpolación de Newton-Gregory, de la forma: $a=3, f(a)=11$, $\Delta=6$, $\Delta^2=2$, $\Delta^3=0$, $\Delta^4=0$, $\Delta^5=0$.

Sustituyendo los elementos no nulos, en la ecuación se tiene:

$$f(x)=\frac{6}{1!}(x-3)+\frac{2}{2!}(x-3)(x-3-1)+11 \quad (14)$$

Realizando la operación de los productos factor común queda

$$f(x)=6x-18+x^2-7x+12+11 \quad (15)$$

Finalmente, ordenando la ecuación, en función del número de bits m resultante, se obtiene

$$f(m)=m^2-m+5 \quad (16)$$

Se obtiene un tratamiento de ecuaciones selectivas en rango, que permite la interpolación y extrapolación, como solución a modelos complejos

$$f(m)=\begin{cases} m^2-m+2 & \forall m\in\{3.5\} \\ m^2-m+6 & \forall m\in\{3.8\} \end{cases}$$

El cual puede ser tratado como un parámetro adaptativo en función del número de bits, considerando errores de aproximación, como se presenta en la ecuación (17)

$$f(m)=\mathrm{m}^2-m+\Delta x \quad (17)$$

Seguidamente, se estimaron los recursos para aplicaciones fractales de códigos Reed Solomon (n,k), para contrastar con los resultados teóricos generados en la ecuación (18)

$$f(m,n,k)=f_{opt}\cdot(n-k)\cdot(m^2-m+\Delta x) \quad (18)$$

La estimación obtenida cumple con la aproximación teórica del modelo fractal y un factor de aproximación f_{opt} que se define de forma analítica, a partir de los reportes de síntesis.

Conclusiones

Finalmente, el modelo propuesto permitió validar la estimación de recursos con ecuaciones obtenidas a través del método numérico en diferencias finitas y su extrapolación a modelos con funciones iteradas, lo que aporta un método para la evaluación de los diseños y su optimización de forma dinámica. En este estudio se obtuvo una optimización de 54.05%, a nivel de utilización de recursos hardware, esto bajo un enfoque teórico matemático, que puede ser aplicado en modelo circuitales fractales. De esta manera, se puede contar con una base para el diseño de sistemas complejos, siendo un importante concepto dado que este operador se presenta con frecuencia en el campo de la ingeniería. Un caso generalizado corresponde a las redes neuronales con arquitectura profunda y códigos Reed Solomon (n,k), que se han considerado como modelos objeto de estudio, también se pueden extrapolar a modelos regenerativos en sistemas de potencia, control avanzado para convertidores de energías renovables y otros sistemas dinámicos.

Esta investigación representa un aporte para la comunidad científica en la producción de conocimiento basado en el modelo desarrollado de componentes circuitales fractales. A la vez de introducir un método de estimación de recursos basado en el operador LFC(n,k), lo que permite simplificar el diseño, así como aplicar métodos numéricos para optimización en áreas de electrónica con hardware reconfigurable. La originalidad se encuentra en el estudio de los modelos descritos en VHDL de una arquitectura concurrente, que puede ser ampliamente utilizada en diversos esquemas en ingeniería, donde se aplica el concepto de funciones iteradas para la estimación de recursos de hardware de los diseños fractales.

Referencias

Allen, J.D. (2008). Energy efficient adaptive reed-solomon decoding system. University of Massachusetts Amherst, USA. Recuperado de http://scholarworks.umass.edu/theses/91/

Biard, L. y Noguet, D. (2008). Reed-Solomon codes for low power communications, 3(2), 13-21.

Imaña, J.L., Sánchez, J., Fernández, M. (2002). Método de multiplicación canónica sobre campos GF (2m) generados por AOPs orientado a hardware reconfigurable. En II Jornadas sobre Computacion Reconfigurable y Aplicaciones JCRA2002, pp. 1-6

Kindap, N. (2004). On an architecture for a parallel finite field multiplier with low complexity based on composite fields. Computers. En IEEE Transactions on. Middle East Technical University. Recuperado de: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=508323

Magaña del Toro, R., Hermosillo-Arteaga, A.R., Romo-Organista, M.P., Carrera-Bolaños, J. (2011). Análisis con elemento finito y remalleo fractal en geotecnia Finite Element Analysis and Fractal Remeshing in Geotechnics. *Ingeniería, Investigación y Tecnología*, 12(1), 103-118. http://dx.doi.org/10.22201/fi.25940732e.2011.12n1.011

Marchesan-Almeida, G. (2007). *Códigos Corretores de Erros em Hardware para Sistemas de Telecomando e Telemetria em Aplicações Espaciais*. Pontifícia Universidade Católica do Rio Grande do Sul.

Otero, S.G., Andalaft, A.C., Vásquez, E.S. (2008). El método de diferencias finitas en evaluación de opciones reales. *Ingeniare: Revista Chilena de Ingenieria*, 16(2), 232-243. https://doi.org/10.4067/S0718-33052008000100013

Paar, C. (1996). A new architecture for a parallel finite field multiplier with low complexity based on composite fields 1 Introduction, 45(7), 856–861.

Quintana-Murillo, J. (2016). *Métodos numéricos en diferencias finitas para la resolución de ecuaciones difusivas fraccionarias*. Tesis Doctoral. Universidad de Extremadura, España. Recuperado de http://dehesa.unex.es/bitstream/handle/10662/4379/TDUEX_2016_Quintana_Murillo.pdf?sequence=1

Rivera, E., y López, H.R. (2012). Evidencia de propiedades fractales en la sucesión de Fibonacci usando wavelets. *Scientia et Technica*, 17(52), 122-128.

Rodríguez-Andina, J.J., De la Torre-Arnanz, E., Valdés-Peña, M.D. (2017). FPGAs fundamentals, advanced features, and applications in industrial electronics. *CRC Press*.

Sandoval, C., y Fedón, A. (2007). Codificador y decodificador digital Reed-Solomon programados para hardware reconfigurable. *Ingeniería y Universidad*, 11(1), 17-32.

Sandoval R,C.E., y Fedón R.A. (2008). Programación VHDL de algoritmos de codificación para dispositivos de hardware reconfigurable. *Revista internacional de métodos numéricos para cálculo y diseño en ingeniería*, 24(4), 3-11.

Sandoval-Ruiz, C. (2012). Codificador RS (n,k) basado en LFCS : caso de estudio RS (7,3). *Rev. Fac. Ing. Univ. Antioquia*, (64), 68-78. Recuperado de http://www.scielo.org.co/pdf/rfiua/n64/n64a07.pdf

Sandoval-Ruiz, C. (2013). *Modelo Optimizado del codificador Reed-Solomon (255,k) en VHDL a través de un LFSR paralelizado*. Tesis Doctoral. Universidad de Carabobo, Venezuela. Recuperado de http://produccion-uc.bc.uc.edu.ve/documentos/trabajos/2000369C.pdf

Sandoval-Ruiz, C. (2016). Modelo de estructuras reconfigurables con registro desplazamiento, para lenguaje descriptor de Hardware VHDL. *Revista Fac Ing UCV*, 31(3), 109-118.

Sandoval-Ruiz, C. (2017a). Analysis of fractal circuits and modeling through iterated functions system for VHDL case study: Reed Solomon encode. *Rev. Ciencia e Ingeniería*, 38(1), 3-16.

Sandoval-Ruiz, C. (2017b). VHDL optimized model of a multiplier in finite fields. *Ingeniería Y Universidad*, 21(2), 195-211. https://doi.org/https://doi.org/10.11144/Javeriana.iyu21-2.vhdl

Sandoval-Ruiz, C.E. (2017c). Logical-Mathematical model of encoder 2D-RS for hardware description in VHDL. *Revista Ingeniería* UC, 24(1), 28-39.

Sandoval-Ruiz, C. (2017d). VHDL Signal processing modules applying neuronal networks for SDR systems. *Revista Fac Ing UCV*, 32(1), en prensa.

Sandoval-Ruiz, C.E. (2017e). Modelo en VHDL de redes neuronales configurables aplicadas a decodificación en radio cognitivo. *Revista Ingeniería UC*, 24(3).

Sandoval-Ruiz, C. (2017f). 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.

Sandoval-Ruiz, C.E. (2018a). Control de Micro-Redes de energía renovable a través de estructuras LFCS reconfigurables en VHDL. *Ciencia y Tecnología*, 18, 71-86.

Sandoval-Ruiz, C.E. (2018b). Códigos Reed Solomon para sistemas distribuidos de energías renovables y smart grids a través de dispositivos electrónicos inteligentes sobre tecnología FPGA. *Memoria Investigaciones en Ingeniería*, 16, 37-54.

Saqib-Nazar, A. (2004). *Implementación eficiente de algoritmos criptográficos en dispositivos de hardware reconfigurable*. México: Centro de Investigación y Estudios Avanzados del Instituto Politécnico Nacional.

Skiba, Y. (2005). *Métodos y esquemas numéricos: un análisis computacional*. Capítulo 9.1. Método de Diferencias Finitas. México: UNAM.

Xilinx. (2011). Xilinx DS251, LogiCORE IP Reed-Solomon Encoder v7.1 Data Sheet. Cadence, 1-16.

Zozaya, A. (2004). *Método de las diferencias finitas y su aplicación a problemas de electrostática.*Recuperado de https://studylib.es/doc/6991583/m%C3%A9todo-de-las-diferencias-finitas-y-su-aplicaci%C3%B3n-a-probl...