REDUCCIÓN DEL NÚMERO DE TRANSISTORES EN OPERACIONES ARITMÉTICAS EN EL SISTEMA NUMÉRICO DE RESIDUOS

CARLOS ARTURO GAYOSO[1], CLAUDIO MARCELO GONZÁLEZ[1], LEONARDO ARNONE[1], JUAN CARLOS GARCIA[1], ANTONIO GARCÍA[2], EDUARDO BOEMO[3]

[1]Laboratorio de Componentes Electrónicos.
Universidad Nacional de Mar del Plata.
cgayoso@fi.mdp.edu.ar

[2]Depto. de Electrónica y Tecnología de Computadores
Universidad de Granada
agracia@ditec.ugr.es

[3]Depto. Ingeniería Informática, ETS Informática
Universidad Autónoma de Madrid
eduardo.boemo@ii.uam.es

ABSTRACT

Recently, a method for implementing arithmetical circuits for performing high speed mathematical perations with integer numbers, called the Residue Number System (RNS), has been proposed. Digital Signal Processing has been traditionally the main field of application for RNS. Different architectures has been used for the implementation of adders and multipliers in this system. One of them is based in the fact that addition can be performed using simple shift and rotation operations. Additionally, in order to reduce the system complexity and, in turn, increase computational velocity, numbers are coded into a binary codeword in which only one bit can be "one". In this work a novel architecture is proposed, which allows to reduce significantly the number of transistors needed to built basic arithmetical circuits. An introduction to the Residue Number System is presented, highlighting its main characteristics, following with the One-Hot Residue (OHR) code system, emphasising its main properties. Finally, improvement to this scheme are presented.

keywords— Residue Number System, One-Hot Residue,

RESUMEN

En los últimos años se ha propuesto el sistema numérico de residuos (*Residue Number System*, RNS) para la implementación de circuitos aritméticos con números enteros de alta velocidad. El principal campo de aplicación del RNS ha sido tradicionalmente el procesamiento digital de señales. Se han sugerido diversas arquitecturas para la construcción de sumadores y multiplicadores en este sistema. Uno de ellos se basa en el hecho de que la suma se puede resolver con operaciones simples de desplazamiento y rotación. Más aún, para reducir la complejidad y aumentar la velocidad de cómputo, se codifica cada valor mediante una combinación binaria en la que sólo uno de los bits puede tomar el valor "1".

En el presente trabajo se propone una arquitectura que permite reducir el número de transistores para construir los circuitos aritméticos básicos de manera considerable. Se comienza con una introducción del RNS destacando sus características esenciales, prosiguiendo con el sistema de codificación *One-Hot Residue* (OHR) enfatizando sus propiedades de mayor interés y se finaliza con la presentación de una variante para mejorar este esquema.

palabras clave— sistema numérico de residuos, aritmética de residuos, *One-Hot*.

REDUCCIÓN DEL NÚMERO DE TRANSISTORES EN OPERACIONES ARITMÉTICAS EN EL SISTEMA NUMÉRICO DE RESIDUOS

CARLOS ARTURO GAYOSO[1], CLAUDIO MARCELO GONZÁLEZ[1], LEONARDO ARNONE[1], JUAN CARLOS GARCIA[1], ANTONIO GARCÍA[2], EDUARDO BOEMO[3]

[1]Laboratorio de Componentes Electrónicos.
Universidad Nacional de Mar del Plata.
cgayoso@fi.mdp.edu.ar

[2]Depto. de Electrónica y Tecnología de Computadores
Universidad de Granada
agracia@ditec.ugr.es

[3]Depto. Ingeniería Informática, ETS Informática
Universidad Autónoma de Madrid
eduardo.boemo@ii.uam.es

RESUMEN

En los últimos años se ha propuesto el sistema numérico de residuos (*Residue Number System*, RNS) para la implementación de circuitos aritméticos con números enteros de alta velocidad. El principal campo de aplicación del RNS ha sido tradicionalmente el procesamiento digital de señales. Se han sugerido diversas arquitecturas para la construcción de sumadores y multiplicadores en este sistema. Uno de ellos se basa en el hecho de que la suma se puede resolver con operaciones simples de desplazamiento y rotación. Más aún, para reducir la complejidad y aumentar la velocidad de cómputo, se codifica cada valor mediante una combinación binaria en la que sólo uno de los bits puede tomar el valor "1".

En el presente trabajo se propone una arquitectura que permite reducir el número de transistores para construir los circuitos aritméticos básicos de manera considerable. Se comienza con una introducción del RNS destacando sus características esenciales, prosiguiendo con el sistema de codificación *One-Hot Residue* (OHR) enfatizando sus propiedades de mayor interés y se finaliza con la presentación de una variante para mejorar este esquema.

palabras clave— sistema numérico de residuos, aritmética de residuos, *One-Hot*.

1. INTRODUCCIÓN

La exigencia de los consumidores en disponer de productos electrónicos de menor tamaño, operados a batería y con mayor autonomía, han puesto la reducción del consumo de energía en el foco de la mayoría de los diseñadores de circuitos integrados. Para lograr este objetivo se han ideado distintas alternativas, poniendo énfasis en alguna etapa en particular del proceso de creación de un *chip*, es decir, en la parte física, tecnológica, circuital, de sistema o en el algoritmo empleado. Entre los métodos algorítmicos se encuentra el empleo de aritméticas alternativas [1], tal es el caso del RNS.

Uno de los puntos cruciales que debe tener en cuenta el diseñador de circuitos integrados de bajo consumo, es que se tenga la menor cantidad de efectos colaterales no deseados, o por lo menos que éstos estén dentro de un rango aceptable. Por ejemplo, algunas técnicas si bien logran disminuir el consumo, provocan una reducción de velocidad (aumento del período de reloj), que puede ser en algunos casos inaceptable para una aplicación determinada. Es por ello que en muchos casos la figura de mérito del diseño a desarrollar no sólo es el consumo, sino el producto retardo-potencia (*delay-power*, DP).

Este trabajo enfoca su atención en la alternativa algorítmica como técnica de reducción del consumo de energía. De manera más precisa, se logra una reducción importante en el número de transistores con respecto al OHR desarrollado en [2], que tiene muy buen comportamiento a la hora de evaluar el producto DP. La simplicidad y regularidad del OHR permite la disminución de caminos críticos con el consiguiente aumento de velocidad. A esto se suma el hecho de que la codificación *One-Hot* de los operandos conduce a una actividad circuital mínima, obteniéndose factores de disipación muy pequeños. Como consecuencia, por ejemplo para sumadores, se obtienen productos DP que son entre un 70 y un 95% menores que los sumadores *ripple-carry* [2]. También en [2] se da un ejemplo concreto de las ventajas del OHR, allí se realiza el diseño de un sintetizador de frecuencia mediante el empleo del OHR, dando como mínimo, una disminución del 90% en el producto DP con respecto al High-Agility Direct Synthesizer.

2. SISTEMA NUMÉRICO DE RESIDUOS (RNS)

El RNS [3] es un sistema para trabajar con números enteros en el cual las operaciones de suma, resta y multiplicación se pueden realizar mediante pequeñas unidades aritméticas que operan en paralelo. No existiendo ni acarreos, ni préstamos, ni productos parciales entre ellos. Los sumadores basados en aritmética tradicional deben propagar la información de acarreo de un bit a otro, de manera que a medida que aumenta la precisión se degradan las prestaciones. El RNS es una técnica eficiente para superar este problema, dado que se trabaja sobre canales independientes sin necesidad de intercambio de información entre ellos. De esta manera los sistemas basados en RNS están compuestos de una serie de canales, cada uno de ellos con un número reducido de bits. Esta característica lo hace apropiado para la realización de un número importante de aplicaciones en procesamiento digital de señales [4].

Un sistema basado en RNS se define mediante un conjunto de enteros relativamente primos entre sí $\boldsymbol{m} = \{m_1, m_2..., m_l\}$, llamados módulos. Su rango dinámico es $M = \prod m_i$ (i=1, 2, …, l), de manera que cualquier entero $0 \le x < M$ queda unívocamente definido por el conjunto de sus l residuos $[x_1, x_2, ..., x_l]$, con $x_i = x \bmod m_i$ (i=1, 2, …, l). Por ejemplo: si $\boldsymbol{m} = \{ 3, 4, 5 \}$ se tiene para $x = 25$, $\boldsymbol{m} = \{ 1, 1, 0 \}$. La condición para que x tenga una única representación RNS es que los módulos m_i deben ser relativamente primos tomados de a pares, es decir mcd(m_i, m_j) = $1 \; \forall i \neq j$.

La principal ventaja del RNS radica en su capacidad para realizar sumas, restas y multiplicaciones a alta velocidad, debido a que la aritmética de residuos se define sobre el anillo de enteros módulo m:

$$Z = (X \lozenge Y) \bmod M \quad \leftrightarrow \quad z_i = (x_i \lozenge y_i) \bmod m_i \quad (i = 1,...,L)$$

Ec. 1

donde ◊ representa suma, resta o multiplicación. En la Ec. 1 se puede apreciar el potencial del RNS, puesto que las operaciones en módulo m se calculan en paralelo sobre l canales independientes. Por ejemplo sea $\boldsymbol{m} = \{ 3, 4, 5 \}$, por lo tanto será $M = 60$ con lo que, si se trabaja sólo con enteros positivos, se pueden representar las cantidades de 0 a 59. En [5] se presenta los siguientes ejemplos:

Decimal	$m = 3$	$m = 4$	$m = 5$	
7	1	3	2	
+ 3	0	3	3	
10	1 mód 3	6 mód 4	5 mód 5	= { 1, 2, 0 }
7	1	3	2	
- 3	0	3	3	
4	1 mód 3	0 mód 4	-1 mód 5	= { 1, 0, 4 }
7	1	3	2	
x 3	0	3	3	
21	0 mód 3	9 mód 4	6 mód 5	= { 0, 1, 1 }

Debido a que no se presentan dependencias de acarreo o datos entre los canales, el rendimiento total del sistema estará dado por la velocidad de procesamiento del canal más lento. Aunque operaciones tales como división o comparación son muy difíciles de realizar, esto no limita la aplicación del RNS y, de hecho, el procesamiento digital de señales se ha transformado en su campo de aplicación preferido. Así, los algoritmos de procesamiento digital de señales, que incluyen elevado número de sumas y productos, pueden ver incrementada su velocidad de funcionamiento mediante su implementación en RNS, como se ha demostrado en aplicaciones tales como transformadas discretas, filtrado digital o procesamiento de imágenes [6][7].

3. REPRESENTACIÓN ONE-HOT RESIDUE (OHR)

La representación OHR toma ventaja de dos aspectos de la aritmética en módulo, la primera aplicable a la suma y resta y la segunda a la multiplicación. De la Tabla I, suma en módulo 7, se deduce que las filas consecutivas se generan desplazando y rotando la anterior un lugar a la izquierda. De manera que la suma se puede convertir en simples operaciones de desplazamiento y rotación. La resta se puede generar de idéntica manera, pero con rotaciones hacia la derecha.

+	**0**	**1**	**2**	**3**	**4**	**5**	**6**
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

Tabla I. *Tabla de suma en módulo 7.*

Para la multiplicación en aritmética de residuos se emplean los campos de Galois [8], que se pueden usar sólo en el caso de módulos primos (p). Se dice que un entero $g < p$ es una raíz primitiva de p si se cumple:

$$\left| g^{\,p-1} \right|_p = 1 \qquad \textbf{Ec. 2}$$

y de forma tal que las clases de residuos en módulo p; g^1, g^2, g^3, g^4,... , $g^{p-1} = 1$ son todas distintas, es decir g mód p tiene orden $p - 1$. Los campos de Galois tienen la propiedad de que todos sus elementos, salvo el cero, se pueden generar a partir de su raíz primitiva mediante la siguiente ecuación:

$$\left| q_n \right|_p = \left| g^{i_n} \right|_p \qquad \textbf{Ec. 3}$$

Por ejemplo: si $p = 19$ mediante $g = 2$ se puede establecer una relación entre el campo G(p) = { 1, 2,... 18 }, y el conjunto índice i = { 0, 1, 2,... 17 } dada por la ecuación de generación anterior, con lo que se puede establecer la siguiente tabla de equivalencias:

q_n	i_n	q_n	i_n	q_n	i_n	q_n	i_n
1	0	13	5	17	10	12	15
2	1	7	6	15	11	5	16
4	2	14	7	11	12	10	17
8	3	9	8	3	13	0	-
16	4	18	9	6	14		

Tabla II. *Isomorfismo entre el grupo multiplicativo q y el aditivo i.*

Con lo que para realizar la operación $|7 \times 9|_{19} = 6_{19}$, primero se deben buscar los índices (indexar) de 7 y 9, que son 6 y 8. Sumarlos en módulo $p - 1 = 18$, es decir $|6 + 8|_{18} = 14_{18}$, y buscar (desindexar) a que número corresponde este índice, en este caso 6_{19}, que es el resultado.

De esta manera, mediante campos de Galois, las multiplicaciones en el RNS se convierten en operaciones de indexación, suma en módulo y desindexación.

La representación OHR de un determinado número es sencilla. Para un dado módulo m cada operando se representa mediante $n = m$ bits y de manera tal que se tiene un "1" sólo en la posición del valor que se quiere representar. Por ejemplo: si $m = 7$ y se quieren representar los enteros 6_7, 4_7, 0_7, se tiene; $6_7 = 1000000_{7OHR}$, $4_7 = 0010000_{7OHR}$, $0_7 = 0000001_{7OHR}$. Esto nos indica que realizar una operación en el OHR es equivalente a cambiar el valor de a lo sumo dos líneas, con la consecuente actividad mínima del circuito y ahorro de energía.

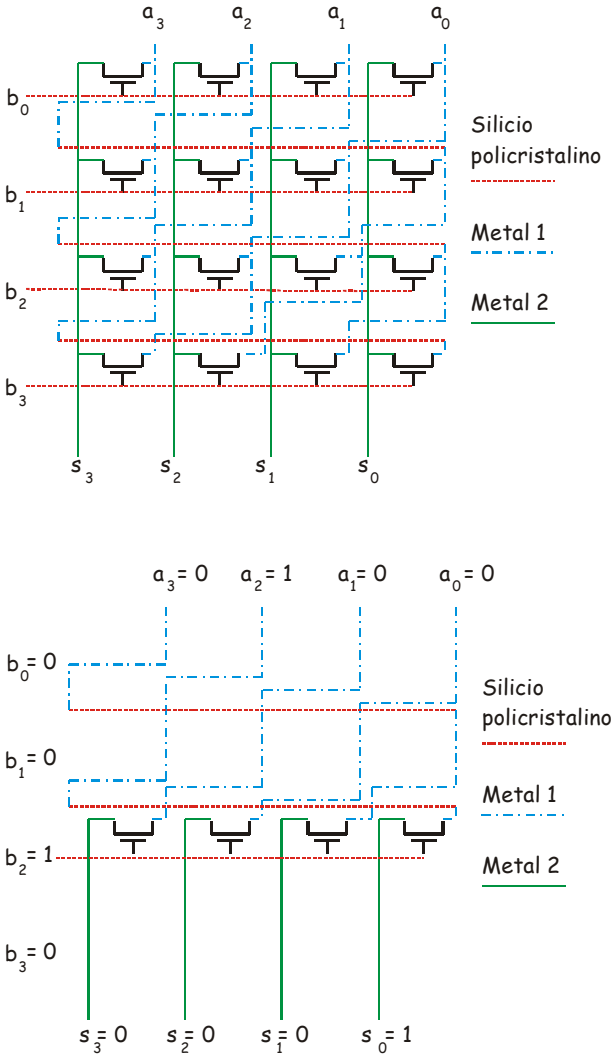

Fig. 1 *Circuito sumador para m = 4 y ejemplo con a = 2 y b = 2.*

La operación de suma mediante codificación OHR consiste entonces en rotar el primer operando un número de veces igual al valor del segundo. Esta operación se puede realizar mediante un desplazador de tonel (*barrel shifter*), como se puede apreciar en la Fig. 1.

La indexación y desindexación, para las operaciones de multiplicación, no necesitan ningún tipo de hardware pues se realiza simplemente mediante la

permutación de las líneas de los *buses* de cada canal RNS. Con lo que el hardware empleado para una operación de multiplicación es el mismo que el necesario para una suma. Lo mismo ocurre con el cálculo de valores inversos. Más aun, la conversión de módulo, es decir representar un residuo en otro módulo, consiste en el agregado de unas pocas compuertas OR. Finalmente, otras dos característica importantes del OHR, son las siguientes: si se desea sumar o multiplicar por un valor constante tampoco se debe agregar hardware, simplemente se permutan las señales de cada canal.

En [6] se muestran los productos DP para sumadores y multiplicadores en OHR contrastados con aritmética binaria. El trabajo se realizó mediante simulación Spice versión 3f4 LEVEL 2, para tecnología de 1,2 µm y con transistores de tamaño mínimo. Para los circuitos aritméticos binarios se eligió el *ripple-carry adder* debido a su menor consumo entre distintas alternativas, en tanto que, para la multiplicación, se tomó la arquitectura en árbol de Wallace. En [6] la comparación determina que la reducción en el producto DP va desde un 35 a un 95 %, para la suma y la multiplicación respectivamente, con respecto a la aritmética binaria.

4. MEJORA PROPUESTA AL OHR

El problema que presenta la implementación de sumadores y multiplicadores mediante el algoritmo de rotación es que el número de transistores crece con m^2, dificultad que se acentúa a medida que m aumenta su valor.

En [9] se presenta una propuesta alternativa para los casos en que se debe trabajar con m elevado. En estas condiciones se debe elegir un m que sea compuesto, es decir $m = m' \times m''$. Con lo que se logra que el área sea proporcional a $(m')^2 + (m'')^2 +$ (*lógica de decodificación*) en lugar de serlo a m^2. Según este método, dado un conjunto $\boldsymbol{m} = \{ m_1, m_2, \ldots m_n \}$, y siendo m_n el módulo mayor, éste se debe seleccionar de manera tal que se pueda descomponer en m_n' y m_n'', para realizar la multiplicación en el canal n mediante el algoritmo de desplazamiento con un ahorro considerable de área. Ahora bien, si m_n se puede descomponer ¿no sería mejor partir desde un comienzo con una selección así: $\boldsymbol{m} = \{ m_1, m_2, \ldots m_n', m_n'' \}$, de manera tal que no sólo mejoran las prestaciones de los multiplicadores sino que también lo hacen las de los sumadores al trabajar con menos bits en el canal n?

Para lograr una reducción importante en el número de transistores necesarios para construir un desplazador de tonel, cuando m es grande, y sin la necesidad de que m se pueda descomponer se ideó una variante simple pero eficaz.

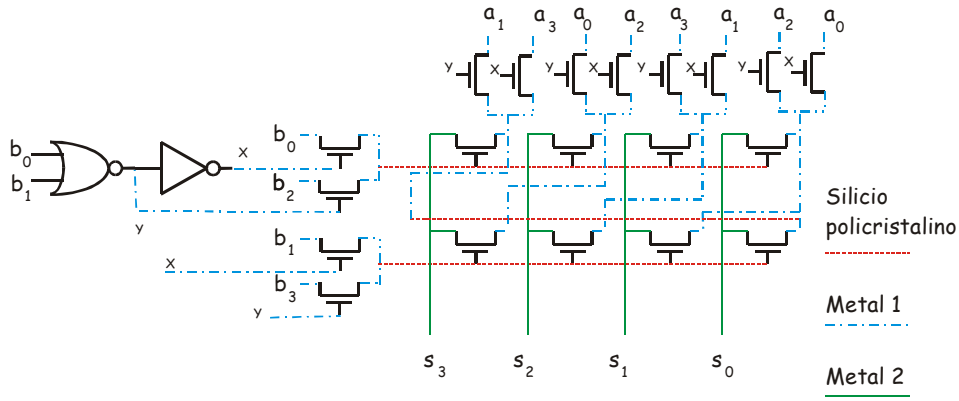

Fig. 2 *OHR_2 para m = 4.*

La idea es construir un desplazador de tonel de m x $m/2$ transistores en lugar de m x m. Luego se ve el "1" está en la parte más significativa o menos significativa del segundo operando, cuyo valor indica el número de desplazamientos-rotaciones que se deben realizar. Si el "1" está en la parte menos significativa el desplazador de tonel funciona como de costumbre, pero si está en la mitad más significativa se produce un desplazamianto anticipado de $m/2$ posiciones. En la Fig. 2 se muestra un ejemplo para $m = 4$. Se debe notar que este ejemplo es sólo a título ilustrativo, pues la reducción en el número de transistores

se aprecia a medida que el módulo crece. En detalle. Sean los números a sumar, o multiplicar, $\boldsymbol{a} = \{ a_3, a_2, a_1, a_0, \}$ y $\boldsymbol{b} = \{ b_3, b_2, b_1, b_0, \}$. La compuerta NOR detecta si b_1 o b_0 es igual a "1", en caso que así sea se tiene un "1" en el nodo x, con lo que la información de las rotaciones a realizar, contenidas en los bits b_1, b_0, pasa a las filas. Simultáneamente se activan los transistores NMOS verticales que dejan al número *a* sin modificar. En caso contrario las filas reciben la información de los bits b_3, b_2 y se produce una rotación-desplazamiento de 2 posiciones, previa a la que realizará por el desplazador de tonel.

Un análisis cuantitativo del número de transistores necesarios para construir el circuito propuesto es la siguiente. Se necesitan *m* x *m* / 2 transistores para el núcleo, 2 x *m* para el desplazamiento-rotación anticipado, 2 x (*m* / 2) para la compuerta NOR, 2 para el inversor y (*m* / 2) x 2 para las filas. Matemáticamente:

$$N_T \propto \frac{m^2}{2} + 4m + 2 \qquad \textbf{Ec. 3}$$

En la Fig. 3 se ilustra el número de transistores requeridos para una y otra realización, en tanto que, en la Fig. 4, el ahorro de área con el método propuesto, con el nombre sugerido de OHR_2.

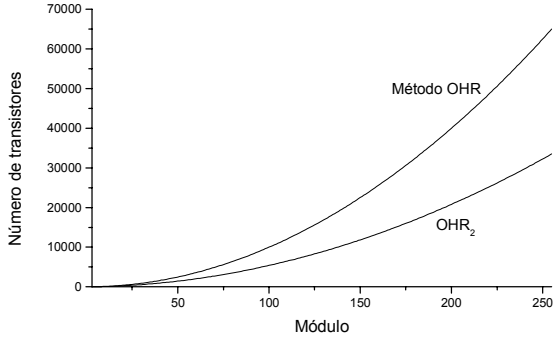

Fig. 3 *Número de transistores requeridos para realizar sumadores en módulo m según OHR y OHR_2.*

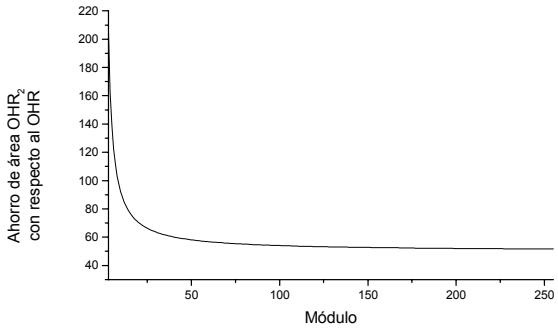

Fig. 4 *Ahorro de área del método propuesto(OHR_2) con respecto al OHR en función del módulo.*

Este esquema se puede extender de la siguiente forma (OHR_3). En lugar de detectar si el "1" ocurre en la parte más o menos significativa del segundo operando, éste se puede dividir en tres partes y aplicar el mismo procedimiento indicado precedentemente. En este caso se necesitan 2 compuertas NOR para detectar en que tercio se encuentra el "1". La cuenta de transistores es: *m* x *m* / 3 transistores para el núcleo, 3 x *m* para el desplazamiento-rotación anticipado, 2 x 2 x (*m* / 3) para las dos compuertas NOR, 4 para dos inversores, 4 para una compuerta NOR de 2 entradas, que indica si el "1" está en el tercio más significativo, y (*m* / 3) x 3 para las filas. Esto es:

$$N_T \propto \frac{m^2}{3} + \frac{24 + 16m}{3} \qquad \textbf{Ec. 4}$$

A medida que *m* crece se puede subdividir aún más, logrando un enorme ahorro de área.

5. CONCLUSIONES Y TRABAJOS FUTUROS

En el presente trabajo se realizó un apretado resumen de las ventajas del RNS y del OHR en cuanto a su potencial de velocidad y bajo consumo. Debido principalmente a la actividad mínima del circuito cuando se realizan operaciones de suma, resta o multiplicación. Más aún, se presentó un mecanismo sencillo para mejorar el OHR que trae aparejado un ahorro de área muy importante en el circuito integrado.

Ya demostrada la ventaja, en cuanto al área se refiere del método propuesto, resta, por el momento, realizar simulaciones Spice de los nuevos esquemas y contrastarlos con los OHR original y binarios para determinar el comportamiento en cuanto al producto DP.

6. REFERENCIAS

[1] C. Nagendra, R. M. Owens, and M. J. Irwin, *"Unifying carry-sum and signed-digit number representations for low power*", *Proc. 1995 International Symposium of Low Power Design*.

[2] W. A. Chren. *"One-hot Residue Coding for Low Delay-Power Product CMOS Design". IEEE Transactions on Circuits and Sistema-II: Analog and Digital Signal Processing". Vol. 45 Nº 3, marzo de 1998.*

[3] N. S. Szabo and R. I. Tanaka, *"Residue Arithmetic and Its Applications to Computer Technology",* McGraw-Hill, NY, 1967.

[4] M. A. Soderstrand, W. K. Jenkins, G. A. Jullien and F. J. Taylor, "*Residue Number System Arithmetic: Modern Applications in Digital Signal Processing"*, IEEE Press, 1986.

[5] Fred J. Taylor, *"Residue Arithmetic: A Tutorial with Examples"*, IEEE Computer Magazine, Mayo, 1984.

[6] W. A. Chren, "*RNS-Based Enhancements for Direct Digital Frequency Synthesis"*, IEEE Transactions on Circuits and Systems II, vol. 42. no. 8, pp. 516-524, Aug. 1995.

[7] J. Ramírez, A. García, P. G. Fernández, L. Parrilla and A. Lloris, "*RNS-FPL Merged Architectures for Orthogonal DWT*", Electronics Letters, vol. 36, no. 14, pp. 1198-1199, Jul. 2000.

[8] Stephen S. Yau, Jackson Chung, *"On the Design of Modulo Arithmetic Units Based on Cyclic Groups",* IEEE Transactions on Computers, Vol. 25 Nº 11, Noviembre 1976.

[9] G. Lakhani, *"Some fast Residual Arithmetics Adders",* International Symposium of Electronics, 1994 Vol. 77, Nº 2.