UNA ARQUITECTURA PARA LA TRANSFORMADA NUMÉRICA DE MERSENNE

Oscar N. Bria Horacio A. Villagarcía Daniel A. Gil

Dpto. de Informática - Cs. Ex. - UNLP
CeTAD - Fac. de Ing. - UNLP
o.bria@ieee.org

Casilla de Correos 317
(1900) La Plata
ARGENTINA

Resumen

La convolución exacta de secuencias de números enteros es una de las operaciones más importantes del procesamiento digital de señales.
Cuando se requiere exactitud no puede usarse el tradicional método de la DFT para acelerar el proceso de cálculo debido a los factores de peso trascendentes presentes en las transformadas de Fourier [1] [5].
La Transformada Numérica de Mersenne (MNT) es una alternativa a la aplicación directa de la convolución, que podría resultar en arquitecturas más simples (menos complejas) según se muestra en [2].
En este trabajo se presenta un arquitectura simple que implementa la MNT, basada unicamente en registros de desplazamientos y sumadores en complemento a uno.
Los registros de desplazamientos resuelven las multiplicaciones en forma cableada, representando así una complejidad de $\mathcal{O}(1)$.
Los sumadores complemento a uno son una variante carry look-ahead, los cuales presentan un retardo moderado y son fáciles de diseñar.
La arquitectura aquí presentada ha sido descripta en VHDL y simulada.

Palabras Claves: Convolución Circular de Números Enteros, Transformada Numérica de Mersenne, Modelaje en VHDL.

UNA ARQUITECTURA PARA LA TRANSFORMADA NUMÉRICA DE MERSENNE

1 La Transformada Numérica de Mersenne

Para una introducción a la transformada numérica de Mersenne (MNT) ver [5] y [1].

La MNT directa se define como:

$$A_k = \left\langle \sum_{n=0}^{p-1} a_n 2^{nk} \right\rangle_{M_p} \qquad ; \qquad 0 \leq k \leq p-1$$

La MNT inversa se define como: $a_n = \left\langle \hat{N} \sum_{k=0}^{p-1} A_k 2^{-kn} \right\rangle_{M_p}$, con $0 \leq n \leq p-1$ y $\hat{N} = \frac{1-M_p}{p} = \frac{2-2^p}{p}$

La notación $\langle y \rangle_x$ equivale a "y módulo x". M_p es el número de Mersenne de base 2 y generado por el primo p. $M_p = 2^p - 1$.

La MNT goza de la, así llamada, propiedad de la convolución. Es decir la convolución circular se transforma en producto en el espacio transformado.

El siguiente es un ejemplo numérico del uso de la MNT para el cálculo de la convolución circular de números enteros:

Para $M_p = 127$ (base 2 y $p = 7$) y $\hat{N} = -18$,

1. $a_n = \{1, 2, 3, 4, 3, 2, 1\}$ $\qquad$ $A_k = \{16, -29, -14, -40, 61, -56, -58\}$.
2. $b_n = \{1, 1, 0, 0, 0, 0, 0\}$ $\qquad$ $B_k = \{2, 3, 4, 9, 17, 33, -62\}$.
3. Multiplicando A_k por B_k se obtiene $C_k = \{32, 40, 57, 21, 21, 57, 40\}$.
4. Calculando la transformada inversa se obtien $c_n = \{2, 3, 5, 7, 7, 5, 3\}$.

Puede verse que los factores de peso involucrados son potencias enteras de dos, las cuales pueden resolverse facilmente con rotaciones si se usa representación numérica binaria.

Además, de la definición de la MNT directa puede deducirse que las operaciones de suma pertinentes pueden resolverse con aritmética modular de complemento a uno [5].

La Figura 1 representa el detalle de las matrices de bits rotadas que intervienen en el cálculo de los primeros tres coeficientes A_k del ejemplo anterior.

$a_0 = 1$	$\circ\ \circ\ \circ\ \circ\ \circ\ \circ\ 1$	< 0
$a_1 = 2$	$\circ\ \circ\ \circ\ \circ\ \circ\ 1\ \circ$	< 0
$a_2 = 3$	$\circ\ \circ\ \circ\ \circ\ \circ\ 1\ 1$	< 0
$a_3 = 4$	$\circ\ \circ\ \circ\ \circ\ 1\ \circ\ \circ$	< 0
$a_4 = 3$	$\circ\ \circ\ \circ\ \circ\ \circ\ 1\ 1$	< 0
$a_5 = 2$	$\circ\ \circ\ \circ\ \circ\ \circ\ 1\ \circ$	< 0
$a_6 = 1$	$\circ\ \circ\ \circ\ \circ\ \circ\ \circ\ 1$	< 0
$A_0 = 16$	$\circ\ \circ\ 1\ \circ\ \circ\ \circ\ \circ$	

	$\circ\ \circ\ \circ\ \circ\ \circ\ \circ\ 1$	< 0
	$\circ\ \circ\ \circ\ \circ\ 1\ \circ\ \circ$	< 1
	$\circ\ \circ\ \circ\ 1\ 1\ \circ\ \circ$	< 2
	$\circ\ 1\ \circ\ \circ\ \circ\ \circ\ \circ$	< 3
	$\circ\ 1\ 1\ \circ\ \circ\ \circ\ \circ$	< 4
	$1\ \circ\ \circ\ \circ\ \circ\ \circ\ \circ$	< 5
	$1\ \circ\ \circ\ \circ\ \circ\ \circ\ \circ$	< 6
$A_1 = -29$	$1\ 1\ \circ\ \circ\ \circ\ 1\ \circ$	

	$\circ\ \circ\ \circ\ \circ\ \circ\ \circ\ 1$	< 0
	$\circ\ \circ\ \circ\ 1\ \circ\ \circ\ \circ$	< 2
	$\circ\ 1\ 1\ \circ\ \circ\ \circ\ \circ$	< 4
	$\circ\ \circ\ \circ\ \circ\ \circ\ 1\ \circ$	< 6
	$\circ\ \circ\ \circ\ \circ\ 1\ 1\ \circ$	< 8
	$\circ\ \circ\ 1\ \circ\ \circ\ \circ\ \circ$	< 10
	$\circ\ 1\ \circ\ \circ\ \circ\ \circ\ \circ$	< 12
$A_2 = -14$	$1\ 1\ 1\ \circ\ \circ\ \circ\ 1$	

Figura 1: Matrices para A_0, A_1, y A_2

2 Complejidad de la MNT

2.1 Funcionamiento de la Matriz de Datos de la MNT

Se llama matriz absoluta a la estructura de datos sugerida directamente por la fórmula que define la MNT. Ver las Figuras 1 y 2.

Se llama matriz diferencial a la estructura que aparece después de observar la recursividad de las rotaciones. Ver la Figura 3.

Las siguientes observaciones son inmediatas:

1. La matriz absoluta es la estructura natural en una implementación que calcule en forma paralela los p coeficientes de cada bloque de entrada. En ese caso, cada una de las p matrices absolutas puede implementarse en forma totalmente cableada.

2. La matriz diferencial es la estructura a considerar cuando los coeficientes de cada

bloque se van a calcular en forma secuencial. En este caso todas las filas de la matriz pueden actualizarse en un sólo paso si se cablea convenientemente.

Un ejemplo puede verse en la Figura 4 para $p = 7$. Se observa que el arreglo puede implementarse con $p - 1$ registros de desplazamientos convenientemente cableados.

El tiempo de rotación diferencial es de $\mathcal{O}(1)$, es decir no depende de la dimensión de los datos. En la práctica la matriz diferencial se puede implementar usando varios registros de desplazamientos.

3. Existe una alternativa más económica, que consiste en no usar una estructura matricial de p registros de p bits. Se puede usar una sóla fila dinámica consistente de un registro programable shift-barrel, y memoria.

En este trabajo se ha decidido continuar en la dirección de la segunda alternativa. Es decir, la estructura de datos es una matriz diferencial, la cual permite resolver las multiplicaciones pertinentes a cada coeficiente en un tiempo de $\mathcal{O}(1)$.

2.2 Retardo y Costo de los Operadores Aritméticos

El retardo y costo de diferentes arquitecturas de operadores aritméticos se resume en la Tabla 1. Para una justificación de los resultados ver [3].

Obsérvese que para dos entradas un multiplicador puede hacerse tan rápido como un sumador. A su vez un multiplicador múltiple puede hacerse tan rápido como un sumador simple. Por supuesto el precio de la rapidez es la mayor complejidad de la arquitectura.

2.3 Complejidad Operacional de la MNT

Si se considera que el cálculo de los p coeficientes de la MNT directa se efectúa en forma secuencial, entonces la estructura de datos adecuada es la matriz diferencial.

El cálculo de cada coeficiente involucra p operaciones de multiplicación y una suma de p operandos:

- La matriz diferencial, convenientemente cableada reduce la complejidad de las p operaciones de multiplicación a $\mathcal{O}(1)$.

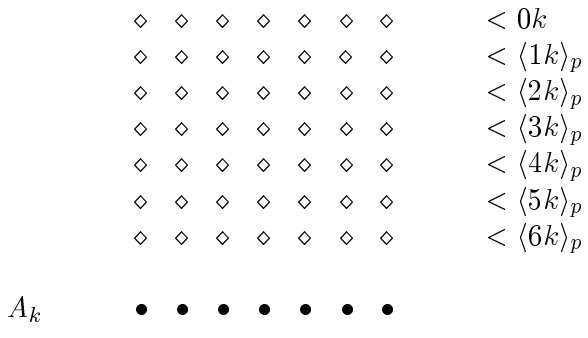

Figura 2: Matriz Absoluta

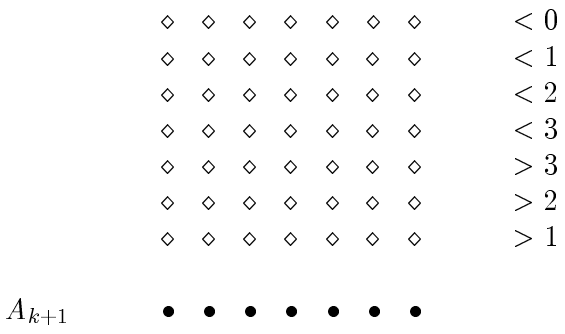

Figura 3: Matriz Diferencial

```
6    5    4    3    2    1    0
6 <- 5 <- 4 <- 3 <- 2 <- 1 <- 0
5 <- 3 <- 1 <- 6 <- 4 <- 2 <- 0
4 <- 1 <- 5 <- 2 <- 6 <- 3 <- 0
4 -> 1 -> 5 -> 2 -> 6 -> 3 -> 0
5 -> 3 -> 1 -> 6 -> 4 -> 2 -> 0
6 -> 5 -> 4 -> 3 -> 2 -> 1 -> 0
```

Figura 4: Cableado de la Matriz de Datos ($p = 7$)

- Si se considera como operador elemental de suma uno de p entradas, entonces obviamente la complejidad operacional de cada suma múltiple es también de $\mathcal{O}(1)$. Si se considera como operación elemental la suma de dos operandos, entonces la complejidad, para cada coeficiente, es de $\mathcal{O}(P)$.

Los resultados anteriores se han resumido en la Tabla 2.

2.4 Complejidad Total de la MNT

Todas las matrices observadas son cuadradas. Se deduce que la MNT es una transformada de bloques que tiene la particularidad de que el número de bits utilizados para representar y procesar los datos es igual al tamaño del bloque. Esto es una desventaja cuando quiere procesarse un bloque relativamente largo comparado con la precisión requerida por un problema dado.

Desde el punto de vista del análisis de retardos y costos, lo importante es hacer intervenir este hecho, pues la dimensionalidad de los datos es del mismo orden que el número de bits utilizados en su representación para el cálculo.

En la Tabla 3 se presentan las complejidades totales en cuanto a retardo y costo para algunas combinaciones de número y tipos de operadores. En todos lo casos se ha supuesto el uso de una matriz diferencial. Obsérvese que el operador II es un operador de dos entradas, y los operadores IX y XII son operadores múltiples (de dos vectores con p entradas cada uno).

	Operador	Retardo	Costo
I	Sumador ripple-carry	$\mathcal{O}(N)$	$\mathcal{O}(N)$
II	Sumador carry look-ahead	$\mathcal{O}(logN)$	$\mathcal{O}(N)$
III	Multiplicador shift & add (basado en I)	$\mathcal{O}(N^2)$	$\mathcal{O}(N)$
IV	Multiplicador shift & add (basado en II)	$\mathcal{O}(NlogN)$	$\mathcal{O}(N)$
V	Multiplicador ripple-carry	$\mathcal{O}(N)$	$\mathcal{O}(N^2)$
VI	Multiplicador carry-save	$\mathcal{O}(N)$	$\mathcal{O}(N^2)$
VII	Multiplicador fast	$\mathcal{O}(logN)$	$\mathcal{O}(N^2)$
IIX	Sumador de N entradas (basado en I)	$\mathcal{O}(NlogN)$	$\mathcal{O}(N^2)$
IX	Sumador de N entradas (basado en II)	$\mathcal{O}((logN)^2)$	$\mathcal{O}(N^2)$
X	Sumador de N entradas (basado en V)	$\mathcal{O}(N)$	$\mathcal{O}(N^2)$
XI	Sumador de N entradas (basado en VI)	$\mathcal{O}(N)$	$\mathcal{O}(N^2)$
XII	Sumador de N entradas (basado en VII)	$\mathcal{O}(logN)$	$\mathcal{O}(N^2)$

Tabla 1: Retardo y Costo de Operadores Aritméticos

Nro total de operaciones de 2 entradas	$\mathcal{O}(p^2)$
Nro total de operaciones de p entradas	$\mathcal{O}(p)$

Tabla 2: Complejidad Operacional de la MNT

3 Unidades de la Arquitectura

Para la definición de la arquitectura se ha seguido un diseño top-down, el cual corresponde cuando se usa a VHDL como herramienta de especificación.

La arquitectura global para la transformada numérica de Mersenne directa (dMNT) se puede ver en la Figura 5. Esta arquitectura tiene tres unidades constitutivas:

- Unidad de matriz de datos, MTU.
- Unidad aritmética complemento a uno, A1U.
- Unidad de control central, CCU.

3.1 Unidad de matriz de datos, MTU

Entonces, se define a la MTU como una matriz $p \times p$, donde cada fila es un registro de desplazamientos (sr). Estos sr's tienen un arreglo especial cableado para resolver los desplazamientos requeridos por la MNT en $\mathcal{O}(1)$ para cada uno de los coeficientes.

3.2 Unidad aritmética complemento a uno, A1U

Se impone la búsqueda de una arquitectura de complejidad intermedia con p sumadores, para que el costo de los sumadores sea comparable al de los registros fila de la MTU. Por lo tanto ninguna de las alternativas sugeridas en la Tabla 3 resulta ser conveniente.

Una alternativa adecuada es usar un arreglo de p sumadores de dos entradas conectados en forma piramidal. Este arreglo, para $p = 5$, puede observarse en la arquitectura detallada para la Transformada Numérica de Mersenne que se muestra en la Figura 6.

Operador	II	IX	XII
Retardo total	$\mathcal{O}(p^2 \log p)$	$\mathcal{O}(p(\log p)^2)$	$\mathcal{O}(p \log p)$
Costo total	$\mathcal{O}(p)$	$\mathcal{O}(p^2)$	$\mathcal{O}(p^2)$

Tabla 3: Complejidad Total de la MNT

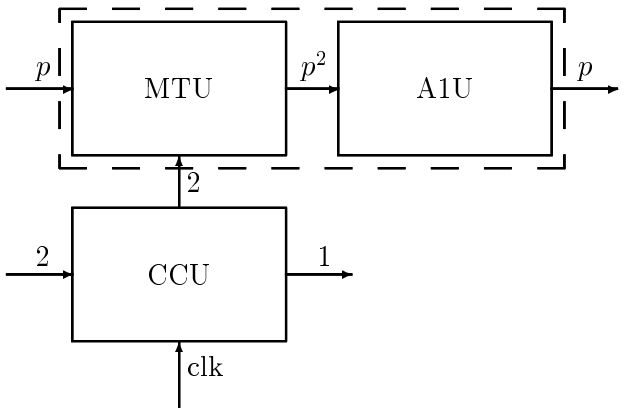

Figura 5: Arquitectura Global

3.3 Unidad de control central, CCU

Sólo interviene en forma directa sobre MTU, pues la A1U es asincrónica. La CCU es independiente de p. A esta unidad se la implementa como una máquina de número finito de estados.

4 Retardo y Costo de la Arquitectura

En la Figura 6 se ha implementado la unidad A1U como una cascada de sumadores binarios de p cifras. El retardo de una implementación como esta es $\mathcal{O}(r \log p)$, donde r es el retardo individual de cada sumador. Si se consideran sumadores carry look-ahead entonces el retardo individual es $\mathcal{O}(\log p)$.

De la simple observación de la Figura 6 puede deducirse que la profundidad del bloque sumador es $\lceil \log_2 p \rceil$. El retardo de cómputo de un vector de largo p es $\mathcal{O}(p\ (\log p)^2)$, y el costo de la implementación es $\mathcal{O}(p^2)$.

Un estudio más detallado del retardo nos lleva a la siguiente ecuación conservadora (se ha incluído el tiempo de carga de los registros):

$$R = 2p(1 + \lceil \log_2 p \rceil^2)$$

El número S de registros de desplazamientos de p bits:

$$S = p - 1$$

El número A de sumadores binarios ca1 de p bits:

$$A = p - 1$$

Por ejemplo, para $p = 7$ se tiene $R(7) = 140$ unidades de tiempo, $S(7) = 6$ y $A(7) = 6$.

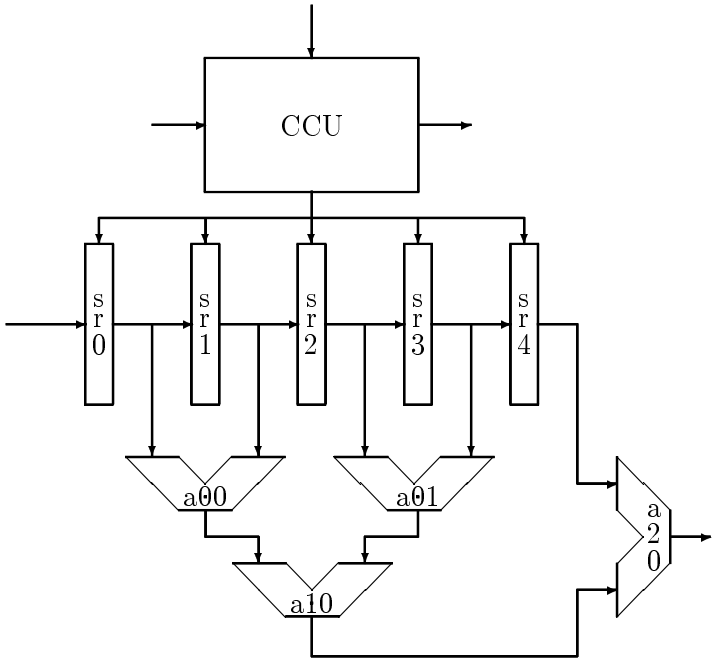

Figura 6: Arquitectura Detallada ($p = 5$)

5 Declaración de Entidades Globales

A continuación se muestran las entidades VHDL que definen las interfases de los tres bloques constitutivos de la arquitectura global (ver la Figura 5) que se viene discutiendo. En estas declaraciones no se ha usado la capacidad de expresión más compacta de VHDL a fin de mantener la claridad.

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY ccu IS -- Central Control Unit.
    PORT
        (
         reset     :  IN   STD_ULOGIC;
         input     :  IN   STD_ULOGIC;
         clk       :  IN   STD_ULOGIC;
         read      :  OUT  STD_ULOGIC;
         rotate    :  OUT  STD_ULOGIC;
         output    :  OUT  STD_ULOGIC
        );
END ccu;

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY mtu7 IS -- Matrix Unit.
    PORT
        (
        input                    :  IN   STD_ULOGIC_VECTOR(7-1 DOWNTO 0);
        read                     :  IN   STD_ULOGIC;
        rotate                   :  IN   STD_ULOGIC;
        o6, o5, o4, o3, o2, o1, o0:  OUT  STD_ULOGIC_VECTOR(7-1 DOWNTO 0)
        );
END mtu7;
```

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY m1u7 IS -- Multiple Ca1 Sum Unit.
    PORT
        (
        x6, x5, x4, x3, x2, x1, x0:  IN   STD_ULOGIC_VECTOR(7-1 DOWNTO 0);
        y6, y5, y4, y3, y2, y1, y0:  IN   STD_ULOGIC_VECTOR(7-1 DOWNTO 0);
        output                    :  OUT  STD_ULOGIC_VECTOR(7-1 DOWNTO 0)
        );
END m1u7;
```

Por supuesto la entidad m1u7 puede tener diferentes arquitecturas, e.g.:

- Una simple cascada de s1u7 (Figura 6). Si se usa una arquitectura carry look-ahead como la vista, entonces se tendrá un retardo de $\mathcal{O}(plog_2p)$ y un costo de $\mathcal{O}(p)$.
- Una arquitectura ad hoc de sumador múltiple con un retardo de $\mathcal{O}(log_2p)$ y un costo de $\mathcal{O}(p^2)$.

6 Conclusiones

Se ha definido una arquitectura de complejidad intermedia para la transformada directa de Mersenne, la cual resulta simple de especificar en un lenguaje de descripción de hardware.

La arquitectura detallada de la Figura 6 ha sido completamente especificada y simulada en VHDL [6]. Dicha arquitectura está a nivel de detalle RTL (Register Transfer Level), por lo que es adecuada para la síntesis automática haciendo uso de un compilador de silicio [4].

Precisamente, el próximo paso es sintetizar la especificación anterior en VLSI usando las herramientas de síntesis de Compass^{TM} y Altera^{TM} disponibles en nuestro laboratorio.

Referencias

[1] Oscar N. Bria y Horacio A. Villagarcía Wanza,"VHDL Functional Model for Integer Circular Convolution based on the Mersenne Number Transform," *Anales del Tercer Workshop Iberchip*, Departamento de Ingeniería Eléctrica, CINVESTAV - Instituto Politécnico Nacional. México, DF, México, 19 al 21 de Febrero de 1997.

[2] Oscar N. Bria, "Complejidad Computacional de la Transformada de Mersenne," *Reporte Técnico onb98-1 - CeTAD*, Facultad de Ingeniería, UNLP, 1998.

[3] M. Davio, J.-P. Deschamps and A. Thayse, Digital Systems with Algorithm Implementation, John Wiley & Sons, 1983.

[4] Lluís Terés, *VHDL Lenguaje Estándar de Diseño Electrónico*, McGraw Hill, 1997.

[5] John Proakis, Charles Rader, Fuyun Ling, and Chrysostomos Nikias, *Advanced Digital Signal Processing*, MacMillan, 1992.

[6] Scott Thibault, *GMVHDL Compiler Demostration Version 4.1*, Green Mountain Computing Systems, 1995.