Metodología de prototipado rápido para sistemas embebidos utilizando lógica programable

Melo, Hugo Maximiliano
Gutiérrez, Francisco Guillermo
Cassasnovas, Marcelo Oscar
Picco, Juan Eduardo
Universidad Tecnológica Nacional, Facultad Regional Córdoba

Abstract
"Se presenta una metodología para el desarrollo de prototipos con lógica programable. Se describen las herramientas de alto nivel utilizadas y se desarrolla, paso a paso, un módulo que luego es incorporado a un sistema embebido. Como ejemplo se utiliza un banco de filtros para la transformada Wavelet".

Palabras Clave
FPGA, DSP, Wavelet, Prototipado Rápido, Filtros FIR, XPS, EDK, SDK.

Introducción

La velocidad con que se actualizan y modifican las tecnologías en electrónica, informática y comunicaciones hace que sea necesario disponer de herramientas que permitan la validación de hipótesis mediante un prototipado rápido, durante la etapa de desarrollo de sistemas complejos que utilizan sistemas embebidos. No sólo debido a la abstracción del hardware, si no al momento de probar y validar nuevos tipos de tecnologías.

El método descripto en el presente trabajo hace uso de uno de los programas más difundidos en el área de Ciencias de ingeniería, MatLab, perteneciente a la empresa MathWorks y su módulo asociado Simulink, que es una plataforma versátil de diseño y simulación de sistemas dinámicos, lo que permitió que integrantes del proyecto con poca experiencia en la metodología de desarrollo en lógica programable, pudiesen probar ideas y realizar simulaciones que con poco esfuerzo pueden ser llevadas al hardware.

A partir del año 2005 los dispositivos de lógica programable han incrementado su participación en el mercado de los sistemas embebidos. Básicamente estos dispositivos están constituidos por celdas lógicas que pueden ser configuradas para realizar distintas operaciones en forma concurrente, dando una potencia de cálculo inigualable con dispositivos secuenciales como son los microcontroladores. Esta potencialidad implica mayores complicaciones al momento del desarrollo y verificación de sistemas basados en este tipo de tecnología, ya que el diseñador se encuentra configurando hardware mediante software, a esto hay que sumarle las altas velocidades para las cuales vienen preparados estos dispositivos.

Las FPGAs, que son los dispositivos lógicos programables con mas capacidad y difusión en el mercado actual, incluyen cada vez con mas frecuencia a procesadores embebidos en silicio permitiendo combinar hardware configurable con procesadores ya verificados en silicio capaces de correr software convencional. Estos procesadores van desde PowerPC hasta los ARM® dual-core Cortex™-A9 embebidos en la familia Zynq-7000 de la empresa XILINX.

Las empresas fabricantes de componentes de hardware reconfigurable han invertido mucho esfuerzo en el desarrollo de herramientas de alto nivel que faciliten la implementación de sistemas complejos en sus dispositivos. Un ejemplo de este tipo de software es el System Generator de la empresa XILINX. Este programa contiene un conjunto de bloques funcionales para utilizar con el SimuLink. Emplea el concepto de "cajas negras" y de abstracción de hardware, con el objetivo de poder llevar a cabo un diseño en bloques funcionales y así obtener una concepción de alto nivel que pueda ser simulada utilizando recursos ya disponibles en SimuLink. El System Generator puede ser también utilizado como herramienta de integración, ya que es posible definir nuevos bloques con códigos VHDL propietarios.

Una vez implementado y simulado el prototipo es necesario bajar el sistema a la plataforma de Hardware seleccionada. La empresa XILINX propone como herramienta de alto nivel para desarrollo de sistemas embebidos el EDK (Embedded Development Kit). Esta plataforma de software se compone del XPS (Xilinx Platform Studio) para el desarrollo del hardware y el SDK (Software Development Kit) para el desarrollo de software. El SDK esta basado en la plataforma Eclipse e incorpora todos los plugins necesarios para el desarrollo y depuración de software en los procesadores embebidos en la lógica programable.

Estas herramientas están pensadas para ayudar al diseñador y poder disminuir el tiempo de desarrollo. El EDK permite diseñar sistemas mediante gráficos, utilizando bloques funcionales, como en este caso el bloque generado por SimuLink y System Generator. Cada bloque funcional es un dispositivo de hardware distribuido en forma de IP (Intellectual Property). Los mismos hacen uso de las celdas configurables de las FPGAs para generar el dispositivo físico.

Los códigos de MatLab auxiliares serán reemplazados por códigos de programa implementado en los PowerPC disponibles en la plataforma.

El sistema fue desarrollado sobre la placa de desarrollo XUPV2P que contiene una FPGA Virtex-II Pro de XILINX.

Wavelet

La teoría de Wavelets trabaja de manera similar a la Teoría de Fourier, la cual dice que una señal puede ser representada por una serie de funciones sinusoidales y de esta forma es más sencillo su análisis, dado que mantiene las ideas principales de un análisis tiempo-frecuencia utilizando una ventana de análisis diferente.

Las Wavelets son familias de funciones que se emplean para el análisis de la señal de interés con el objeto de obtener determinadas características de espacio, tamaño y dirección. La familia está definida por:

$$h_{a,b} = \frac{h\left(\frac{x-b}{a}\right)}{\sqrt{|a|}} \forall a, b \in \Re \wedge a \neq 0$$

La familia Wavelet se genera desde una función madre h(x), que puede ser modificada y caracterizada mediante la modificación de los parámetros a y b para obtener traslaciones y escalado temporal. De esta manera se trata de lograr la mejor concentración en información de tiempo y frecuencia [1].

Las transformadas Wavelet se clasifican en Transformadas Wavelet Discretas (DWT) y Transformadas Wavelet Continuas (CWT) [2].

Los principales usos que se le brinda a la DWT están ligados a la compresión de imágenes, aprovechando su capacidad de separar la información redundante, mientras que la CWT esta principalmente abocada a la búsqueda de discontinuidades de señales, y como principal campo de acción en aplicaciones médicas.

Tipos de Wavelet

Según la aplicación se selecciona la Wavelet madre *Figura 1* de la cual se derivan las familias de señales, al aplicarles propiedades de dilatación, contracción y desplazamiento temporal.

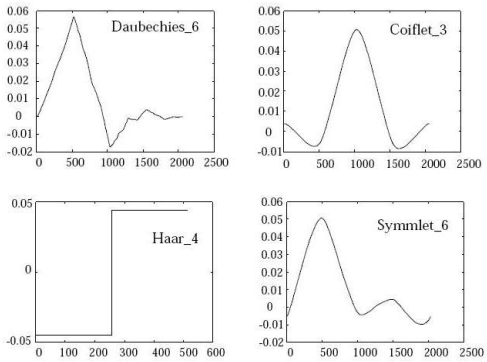

Figura 1: Funciones de Wavelet madre

Transformada Wavelet Discreta en 2-D

Para realizar la Transformada Wavelet Discreta (DWT) se puede aplicar la metodología de bancos de filtros. Estos bancos son filtros pasa bajos y pasa altos seguido de etapas de down sampling que generan la descomposición. En la *Figura 2 s*e presenta un esquema de banco de filtros para una descomposición simple. De manera similar,la reconstrucción es realizada por bancos de filtros y up sampling de la señal.

El decimado (Down Sampling) y undecimado (Up Sampling) indican decremento o incremento, respectivamente, del número de muestras, lo cual se logra eliminando una muestra o intercalando un cero entre ellas, antes del procedimiento [3].

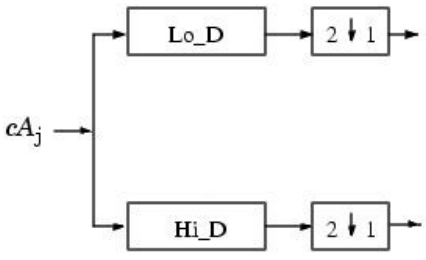

Figura 2: Descomposición Simple

Una imagen se puede representar por una matriz de datos en donde cada elemento representa un píxel. En caso de ser una imagen en color, la misma puede representarse por sus componentes RGB o YCrCb. Para aplicar la transformada Wavelet en dos dimensiones utilizando el método de filtros separables, es necesario recorrer la matriz de dos maneras, primero por filas y luego por columnas como puede verse en la *Figura 3*.

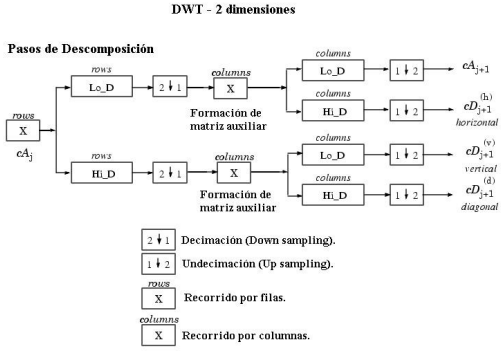

Figura 3: Wavelet 2-D

La energía normalizada de una sub-imagen formada por N coeficientes de Wavelet se define como:

$$E_{ni} = \frac{1}{N} \cdot \sum_{j,k} [D_{ni} \cdot (b_j, b_k)]^2$$

La característica de energía Wavelet {Eni} n=1...d, i= H, V, D refleja la distribución de energía a lo largo del eje de frecuencia sobre una escala y en una orientación determinada.

La energía de las imágenes se concentra en las frecuencias bajas. Una imagen tiene un espectro cuya amplitud se reduce con el incremento de las frecuencias. Estas propiedades quedan reflejadas en la Transformada Wavelet Discreta de la imagen [4].

En compresión y en algunas otras aplicaciones de la transformada se hace necesario aplicar una técnica multinivel. Esta se obtiene aplicando sucesivamente las transformadas a la parte de aproximación de la etapa anterior como puede verse en la *Figura 4*.

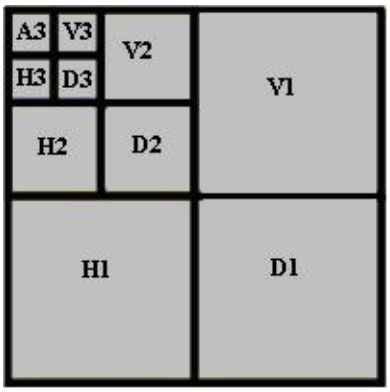

Figura 4: Niveles de Wavelet en 2-D

Implementación

Para la implementación de la transformada Wavelet en 2-D es necesario recorrer las matrices de datos por filas y luego por columnas. A modo de prueba de concepto se optó por implementar modularmente las distintas etapas de la transformada.

Cada una de las etapas se presenta con un módulo independiente y la unión entre ambos se realiza con un código de MatLab que posteriormente será reemplazado en la FPGA por rutinas de manipulación de datos

A continuación se describe el método empleado y los resultados obtenidos

Filtro FIR

Filtro FIR en su forma directa.

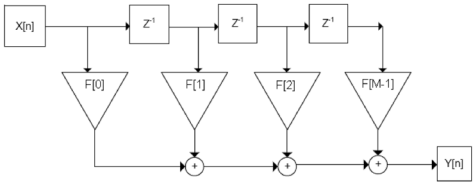

Figura 5: Estructura del filtro FIR

La salida Y[n] del filtro no depende de los valores previos de la misma *(Figura 5)*, solo de los valores actuales y/o previos de la entrada X[n], es decir, es un sistema no recursivo. Esto hace que los FIR (Finite Impulse Response) sean siempre estables, teniendo todos sus polos en el origen.

Se destaca que el parámetro M-1 es el orden del polinomio y orden del filtro, mientras que la cantidad de coeficientes del filtro es M.

Los filtros FIR a implementar son los que permiten realizar la transformada Wavelet por el método de bancos de filtros. Estos coeficientes se obtienen desde la ventana de comando de MatLab con la siguiente expresión:

```
[LO_D,HI_D,LO_R,HI_R]= FILTERS('db3')
```

'db3' le indica a la función de MatLab que la Wavelet madre es una Daubechies 3. Los filtros resultantes para esta Wavelet son de orden 5 con un total de 6 coeficientes.

A continuación se realiza el esquemático de la *Figura 6* en entorno SimuLink, utilizando bloques propios de SimuLink y System Generator.

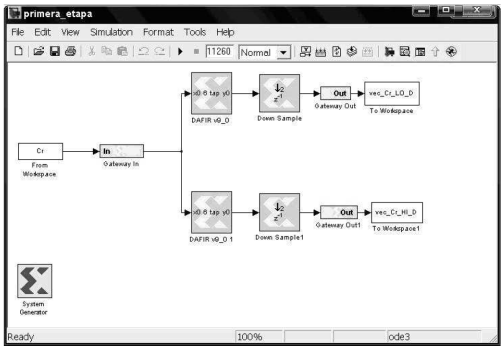

Figura 6: Primera Descomposición Simple

Se toma como entrada la componente de Crominancia roja (Cr) de una imagen de 100 x 100 píxeles.

Luego de la decimación de la primera descomposición simple, los filtros arrojan N+2 coeficientes con valor numérico, donde N es el número que se espera luego de una decimación. Estos dos valores podrían ser interpretados como extras, producto de:

$$N+2=\left.\frac{Datos\ de\ Entrada}{2}+\frac{Orden\ del\ filtro}{2}\right|_{Truncación\ parte\ entera}$$

Ejemplo:

Cantidad de datos a la entrada= 10000

Orden del filtro = 5

$$\frac{10000}{2}+\frac{5}{2}\Big|_{Truncación\ parte\ entera}=5002$$

Cantidad de coeficientes a la salida = 5002.

Propuesta de implementación

El presente trabajo propone implementar rápidamente en hardware un prototipo funcional del sistema que permita hacer una evaluación conceptual y funcional del diseño, sin atacar aún el problema de optimización del mismo.

Para realizar los distintos barridos necesarios de la matriz de datos, se utiliza programación directa en MatLab, y se aplica la señal a SimuLink directamente en forma de vector, lo cual hace transparente el recorrido de la matriz para este último.

MatLab esta diseñado para trabajar con matrices, por lo que las operaciones con este tipo de arreglo de datos son extremadamente simples de realizar, la mayoría de ellas se reducen a operadores, como la que devuelve un vector, a partir de recorrer la matriz por columnas. Para obtener el barrido horizontal y vertical se utiliza la misma función pero aplicada a la matriz original o a su transpuesta. Para poder hacer uso de este método es necesario que la matriz sea cuadrada y además debe respetar la apariencia de la imagen original.

Como método de prueba se trabajó sobre la siguiente proposición: Si se trabaja con una matriz de datos que tiene 10000 elementos, por ejemplo una imagen de 100 x 100 pixeles, a la salida de la primera descomposición se obtienen 5002 datos, lo cual no es compatible con una matriz rectangular. Esto es por una característica propia del filtro FIR y depende directamente del orden del mismo. A los efectos de lograr una matriz rectangular se ensayaron las siguientes soluciones:

a) Se agregaron 98 ceros para hacer compatible el número de datos con una matriz rectangular, formato de imagen 51x100 pixeles, necesaria para la siguiente etapa. Esto dio como resultado una alteración de la imagen reconstruida ya que los ceros quedaban embebidos en el análisis.

b) Se optó por truncar el número de datos, considerando válidos 5000 datos. Durante los ensayos se determinó que no se puede descartar cualquier dato, ya que esto repercute en los resultados de la posterior reconstrucción.

Para cada descomposición simple, se optó por tomar como válidos los N primeros datos, eliminando M datos de la descomposición, el valor de M se obtiene truncando la parte entera de la siguiente relación:

$$M=\frac{Orden\,del\ filtro}{2}\Big|_{Truncación\ parte\ entera}$$

La cantidad N de datos útiles se calcula con la siguiente fórmula:

$$N=\frac{Datos\,de\,Entrada}{2}$$

De esta forma se obtienen los datos para formar la matriz rectangular necesaria para los siguientes pasos.
Este método se utilizó tanto en la etapa de descomposición como en la de reconstrucción.

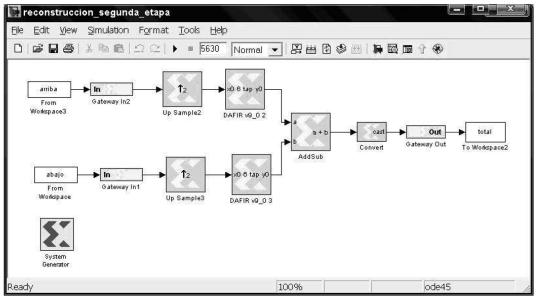

Figura 7: Reconstrucción Simple

Se debe tener en cuenta que para una reconstrucción correcta, utilizando el presente método, se debe aplicar el recorte de datos de manera invertida. Es decir si en la etapa de descomposición se utilizaron los primeros N datos, en la etapa de reconstrucción se deben utilizar los últimos N' datos.

$$N' = (Datos\ de\ entrada) \cdot 2$$

Dejando de tener en cuenta una cantidad M de coeficientes de salida.

Distorsión en fronteras

La teoría de banco de filtros utilizada para la implementación de la transformada Wavelet, está planteada para señales infinitas. Al trabajar con imágenes, señales finitas, se producen distorsiones en los límites de las señales [5].

Se han propuesto varios métodos para solucionar este problema. Todos ellos proponen extender la señal de alguna manera. La bibliografía consultada propone entre otros el método de convolución circular y la reflexión simétrica, que se obtienen mediante la reflexión y la repetición simétrica de las muestras en la frontera de la imagen. MatLab también plantea la posibilidad de relleno con ceros.

Estas ampliaciones no son arbitrarias y dependen exclusivamente del orden del filtro. Implementando a la extensión, la reconstrucción continúa generando distorsión en los bordes, sin embargo es bastante fácil apreciar la reflexión simétrica a la salida de los filtros. Eliminando dicha distorsión, se obtiene la salida recuperada perfecta, que puede ser verificada con MatLab a través de:

```
[Ap De]= dwt (entrada, 'db3');
```

Donde "entrada" es un vector de valores finitos, "db3" corresponde al tipo de onda utilizado para el cálculo de coeficientes y las salidas son "Ap" y "De" corresponden a la Aproximación y Detalle respectivamente.

Resultados

Al comparar la imagen reconstruida con la imagen original utilizando el método implementado, se hallaron errores en el margen superior izquierdo de la imagen, de manera más específica en una submatriz de n x n donde n es la cantidad de coeficientes del filtro implementado para Daubechies 3.

Como solución a este problema se agregaron marcos a la imagen original. En esta experiencia para prototipado rápido se utilizó un marco cuyo valor numérico era el uno, esto permitió apreciar el comienzo de la imagen al finalizar el marco, y pese a que no se empleó ninguna extensión de frontera se logró una reconstrucción perfecta. Esto se debe a que los errores de los procesos de truncación de información para la formación de matrices auxiliares de recorrido, descriptos anteriormente, se sitúan dentro del perímetro del marco, Fig. 8, que posteriormente es eliminado.

Figura 8: Márgenes agregados

Incorporación del módulo Wavelet a un sistema embebido en FPGA con XPS

La etapa final del desarrollo es la incorporación del módulo desarrollado en un sistema real.

La plataforma de trabajo con la que se cuenta está basada en una FPGA VirtexII-Pro de XILINX, la cual tiene incorporado en silicio dos procesadores PowerPC.

La misma empresa propone como herramienta de alto nivel para desarrollo de sistemas embebidos el EDK (Embedded Development Kit). Esta plataforma de software se compone de:

- XPS (Xilinx Plataform Studio) para el desarrollo del hardware.
- SDK (Software Development Kit) para el desarrollo de software.

Con el XPS (Xilinx Plataform Studio) utilizando el EDK (Embedded Development Kit). Una vez generada la descripción del Hardware, se utiliza el SDK (Software Development Kit) para generar y depurar el software.

Del trabajo realizado con System Generator resulta un módulo que tendrá dos registros de entrada/salida de 8 bits, una entrada para el reloj del sistema y una entrada de chip enable. Para portar el mismo se ejecuta el asistente "Create or import Peripheral" del entorno XPS [6].

Los pasos para implementar un sistema embebido utilizando el entorno XPS son:

a) Generar gráficamente la plataforma base, Micro, bus, controladores de memorias, periféricos de IO genéricos, a través de un asistente.

b) Generar un nuevo IP para poder incorporar la entidad principal, en este caso el de los bloques Wavelet. Esto se debe contemplar dentro de las funciones que se seleccionan durante el asistente para generación de la interfaz funcional para propiedad intelectual (IPIF). La existencia de los dos registros accesibles por software serán los mismos que necesita el módulo generado por System Generator.

c) Instanciar las fuentes de los archivos generados por System Generator en los archivos "user:logic.hdl" y "top_entidad.hdl.".

Una vez verificada la incorporación, mediante la síntesis de las fuentes, se agrega el IP desde el repositorio en el entorno EDK, se conecta al bus correspondiente y se generan las posiciones de memoria del sistema.

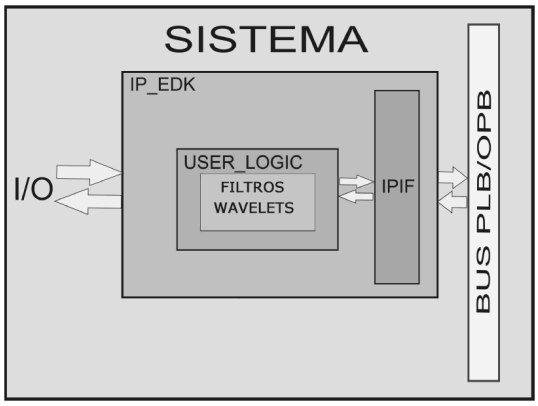

Figura 9: Esquema del módulo Encapsulado en el IPIF

Una vez incorporado el filtro al sistema, se crea una rutina en SDK, la cual escribirá en el registro de entrada del filtro datos enviados por el puerto serie de una PC conectada al sistema. Los datos procesados se leen del registro de salida y se envían a una PC por el puerto serie, los mismos son almacenados y posteriormente contrastados con los resultados de las simulaciones realizadas en SimuLink.

Conclusión y futuros trabajos

Las pruebas realizadas con la metodología utilizada demostró ser efectiva para la implementación de módulos IP, la interacción entre las herramientas de alto nivel demostró ser robusta y confiable. La posibilidad de utilizar la potencia de MatLab en el desarrollo de algoritmos y verificaciones abre la puerta a la implementación de hipótesis de manera rápida para validación o refutación, acortando los tiempos de análisis, diseño y desarrollo. La evolución de las herramientas de desarrollo de sistemas embebidos en plataformas FPGA muestra un crecimiento importante y sostenido de este campo en este tipo de tecnologías.

En Centro Universitario de Automación y Robótica (CUDAR), actualmente se está trabajando en proyectos relacionados con la medición óptica de pelos de camélidos. Donde se ha utilizado la transformada Wavelet para separar los detalles de las

imágenes y de esta forma mejorar las mediciones.

Referencias

[1] Jalali, Payman. “Wavelets and aplication.” Energy Tecnology Department. Lappeenranta University of Tecnology.

[2] Kobayashi, Mei. “Wavelets Analisys: Application in Industry” Tokyo Research Laboratory.

[3] Burrus, Sidney; Gopinath, Ramesh; Guo Haitao. “Introduction to Wavelets and Wavelets Transforms”. Electrical and computer engenieering departament. Rice University. Houston, Texas.

[5] Borja José García Menéndez; Eva Mancilla Ambrona; Ruth Montes Fraile. “Optimizacion de la transformada Wavelet Discreta” Universidad complutense de Madrid – Facultad de informática 2004 – 2005.

[6] Strang ,Gilbert; Nguyen, Truong. “Wavelets and Filter Banks”.

[7] Perez, Alejandro; Gutierrez, Francisco, Rodolfo Cavallero, Nicolas Ravotti “Desarrollo de sistemas embebidos en FPGAs. Diseño e incorporación de periféricos”.