Implementación de la FFT mediante FPGA

César Pedraza, José Paternina.

Universidad Santo Tomás, Bogotá - Colombia.

cesarpedraza@correo.usta.edu.co, josepaternina@correo.usta.edu.co

ABSTRACT

This document pretends illustrate one way to implement the Fast Fourier Transform using the Cooley Tukey algorithm. The proposal is based in a FPGA architecture to get more performance than a PC with a high level programmation language. It shows the different parts of hardware necessary for the calculus of the FFT of 64 and 128 points, with the possibility for 256 and 512 points.

RESUMEN

Este documento pretende ilustrar una forma de implementar la transformada rápida de Fourier usando el algoritmo de Cooley Tukey. La propuesta está basada en una arquitectura FPGA para obtener más rendimiento que un PC con un lenguaje de alto nivel. Este muestra las diferentes partes de hardware necesarias para el cálculo de la FFT de 64 y 128 puntos, con la posibilidad de ampliarla a 256 y 512 puntos.

1. INTRODUCCIÓN

ESTE documento describe una propuesta para la implementación de la transformada rápida de Fourier en hardware mediante dispositivos lógicos programables. Para realizar dicho sistema es necesario comprender algunas de las especificaciones de los PLDs, así como la comprensión del algoritmo de mariposa de diezmado en tiempo base dos para el cálculo de la fft. Actualmente los FPGAs son los dispositivos lógicos programables con más amplio uso en la ingeniería. La compañía Xilinx es una de las más conocidas en el diseño de dispositivos lógicos programables de alta densidad, razón por la cual se implementó el sistema para un FPGA de la familia Spartan II-E.

2. MARCO TEÓRICO

Algoritmo para la FFT de Cooley-Tukey.

La transformada discreta de Fourier es una de las técnicas más populares para convertir señales del dominio del tiempo al dominio de la frecuencia. La DFT (transformada discreta de Fourier) puede ser representada mediante la ecuación 1.

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} \qquad (1)$$

En donde x(k) son las muestras de la señal en el dominio del tiempo, X(n) son las muestras obtenidas en el dominio frecuencial, N es el número de puntos y W_N son los factores de giro.

La transformada rápida de Fourier (Fast Fourier Transform) es un algoritmo para el cálculo de la DFT desarrollado por Tukey y Cooley en 1965 el cual reduce el número de sumas y multiplicaciones respecto al algoritmo original.

Existen básicamente dos tipos de algoritmos para la FFT, el de diezmado en tiempo y diezmado en frecuencia.

Básicamente el algoritmo FFT toma el de la DFT y lo separa en dos partes, uno con índices pares y otro con impares. Ec 2. [2]

$$X(k) = \sum_{n=0}^{(N/2)-1} x_1(m) W_{N/2}^{mk} + W_N^k \sum_{n=0}^{(N/2)-1} x_2(m) W_{N/2}^{km} \qquad (4)$$

Como resultado el cálculo de la transformada termina reducido al cálculo varias de dos puntos como se observa en la figura 1.

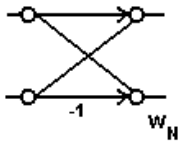

Figura 1. Cálculo de una mariposa base 2.

3. DISEÑO DEL PROCESADOR FFT

El procesador de la FFT se ha dividido en bloques de

hardware como se observa en la figura, cada uno de los cuales son descritos a continuación. La figura 2 muestra un diagrama simplificado del procesador al cual ingresan las muestras a una memoria interna del PLD.

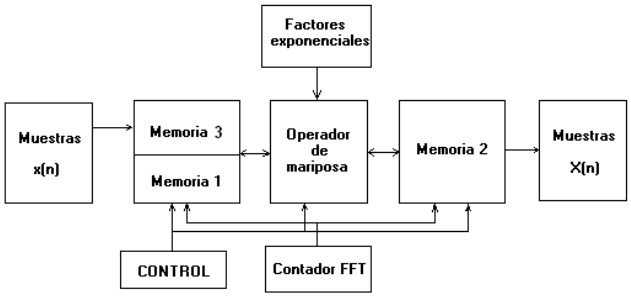

Fig 2. Diagrama de bloques del procesador FFT.

3.1. Representación de punto fijo.

El procesador FFT requiere la manipulación de números reales desde el momento en que se opera una muestra con un factor de giro, debido a que este último consiste en un número complejo cuyas partes son menores o iguales a uno. Para representar los factores de giro se usan 8 bits para la parte real y ocho para la parte imaginaria. Figura 3.

Bit	7	6	5	4	3	2	1	0
Peso	Signo	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}

Figura 3. Representación en punto fijo de los factores de giro.

3.2. Operador mariposa.

Cada una de las mariposas de la transformada requiere de una multiplicación compleja, por lo que es necesario determinar las partes de un dispositivo que realice esta operación. La figura 4 muestra que se debe realizar cuatro multiplicaciones binarias y dos sumas.

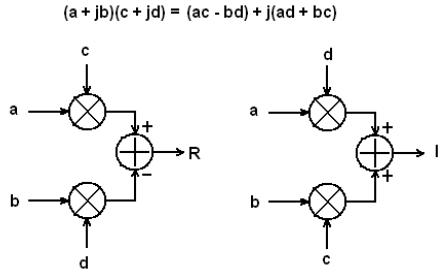

Figura 4. Partes del multiplicador complejo.

Este dispositivo es uno de los que requiere más recursos de hardware para ser implementado, debido a que exige 256 operaciones lógicas and y 16 sumas aritméticas de 16 bits por cada multiplicador, por lo tanto es conveniente que el sistema comparta uno solo que deberá ser usado para calcular todas las mariposas de la FFT [1].

El operador mariposa consiste en un bloque de hardware encargado de realizar un cálculo de una de estas, por lo que requiere de un multiplicador complejo y dos sumadores. La figura 5 muestra los bloques requeridos para realizar este proceso.

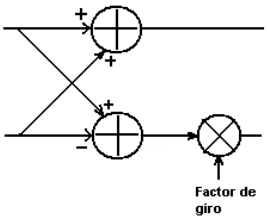

Fig 5. Operador mariposa.

3.3. Bloques de memoria.

Para almacenar los datos procesados de forma temporal se usan los bloques de memoria dispuestos en el FPGA. Dichos bloques pueden ser configurados de distintas formas, variando la longitud de la palabra y su cantidad de posiciones. Esta memoria es síncrona y de puerto dual, es decir, es posible leer datos y escribirlos en un mismo pulso de reloj. La configuración S16_S16 tiene la capacidad de manejar dos bloques independientes cada uno con una palabra de 16 bits y 256 posiciones [6].

Para efectos del procesador FFT se ha determinado usar dos bloques S16_S16 identificados como memoria 1 y 2, con el fin de guardar la parte real y compleja resultantes de cada etapa de procesamiento. Una tercera memoria de ocho bits de palabra es la que se encargará de almacenar las muestras que ingresan al procesador y que proceden de un conversor analógico digital.

3.4. Sistema de control.

Habiendo determinado la forma en que se implementa los cálculos para una mariposa, se procede a diseñar el sistema que realizará los cálculos completos de la FFT.

La figura 6 muestra el diagrama de mariposa para la transformada de interés, donde se identifica claramente la necesidad de realizar seis etapas de cálculos para completar el algoritmo. La primera toma las muestras con direccionamiento de bit reverso y luego de procesarlas almacena el resultado en la memoria 2. La segunda toma los resultados de la primera etapa y calcula las mariposas una a una guardándolas en una primera memoria. Las siguientes etapas realizan la misma operación leyendo los resultados de la etapa anterior, pero teniendo en cuenta

que los factores de giro son distintos.

El proceso anterior es realizado mediante un sistema de control basado en una máquina de estados, que direcciona las memorias, genera los índices de escritura y lectura de las mismas, especifica el factor de giro a usar en determinado instante de cálculo y controla la salida de las muestras de la FFT.

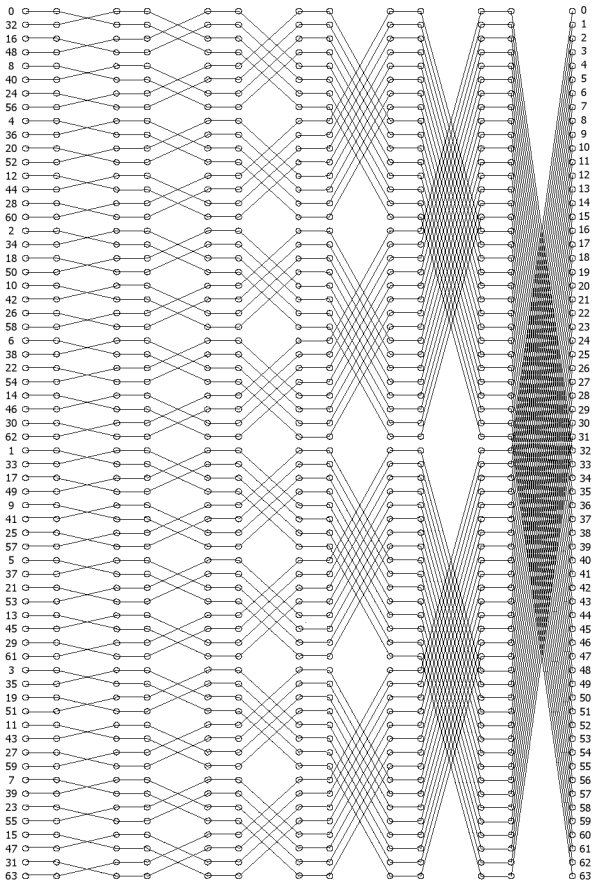

Fig 6. Diagrama de mariposa para una FFT de base 2 de 64 puntos con diezmado en tiempo.

Este sistema de control tiene la característica de leer información de una memoria para procesarla, al tiempo que guarda los datos de un proceso anterior.

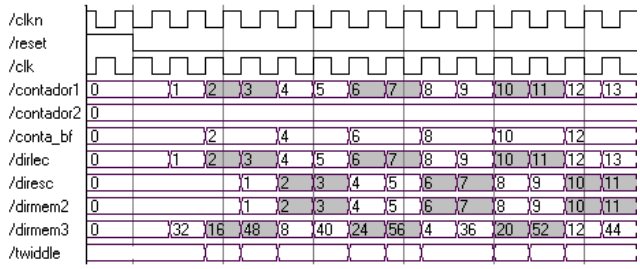

Fig 7. Simulación de parte del sistema de control.

La figura 7 muestra algunas de las señales del sistema de control durante los primeros ciclos de reloj, durante los cuales se inician los cálculos de la etapa 1, y se aprecia que las direcciones de las memorias 2 y 3 se encuentran desfasadas dos ciclos de reloj, debido a que se está leyendo la memoria 3 para tomar las muestras, y luego de procesarlas mediante el operador de mariposa se almacena en la memoria 2. El bus de dirección de la memoria 3 se encuentra con los bits en reverso, razón por la cual se observan las direcciones con los valores observados en la figura 7. La señal contador2 define la etapa de la transformada que se está realizando y la señal clkn corresponde a la señal de escritura de la memoria 2.

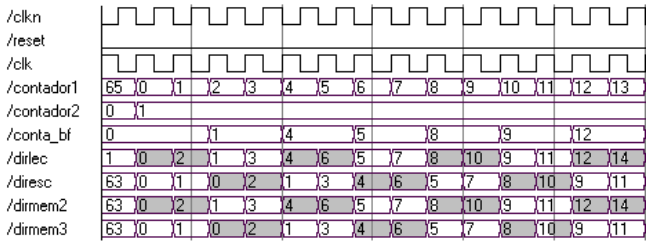

Fig 8. Simulación de parte del sistema de control.

Una vez finalizada la primera etapa, la máquina de estados inicia el proceso de cálculo de la segunda. La figura 8 muestra dicho cambio, en donde la señal contador2 indica que se inicia la segunda fase de la FFT, además que el direccionamiento se realiza de dos en dos como se observa en la figura 6. De igual forma las señales de control de escritura y de lectura se encuentran desfasadas dos ciclos de reloj para el pipelining.

3.5. Sistema de adquisición.

Luego de procesar la información es necesario extraerla del FPGA con el objeto de mostrarla. Para tal efecto se ha determinado leer la memoria donde quedaron almacenadas la parte real e imaginaria de la FFT representadas en 16 bits cada una, mediante el puerto paralelo de un PC. Finalmente en este se calcula la magnitud del vector complejo de cada muestra que corresponde a la magnitud de la componente frecuencial de la señal que ingresó en el dominio del tiempo.

4. RESULTADOS

Luego de realizar simulaciones y pruebas de laboratorio resulta prometedor el desempeño del procesador FFT que se ha diseñado. Inicialmente se cuenta con un oscilador de 50MHz lo que lleva al sistema a realizar un cálculo

completo de la FFT de 64 puntos en 7.92µs, para un total de 126262 transformadas por segundo.

Adicionalmente se encuentra abierta la posibilidad de aumentar la frecuencia de la señal de reloj y adicionar un nuevo pipelining en el cual se puedan calcular etapas de la FFT en paralelo, lo que requeriría más memoria y recursos del FPGA que actualmente se encuentran disponibles.

Se realizaron pruebas en matlab que confirman la efectividad del algoritmo y los procesos usados en el FPGA. Dichas pruebas se usaron con el fin de determinar la resolución del análisis espectral obtenido. La figura 9 muestra los resultados obtenidos al analizar una señal compuesta de dos señales puras de 100 y 200 Hz con ruido. Se observa también un error en el cálculo de la FFT a causa de la precisión usada en el sistema.

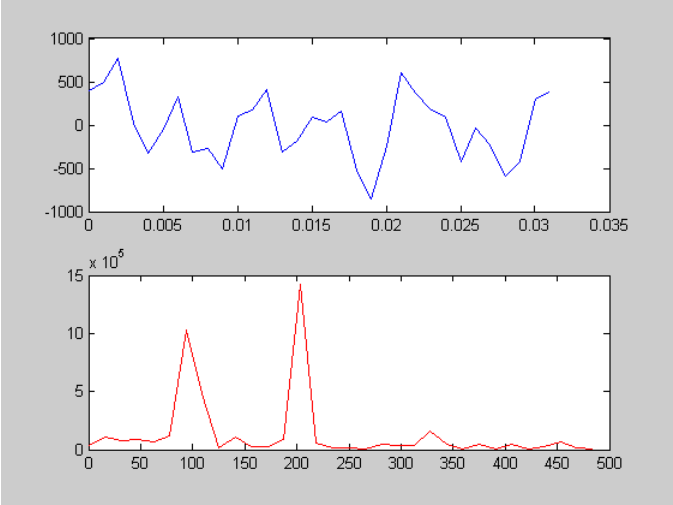

Fig 9. Simulación del algoritmo implementado para la FFT de 64 puntos.

5. CONCLUSIONES

- En el procesamiento digital de señales, es indudable que el proceso de multiplicación es el que más recursos abarca, desde el punto de vista de tiempo como de hardware, por lo tanto, lograr diseñar un hardware eficiente que multiplique, mejora en gran medida los resultados y más aún cuando se trata de procesamiento en tiempo real.
- Es evidente que el procesamiento de señales necesita de hardware muy especializado (DSP), un hardware genérico como lo es un FPGA puede reemplazar con creces uno de estos dispositivos, y en la mayoría de casos a un costo más módico, sin sacrificar velocidad de procesamiento ni complejidad de los problemas a ser resueltos.
- En el diseño de sistemas con FPGA el tiempo de desarrollo y depuración son más extensos que los demás sistemas, dado que estos dispositivos son elementos de hardware genéricos.
- Como gran ventaja del desarrollo de procesadores digitales de señales en FPGA encontramos la posibilidad de implementar sistemas concurrentes aumentando el horizonte de velocidad de procesamiento en tiempo real.
- Uno de los puntos críticos en el diseño de procesadores digitales de señales son las memorias. Procesar grandes cantidades de información requiere de capacidades de memorias algo considerables. Actualmente se cuenta con algoritmos de aritmética distribuida que minimizan el uso de los recursos combinacionales, pero requieren así mismo de cantidades de memoria superiores. Es importante avanzar en la comprensión e investigación de algoritmos nuevos que permitan optimizar más algunos de los procesos requeridos en el procesamiento digital de señales.

6. REFERENCIAS

[1] A. Bekerman, V. Owall, and M. Torkelson, "A Low Depth Complex Multiplier Using Distributed Arithmetic*"IEEE Journal of solid state circuits*, vol. 35, pp. 656-659, Apr. 2000.J. Proakis, D Manolakis, Tratamiento digital de señales principios, algoritmos y aplicaciones. 2da ed., Prentice may., pp. 457-507.

[2] Kamal B. S. "A hardware efficient architecture for Fast Fourier Transform", University of Maryland Baltimore Country. Sin publicar.

[3] Shousheng He, M. Torkelson, "A new approach to pipeline FFT processor", in proc. 10th International Parallel Processing Symposium, IPPS April 1996 .

[4] B.M. Baas, "Low-power high-performance 1024-point FFT processor", IEEE Journal of Solid-State Circuits, Vol. 34, No. 3, March 1999.

[5] Xilinx programmable databook. DS077-2 . November 15 2001.

[6] Shousheng He, M. Torkelson, "Designing pipeline FFT processor for FDM (de)modulation", in Proc. International Symposium on Signals, systems and Electronics, ISSSE Oct 1998.

[7] Spartan-IIE 1.8V FPGA Family: Functional Description. DS077-2 (v1.0) November 15, 2001