Implementação do Algoritmo de Cooley Tukey para o Cálculo da Transformada Rápida de Fourier em hardware

Felipe Moraes Henes[1], **Robert Torrel**[1], **Vitor Righi**[1], **Rubén Edgardo Panta Pazos**[1]

[1]Curso de Engenharia de Computação – Universidade de Santa Cruz do Sul (UNISC)
Av. Independência – 96.815-900 – Santa Cruz do Sul – RS – Brasil

{fhenes,roberttorrel,vitorrighi,rpazos}@unisc.br

Abstract. *This paper describes the main methodology for made an ASIC Design Flow, in a Fast Fourier Transform (FFT) calculation module, and the main results in this process. Also talk a bit about the Cooley Tukey algorithm, responsible for solving the problem*

Resumo. *Este artigo descreve a principal metodologia adotada para a realização do fluxo de projeto de um ASIC, aplicado em um módulo responsável pelo cálculo da Transformada Rápida de Fourier (FFT), bem como os principais resultados obtidos neste processo. Além disso fala-se um pouco sobre o algoritmo de Cooley Tukey, este responsável pela resolução do problema.*

1. Introdução

A constante evolução tecnológica, principalmente na área de processamento digital de sinais, leva cada dia a uma maior necessidade de compressão de sinal, visto que o volume de dados, usados por aplicações deste tipo, vem crescendo exponencialmente, hora necessário para garantir a qualidade, como no caso de vídeos, imagens, fotos, hora para garantir uma transmissão em tempo hábil de uma grande gama de dados.

Outro ponto importante de ser analisado, é o fato de que a cada dia possuímos mais aparelhos eletrônicos que se comunicam sem a necessidade de estabelecer uma conexão com fios, porém estes dados são transmitidos em um meio físico muito suscetível a erros, ruídos e que na maioria das vezes não possibilita, sem a aplicação de técnicas de otimização, o tráfego de grandes quantidades de dados.

O projeto proposto visa implementar um ASIC para tratamento digital de sinais elétricos, para isto aplica a Transformada Discreta de Fourier sobre estes sinais, por meio de um algoritmo de FFT (Fast Fourier Transform), que garante o cálculo da transformada discreta de fourier (DFT), em um bom tempo de execução, o que não acontece se for realizar os cálculos da maneira tradicional.

2. Fast Fourier Transform (FFT)

A grande demanda de tempo computacional, e o elevado número de operações (multiplicações e somas) complexas necessários para se realizar os cálculos da Transformada Discreta de Fourier, pelo modo clássico de resolução, levou a neccessidade da criação de um modo mais eficiente para se realizar estes cálculos.

A (FFT) Transformada Rápida de Fourier é um método simples de calcular a (DFT) Transformada Discreta de Fourier que é usada para converter um sinal de entrada

no dominio discreto de tempo, *x(n)*, para o dominio discreto de frequencia *X(r)* e vice versa [Pilz et al. 2005].

A Transformada Rápida de Fourier, ou FFT foi implementada com o objetivo de diminuir a complexidade (temporal) necessária para calcular uma DFT (Transformada Discreta de Fourier), visando aplicações em tempo real.

É um algoritmo, ou seja, uma sequência de operações matemáticas, que retornam o resultado da transformada discreta de Fourier, em um tempo de execução satisfatório.

Existem vários algoritmos, desde os mais complexos, utilizando princípios de física quântica, até os mais simples, que em alguns casos inserem erros indesejáveis nas operações.

Segundo [Banerjee et al. 2001], a forma tradicional de se resolver uma DFT realiza aproximadamente (N2) operações complexas (multiplicações e adições) para chegar a resposta do problema, enquanto que os algoritmos de FFT em geral, levam aproximadamente (N*log2*N).

Sendo assim, a vantagem cresce exponencialmente com o numero de iterações, como pode se observar na figura abaixo.

N	N^2 (DFT)	$N.\log_2 N$ (FFT)	Vantagem
2	4	2	2
4	16	8	2
8	64	24	2,67
16	256	64	4
32	1024	160	6,4
64	4096	384	10,67
128	16384	896	18,29
256	65536	2048	32
512	262144	4068	56,89
1024	1048576	10240	102,4
2048	4194304	22528	186,18
4096	16777216	49512	341,33
8192	671088964	106496	630,15

Figure 1. Comparativo Entre os modos de resolução do problema

3. O Algoritmo de Cooley Tukey

O algoritmo de Cooley Tukey é um dos mais antigos algoritmos para o cálculo da FFT, ele está baseado no princípio de dividir para conquistar, fazendo uso de somas e multiplicações, utiliza as raízes das unidades para realizar o cálculo, e estas contém números complexos.

A seguir um breve de como o algoritmo de Cooley Tukey procede para realizar os cálculos:

1. Rearranjo e leitura de bits
   Considere-se o sinal S = [f (0), f (1), f (2), f (3)] , de quatro elementos. Conforme o rearranjo e leitura de bits, têm-se:
2. FFT para um sinal de 4 elementos

Numeração Original		Arranjo original	Numeração bits invertidos		Arranjo reordenado
0	**0**	*f*(0)	**0**	**0**	*f*(0)
0	**1**	*f*(1)	**1**	**0**	*f*(2)
1	**0**	*f*(2)	**0**	**1**	*f*(1)
1	**1**	*f*(3)	**1**	**1**	*f*(3)

Figure 2. Rearranjo inicial dos dados

(a) Calcula-se as 4 raízes da unidade, W4 = [1, -i, -1, i].
(b) Efetua-se uma reordenação conforme o quadro acima.
(c) São calculadas as duas DFT para S1 = [f (0) f (2)] e S 2 = [f (1) f (3)] , denotadas com T1 e T2, respectivamente.

4. Arquitetura Proposta

A Arquitetura proposta, ilustrada na figura 3, consiste em três blocos principais. Um bloco referente ao carregamento dos dados de entrada, outro bloco responsável por realizar as operações e um último que realiza o descarregamento dos dados do circuito.

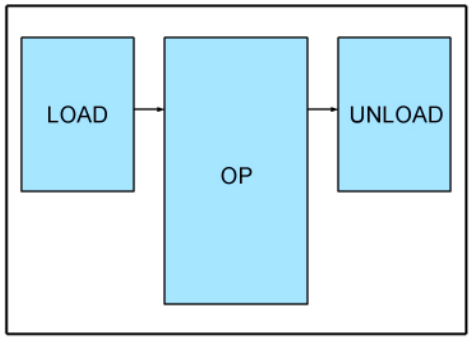

Figure 3. Arquitetura Básica do Sistema

O bloco principal (bloco OP), é o bloco responsável pela parte operativa do circuito, nele, além de blocos de controle, está a ULA (Unidade Lógica Aritimética).

Esta unidade possui dois núcleos de processamento de dados, o que possibilita a realização de duas computações por ciclo de relógio, aumentando assim o *throughput* do sistema, visto que esta é a parte mais complexa do circuito, e por isto consome maior tempo. A figura 4 mostra os blocos principais da parte operativa do circuito (OP).

5. Resultados obtidos

Os Princpais resultados obtidos dizem respeito a simulação do circuito pós síntese lógica com parâmetros de timming observados.

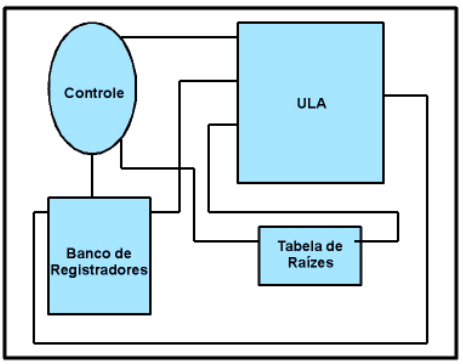

Figure 4. Arquitetura do Circuito Operativo

O circuito opera sobre dois clocks, um clock principal de 20MHz e outro para as memórias de 40MHz. Na realização da síntese lógica o clock principal passou com uma pequena folga positiva de 685 ps enquanto o clock secundário teve uma folga de 690 ps.

Além disso, a área do die ficou em torno de 3 x 3 mm aproximadamente, a tecnologia utilizada foi de 0,35 um.

Ainda em simulação, o circuito apresenta uma vazão de aproximadamente 1000 operações por segundo. Por operações entende-se o ciclo completo do circuito incluindo carregamento, processamento e descarregamento dos dados.

6. Conclusões

Por final conclui-se que o algoritmo de Cooley-Tukey, mesmo sendo um método clássico, é um ótimo meio para o cálculo da Transformada Rápida de Fourier, pois os resultados das simulações foram satisfatórios, e o circuito se mostrou eficiente e rápido, aliado a isto a arquitetura definida, apresentou uma boa vazão, como citado acima.

References

Banerjee, A., Sundar Dhar, A., and Banerjee, S. (2001). FPGA realization of a CORDIC based FFT processor for biomedical signal processing. *Microprocessors and Microsystems*, 25(3):131–142.

Pilz, N., Lerner, B., and Adamson, K. (2005). Parallel FFT implementations on fixed-point DSP-cores with subword-parallelism. In *IEE conference publication*, volume 511, page 338. Citeseer.