Implementação de uma FFT com complexidade multiplicativa abaixo do limitante de Heideman-Burrus

P.A.L. Sá de Melo, H.M. de Oliveira

***Abstract*— The computational implementation of a new fast algorithm for computing DFT based on a matrix Laurent series [1] is presented via Simulink™. The arithmetic complexity, expressed by the number of nontrivial real floating-point multiplications, achieves values below the standard Heideman-Burrus bound [2]. The case *N*=16 is presented in details, requiring merely 12 real multiplications and 101 additions.**

***Resumo*— A implementação computacional em plataforma Simulink® de uma nova transformada rápida de Fourier com base em Séries de Laurent matriciais [1] é apresentada. A complexidade aritmética, expressa em multiplicações reais não-triviais, atinge valores inferiores àqueles estabelecidos na cota padrão de Heideman-Burrus [2]. O exemplo *N*=16 é apresentado detalhadamente, com apenas 12 multiplicações reais e 101 adições.**

***Palavras Chave*— algoritmos rápidos, Fourier, FFT, séries de Laurent, cota de Heideman-Burrus.**

I. INTRODUÇÃO

A transformada de Fourier, contínua ou discreta, desempenha um papel muito importante em diversas áreas da Engenharia, principalmente em processamento de sinais.

Este artigo apresenta uma implementação em Simulink®, ferramenta de programação visual do *software* MATLAB®, de um novo algoritmo rápido para o cálculo da Transformada Discreta de Fourier (DFT) [1]. Seja N o número de amostras no domínio do tempo de um sinal $v=(v_n)$, n=0,1,2,...,N-1. A DFT do sinal v é dada pela sequência $V=(V_k)$, de comprimento N, no domínio da frequência, expressa por:

$$V_k := \sum_{n=0}^{N-1} v_n \exp\left(-\frac{j2\pi kn}{N}\right). \qquad (1)$$

Esta transformada pode ser calculada através de algoritmos rápidos, tais como o algoritmo Cooley-Tukey, e o algoritmo Winograd-Fourier [3].

Em 1986, Heideman investigou a complexidade aritmética da DFT e deduziu cotas inferiores sobre o número de multiplicações complexas necessárias para calculá-la [2]. Para potências de 2, esta cota é simplificada para :

$$\mu_{DFT}(N) = 4N - 2\log_2^2 N - 2\log_2 N - 4. \qquad (2)$$

II. A DFT COMO UMA SÉRIE DE LAURENT MATRICIAL

de Oliveira, Campello de Souza e de Oliveira propõem, em um artigo submetido neste mesmo Simpósio [1], um algoritmo para cálculo da DFT com base na decomposição da matriz da DFT em série matricial de Laurent [4].

Define-se inicialmente uma matriz M:=($k.n$ (mod N)). Define-se também um operador χ_l sobre uma matriz $N\times N$ para l=0,1,2,…,N-1, que resulta numa matriz cujos elementos são $\left(\delta_{l,m_{k,n}}\right)$, em que δ denota o símbolo de Kronecker. Desta forma, são construídas classes C_m e as respectivas matrizes M_m expressas por:

$$C_m := \left\{x \in Z_N \,|\, 4x \equiv 4m \,(\mathrm{mod}\, N)\right\};$$

$$M_m = 1.\chi_m(M) - j.\chi_{m+N/4}(M) - 1.\chi_{m+N/2}(M) + j.\chi_{m+3N/4}(M),$$

$$m = -\left\lfloor \left(\frac{N}{4}-1\right)/2 \right\rfloor, \ldots, -2, -1, 0, +1, +2, \ldots, +\left\lceil \left(\frac{N}{4}-1\right)/2 \right\rceil$$

A matriz da transformada de Fourier pode assim ser avaliada como [1]:

$$\Re eDFT = \left\{ \Re e(M_0) + \sum_{m=1}^{(N/4-1)/2} \Re e(M_m + M_{-m}).\cos\frac{2\pi mn}{N} \right\} + \left\{ \sum_{k=1}^{(N/4-1)/2} \Im m(M_m - M_{-m}).\sin\frac{2\pi mn}{N} \right\}, \qquad (3a)$$

$$\Im mDFT = \left\{ \Im m(M_0) + \sum_{m=1}^{(N/4-1)/2} \Im m(M_m + M_{-m}).\cos\frac{2\pi mn}{N} \right\} - \left\{ \sum_{k=1}^{(N/4-1)/2} \Re e(M_m - M_{-m}).\sin\frac{2\pi mn}{N} \right\}. \qquad (3b)$$

III. IMPLEMENTAÇÃO EM SIMULINK® PARA N=16

O algoritmo apresentado foi implementado na plataforma Simulink® para N=16. Seguindo os passos do algoritmo, constroem-se as seguintes classes:

$$C_0 = \{0,4,8,12\} \qquad C_1 = \{1,5,9,13\}$$

$$C_{-1} = \{15,3,7,11\} \qquad C_2 = \{2,6,10,14\}.$$

A partir destas classes, obtêm-se as seguintes matrizes escalonadas, em que *rref* indica a forma escalonada padrão, 0_n indica um bloco de n zeros, e ";" indica uma separação de linhas:

rref $[\Re e(M_0)] =$
$\{1, 0_{15}\ ;\ 0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1\ ;\ 0_2,1,0_3,1,0_3,1,0_3,1,0\ ;\ 0_4,1,0_7,1,0_3\}$

rref $[\Re e(M_1 + M_{-1})] =$
$\{0,1,0_5,-1,0,-1,0_5,1\ ;\ 0_3,1,0,-1,0_5,-1,0,1,0_2\}$

rref $[\Re e(M_2)] =$
$\{0,1,0_3,-1,0_3,1,0_3,-1,0_2\ ;\ 0_2,1,0_7,-1,0_5\ ;\ 0_3,1,0_3,-1,0_3,1,0_3,-1\ ;\ 0_6,1,0_7,-1,0\}$

rref $[\Im m(M_1 - M_{-1})] =$
$\{0,1,0_5,-1,0,-1,0_5,1\ ;\ 0_3,1,0,-1,0_5,-1,0,1,0_2\}$

rref $[\Im m(M_2)] =$
$\{0,1,0_3,-1,0_3,1,0_3,-1,0_2\ ;\ 0_2,1,0_7,-1,0_5\}$.

H.M de Oliveira e P.A.L. Sá de Melo são membros do CODEC, Universidade Federal de Pernambuco (UFPE), C.P. 7800, CEP: 50711-970, Recife-PE, Brasil. (e-mails: hmo@ufpe.br, pmelo@hotmail.com).

As componentes do vetor de entrada são combinadas como indicado pelas linhas das matrizes escalonadas e ponderadas de acordo com a equação 3 (c.f. Figura 1.a). Deve-se então encontrar, para cada linha das matrizes da série, quais combinações lineares das linhas das matrizes escalonadas resultam nessas linhas. Estas combinações estão expostas na Figura 1.b, para o cálculo da parte real da DFT.

Tabela 1. Complexidade aditiva do algoritmo FFT baseado em série matriciais de Laurent para *N*=16.

Número de adições reais ponto-flutuante	
Combinações com base nas matrizes escalonadas	42
Avaliação das componentes reais	39
Avaliação das componentes imaginárias	20
Total	101

A complexidade aditiva para o algoritmo apresentado é $\alpha(16)=101$, podendo ainda ser reduzida através do aproveitamento de simetrias adicionais que surgem. A complexidade multiplicativa é $\mu(16)=12$, valor abaixo do limitante de Heideman-Burrus para comprimento 16, que é 20 (equação 2). A complexidade obtida é inferior à calculada pela cota em oito multiplicações. Isto se justifica devido à presença de componentes do espectro nas freqüências de $\pm\pi/4$ e $\pm 3\pi/4$, cujas funções seno e cosseno são idênticas, reduzindo o número de multiplicações em quatro unidades, e também devido ao surgimento de linhas idênticas nas matrizes escalonadas no cálculo da parte real e parte imaginária da transformada. A extensão do procedimento para outros comprimentos pode ser vista no artigo companheiro [1].

IV. CONCLUSÕES

Um novo algoritmo rápido (FFT) para calcular a Transformada Discreta de Fourier (DFT) foi implementado em Simulink®, descrevendo as etapas e corroborando que é possível atingir um número de multiplicações ponto-flutuante menor do que aquele preconizado pela cota inferior de Heideman. O procedimento é sistemático e fácil de reproduzir, permitindo manipular com maior variedade de comprimentos de bloco do que o tradicional radix-2 [3]. Comenta-se sobre argumentos explorados na diminuição da complexidade, os quais são válidos para outros comprimentos de blocos. Mas o objetivo específico é validar o algoritmo [1] proposto, mostrando que são possíveis implementações de complexidade abaixo do limitante. As simetrias existentes indicam que há muito a se fazer na concepção de *hardware* dedicados. Esta nova FFT é de fácil implementação, seja em DSP ou circuitos integrados, sendo atrativo explorar implementações em FPGA, dado o paralelo entre Simulink e FPGA.

AGRADECIMENTOS

Este trabalho recebeu apoio financeiro parcial do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

REFERÊNCIAS

[1] de Oliveira, H.M., Campello de Souza, R.M., and de Oliveira R.C., A Matrix Laurent Series-based Fast Fourier Transform for Blocklengths N≡4 (mod 8), *XXVII Simpósio Brasileiro de Telecomunicações* - SBrT, Blumenau, SC, 2009 companion paper (submitted).

[2] Heideman, M.T., Burrus, C.S., On the Number of Multiplications Necessary to Compute a length-2^n DFT, *IEEE Trans. Acoust., Speech, Signal Processing*, vol.34,pp.91-95, 1986.

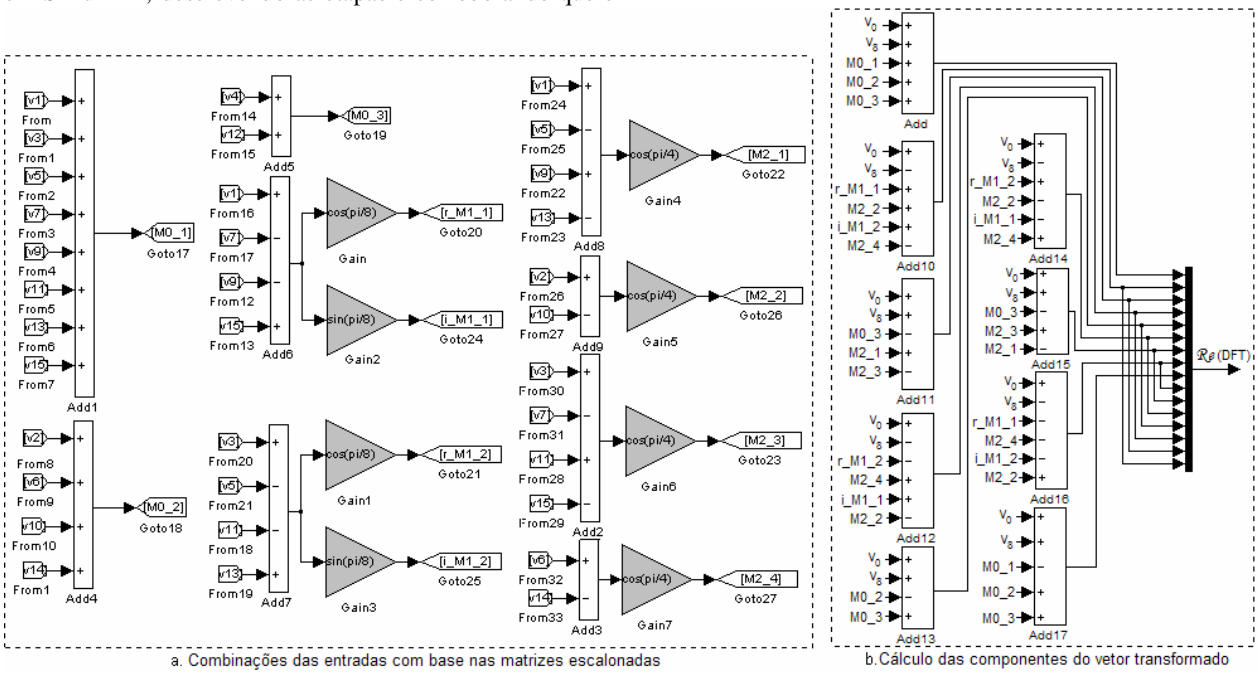
[3] Blahut, R. E., *Fast Algorithms for Digital Signal Processing*, Addison-Wesley, 1985.

[4] Adel-Ghaffar, K.A.S., Long Division from Laurent Series Matrices and the Optimal Assignment Problem, *Linear Algebra and its Application*, vol.280, No.2-3, Sept pp.189-197, 1998

APÊNDICE

O código fonte para a implementação *simulink* encontra-se disponível (*freeware*) na URL:

http://www2.ee.ufpe.br/codec/Procedure_FFT.htm



a. Combinações das entradas com base nas matrizes escalonadas

b. Cálculo das componentes do vetor transformado

Figura 1. Implementação em Simulink do cálculo da parte real da DFT (*N*=16) com uso do algoritmo FFT apresentado. Os blocos triangulares (Fig.1.b) destacam as 8 multiplicações reais, que, somadas com outras 4 do cálculo da parte imaginária, resulta em um total de 12 multiplicações reais ponto-flutuante, número inferior às 20 multiplicações da cota de Heideman-Burrus para *N*=16.