UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE ELÉTRICA
CURSO DE ENGENHARIA ELÉTRICA

CALLEBE SOARES BARBOSA

# IMPLEMENTAÇÃO DO ALGORITMO RADIX-2 PARA CÁLCULO DA FFT EM FPGA

TRABALHO DE CONCLUSÃO DE CURSO

PATO BRANCO
2017

CALLEBE SOARES BARBOSA

# IMPLEMENTAÇÃO DO ALGORITMO RADIX-2 PARA CÁLCULO DA FFT EM FPGA

Trabalho de Conclusão de Curso de graduação, apresentado à disciplina de Trabalho de Conclusão de Curso 2, do Curso de Engenharia Elétrica da Coordenação de Engenharia Elétrica - COELT - da Universidade Tecnológica Federal do Paraná - UTFPR, Câmpus Pato Branco, como requisito parcial para obtenção do título de Engenheiro Eletricista.

Orientador: Prof. Dr. Fábio Luiz Bertotti

PATO BRANCO

2017

**TERMO DE APROVAÇÃO**

O Trabalho de Conclusão de Curso intitulado **IMPLEMENTAÇÃO DO ALGORITMO RADIX-2 PARA CÁLCULO DA FFT EM FPGA** do acadêmico **Callebe Soares Barbosa** foi considerado **APROVADO** de acordo com a ata da banca examinadora **N° 123** de **2017**.

Fizeram parte da banca examinadora os professores:

**Prof. Dr. Fábio Luiz Bertotti**

**Prof. Dr. Stephen Hawking**

**Dr. Brian Greene**

**Prof. Dr. Michio Kaku**

**Prof. Richard Feynman**

Aqui vai o texto da dedicatória. Aqui vai o texto da dedicatória. Aqui vai o texto da dedicatória.

*Acreditar é mais fácil do que pensar. Daí existem muito mais crentes do que pensadores.*

Bruce Calvert

## AGRADECIMENTOS

Aqui são os agradecimentos.

## RESUMO

Escreva aqui o texto de seu resumo... UTFPRTEX

**Palavras-chave**: LATEX, FFT, Transformada Rápida de Fourier, Sistemas Digitais, FPGA, Spartan 3E.

**ABSTRACT**

Write here the English version of your Resumo...

**Keywords**: LaTeX, FFT, Fast Fourier Transform, Digital System, FPGA, Spartan 3E, Digital Signal Process.

## LISTA DE FIGURAS

## LISTA DE TABELAS

## LISTA DE SÍMBOLOS

| | |
|---|---|
| $\phi$ | |
| $\vec{\alpha}$ | alpha |
| $\upsilon\omega\psi_{n-1}^{jk}$ | Função de teste da lista de símbolos. Está é uma descrição longa para um único símbolo |
| $\nabla$ | Gradiente |
| $\upsilon\omega\psi_{n-1}^{jk}$ | teste |

**SUMÁRIO**

# 1 INTRODUÇÃO

A Transformada Discreta de Fourier ou DFT *(Discrete Fourier transform)*, segundo Bingham (1990), é um recurso amplamente utilizado em aplicações como processamento de imagens, comunicação rede de área local sem-fio ou WLAN *(Wireless Local Area Network)*, multiplexação por divisão de frequências ortogonais ou OFDM (Orthogonal Frequency Division Multiplexing), e medições de diferentes espectros.

O calculo da DFT é uma tarefa que exige muitos recursos computacionais e que requer um projeto preciso para uma implementação eficiente (WANG *et al.*, 2010). A DFT tem como base a própria série trigonométrica de Fourier discretizada. Segundo Lathi (2007, p. 719), graças a um algoritmo desenvolvido por Cooley e Tukey o número de cálculos para executar uma DFT foi drasticamente reduzido, possibilitando um tempo de execução aceitável.

Segundo Zhou *et al.* (2009) os dispositivos conhecidos como FPGA *(Field Programmable Gate Array)* são cada vez mais usados em implementação de hardware para telecomunicações, por exemplo, devido a sua capacidade de alcançar um elevado desempenho, aliado a sua flexibilidade de configuração. Desta forma, o uso de FPGA para implementar um hardware específico para o calculo da DFT torna-se relevante.

## 1.1 JUSTIFICATIVA

Como afirma He e Guo (2008), o algoritmo da FFT e utilizada em larga escala como componente chave em sistemas de processamentos de sinais. De tal forma que a FFT não é somente usada em aplicações de telecomunicações e processamento de dados audiovisuais (BINGHAM, 1990), mas e o algoritmo numérico mais comum em diversas áreas, como engenharia, medicina, física e matemática (VANMATHI *et al.*, 2014).

No campo da engenharia biomédica, a FFT tem sido empregada na avaliação de parâmetros biológicos, como a bioimpedância (AMARAL *et al.*, 2011). Segundo Martinsen e Grimnes (2011), a análise da bioimpedância já e usada para monitorar atividades bioelétricas cerebrais, diagnosticar câncer de pele, dermatite, hiperidrose, ava-

liar a composição corporal, avaliar condição nutricional, detectar processo de rejeição de órgãos transplantado e ainda monitorar recém-nascidos através tomografia de impedância elétrica ou EIT *(Electrical Impedance Tomography)*. Sendo a EIT a única técnica de obtenção de imagens que não afeta o sistema imunológico de recém-nascidos (TRIANTIS *et al.*, 2011).

Para Ibrahim *et al.* (2016), a FPGA e uma boa escolha para a implementação do algoritmo da FFT devido a grande variedade de recursos de *hardware* sintetizáveis, além de possuir recursos de programação paralela que permite o processamento paralelo de sinais, conferindo assim uma maior rapidez na execução do algoritmo. Portanto implementar a FFT, uma ferramenta tao útil em diversas áreas da ciência, em uma plataforma vantajosa como a FPGA, é o principal motivador deste trabalho.

# 2 REVISÃO DE LITERATURA

Um sinal é um conjunto de dados ou informações, que podem descrever diversos tipos de fenômenos, como um sinal de telefonia ou de voz (Figura 2), o registro de vendas de uma empresa ou ainda os valores de fechamento de uma bolsa de valores (Figura 1) (LATHI, 2007, p. 75). Segundo Oppenheim e Willsky (2010, p. 1) existe uma linguagem adequada para descrever sinais e um conjunto poderoso de ferramentas para analisá-los, capaz de se aplicar a problemas oriundos de diversos domínios. A série de Fourier e a Transformada Rápida de Fourier são ambas exemplos destas ferramentas, e serão apresentadas nos próximos capítulos.


**Figura 1: Índice BM & FBOVESPA, São Paulo - Brasil**
**Fonte: TradingView (2017)**

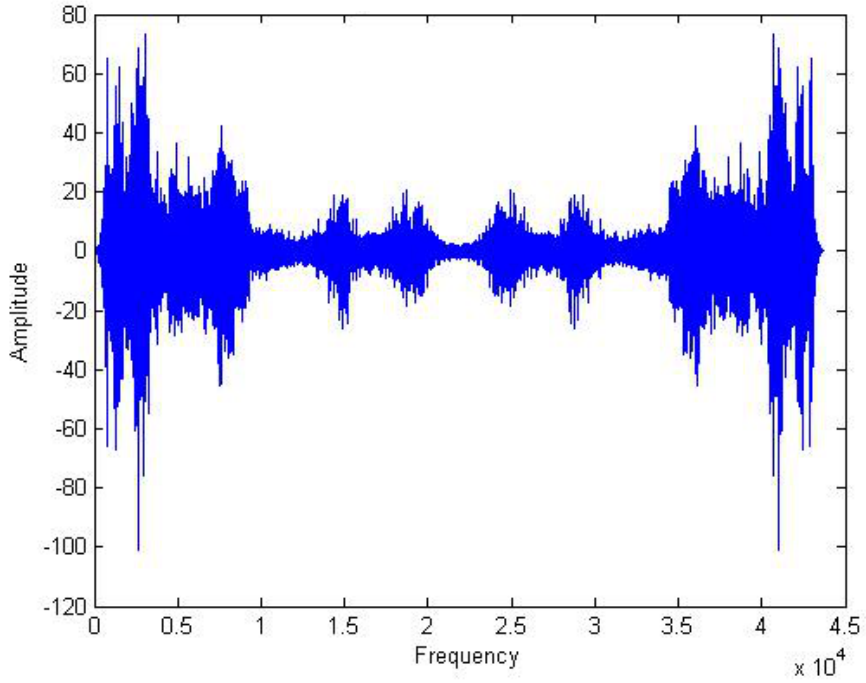


**Figura 2: Sinal Referência de Voz**
**Fonte: NirmalRaj e Vigneswaran (2015)**

## 2.1 SÉRIE DE FOURIER

Segundo Lathi (2007, p. 530) "Um sinal periódico $x(t)$ com período $T_0$ pode ser descrito como a soma de senoides de frequência $f_0$ e todas as suas harmônicas", conforme visto em (1). O qual e conhecida como série trigonométrica de Fourier de um sinal periódico $x(t)$, sendo $\omega_0$ a frequência fundamental.

$$x(t) \; = \; a_0 + sum_{n=1}^{\infty} a_n cos(n\omega_0 t) + b_n sen(n\omega_0 t) \tag{1}$$

E possível determinar os coeficientes $a_n$ e $b_n$ de (1 )pelas equações (2) e (3), em que o coeficiente $T_0$ representa o período relativo a $f_0$.

$$a_n \; = \; \frac{2}{T_0} \int_{T_0} x(t) cos(n\omega_0 t) dt \tag{2}$$

$$a_n \; = \; \frac{2}{T_0} \int_{T_0} x(t) cos(n\omega_0 t) dt \tag{3}$$

A expressão da série de Fourier em termos exponenciais $e^{j\omega_0 t}$ e $e^{?j\omega_0 t}$ é facilmente obtida a partir da forma trigonométrica (LATHI, 2007). A forma exponencial

da serie de Fourier é dada por (4), em que o coeficiente $C_n$ e análogo aos coeficientes $a_n$ e $b_n$ da serie trigonométrica, sendo obtido por (5.) Nota-se que diferente dos coeficientes $a_n$ e $b_n$, o coeficiente $C_n$ pode representar um valor complexo.

$$x(t) = \sum_{-\infty}^{\infty} C_n e^{jn\omega_0 t} \tag{4}$$

$$C_n = \frac{1}{T_0} \int_{T_0} x(t) e^{jn\omega_0 t} dt \tag{5}$$

### 2.1.1 ESPECTRO EXPONENCIAL DE FOURIER

Segundo Lathi (2007, p. 556), o espectro exponencial de Fourier e traçado a partir dos coeficientes $C_n$ e das frequências $n\omega_0$ da forma exponencial da serie de Fourier. Então e necessário expressar o espectro em função da parte real e da parte imaginaria, ou do modulo e do ângulo. A forma de $C_n$ em modulo é angulo geralmente é mais útil para se expressar o espectro. Logo são taçados dois gráficos para o espectro exponencial de Fourier, um que relaciona $|C_n|$ com $n\omega_0$, e outro que relaciona $\angle C_n$ com $n\omega_0$.

Para Lathi (2007, p. 533) os dois gráficos juntos formam o espectro de frequência, o qual revela os conteúdos de frequência do sinal $x(t)$, com suas amplitudes e fase. Conhecendo-se este espectro não só é possível analisar o sinal $x(t)$, como também reconstruí-lo de forma fácil.

## 2.2 SÉRIE DE FOURIER EM TEMPO DISCRETO

Ate aqui foi apresentada a forma continua da série de Fourier, porém para ser útil em uma aplicação computacional é necessário encontrar sua forma discreta, ou DFT *(Discrete Fourier Transform)*. Segundo HAYKIN e Veen (2001, p. 314) a DFT é a única representação de Fourier que pode ser calculada por um computador, sendo amplamente usada para manipular sinais.

O primeiro passo para se obter uma DFT e considerar o teorema da Amostragem. Tal teorema afirma que um sinal real $x(t)$, cujo o espectro e limitado em $\phi\ Hz$, pode ser reconstruído a partir de suas amostras tomadas uniformemente a uma taxa $f_s > 2\phi$ (LATHI, 2007, p. 679). Em seguida, a amostragem de $x(t)$, feita a uma frequência $f_s$, pode ser obtida pela multiplicação de $x(t)$ por um trem de impulsos $\delta(t)$. Sendo tais impulsos unitários e periódicos, repetidos a cada $T = 1/f_s$ segundos, por um numero

total de amostras $N_0$, a amostragem pode ser definida por:

$$\overline{x}(t) \;=\; x(t)\delta_T(t) \;=\; \sum_{n=0}^{N_0-1} x(nT)\delta(t-nT) \tag{6}$$

Por conveniência, deseja-se obter um espectro do sinal amostrado $x(t)$ em função de $\omega$ ou expresso em termos de frequência. Para tal, segundo Lathi (2007, p. 681), o trem de impulsos $\delta(t)$ e um sinal periódico que pode ser descrito pela série trigonométrica de Fourier da seguinte forma:

$$\delta_T(t) \;=\; \frac{1}{T}[1 + 2cos(\omega_s t) + 2cos(2\omega_s t) + 2cos(3\omega_s t) + \dots] \tag{7}$$

Logo, multiplicando $x(t)$ por $\delta_T(t)$, obtêm-se:

$$\overline{x}(t) \;=\; x(t)\delta_T(t) \;=\; \frac{1}{T}[x(t) + 2x(t)cos(\omega_s t) + 2x(t)cos(2\omega_s t) + 2x(t)cos(3\omega_s t) + \dots] \tag{8}$$

?

Segundo Oppenheim e Willsky (2010, p. 125), a transformada de Fourier do primeiro termo $x(t)$, em (8), é $X(\omega)$. Já a transformada de Fourier do segundo termo 2x(t)cos(!st) é $X(\omega ? \omega_s) \;+\; X(\omega + \omega_s)$, e do terceiro termo $2x(t)cos(2\omega_s t)$ é $X(\omega ? 2\omega_s) + X(\omega + 2\omega_s)$. E assim, semelhantemente a transformada de Fourier dos demais termos da serie que descreve (8), representam o espectro $X(\omega)$ deslocado em $n\omega_s$ e $?n\omega_s$. Assim,

$$\overline{X}(\omega) \;=\; \frac{1}{T}\sum_{\infty}^{-\infty} X(\omega - n\omega_s) \tag{9}$$

Desde que a frequência de amostragem $f_s$ garanta o critério do teorema da Amostragem, o sinal $\overline{X}$ será constituído de repetições não sobrepostas de $x(\omega_0)$, a um intervalo de tempo $T \;=\; 1/f_s$. Logo tanto $\overline{X}(\omega)$, quanto $\overline{x}(t)$ são periódicas e equivalentes, porém com representações distintas do especto amostrado. Sendo assim, através da propriedade de deslocamento no tempo da transformada de Fourier (10) e da e (6), obtém-se (11) Oppenheim e Willsky (2010, p. 125):

$$\delta(t - nT) \longleftrightarrow e^{-jn\omega T} \tag{10}$$

$$\overline{x}(t) \;=\; \sum_{n=0}^{N_0-1} x(nT)e^{-jn\omega T} \tag{11}$$

Segundo Lathi (2007, p. 705), a transformada de $\overline{x}(t)$ pode ser aproximada, considerando um certo *aliasing* negligenciável, para $X(\omega)/T$. Portanto:

$$X(\omega) \;=\; T\sum_{n=0}^{N_0-1} x(nT)e^{jn\omega T} \;\; |\omega| \leq \frac{\omega_s}{2} \tag{12}$$

Analisando a propriedade periódica de $x(t)$ e $X(\omega)$, e considerando $x(nT)$ e $X(r\omega_0)$ a n-ésima e r-ésina amostra de $x(t)$ e $X(\omega)$, respectivamente, são definidas as seguintes variáveis:

$$x_n \;=\; Tx(nT) \tag{13}$$

$$x_n \;=\; \frac{T_0}{N_0}x(nT) \tag{14}$$

$$X_r \;=\; X(\omega) \tag{15}$$

$$\omega \;=\; r\omega_0 \tag{16}$$

$$X_r \;=\; X(r\omega_0) \tag{17}$$

$$\omega_0 \;=\; 2\pi f_0 \;=\; \frac{2\pi}{T_0} \tag{18}$$

Assim, substituindo (17) e (14) em (12), e fazendo $\omega_0 T \;=\; \Omega_0 \;=\; 2pi/N_0$, se obtêm a seguinte expressão para a transformada discreta de Fourier (OPPENHEIM; WILLSKY, 2010, p. 125):

$$X_r \;=\; \sum_{n=0}^{N_0-1} x_n e^{j\omega_0 nr} \tag{19}$$

Onde:

$$\Omega_0 = \frac{2\pi}{N_0} \tag{20}$$

Para compactar a expressão de (19) se faz a substituição da expressão exponencial pela variável $W$, de modo que $W_{N_0} = e^{?2\pi/N_0} = e^{-j\Omega_0}$. Logo a expressão para DFT é dada por (21) (MEYER-BAESE, 2007, p. 344):

$$X_r = \sum_{n=0}^{N_0-1} x_n e^{j\omega_0 nr} \tag{21}$$

Onde:

$$0 \leq k \leq N_0 - 1 \tag{22}$$

## 2.3 TRANSFORMADA RÁPIDA DE FOURIER

Para se calcular uma DFT de $N_0$ valores usando apenas (21), e necessário realizar um total de $N_0^2$ multiplicações e $N_0(N_0 - 1)$ somas utilizando números complexos. Deste modo, quando $N_0$ assume um valor elevado, muitos recursos computacionais são necessários, até chegar ao ponto de que esse algoritmo se torna impraticável.

Para que se possa reduzir o numero de operações matemáticas necessárias para calcular a DFT é que surgiu o algoritmo criado por J.W. Cooley e John Tukey, conhecido como Transformada Rápida de Fourier ou FFT *(Fast Fourier Transform)* Lathi (2007, p. 719). Para reduzir o numero de cálculos, a FFT se utiliza da propriedade linear da transformada de Fourier. Já que, segundo Oppenheim e Willsky (2010, p. 119), a transformada de Fourier de um sinal pode ser dada pela combinação linear da transformada de Fourier de segmentos menores do mesmo sinal. Logo, é possível aplicar a DFT o paradigma da Divisão e Conquista, o qual é um recurso muito utilizado em algoritmos de ordenação.

Segundo Cormen *et al.* (2002, p. 21) um algoritmo de Divisão e Conquista realiza o desmembramento de um problema em vários subproblemas que são idênticos ao original, porém menores em sua faixa de ação, o que os torna mais simples de resolver. Em seguida, resolvem-se os subproblemas recursivamente e combinam-se essas soluções de modo a obter a solução para o problema original.

De modo muito semelhante o algoritmo da FFT prevê uma divisão recursiva da DFT em dois blocos: bloco par e o bloco ímpar, como mostrado em (23) (CHU;

GEORGE, 1999, p. 35). Nesta mesma equação os limites dos somatórios de ambas as parcelas ímpar e par foram redefinidas para englobar apenas metade dos $N_0$ pontos, bem como os expoentes de $W$ foram ajustados.

$$X_r = \underbrace{\sum_{n=0}^{\frac{N_0}{2}-1} x_{2n} W_{N_0}^{2nr}}_{Parcela\ Par} + \underbrace{\sum_{n=0}^{\frac{N_0}{2}-1} x_{2n+1} W_{N_0}^{(2n+1)r}}_{Parcela\ Ímpar} \tag{23}$$

Utilizando algumas das propriedades geométricas de $W$, já que o mesmo representa um numero complexo, pode-se realizar simplificações importantes em 24. Primeiro nota-se que $W_{N_0/2} = W_{N_0}^2$, logo:

$$X_r = \underbrace{\sum_{n=0}^{\frac{N_0}{2}-1} x_{2n} W_{N_0}^{2nr}}_{G_r} + \underbrace{\sum_{n=0}^{\frac{N_0}{2}-1} x_{2n+1} W_{N_0}^{(2n+1)r}}_{H_r} \tag{24}$$

Como $G_r$ e $H_r$ são DFTs com $N_0/2$ pontos cada, então ambos possuem um período de $N_0/2$. Com base na propriedade periódica destas DFTs pode-se utilizar as simplificações (25) e (26) para reduzir o número de cálculos na DFT (LATHI, 2007, p. 721).

$$G_{r+\frac{N_0}{2}} = G_r \tag{25}$$

$$H_{r+\frac{N_0}{2}} = H_r \tag{26}$$

$$W_{N_0}^{r+\frac{N_0}{2}} = W_{N_0}^{\frac{N_0}{2}} = e^{-j\pi} W_{N_0} = -W_{N_0}^r \tag{27}$$

Alem disso, a expressão em (27) pode ser assumida para se reduzir o numero de cálculos da FFT. Portanto, usando (28) e (29) se obtém, respectivamente, os primeiros $N_0/2$ pontos e os últimos $N_0/2$ pontos da FFT.

$$W_{N_0}^{r+\frac{N_0}{2}} = W_{N_0}^{\frac{N_0}{2}} = e^{-j\pi} W_{N_0} = -W_{N_0}^r \tag{28}$$

$$W_{N_0}^{r+\frac{N_0}{2}} = W_{N_0}^{\frac{N_0}{2}} = e^{-j\pi} W_{N_0} = -W_{N_0}^r \tag{29}$$

Portanto, uma DFT pode ser calculada combinando duas DFTs de $N_0/2$, tal

como mostrado em (28) e (29). É comum na literatura representar este processo de cálculo de DFT feito pelo algoritmo da FFT pelo diagrama da Figura (3). Este diagrama é conhecido como *Butterfly* de Fluxo do Sinal (CHU; GEORGE, 1999, p. 36).

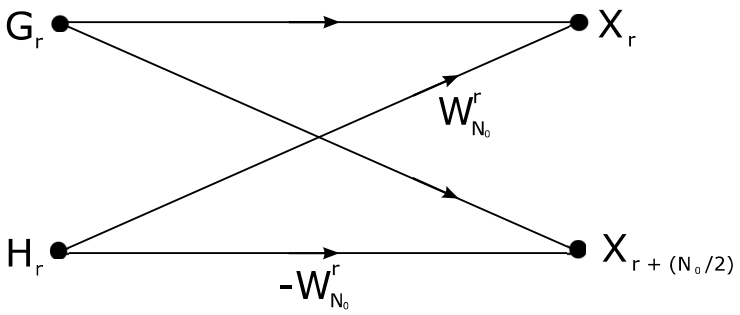


**Figura 3: Butterfly do Fluxo do Sinal**
**Fonte: Lathi (2007, p. 721)**

Aliando o conceito de divisão em conquista ao método de cálculo da DFT usando o diagrama *Butterfly*, a representação do algoritmo da FFT pode ser facilmente representado pelo diagrama da Figura (4) (LATHI, 2007, p. 722). Nesta figura, a FFT é feita para apenas 8 amostras de sinal $X$.

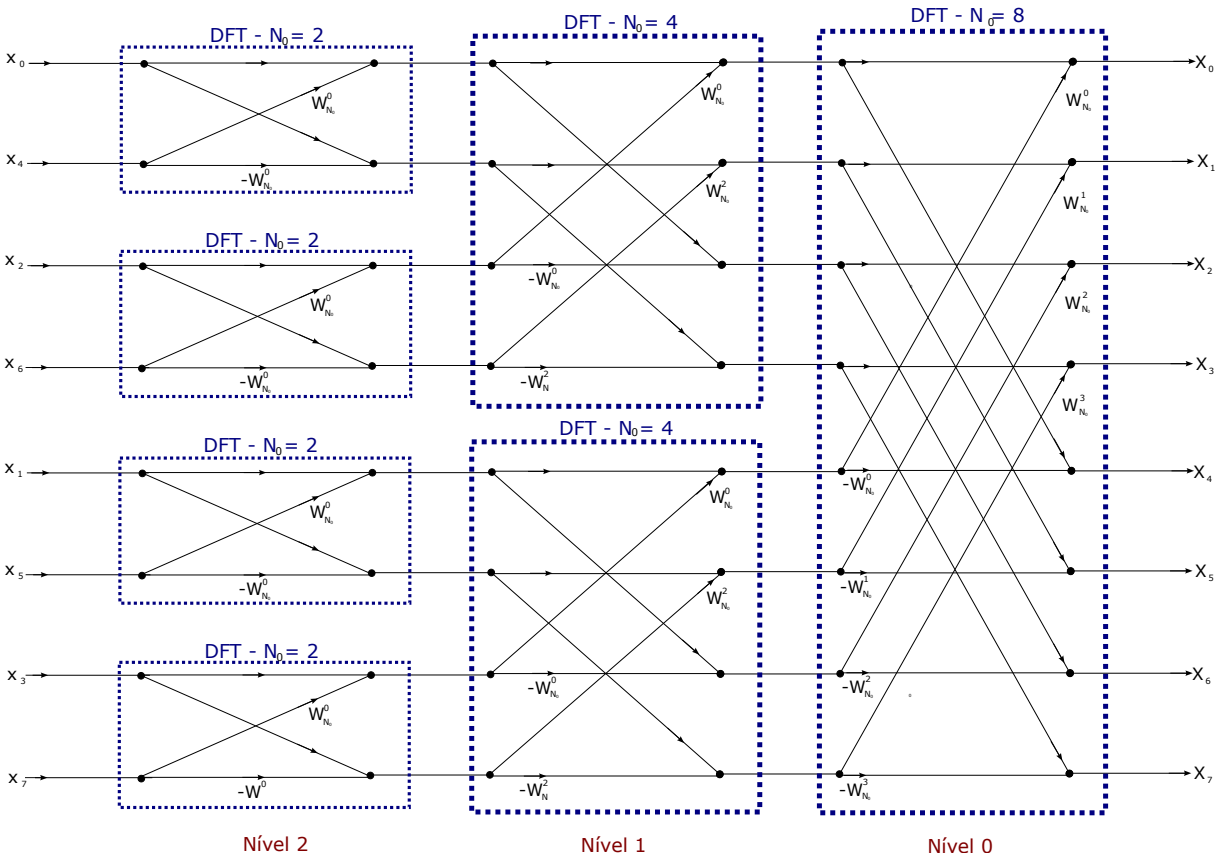


**Figura 4: FFT de 8 Pontos**
**Fonte: Lathi (2007, p. 722)**

Apos se dividir uma DFT de tamanho $N_0$ em duas DTFs de tamanho $N_0/2$, e subdividindo cada uma das DFTs de tamanho $N_0/2$ em duas $N_0/4$ (LATHI, 2007, p. 721). E assim o procedimento continua ate que se atinja um nível em que as DFTs tenham tamanho $N_0/2^n = 2$, ou seja quando se atinge DFTs que possuam um custo de cálculo mínimo.

Um fato importante sobre o algoritmo da FFT e que o valor $N_0$ pode ser escolhido segundo a relação $N_0 = r_n$, onde $n$ e o numero de níveis necessários para calcular a FFT, e $r$ e o mínimo tamanho DFT. Os algoritmos de FFT mais usados possuem a base $r$ igual a 2 ou a 4. Consequentemente, estes algoritmos são conhecidos respectivamente como Radix-2 e Radix-4 (MEYER-BAESE, 2007, p. 365). Neste trabalho o foco sera o algoritmo com a base $r$ igual a 2, ou seja o Radix-2.

Na Figura 2 as DFTs estão agrupadas por níveis, onde cada nível abrange as DFTs de mesmo tamanho $N_0/n$, e no ultimo nível a esquerda há quatro DFTs de tamanho 2. O numero de níveis necessários em uma FFT de $N_0$ pontos é $log_2 N_0$. Os valores de $X$ a direita estão ordenados de forma crescente, por ? em os valores de $x$ a esquerda estão ordenados de forma diferente. Está ordenação é conhecida como *Bit-Reverse* (CHU; GEORGE, 1999, p. 51).

Segundo Chu e George (1999, p. 51), quando se divide o processo de cálculo de uma DFT em duas, sendo uma responsável pelos valores pares e a outra pelo valores ímpares, conforme as conexões entres o níveis vão ocorrendo há, permutações entre os sinais. O processo de *Bit-Reverse* prevê estas permutações, e através dele é possível saber qual a ordem adequada dos sinais na entrada da FFT. Para aplicar o conceito de *Bit-Reverse*, basta considerar um elemento $x_k$ de ordem $n$ e escreve-lo na base binaria com $log_2 N_0$ bits. Em seguida, para determinar onde $x_k$ será ocupado, basta converter novamente para base decimal o número binário obtido lendo esse na ordem inversa dos bits.

Ao final de todo o processo de simplificação do cálculo da DFT, o algoritmo da FFT necessita apenas realizar $(N_0/2)\ log_2\ N_0$ multiplicações e $N_0\ log_2\ N_0$ somas complexas (LATHI, 2007, p. 720). Desta forma, reduz-se assim a complexabilidade do algoritmo, tornando a DFT praticável até para valores elevados de $N_0$.

## 2.4 FPGA

Como afirma Meyer-Baese (2007, Prefácio), muitos algoritmos de processamento de sinais, como FFT *(Fast Fourier Transform)* e os filtros FIR ou IIR, imple-

mentados anteriormente em PDSPs ou em Circuitos Integrados de Aplicação Especifica ou ASIC *(Application Specific Integrated Circuits)*, agora estão sendo implementados em FPGAs.

Moore (2007)[p. 4] define a FPGA como um dispositivo semicondutor capaz de ser totalmente redefinido após sua fabricação, permitindo ao desenvolvedor reconfigurar produtos e funções já implementadas, adaptando o *hardware* a novas funções. De forma prática, a FPGA permite uma flexibilidade em um projeto, podendo mudar a forma como ele é implementado, sem a necessidade de se construir um *hardware* novo.

## 2.5 ALGORITMO CORDIC

Na equação (28) nota-se que o calculo a FFT depende essencialmente de um multiplicação complexa entre $W_{N_0}^r$ e $H_r$, o que de fato nada mais é do que a operação trigonométrica de rotacionar o vetor $H_r$ pelo ângulo de $2\pi r/N_0$. Esta tarefa pode ser realizada através da representação dos vetores na forma retangular, e utilização de 4 multiplicadores e 2 somadores(DESPAIN, 1974, p. 1). Porém em termos de complexidade hardware, multiplicadores são mais elaborados e em muitos dispositivos FPGA possuem apenas algumas dezenas. Logo com o objetivo de oferecer uma solução para funções trigonométricas mais simples Jack E. Volder(**??**) desenvolveu a técnica de computação trigonométrica CORDIC (*COordinate Rotation DIgital Computer*).

### 2.5.1 CORDIC TRADICIONAL

O algoritmo CORDIC tem como base as micro-rotações do vetor alvo, de tal modo que cada micro-rotação possa ser feita por somas e deslocamentos de bits. Para tal considere o vetor $H = x + iy$, e a matriz de rotação $A$ em função do angulo $\theta$. A rotação do vetor é dada pela equação (30)(GARRIDO *et al.*, 2016).

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{30}$$

Isolando o termo $cos(\theta)$ da equação (30), é obtido o sistema (31), o qual é a base da técnica CORDIC convencional (EL-MOTAZ *et al.*, 2014).

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = cos(\theta) \begin{bmatrix} 1 & -tan(\theta) \\ tan(\theta) & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{31}$$

Segundo (**??**), ao invés de rotacionar o vetor $H$ pelo angulo $\theta$, o algoritmo CORDIC rotaciona $H$ por ângulos muito menores, de tal forma que:

$$\sum_{n=0}^{N-1} \theta_n \ \simeq \ \mu\theta \tag{32}$$

e ainda,

$$\theta_n \ = \ tan^{-1}(2^{-n}) \tag{33}$$

Logo aproximando $tan(\theta_n)$ para $\theta_n$, modifica-se a equação (31) e se obtêm:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \prod_{n=0}^{N-1} K_c \begin{bmatrix} 1 & -\mu 2^{-n} \\ \mu 2^{-n} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{34}$$

$$k_c = cos(tan^{-1}(2^{-n})) = \frac{1}{\sqrt{1+2^{-2n}}} \tag{35}$$

onde

$$\mu \in \{-1, 1\} \tag{36}$$

$K_c$ é calculado a cada micro-rotação $n$, assim seu valor total é dado pelo produto de todos os $N$ ganhos. O número total de micro-rotações ($N$) é escolhido de acordo com o SQNR (*Signal-to-Quantization-Noise Ratio* ) admissível a cada rotação. Segundo (**??**), considerando $N \rightarrow \infty$, $K_c$ é aproximadamente igual a $0,6073$, e seu inverso $1,647$ é conhecido como ”Ganho CORDIC”. Tal ganho é aproximadamente o mesmo para todas as rotações e em muitos sistemas ele não precisa ser compensado.

Portanto o Algoritmo CORDIC se resume a operações de deslocamento de *bit* e somas. A direção das micro-rotações é determinada por $\mu$, que depende diretamente sinal do ângulo $\theta_n$ a se rotacionar. Em aplicações onde o ângulo a se rotacionar é previamente conhecido, que é o caso da FFT, as sequências de rotações $\mu$ podem ser armazenado em uma ROM (EL-MOTAZ *et al.*, 2014). Tornando as micro-rotações como interações $n$, e armazenando em $z$ os ângulos rotacionados a cada interação $\theta_n$, o algoritmo CORDIC é dado por:

$$x(n+1) = x(n) - [\mu 2^{-n}]y(n) \tag{37}$$

$$y(n+1) = y(n) - [\mu 2^{-n}]x(n) \tag{38}$$

$$z(n+1) = z(n) - tan^{-1}[\mu 2^{-n}] \tag{39}$$

### 2.5.2 EEAS-CORDIC

No algoritmo CORDIC cada ângulo de rotação $\theta_n$ é necessariamente determinado de maneira sequencial, após cada interação $n$. Em aplicações onde os ângulos de rotação são conhecidos é possível contornar está restrição através de métodos *Angle Recording* (AR) (WU; WU, 2001).

Os AR tem como objetivo reduzir o numero de interações CORDIC através substituição dos ângulos de rotação pela combinação lineares de um conjunto elementar de micro-rotações. Com foco na redução do erro $\epsilon$ oriundo da aproximação (32), o método *Elementary-Angle-Set* (EASR) é um AR que expande o conjunto de combinações lineares adicionando zero ao conjunto de $\mu$, obtendo uma melhor aproximação para certos valores de $\theta$ e uma redução de até 50% no número de interações (MEHER *et al.*, 2009). Deste modo o erro de aproximação é expresso por:

$$\epsilon = \left| \theta \; - \; \sum_{n=0}^{N} \mu\theta_n \right| \; = \; \left| \theta \; - \; \sum_{n=0}^{N} \mu \; tan^{-1}(2^{-n}) \right| \tag{40}$$

Onde:

$$\mu \in \{-1, \; 0, \; 1\} \tag{41}$$

Por outro lado o *Extended Elementary-Angle-Set Recoding* (EEASR) apresenta um método baseado no EASR que estende o conjunto de ângulos elementares (*Elementary-Angle-Set*, EAS), para aumentar a possibilidades de decomposição do ângulo de rotação(WU *et al.*, 2003). Para perceber a modificação que o EEASR propõem nota-se primeiramente que o conjunto de ângulos elementares no CORDIC tradicional é definido como:

$$S_1 = \{tan^{-1}(\mu 2^{-s})\} \tag{42}$$

$$: \; \mu \in \{-1, 0, 1\}, \; s \in \{0, 1, \ldots, N-1\} \tag{43}$$

Como é possível notar em (43) os ângulos elementares dependem de apenas um termo potência de dois, ou *Signed Power of Two* (SPT). Segundo (WU *et al.*, 2003) para aumentar a precisão dos ângulos elementares e consequentemente reduzir o numero de interações pode-se adicionar mais um termo SPT em (43). Assim,

$$S_2 = \{tan^{-1}(\mu_0 2^{-s_0} + \mu_1 2^{-s_1})\} \tag{44}$$

$$: \ \mu_0, \mu_1 \in \{-1, 0, 1\}, \ s_0, s_1 \in \{0, 1, \ldots, N-1\} \tag{45}$$

Portanto alterando a equação (34) com base em 45 é obtido seguinte expressão:

$$x(n+1) = x(n) - [\mu_0 2^{-s_0(n)} + \mu_1 2^{-s_1(n)}]y(n) \tag{46}$$

$$y(n+1) = y(n) - [\mu_0 2^{-s_0(n)} + \mu_1 2^{-s_1(n)}]x(n) \tag{47}$$

$$z(n+1) = z(n) - tan^{-1}[\mu_0 2^{-s_0(n)} + \mu_1 2^{-s_1(n)}] \tag{48}$$

Ao adicionar mais termos tanto a $\mu$ e a $S_1$ cada operação de rotação do algoritmo CORDIC não tem mais o ganho constante $K_c$, e passa a ter um ganho na forma:

$$K_c \ = \ \frac{1}{\sqrt{1 + [\mu_0 2^{-s_0(n)} + \mu_1 2^{-s_1(n)}]^2}} \tag{49}$$

$$K_c \ = \ \prod_{n=0}^{N-1} K(n)_c \tag{50}$$

### 2.5.3 MSR-CORDIC

O ganho CORDIC $K_c$ apresentado anteriormente sempre se estabelece acima de 1 em qualquer das variações do algoritmo CORDIC apresentados. Com um valor maior que a unidade $K_c$ demanda uma redução de escala no resultado do algoritmo, esta redução provoca uma degradação no valor de SQNR. Logo para evitar esta degradação é necessário manter o modulo do vetor de entrada o mais perto da unidade a cada interação (LIN; WU, 2005). Assim em (LIN; WU, 2005) foi reformulada a expressão 48 da seguinte forma:

$$x(n+1) = \left[\sum_{j=1}^{J} \mu_j 2^{-s_j(n)}\right] x(n) - \left[\sum_{i=1}^{I} \mu_i 2^{-s_i(n)}\right] y(n) \tag{51}$$

$$y(n+1) = \left[\sum_{i=1}^{I} \mu_i 2^{-s_i(n)}\right] x(n) + \left[\sum_{j=1}^{J} \mu_j 2^{-s_j(n)}\right] y(n) \tag{52}$$

$$z(n+1) = z(n) - tan^{-1}\left[\frac{\sum_{i=1}^{I} \mu_i 2^{-s_i(n)}}{\sum_{j=1}^{J} \mu_j 2^{-s_j(n)}}\right] \tag{53}$$

Onde

$$N = I + J \tag{54}$$

O *Mixed Scaling Rotation* (MSR) busca realizar em uma única interação a operação de redução de escala e de micro-rotação. Para tal ele adiciona mais graus de liberdade através da inserção de termos SPT em $x(n)$ da parcela $x(n+1)$, e em $y(n)$ da parcela $y(n+1)$. Consequentemente ao inserir tais termos na função tanto $\theta_n$ quanto $K_c$ precisam ser corrigidos (KUO *et al.*, 2003). Portanto $K_c$ passa a ser:

$$K(n)_c = \frac{1}{\sqrt{\left(\sum_{i=1}^{I} 2^{-s_i(n)}\right)^2 + \left(\sum_{j=1}^{J} 2^{-s_j(n)}\right)^2}} \tag{55}$$

$$K_c = \prod_{n=0}^{N-1} K(n)_c \tag{56}$$

Como pode ser percebido em (55) e (53), através da escolha adequada dos conjuntos de termos $S_i$ e $S_j$, e dos valores de $I$ e $J$ é possível aproximar $K_c$ da unidade e ao mesmo tempo reduzir o erro $\epsilon$. Com mais graus de liberdade o MSR possui uma densidade combinacional dentro do circulo unitário maior do que o CORDIC tradicional e o EEAS-CORDIC (KUO *et al.*, 2003).

### 2.5.4 ESTRATÉGIA DE PROJETO DOS PARÂMETROS CORDIC

Como apresentado anteriormente tanto o algoritmo EEAS-CORDIC quanto o MSR-CORDIC possuem parâmetros de projeto a serem selecionados para que o processo de rotação seja realizado minimizando interações e maximizando a precisão. Em (WU *et al.*, 2003) o autor propões um método de otimização para parâmetros EEAS chamado *Trellis-Based Search*(TBS). A partir do numero máximo de interações $N$, do

ângulo de rotação $\theta$ e de $W$, que representa numero de bits de $\theta$, o TBS emprega um método de otimização para encontrar os melhores parâmetros $S^0(n)$, $S^1(n)$, $\mu_0$ e $\mu_1$ (WU *et al.*, 2003). A função de otimização usado em *TBS* é dada pela minimização do erro $\epsilon$, o qual está descrita em 40, e as restrições são dadas pelos conjuntos de ângulos elementares $S_2$ apresentados em (45).

*TBS* é baseado no efeito que os diferentes arranjos dos ângulos elementares possíveis tem sobre $\epsilon$. Em (WU *et al.*, 2003) o autor utiliza um exemplo base para explicar o algoritmo, o qual também será usado aqui.

Considerando um ângulos de rotação $\theta = \pi/3$, o máximo de interações $N = 4$ e a resolução de $\theta$ com $W =$ bits. O algoritmo segue o seguintes passos:

1. *Inicialização:* Inicialmente é definido $z(S_2)$ como sendo o numero de elementos no conjunto de ângulos elementares, $r(k)$ representando cada elemento de $S_2$ individualmente, $1 \leq k \leq z(S_2)$ e $S_2$ sendo o conjunto dos ângulos elementares não redundantes. $\phi(n, k)$ representa a estrutura de acumulação do algoritmo, e será usada para se alcançar o resultado. Defini-se $z(S_2) = 15$ combinações e $\phi(1, k) = r(k)$.

2. *Acumulação:* A cada elemento do vetor $\phi(i + 1, k)$ é atribuído o valor de $\phi(1 : z(S_2), i)$ que corresponde ao mínimo de $\|\phi(1 : z(S_2), i) + r(k) - \theta\|$. O índice $i$ varia $1 \leq i \leq R_m - 1$, e $k$ varia de $1 \leq k \leq Z(S_2)$.

3. *Determinação do Ótimo Global:* Ao final do processo de acumulação os elemento da coluna $R_m$ de $\phi$ apresentam os resultados globais. Logo dentro desta coluna é determinado o valor que mais se aproxima de $\theta$, defini-se este como $\theta_{TBS}$.

4. *Determinação do Caminho Solução:* A partir do elemento $\theta_{TBS}$ da coluna $R_m$ de $\phi$ é traçado o caminho reverso até a coluna $1$ de $\phi$. A começar por $(k' = \theta_{TBS})$ e $(i = R_m)$ é escolhido o valor da coluna $(i - 1)$ que minimiza $\|\phi(1 : z(S_2), i - 1) + r(k') - \theta\|$, em seguida se atribui a $k'$ a posição $k$ do valor minimizante. Os valores $k'$ são armazenados em um vetor e representam as combinações de ângulos elementares que minimizam o erro $\epsilon$.

A tabela 1 apresenta os valores encontrados para $\phi$ na execução do exemplo apresentado. Em destaque estão os valores considerados o conjunto minimizante solução $S_2$.

**Tabela 1: Matriz $\phi$ para o dado Exemplo**

| | i=1 | i=2 | i=3 | i=4 |
|---|---|---|---|---|
| k=1 | -0,7854 | 0,3218 | 0,3556 | 0,3218 |
| k=2 | -0,4636 | 0,6435 | 0,6774 | 0,6435 |
| k=3 | -0,245 | 0,8622 | 0,8961 | 0,8622 |
| k=4 | -0,6435 | 0,4636 | 0,4975 | 0,4636 |
| k=5 | 0 | 1,1071 | 1,0304 | 1,0304 |
| k=6 | 0,4636 | 1,1071 | 1,1071 | 0,9612 |
| k=7 | 0,245 | 1,0304 | 1,1071 | 1,141 |
| k=8 | 0,7854 | 1,0304 | 0,9965 | 1,0304 |
| k=9 | -0,9828 | 0,1244 | 0,1582 | 0,1244 |
| k=10 | -0,8961 | 0,2111 | 0,245 | 0,2111 |
| k=11 | -1,1071 | 0 | 0,0339 | 0 |
| k=12 | 0,9828 | 0,9828 | 1,1071 | 1,0167 |
| k=13 | 0,8961 | 1,141 | 1,0204 | 1,0543 |
| k=14 | 0,6435 | 1,1071 | 1,1071 | 0,9991 |
| k=15 | 1,1071 | 1,1071 | 1,1071 | 1,141 |

**Fonte: Autoria Própria**

Para o algoritmo MSR-CORDIC porém é mais complicado determinar todos os parâmetros $S_j$, $S_i$, $\mu_i$, $\mu_j$, $I$ e $J$. Primeiramente nota-se que o conjunto dos ângulos fundamentais deixa de ser representado por (45) e passa a ser (LIN; WU, 2005):

$$S_{N_{SPT}} = \frac{\sum_{i=1}^{I} \mu_i 2^{-s_i(n)}}{\sum_{j=1}^{J} \mu_j 2^{-s_j(n)}} \tag{57}$$

$$: \ \mu_i, j \in \{-1, 0, 1\}, \tag{58}$$

$$s_i, s_j \in \{0, 1, \ldots, N-1\} \tag{59}$$

Portanto o conjunto de soluções ótimas de ângulos elementares para a redução do erro $\epsilon$ expande, o que em primeiro plano possibilidade uma maior oportunidade de conjuntos que otimizam o função do erro $\epsilon$. Entretanto como é desejado manter o ganho $K_c$ perto da unidade, não há apenas uma função objetivo a ser considerada, existe ainda a função do erro entre $K_c$ e a unidade, o qual pode ser expressa como:

$$\epsilon_{Kc} = \left\| 1 - \prod_{n=0}^{N-1} \frac{1}{\sqrt{\left(\sum_{i=1}^{I} 2^{-s_i(n)}\right)^2 + \left(\sum_{j=1}^{J} 2^{-s_j(n)}\right)^2}} \right\| \tag{60}$$

Como ambos os erros $\epsilon$ e $\epsilon_{Kc}$ tem impactos distintos no erro total de cálculo de uma unidade CORDIC, sendo que o primeiro impacta no ângulo do vetor de saída e o segundo impacta no módulo deste ângulo. Assim neste artigo será usado uma adaptação do algoritmo TBS já apresentado.

Na modificação proposta aqui várias instâncias do algoritmo TBS são calculados para diferentes combinações de $I$ e $J$, sempre respeitando a restrição $N = I+J$. Cada combinação $I$ e $J$ gera diferentes combinações de $S_{N_{SPT}}$. Assim cada instância TBS gera um valor de erro $\epsilon$ e $\epsilon_{Kc}$, então é selecionada a instância que possui o menor valor de $\epsilon + \epsilon_{Kc}$ como a melhor solução global.

## 2.6 ZYNQ-7000

# 3 IMPLEMENTANDO PROCESSADOR CORDIC

# 4 IMPLEMENTANDO FFT NO ZYNQBERRY - TE0726

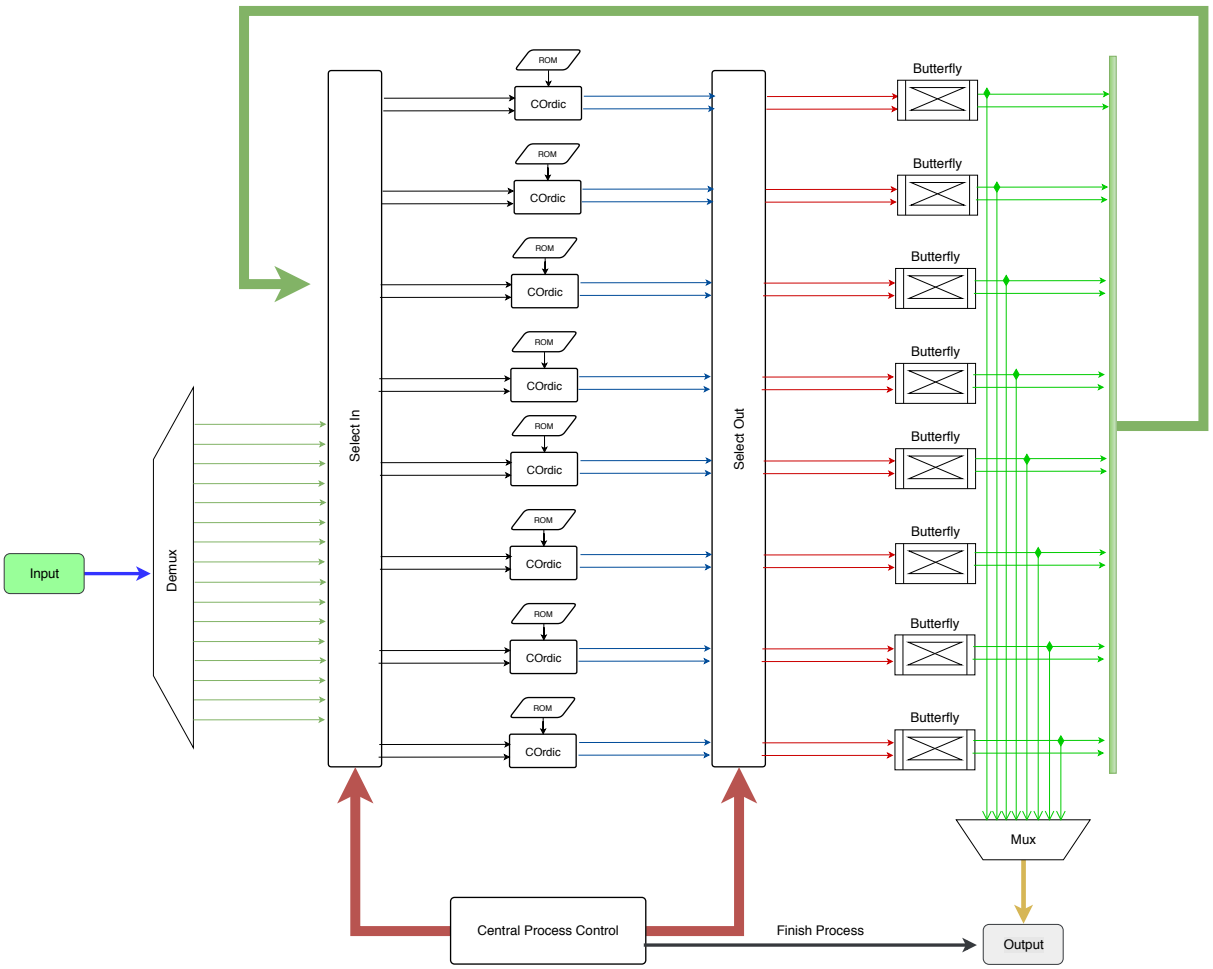


**Figura 5: Arquitetura Implementada FFT de 16 Pontos**
**Fonte: Autoria Própria**

**Tabela 2: Exemplo de Tabela**

| Posição | Município | População |
|---|---|---|
| 1 | São Paulo | 11.376.685 |
| 2 | Rio de Janeiro | 6.690.290 |
| 3 | Salvador | 2.710.968 |
| 4 | Brasília | 2.648.532 |
| 5 | Fortaleza | 2.500.194 |

**Fonte: (??)**

### 4.0.1 TESTES E EXEMPLOS DE LISTA DE SIGLAS

Testes da lista de siglas:

(TDLS1).

(EASREFER).

(ER).

Esta é uma sigla em que os parênteses não aparecem: SQPNA.

(DMA).

É preciso rodar duas vezes o Latex para que a lista de siglas seja atualizada.

### 4.0.2 TESTES E EXEMPLOS DE LISTA DE SÍMBOLOS

$$\phi = \vec{\alpha} \otimes \upsilon\omega\psi_{n-1}^{jk} \tag{61}$$

Onde:

$\phi$: Angulo phi.

$\vec{\alpha}$: Alfa.

$\upsilon\omega\psi_{n-1}^{jk}$: Um simbolo grande.

Este símbolo ($\nabla$) foi inserido no texto, fora do ambiente *equation*.

$\upsilon\omega\psi_{n-1}^{jk}$

### 4.0.3 EXEMPLOS DE CITAÇÃO DE REFERÊNCIAS BIBLIOGRÁFICAS

(**??**)

(**????**)

Conforme **??**) a metodologia proposta é, bla bla bla...

(**??**)

# 5 NÍVEL 1 - TESTES DE CAPITULAÇÃO - PRIMÁRIO

## 5.1 NÍVEL 2 - SECUNDÁRIO

### 5.1.1 NÍVEL 3 - TERCIÁRIO

#### 5.1.1.1 NÍVEL 4 - QUATERNÁRIO

##### 5.1.1.1.1 NÍVEL 5 - QUINÁRIO

# REFERÊNCIAS

AMARAL, Carlos Eduardo Ferrante do; LOPES, Heitor S; ARRUDA, Lúcia V; HARA, Marcos S; GONçALVES, Antonio J; DIAS, Adilson A. Design of a complex bioimpedance spectrometer using dft and undersampling for neural networks diagnostics. **Medical engineering & physics**, Elsevier, v. 33, n. 3, p. 356–361, 2011.

BINGHAM, J. A. C. Multicarrier modulation for data transmission: an idea whose time has come. **IEEE Communications Magazine**, v. 28, n. 5, p. 5–14, May 1990. ISSN 0163-6804.

CHU, Eleanor; GEORGE, Alan. **Inside the FFT black box: serial and parallel fast Fourier transform algorithms**. 1. ed. [S.l.]: CRC Press, 1999.

CORMEN, Thomas H; LEISERSON, Charles E; RIVEST, Ronald L; STEIN, Clifford. **Algoritmos: teoria e prática**. 2. ed. [S.l.]: Editora Campus, 2002.

DESPAIN, Alvin M. Fourier transform computers using cordic iterations. **IEEE Transactions on Computers**, IEEE, v. 100, n. 10, p. 993–1001, 1974.

EL-MOTAZ, M. A.; NASR, O. A.; OSAMA, K. A cordic-friendly fft architecture. In: **2014 International Wireless Communications and Mobile Computing Conference (IWCMC)**. [S.l.: s.n.], 2014. p. 1087–1092. ISSN 2376-6492.

GARRIDO, M.; KäLLSTRöM, P.; KUMM, M.; GUSTAFSSON, O. Cordic ii: A new improved cordic algorithm. **IEEE Transactions on Circuits and Systems II: Express Briefs**, v. 63, n. 2, p. 186–190, Feb 2016. ISSN 1549-7747.

HAYKIN, SIMON S; VEEN, Barry Van. **Sinais e sistemas**. [S.l.]: Bookman, 2001.

HE, Hongjiang; GUO, Hui. The realization of fft algorithm based on fpga co-processor. In: IEEE. **Intelligent Information Technology Application, 2008. IITA'08. Second International Symposium on**. [S.l.], 2008. v. 3, p. 239–243.

IBRAHIM, Muhammad; KAMAL, Mohsin; KHAN, Omar; ULLAH, Khalil. Analysis of radix-2 decimation in time algorithm for fpga co-processors. In: IEEE. **Computing, Electronic and Electrical Engineering (ICE Cube), 2016 International Conference on**. [S.l.], 2016. p. 154–157.

KUO, Jen-Chih; WEN, Ching-Hua; LIN, Chih-Hsiu; WU, An-Yeu (Andy). Vlsi design of a variable-length fft/ifft processor for ofdm-based communication systems. **EURASIP Journal on Advances in Signal Processing**, v. 2003, n. 13, p. 439360, Dec 2003. ISSN 1687-6180. Disponível em: <https://doi.org/10.1155/S1110865703309060>.

LATHI, Bhagwandas Pannalal. **Sinais e Sistemas Lineares**. 2. ed. [S.l.]: Brasil: Bookman, 2007.

LIN, Chih-Hsiu; WU, An-Yeu. Mixed-scaling-rotation cordic (msr-cordic) algorithm and architecture for high-performance vector rotational dsp applications. **IEEE Transactions on Circuits and Systems I: Regular Papers**, v. 52, n. 11, p. 2385–2396, Nov 2005. ISSN 1549-8328.

MARTINSEN, Orjan G; GRIMNES, Sverre. **Bioimpedance and bioelectricity basics**. [S.l.]: Academic press, 2011.

MEHER, P. K.; VALLS, J.; JUANG, T. B.; SRIDHARAN, K.; MAHARATNA, K. 50 years of cordic: Algorithms, architectures, and applications. **IEEE Transactions on Circuits and Systems I: Regular Papers**, v. 56, n. 9, p. 1893–1907, Sept 2009. ISSN 1549-8328.

MEYER-BAESE, Uwe. **Digital signal processing with field programmable gate arrays**. 3. ed. [S.l.]: Springer, 2007.

MOORE, Andrew. **FPGAs for Dummies Altera Special Edition**. [S.l.]: Wiley Brand, 2007.

NIRMALRAJ, S.; VIGNESWARAN, T. **A novel method to compress voice signal using compressive sensing**. March 2015. 1-4 p.

OPPENHEIM, Alan V; WILLSKY, Alan S. **Sinais e Sistemas**. 2. ed. [S.l.]: Brasil:Pearson, 2010.

TRADINGVIEW. **Índice BM & FBOVESPA**. 2017. `https://www.tradingview.com/symbols/BMFBOVESPA-IBOV/`. Acessado em 14 de Agosto de 2017.

TRIANTIS, Iasonas F; DEMOSTHENOUS, Andreas; RAHAL, Mohamad; HONG, Hongwei; BAYFORD, Richard. A multi-frequency bioimpedance measurement asic for electrical impedance tomography. In: IEEE. **ESSCIRC (ESSCIRC), 2011 Proceedings of the**. [S.l.], 2011. p. 331–334.

VANMATHI, K; SEKAR, K; RAMACHANDRAN, Remya. Fpga implementation of fast fourier transform. In: IEEE. **Green Computing Communication and Electrical Engineering (ICGCCEE), 2014 International Conference on**. [S.l.], 2014. p. 1–5.

WANG, B.; ZHANG, Q.; AO, T.; HUANG, M. **Design of Pipelined FFT Processor Based on FPGA**. Jan 2010. 432-435 p.

WU, Cheng-Shing; WU, An-Yeu. A novel trellis-based searching scheme for eeas-based cordic algorithm. In: **2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)**. [S.l.: s.n.], 2001. v. 2, p. 1229–1232 vol.2. ISSN 1520-6149.

WU, Cheng-Shing; WU, An-Yeu; LIN, Chih-Hsiu. A high-performance/low-latency vector rotational cordic architecture based on extended elementary angle set and trellis-based searching schemes. **IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing**, v. 50, n. 9, p. 589–601, Sept 2003.

ZHOU, Bin; PENG, Yingning; HWANG, David. Pipeline fft architectures optimized for fpgas. **International Journal of Reconfigurable Computing**, Hindawi Publishing Corp., v. 2009, p. 1, 2009.

## ANEXO A - STELLAR MYSTERY SOLVED, EINSTEIN SAFE

For more than 30 years, Villanova University astronomer Ed Guinan has been plagued, puzzled, and perplexed by DI Herculis. On the surface, this binary star seems pretty much like any other binary star, with two stars going 'round and 'round each other in a predictable, orderly fashion. But there remained a nagging problem that as much as Guinan wanted, he couldn't just sweep under the rug: DI Her was not behaving in accordance with Einstein's general theory of relativity.

Every year, Guinan and his colleagues would observe the 8.5 - magnitude star. The two stars orbit each other in a plane lined up perfectly with Earth's line of sight, and they eclipse each other every 10.55 days. Thanks to these eclipses, which have been recorded since 1900, Guinan could make exceedingly precise measurements of the stars' masses, sizes, luminosities, and orbital characteristics.

Almost every known aspect of the system was hunky-dory. The stars are exactly as massive, large, and bright as theory predicts. But in a series of published papers, Guinan and his Villanova colleague Frank Maloney kept pointing out that the orbit was not behaving in accordance with general relativity, the cornerstone for modern science's understanding of gravity.

For years, Guinan looked for a third star in the system, or some other factor that could be throwing the orbit out of whack, but to no avail. The point in the orbit at which the two stars come closest continues to advance (precess) each orbital cycle at a rate only one-fourth the amount theory predicts. Guinan continually receives mail from armchair theorists who attempt to explain DI Her's anomalous precession with alternative theories of gravity.

Finally, after decades of frustration, a group led by Simon Albrecht (MIT) has solved the problem, and Einstein's theory has survived unscathed. "The monkey has been lifted off my back,"jokes Guinan.

After being informed about DI Her by Guinan, Albrecht and his colleagues took detailed spectra in 2008 using a 1.93-meter telescope in France. These measurements, unlike earlier ones made by Guinan and others, were obtained with a high-resolution spectrograph. With more modern equipment, computers, and techniques, Albrecht revealed that the rotation axes of the two stars are tipped over on their sides with respect to the orbital plane (similar to Uranus's orientation with respect to the Sun), which makes DI Her an oddity among closely separated binary stars.

In this unusual arrangement, gravitational forces created by the misaligned

equatorial bulges of the stars give them an extra ”kick”when they are closest in their elliptical (oval-shaped) orbits. This kick reduces the orbital precession to the observed rate, eliminating any discrepancies with Einstein’s theory. ”We consider the mystery of the anomalous orbital precession of DI Herculis to be solved,”says Albrecht, whose paper will appear in tomorrow’s *Nature*.

Guinan concurs, but he adds, ”I’m relieved that the general relativity problem is solved. But it’s been replaced with a new mystery.”

DI Her’s two members, both hot, massive, B-type stars, should have formed from a common disk of gas and dust. By feeding from the same disk, the spin axes of the two stars should be nearly perpendicular to the orbital plane, an arrangement seen in the large majority of binary stars, especially those like DI Her that have small separations.

”My guess is that there was a third-body interaction that perturbed the system, or the two stars formed separately and formed a binary through a capture process,”says Guinan. ”Another possibility is that our ideas of how binaries form may be off.”

Together with MIT colleague Joshua Winn, Albrecht is observing other binary systems that display similar anomalies. ”We want to understand if DI Her is a unique system or if misalignment is more common among close systems,”says Albrecht. ”We also want to understand what might cause this misalignment. Clearly, there is more going on then we guessed so far.”

Even if Einstein’s theory remains safe and sound, astronomers and physicists continue to subject it to more stringent tests. Given general relativity’s incompatibility with quantum mechanics (the theory that describes the microworld of atoms and light with extraordinary accuracy), scientists expect that Einstein will not have the last word on gravity. Ultimately, it will be replaced by an even deeper ”theory of everything.”

## A.1 EXEMPLO DE SEÇÃO ANEXO

Esta é uma seção, dentro dos anexos.