Universidade Federal do Paraná

Anderson de Lima Luiz

Trabalho de conclusão de curso

# CIRCUITO DIGITAL COMPENSADOR DE ERROS DE RELÓGIO DE CONVERSORES ANALÓGICO-DIGITAIS ENTRELAÇADOS APLICADO À TECNOLOGIA CMOS 130 nm

CURITIBA
2019

ANDERSON DE LIMA LUIZ

# CIRCUITO DIGITAL COMPENSADOR DE ERROS DE RELÓGIO DE CONVERSORES ANALÓGICO-DIGITAIS ENTRELAÇADOS APLICADO À TECNOLOGIA CMOS 130 nm

**Trabalho de conclusão de curso apresentado ao curso de graduação em engenharia elétrica – setor de tecnologia – Universidade Federal do Paraná, como requisito parcial à obtenção do título de bacharel em engenharia elétrica.**

**Orientador: Prof. Dr. Luis H. A. Lolis**

**Coorientadora: Profa. Dra. Sibilla Batista da Luz França**

CURITIBA
2019

UNIVERSIDADE FEDERAL DO PARANÁ

ANDERSON DE LIMA LUIZ

Esta monografia foi julgada adequada para a obtenção do título de bacharel em engenharia elétrica – sendo aprovada em sua forma final pela banca examinadora:

---

Orientador(a): Prof. : Dr. Luis H. A. Lolis
Universidade Federal do Paraná – UFPR

---

Coorientador(a): Profa. : Dra. Sibilla B. L. França
Universidade Federal do Paraná – UFPR

---

Prof. : Dr. Bernardo R. B. A. Leite
Universidade Federal do Paraná – UFPR

---

Prof : Dr. Eduardo G. Lima
Universidade Federal do Paraná – UFPR

Curitiba, 10 de dezembro de 2019

*Dedicatória*

*Agradeço a Deus,*
*que sempre proporcionou tudo que foi preciso para cumprir esta etapa*
*da minha jornada.*

*Agradeço aos meus pais – Isabel de Lima Luiz e Renato Luiz,*
*por construírem as grandes pontes que me permitiram superar todos os*
*obstáculos com perseverança.*

## RESUMO

Os conversores analógicos digitais entrelaçados no tempo (TIADCs) representam uma topologia de conversores analógico-digitais (ADCs) capazes de unir a alta resolução e frequência de amostragem através do entrelaçamento de ADCs de menor velocidade em ângulos de amostragem consecutivos. O TIADC se destaca pela sua relação linear entre o aumento da frequência de amostragem e o aumento de consumo de energia, relação essa que é exponencial em topologias clássicas de ADC. No entanto, ocorrem erros associados ao tipo de operação destacado: offset, ganho e deriva de relógio. Neste projeto, foca-se na teoria necessária para que projetem-se filtros digitais capazes de compensar o erro de deriva de relógio. Posteriormente, é realizada a implementação desses filtros em hardware com a capacidade de retificar erros com até 25% de deriva - respeitando os limites interpolação de Lagrange, método pelo qual os filtros foram desenvolvidos. A validação da funcionalidade dos filtros foi verificada com a geração de sinais de múltiplos tons e fase aleatória, aumentando a frequência desses tons até o limite de Nyquist gerados na plataforma de simulação virtual MATLAB©. A métrica de validação é a relação sinal-distorção alta (SDR). A compensação mostrou-se eficiente até uma banda de frequência elevada - até 80% do limite de Nyquist. Foram testadas diversas configurações de filtros, em ordem e em cálculo de vírgula fixa (número de bits). Os filtros escolhidos para implementação em dispositivos físicos são os que representam o melhor custo benefício tanto em ordem quanto em número de bits analisando os que consomem menos memória e que utilizam janelas de filtragem que proporcionam a SDR mais elevada. A SDR mira é de 61,96 dB – valor necessário para construção de um ADC de 10 bits. O processo de implementação dos cálculos em vírgula fixa foram realizados e validados através do conversor interno ao próprio MATLAB - mostrando que o comprimento de palavra mais eficiente para este uso específico era de um total de 16 bits, com 8 bits na parte fracionária; utilizando filtros de ordem 50. O código foi traduzido da linguagem MATLAB para a linguagem C e posteriormente VHDL. A validação funcional dos filtros em linguagem de descrição de *hardware* é feita em cosimulação entre o *Simulink* e o *ModelSim*. O código em VHDL é avaliado a partir da plataforma *Cadence Incisive Enterprise Simulator* – para verificação funcional executada com o *ModelSim*. O arquivo .gds (o desenho) do layout é feito a partir das ferramentas *Cadence Genus*$^{TM}$ *Synthesis Solution* e *Cadence Innovus Implementation System* – que alocará as portas lógicas da tecnologia escolhida, para confecção do layout. A partir das mesmas ferramentas, o *layout* é analisado para verificação de regras de desenho e obtenção dos resultados para simulação pós-*layout*. O *Cadence Incisive Enterprise Simulator* realiza a simulação lógica pós-*layout* a partir dos resultados de simulação com não idealidades. A área total do circuito com os quatro filtros mostra ter 0,093 $mm^2$ de área – usando 3544 portas lógicas, apresentando atraso total de 1,512 ns e potência de 16,8 mW.



**Palavras-chave**: TIADCs, Síntese digital, filtragem digital.

## ABSTRACT

Time-Interleaved Analog-to-Digital Converters (TIADCs) represent a topology of Analog-to-Digital Converters (ADCs) capable of uniting high resolutions and sampling frequencies by interlacing lower speed ADCs at consecutive sampling angles. TIADCs stands out for its linear relationship between increased sampling frequencies and increased power consumption, which is exponential in classic ADC topologies. However, errors associated with the device's operation may occur: offset, gain and clock skew. In this project, it is focused on the theory needed to design digital filters capable of compensating the clock skew error. Subsequently, these filters are implemented in hardware with the ability to rectify errors with up to 25 % skew – respecting the Lagrange interpolation limits, the method by which the filters were developed. Validation of filter functionality was verified by generating multi-tone and random phase signals, increasing the frequency of these tones to the Nyquist limit generated on the MATLAB© virtual simulation platform. The validation metric is the high signal-distortion ratio (SDR). Compensation proved to be efficient up to a high frequency band - up to 80 % of the Nyquist limit. Several filter configurations were tested, in order and in fixed point calculations (number of bits). The filters chosen for deployment to physical devices are the most cost-effective in both order and number of bits by analyzing those that consume less memory and use filtering windows that provide the highest SDR. The SDR crosshair is 61.96 dB - value required to build a 10-bit ADC. The process of implementation of the fixed point calculations was performed and validated through the internal converter of MATLAB itself - showing that the most efficient word length for this specific use was a total of 16 bits, with 8 bits in the fractional part; using 50-order filters. The code was translated from MATLAB language to C language and later to VHDL. The functional validation of the hardware description language filters is done in cosimulation between Simulink and ModelSim. The VHDL code is evaluated from the Cadence Incisive Enterprise Simulator platform - for the same functional verification performed with ModelSim. The .gds file (the layout) is made from Cadence $Genus^{TM}$ Synthesis Solution and Cadence Innovus Implementation System, tools - which will allocate the logic gates of the chosen technology in order to create the layout. From the same tools, the layout is analysed for drawing rules accordance and obtaining results for post-layout simulations. Cadence Incisive Enterprise Simulator performs post-layout logic simulations from non-ideal simulation results. The total circuit area with the four filters is 0.093 $mm^2$ in area - using 3544 logic gates, with a total delay of 1.512 ns and 16.8 mW of power.



**Keywords**: TIADCs, Síntese digital, filtragem digital.

## LISTA DE ABREVIATURAS E SIGLAS

| | |
|---|---|
| ADC | Conversor analógico-digital |
| CMOS | Metal-óxido-semicondutor complementar |
| DRC | Verificação das regras de desenho |
| FIR | Resposta ao impulso finita |
| FPGA | Matriz de portas programável em campo |
| IIR | Resposta ao impulso infinita |
| LUT | Tabela de pesquisa |
| NMSE | Erro quadrado médio normalizado |
| RTL | Nível de transferência de registro |
| SDF | Forma de atraso padrão |
| SDR | Relação sinal-distorção |
| TIADCs | Conversores analógico-digitais entrelaçados no tempo |

VHDL Linguagem de descrição de *hardware* de circuitos integrados de velocidade muito alta

# Sumário

# 1 Introdução

## 1.1 Justificativa

A grande utilização de aparatos digitais fomentou o desenvolvimento de maiores velocidades de processamento de sinais, assim como a quantidade de transmissão de informação. Métodos mais eficientes de manipulação de dados para impedir o dispêndio em termos de memória e energia devem ser desenvolvidos – [1]. No que concerne a conversão do sinal analógico para o digital, destaca-se a dificuldade de suprir simultaneamente velocidades elevadas e altas resoluções com baixo consumo e baixa superfície de silício. Os conversores analógico-digitais entrelaçados apresentam uma proposta para resolução do problema do alto consumo para um grande desempenho. O desempenho do método mostra-se mais eficiente do que o uso individual de vários ADCs para manipulação de dados [2], todavia três erros importantes se destacam na disfunção do TIADC: erros de offset, ganho e deriva de clock. O último impacta mais na aparição de erros na reconstrução do sinal – logo, é fundamental propor técnicas que possam otimizar o correto funcionamento do clock entre os M conversores utilizados.

Propõe-se a utilização de filtros de resposta ao impulso finita (FIR) para a correção da deriva de relógio dos múltiplos conversores analógico-digitais e otimização da reconstrução do sinal que se deseja amostrar. Dentre os erros presentes no TIADC, a deriva de relógio é a que impacta mais em sinais que demandam uma precisão maior — dado que as regiões onde o sinal varia rapidamente são mais complexas de reconstruir. Sinais que variam bastante em pequenos intervalos de tempo necessitam de sinais de relógio bem sincronizados. Em comparação com os filtros de resposta ao impulso infinita (IIR), os filtros FIR apresentam maior facilidade para correções da deriva de clock – pois apresentam filtragem de fase linear, então a fase pode ser corrigida independentemente da amplitude do sinal – [3]. A frequências baixas, os filtros de resposta impulso finito possuem baixa resolução, então a boa escolha do filtro depende diretamente da programação que é usada para gerar os coeficientes do filtro.

Propõe-se a utilização de filtros digitais implementados em hardware para corrigir a deriva de relógio dos múltiplos conversores analógico-digitais e otimizar a reconstrução do sinal. Entre os principais aspectos estudados, destacam-se a teoria de reconstrução de [4], o estudo da ferramenta de simulação Simulink do ambiente MATLAB®, a construção de blocos comportamentais dentro do ambiente a partir da linguagem de descrição de hardware VHDL, e as plataformas de síntese e análise de circuitos digitais Genus™ Synthesis Solution e Incisive Enterprise Simulator – da Cadence Design Systems.

O estudo realizado difere em relação ao método utilizado no estado da arte da correção de erros de relógio em TIADCs – [5] e [6] – que utiliza métodos de calibração

e estatística para eliminação da deriva de relógio em tempo real. Em [4] o método de calibração é utilizado em TIADCs com SDR de 60 dB e frequência de 2,7 GHz, mas trabalha com filtros polifase. [6] propõe um algoritmo de calibração baseado em estatística para a correção de erros, mas depende de dados anteriores para ser capaz de gerar os filtros.

## 1.2 Objetivos

### 1.2.1 Objetivos gerais

Avaliar a teoria de reconstrução de sinais não uniformemente amostrados como forma de compensação de erro de *deriva de clock* em *TIADCs* utilizando sinais de entrada de múltiplos tons. Para diferentes valores de *clock skew*, ordem, filtragem e também distorções oriundas da implementação em *hardware* – devido à cálculos em vírgula fixa e de síntese em silício – estudando e otimizando as características de consumo de pós-*layout*.

### 1.2.2 Objetivos específicos

Entre os principais aspectos estudados, destacam-se a teoria de reconstrução de sinais não uniformemente amostrados, o estudo da ferramenta de simulação virtual MATLAB®, o desenvolvimento da plataforma de simulação para o teste dos filtros FIR, aplicação de erros de *offset* e *deriva de clock*, assim como a validação do algoritmo em [3].

Utilizando a plataforma virtual MATLAB® avalia-se o desempenho dos filtros FIR utilizando o sinal de múltiplos tons. Diferentes janelas serão utilizadas para reconstrução do sinal e a estimação do SDR. Diferentes frequências de amostragem também são utilizadas para o teste da eficiência dos filtros.

A última análise realizada para a validação do desempenho dos filtros foi a aplicação de testes com diferentes números de bits nos filtros. Os filtros e o algoritmo da operação de filtragem em vírgula fixa foram confeccionados a partir da *toolbox* do *MATLAB®*: *Fixed-Point Converter*.

Confeccionar e validar o código VHDL através da co-simulação entre o ambiente *Simulink* e *ModelSim* e posteriormente testar a mesma validação dentro do ambiente *Cadence Incisive Enterprise Simulator*.

Os objetivos finais do projeto são a confecção e desenho das portas lógicas do circuito através do *Cadence Genus$^{TM}$ Synthesis Solution* – roteamento das trilhas

e vias através do *Cadence Innovus Implementation System*, validação, correção de DRCs e otimização do circuito através da última ferramenta.

# 2 Revisão da literatura

## 2.1 Amostragem e reconstrução

### 2.1.1 Uniforme

A amostragem de um sinal se resume em obter dados discretos de um sinal contínuo, para que posteriormente possam ser reconstruídos. O sinal contínuo no tempo deve ser discretizado com a relação:

$$x[n] = x_c(nT_s), -\infty < n < \infty \tag{1}$$

A função $x_c$ representa o sinal de entrada contínuo, e $x$ representa o sinal discretizado. $T_s$ é conhecido como *período de amostragem* - logo caracteriza o inverso da frequência de amostragem $f_s$.

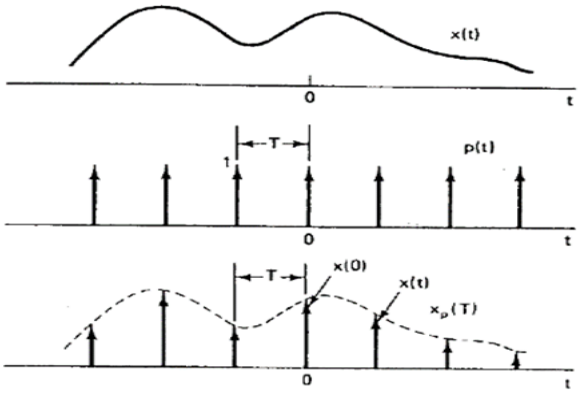


Figura 1: Representação da amostragem de um sinal contínuo.
Fonte: [7]

É essencial analisar a teoria da amostragem de sinais no domínio da frequência. Para que isso seja possível introduz-se a função trem de impulsos $s(t)$:

$$s(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT) \tag{2}$$

Lembrando que $\delta$ representa a função delta de Dirac. Multiplicando a função representante do sinal contínuo ao trem de impulsos, obtém-se o sinal amostrado $x_s$:

$$x_s(t) = x_c(t)s(t) \tag{3}$$

substituindo (2) em (3), tem-se:

$$x_s(t) = x_c(t) \sum_{n=-\infty}^{\infty} \delta(t - nT) \tag{4}$$

discretizando $x_c(t)$, obtém-se:

$$x_s(t) = \sum_{n=-\infty}^{\infty} x_c(nT_s)\delta(t - nT) \tag{5}$$

A transformada de Fourier de uma função trem de impulsos é:

$$S(j\Omega) = \frac{2\pi}{T} \sum_{k=-\infty}^{\infty} \delta(\Omega - k\Omega_s) \tag{6}$$

Onde $\Omega_s$ denota $\frac{2\pi}{T}$, que é a frequência de amostragem contínua em radianos. Aplicando o teorema da convolução:

$$X_s(j\Omega) = \frac{1}{2\pi} X_c(j\Omega) * S(j\Omega) \tag{7}$$

A relação seguinte provê o relacionamento entre a transofmrada de Fourier e a entrada e saída do sinal que intenta-se amostrar:

$$X_s(j\Omega) = \frac{1}{T} \sum_{k=-\infty}^{\infty} X_c(j(\Omega - k\Omega_s)) \tag{8}$$

É importante ressaltar que a frequência de amostragem deve ser duas vezes maior do que a frequência do sinal de onde está sendo feita a amostragem – garantindo a reconstrução mais fidedigna. A este princípio fundamenta-se o *teorema de Nyquist-Shannon* [3] – o teorema é descrito analiticamente abaixo: Considerando que $x_c(t)$ é um sinal limitado em banda, então:

$$X_c(f) = 0, \ para \ |\Omega| \geq \Omega_N \tag{9}$$

*Então, $x_c(t)$ é determinado unicamente por suas amostras $x[n] = x_c(nT), n = 0, 1, 2, \ldots,$ se*

$$\Omega_s = \frac{2\pi}{T} \geq 2\Omega_N \tag{10}$$

A frequência mínima necessária para se fazer a mais verossímil amostragem do sinal, denota-se limite de *Nyquist* - denotada por $\Omega_N$ na equação (8). A figura 2 representa cada uma das funções analisadas no domínio da frequência:

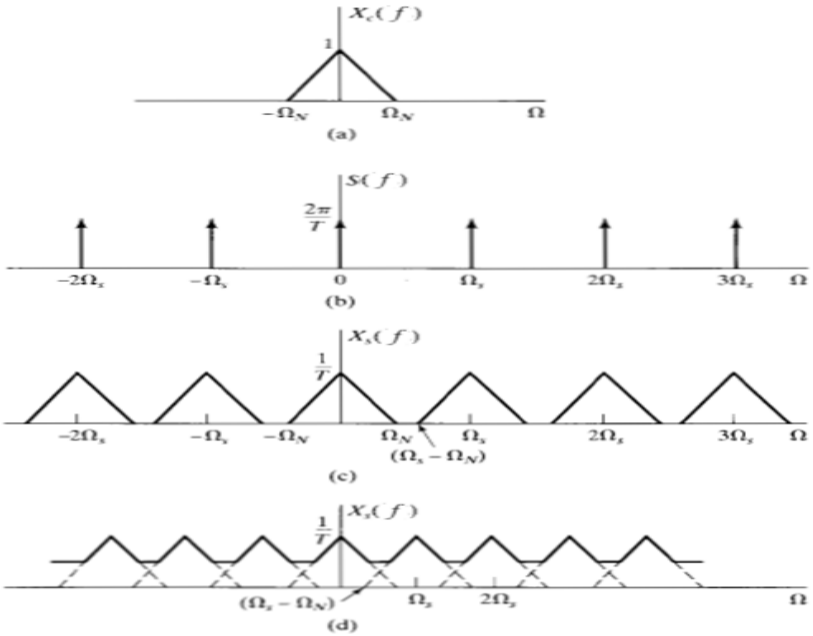


Figura 2: Representação do sinal de entrada com o filtro de banda no domínio da frequência (filtro passa-baixas) (a), trem de impulsos periódico (espectro da função de amostragem) (b), (c) representa a transforada de *Fourier* do sinal amostrado com $\Omega_s > 2\Omega_N$. (d) está expondo o efeito de *aliasing* – um efeito indesejado de sobreposição de sinal que ocorre quando $\Omega_s < 2\Omega_N$.
Fonte: [7]

Para obter o sinal individual para cada conversor desejado, deve-se filtrar o trem de impulsos referido em (c) da figura 2 com um filtro passa-baixas. O filtro de reconstrução ideal é o filtro *sinc*, dado que representa um retângulo no domínio da frequência. O filtro é exposto na subsequente equação:

$$x(t) = \sum_{n=-\infty}^{\infty} x_c(nT_s) sinc(2W(t - nT_s)) \tag{11}$$

Onde $W$ é a largura da banda.

### 2.1.2 Não-Uniforme

A amostragem uniforme é definida quando todos os sub-ADCs do conversor entrelaçado segue amostragem de sinal bem definidas e espaçadas segundo [8]. A amostragem não-uniforme define-se quando as fases de *clock* de cada sub-ADC são assíncronas – a reconstrução do sinal dependerá da multiplexação feita na saída de cada sub-conversor. O domínio analítico da amostragem não-uniforme é corroborado pela interpolação de Lagrange – a qual afirma que a interpolação pelo filtro *sinc* pode

ser aproximada por um filtro passa-baixas, quando os conversores analógico-digitais estão uniformemente espaçados – [4]. É postulado que o sinal pode ser adequadamente amostrado pelo somatório do sinal contínuo multiplicado por uma função de Lagrange:

$$x(t) = \sum_{n=-\infty}^{\infty} x_c(t_n) \frac{G(t)}{G'(t_n)(t - t_n)} \tag{12}$$

onde a função $G$, denota:

$$G(t) = (t - t_0) \prod_{\substack{k=-\infty \\ k\neq 0}}^{\infty} \left(1 - \frac{t}{t_k}\right) \tag{13}$$

A interpolação utilizando (12) é chamada interpolação de Lagrange. A transformada de Fourier de $\frac{G(t)}{G'(t_n)(t-t_n)} = l_n(t)$, é limitada em banda e forma uma sequência biortogonal a $e^{ft_n}$ entre $[\frac{-\pi}{T_N}, \frac{\pi}{T_N}]$, dado que $|t_n - nT_N| \leq d < \frac{T_N}{4}, para\ todo\ n \in Z$ - [4].

A amostragem periodicamente não uniforme pode ser vista como uma combinação de $N$ sequências de amostras uniformes Logo,"*nessa forma de amostragem, os pontos de amostragem são divididos em grupos de N pontos cada. Os grupos tem um período recorrente, que é denotado por T, que é igual a N vezes o período de Nyquist [1]*".(13) pode ser sumarizado como:

$$G(t) = t \prod_{\substack{n=-\infty \\ n\neq 0}}^{\infty} \left(1 - \frac{t}{nT}\right) \prod_{n=-\infty}^{\infty} \left(1 - \frac{t}{nT + t_1}\right) ... \prod_{n=-\infty}^{\infty} \left(1 - \frac{t}{nT + t_{M-1}}\right) \tag{14}$$

Cada um dos produtórios convergem em uma função de senos correspondentes a $sin\left(\frac{\pi(t-t_i)}{T}\right)$, as quais podem ser provadas a partir das relações:

$$sin\left(\frac{\pi t}{t}\right) = kt \prod_{\substack{n=-\infty \\ n\neq 0}}^{\infty} \left(1 - \frac{t}{nT}\right) \tag{15}$$

$$sin\left(\frac{\pi(t - t_i)}{t}\right) = k(t - t_i) \prod_{\substack{n=-\infty \\ n\neq 0}}^{\infty} \left(1 - \frac{t - t_i}{nT}\right) \tag{16}$$

substituindo (15) e (16) na série de produtórios, obtém-se:

$$G(t) = c \prod_{i=0}^{M-1} sin\left(\frac{\pi(t - t_i)}{T}\right) \tag{17}$$

$c$ é uma constante. Derivando a expressão anterior e avaliando-a em $t = t_i + nT$, tem-se:

$$G'(t_i + nT) = c\frac{\pi}{T}(-1)^{nM} \prod_{\substack{q=-\infty \\ q\neq 0}}^{\infty} sin\left(\frac{\pi(t_i - t_q)}{T}\right) \tag{18}$$

Relacionando (18) e (17) e utilizando a postulação de que para a amostragem não-uniforme;

$$t_i + nT, i = 0, 1, ..., M-1, n \in (-\infty, \infty) \tag{19}$$

Obtém-se:

$$x_c(t) = \sum_{n=-\infty}^{\infty} \sum_{i=-\infty}^{M-1} x_c(nT + t_i) \cdot \frac{a_i(-1)^{nM} \prod_{i=0}^{M-1} sin\left(\frac{\pi(t-t_i)}{T}\right)}{\frac{\pi(t-nT-t_i)}{T}} \tag{20}$$

Onde o coeficiente $a_i$ é:

$$a_i = \frac{1}{\prod_{\substack{i=-\infty \\ q\neq i}}^{\infty} sin\left(\frac{\pi(t_i-t_q)}{T}\right)} \tag{21}$$

### 2.1.3 Implementação de filtros

Para converter o filtro de tempo contínuo para um filtro discreto aplicando a identidade de interpolação, é necessário trocar as ordens dos somatórios na equação da reconstrução:

$$f_i(t) = \sum_{k=-\infty}^{\infty} x_c(kT + t_i) \cdot \frac{a_i(-1)^{kM} \prod_{q=0}^{M-1} sin\left(\frac{\pi(t_i-t_q)}{T}\right)}{\frac{\pi(t-kT-t_i)}{T}} \tag{22}$$

reescrevendo a equação anterior como uma convolução e usando a relação: $sin(t - k\pi = (-1)^k sin(t)$:

$$f_i = s_i(t) * h_i(t) \tag{23}$$

sendo:

$$h_i(t) = a_i T \cdot \frac{sin\left(\frac{\pi t}{T}\right)}{\pi t} \prod_{\substack{q=0 \\ q\neq i}}^{M-1} sin\left(\frac{\pi(t + t_i - t_q)}{T}\right) \tag{24}$$

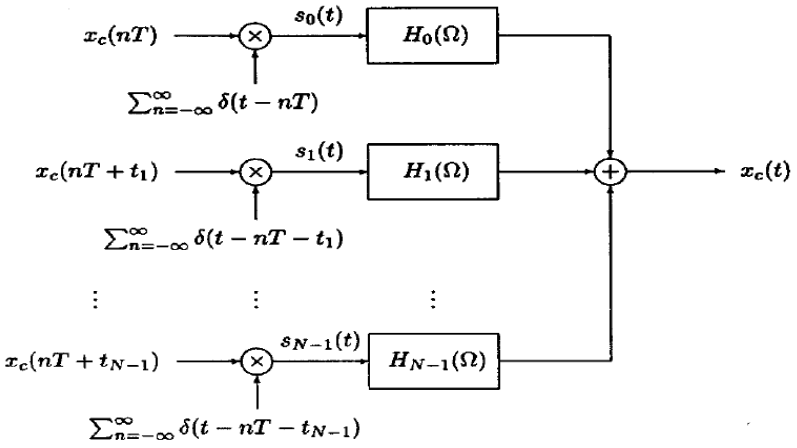


Figura 3: Reconstrução de amostras não-uniformes recorrentes usando um banco de filtros – [4].

e $s_p(t)$ é o trem de impulso das amostras de entrada:

$$s_i(t) = \sum_{k=-\infty}^{\infty} x_c(kT + t_i)\delta(t - kT - t_i) \tag{25}$$

que pode ser escrita como a soma de $M$ convoluções:

$$x_c(t) = \sum_{i=0}^{M-1} s_i(t) * h_i(t) \tag{26}$$

A última equação denota que as amostras são divididas em $M$ subsequências - uma de cada até o enésimo da taxa de *Nyquist* da variação de tempo modificada de $x_c(t)$. As $M$ convoluções, que representam as $M$ filtragens em paralelo, são responsáveis pela reconstrução de sinal.

## 2.2 Conversor analógico-digital entrelaçado

A fundamental propriedade do *ADC* (*Analog-to-Digital Converter*) entrelaçado é a de utilizar confluência de dados entre um número M de conversores para aumentar a velocidade do dispositivo – esses sub-conversores operam em diferentes fases de relógio, configurando uma multiplexação temporal dos sinais digitais vindos de cada ADC individual (A velocidade total de ganho em dados transferidos é diretamente proporcional ao número M de conversores utilizados – todavia, os múltiplos conversores prejudicam a coerência do sinal diminuindo a eficiência da conversão analógico-digital. Ao decorrer do texto serão explanados os problemas que podem afetar o desempenho,

como erros de *offset*, ganho e *clock skew*, chegando aos sub-conversores – assim como métodos que viabilizam reduzir a influência de distorções.

A propriedade de intercalação na operação dos $M$ conversores deve ser tal que os sinais de operação de *clock* para cada $ADC$ devem ser defasados em um ângulo:

$$\Phi = \frac{2\pi}{M} m, m = 1, 2, ..., M - 1 \tag{27}$$

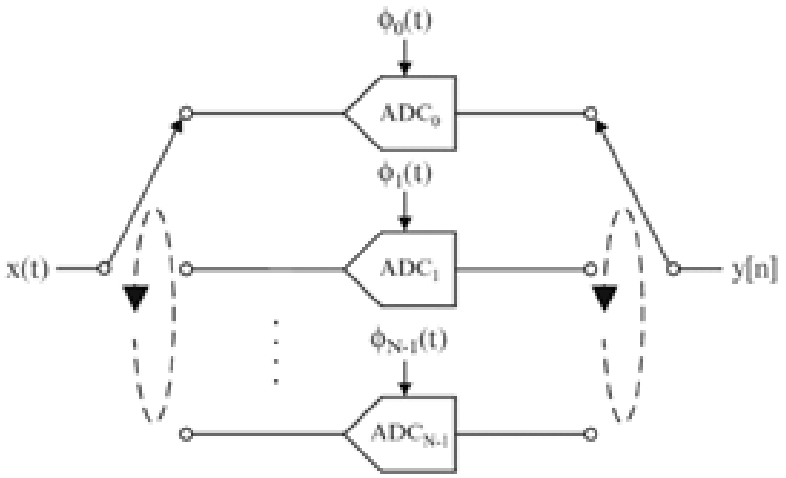


Figura 4: Diagrama representando o funcionamento do TIADC.
Fonte: [8]

Como o conversor final arrecada o sinal vindo de todos os seus subcomponentes, a expressão que mostra o sinal de saída é:

$$y[n] = \sum_{i=0}^{M-1} y_{i[n]} \tag{28}$$

Onde $M$ é a quantidade de sub-ADCs presentes. A expressão (28) simplesmente denota que o sinal final final vai depender do sinal de entrada, como se observa subsequentemente:

$$y[n] = x(nT_s) \tag{29}$$

$T_s$ é o período de amostragem do sinal. A análise do ADC entrelaçado no domínio da frequência apresenta conclusões mais práticas, para análise subsequente, os fenômenos descritos serão expostos utilizando como base a transformada discreta de Fourier:

$$X(f) = \sum_{n=-\infty}^{\infty} x[n] e^{-j2\pi fn} \tag{30}$$

Lembrando que $X(f)$ é uma função periódica de período 1. Como a amostragem do sinal é feita em tempo discreto, a expressão (30) pode ser multiplicada por um delta de Dirac:

$$\delta_i[n] = \sum_{k=-\infty}^{\infty} \delta[n - kM - i] \tag{31}$$

$$y_i[n] = x(nT_s)\delta_i[n] \tag{32}$$

A relação precedente expõe que os conversores vão funcionar selecionando as suas respectivas amostras com ação do delta de Dirac, a partir dessa função selecionam-se as faixas em que cada conversor vai retirar as suas amostras. O sinal de entrada vai ser repartido dependendo do período da taxa efetiva de amostragem $T_s$.

Pela propriedade fundamental da convolução da transformada de Fourier , $Y_i(f)$ é obtida através da convolução das transformadas das funções $\delta_i[n]$ e $x[n]$ - $D_i(f)$ e $X(f)$, respectivamente:

$$Y_i(f) = X(f) * D_i(f) = \frac{1}{M} \sum_{k=-\infty}^{\infty} e^{-j\frac{2\pi k}{M}i} \cdot X\left(f - \frac{k}{M}\right) \tag{33}$$

Um multiplexador é localizado na terminação de cada um dos sub-ADCs para que seja possível reconstruir o sinal a partir da soma dos sinais enviados por cada um deles:

$$Y(f) = \sum_{i=0}^{M-1} Y_i(f) \tag{34}$$

$Y_i(f)$ representa o sina de saída de cada respectivo ADC.
Utilizando o sinal enviado por cada sub-ADC de (34), a expressão torna-se:

$$Y(f) = \frac{1}{M} \sum_{k=-\infty}^{\infty} \sum_{i=0}^{M-1} e^{-j\frac{2\pi k}{M}i} \cdot X\left(f - \frac{k}{M}\right) \tag{35}$$

esta expressão reduz-se a:

$$Y(f) = \sum_{k=-\infty}^{\infty} X(f - k) \tag{36}$$

Para $\frac{k}{M}$ inteiro - e zero para qualquer outro valor. Realizada a transformada inversa, verifica-se a reconstrução do sinal x[n].

### 2.2.1 Apresentação de erros

2.2.1.1 Offset e ganho Cada ADC possui um erro inerente e estático, os $M$ conversores podem somar estas características indesejadas e afetar bastante o desempenho do conversor entrelaçado. O erro pode ser adicionado a (36) como um outro somatório representando o deslocamento do sinal:

$$Y(f) = \sum_{k=-\infty}^{\infty} X(f-k) + \sum_{i=0}^{M-1} O_i(f) \tag{37}$$

Os erros representados por $O_i$ independem do sinal de entrada, surgem apenas devido aos *offsets* inerentes às não idealidades dos próprios conversores analógico-digitais menores.

O desvio de ganho pode ser representado analiticamente pela relação:

$$Y(f) = \frac{1}{M} \sum_{k=-\infty}^{\infty} \sum_{i=0}^{M-1} G_i \cdot e^{-j\frac{2\pi k}{M}i} \cdot X\left(f - \frac{k}{M}\right) \tag{38}$$

onde $G_i$ representa o erro de ganho de cada conversor. Vale notar que se $G_i$ for 1, então a função de transferência se comportará como no modelo ideal, considerando que não existam erros de offset entre os conversores.

2.2.1.2 Deriva de *clock (clock skew)* Devida a própria natureza do funcionamento do TIADC – pequenos erros no instante de aquisição do sinal analógico devido a pulsos de clock alocados podem distorcer bastante o sinal que se objetiva reproduzir. A equação que descreve o comportamento do sinal de saída considerando irregularidades no sinal de clock podem ser observadas a seguir:

$$Y(f) = \frac{1}{M} \sum_{k=-\infty}^{\infty} \sum_{i=0}^{M-1} e^{-j\frac{2\pi k}{M}\tau_i} \cdot e^{-j\frac{2\pi k}{M}i} \cdot X\left(f - \frac{k}{M}\right) \tag{39}$$

Onde $\tau_i$ corresponde ao tempo de atraso ou adiantamento tomando como referência o tempo para o funcionamento ideal.

A figura seguinte mostra como o este tipo de erro influencia a amostragem do sistema:

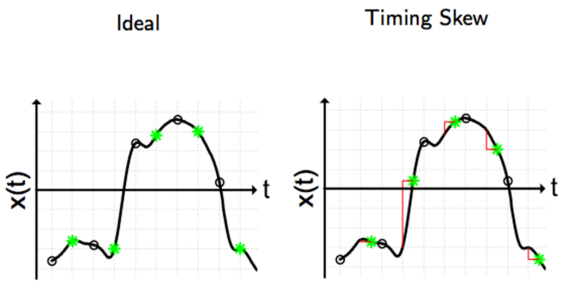


Figura 5: Representação do desvio (b) ao sinal ideal (a) devido a derivação de clock. [8]

## 2.3 Filtragem digital

Para análise desta seção supomos que todos os sinais têm energia finita e estão limitados em banda a largura $W$. Destaca-se também que $T_Q$ é o período de Nyquist-dado por $\frac{\pi}{W}$. Assume-se também que a função $h_1(t)$ para $l = 0, 1, \ldots, M-1$ representa as respostas aos impulsos dos filtros de tempo contínuo com as correspondentes respostas em frequência sendo $H_l(\Omega)$ para l variante entre o mesmo intervalo e limitado em banda à W. Para qualquer M e T tais que $\frac{T}{T_Q} - \frac{1}{M} = k$ para algum inteiro k, então a identidade de interpolação será:

$$H(f) = \frac{M}{T_Q} \sum_{l=0}^{M-1} \left( H_l \left( \frac{Mf}{T_Q} - \frac{2\pi l}{T_Q} \right) + H_l \left( \frac{(Mf - 2\pi(l-M))}{T_Q} \right) \right), \; para |f| \leq \pi \tag{40}$$

### 2.3.1 Design do banco de filtros para compensação de erros de relógio

Na reconstrução no domínio contínuo, cada sinal $s_i(t)$ está atrasado no tempo – denotando que impulsos de Dirac responsáveis pela amostragem estão deslocados no tempo. Cada filtro de reconstrução $H_i(\Omega)$ compensa o erro de relógio com o produto de senos apresentado em (24). Na reconstrução baseada no tempo discreto, as amostras $x_i[n]$ são armazenadas em paralelo, onde o sinal atrasado é amostrado – mas não deslocado no tempo – em relação a variável de tempo discreto "$n$"(Fig. 6):

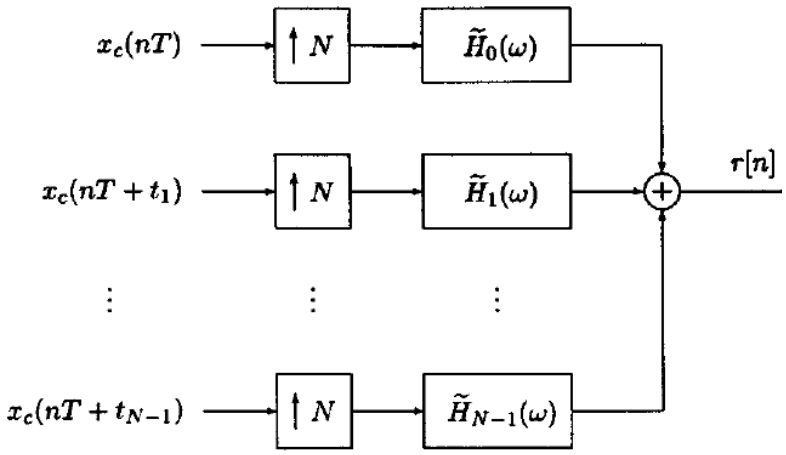


Figura 6: Interpolação usando um banco de filtros de tempo discreto. – [1].

$$x_i[n] = \sum_n x_c(nT + t_i). \tag{41}$$

Os sinais aumentam a frequência de amostragem até o valor da frequência efetiva do conversor analógico-digital $f_{TIADC} = N/T$,

$$\tilde{x}_i[n] = \begin{cases} x_i[n/N] & n = \pm 0, \pm N, \pm 2N, ... \\ 0 & \text{caso contrário,} \end{cases} \tag{42}$$

filtrado pelos filtros de interpolação $\tilde{h}_i[n]$ e então combinados na saída,

$$r[n] = \sum_{i=0}^{N-1} \tilde{x}_i[n] * \tilde{h}_i[n] \tag{43}$$

onde $r[n]$ é o sinal compensado efetivo, que é o sinal amostrado uniformemente em $f_{TIADC}$. O atraso observado na figura 3 é não inteiro em relação à variável de tempo discreto "$n$". Pode-se observar que os atrasos não inteiros são então integrados no filtro de reconstrução:

$$\tilde{H}_i(\omega) = \frac{N}{T} H_i \left( \frac{N\Omega}{T} \right) e^{-jt_i N\Omega/T}. \tag{44}$$

O filtro de interpolação de resposta ao impulso é obtido por atrasar $h_i(t)$ por $t_i$ e amostrar $h_i(t - t_i)$ em $f_{TIADC}$:

$$\tilde{h}_i[n] = \sum_n h_i \left( \frac{nT}{N} - t_i \right) \tag{45}$$

### 2.3.2 FIR/Janelamento

Filtros *FIR (Finite impulse response)* são – majoritariamente – implementados com o método de janelamento; o qual consiste em truncar a resposta de um filtro de tempo contínuo ideal, afinal este filtro é muito longo e a parte que será janelada pode desempenhar o filtro de maneira satisfatória.

O filtro deve ser amostrado nos instantes:

$$h_i[n] = h_i(t - nT_Q - t_i),\ n \in (-\infty, \infty) \tag{46}$$

A maneira mais simples de obter um filtro FIR causal de um sistema com desejada resposta impulso - com auxílio de "$h_i[n]$"; lembrando que esta representa um filtro passa-baixas, devido a presença da função *sinc*; é truncar a resposta usando uma janela quadrada, o que resulta em:

$$f(n) = \begin{cases} h_i[n], & 0 \leq n < M \\ 0, & \text{caso contrário} \end{cases} \tag{47}$$

Onde M é o comprimento desejado para o filtro resultante. Em forma geral $h[n]$ é representado como o produto de uma resposta de impulso desejada e uma janela de duração finita $w[n]$:

$$h[n] = h_i[n].w[n] \tag{48}$$

O truncamento no domínio da frequência denota a convolução do espectro da janela com a desejada resposta em frequência ideal. Diferentes janelas impactam na resposta em frequência do filtro de maneiras diferentes. O que caracteriza os diferentes *tipos de janelas são os respectivos tempos de decaimento*. Uma janela de decaimento lento tempo um menor número de lóbulos secundários e uma frequência de decaimento veloz. A escolha da janela é influenciada pelo tipo de sinal de entrada que se trabalha. Abaixo encontram-se os tipos de janelas que foram usadas nesse trabalho:

*Retangular*

$$f(n) = \begin{cases} 1, & 0 \leq n \leq M \\ 0, & \text{caso contrário} \end{cases} \tag{49}$$

*Blackman*

$$f(n) = \begin{cases} 0,42 - 0,5cos\left(\frac{2\pi n}{M}\right) + 0,08cos\left(\frac{4\pi n}{M}\right), & 0 \leq n \leq M \\ 0, & \text{caso contrário} \end{cases} \tag{50}$$

## 2.4 Aritmética de vírgula fixa

A representação de números em vírgula fixa permite operações de números fracionários utilizando *hardware* de baixo custo e é fundamental para aplicação de filtros digitais em FPGAs e/ou ASICs. Para abaixar o custo da implementação, processadores de sinal digital são confeccionados para realizar operações aritméticas apenas com números inteiros. Para representar números fracionários nesses processadores, é possível usar uma "vírgula"binária implicada. Um número de vírgula fixa tem um número específico de dígitos reservados para parte inteira e outra para a fracionária. Independentemente do módulo do número, ele sempre será representado com o mesmo número de dígitos para cada porção. As operações aritméticas do dispositivo sempre terão um conhecimento inato de onde o ponto decimal está, quando executadas. A operação com números negativos precisa ser especificada no tratamento dos mesmos pelo próprio *designer*. A utilização do número negativo requer a eliminação de um bit de precisão para indicar se um número considera o sinal do dado tratado. Enquanto os números positivos são convertidos aos inteiros convencionalmente, os números negativos necessitam da utilização da técnica de complemento de dois para conversão.

## 2.5 Linguagem de descrição de *hardware*

As linguagens de descrição de hardware são utilizadas para o projeto de circuitos integrados digitais para aplicações específicas. Utilizando *softwares* de confecção de *layout*, os circuitos integrados são então fabricados utilizando técnicas especiais de corte a laser e tratamento químico – [9].

As linguagens de descrição de *hardware* utilizadas são: VHDL (*Very-High-Speed Integrated Circuit Hardware Description Language*) e verilog.

# 3 Metodologia

Além do material bibliográfico apresentado - são utilizadas as plataformas virtuais para realização das simulações:

- *MATLAB®*;
- *Simulink*;
- *ModelSim*;
- *Cadence Incisive Enterprise Simulator*;
- *Cadence Genus™ Synthesis Solution* e
- *Cadence Innovus Implementation System.*

A partir destas desenvolveram-se os modelos matemáticos do comportamento do *TI-ADC* na implementação em silício física.

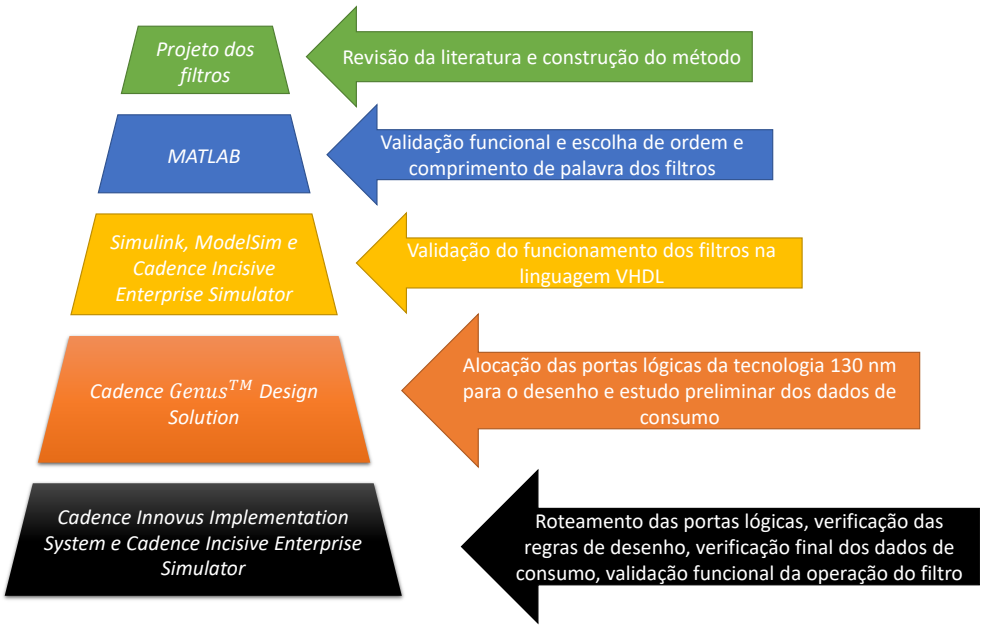


Figura 7: Diagrama representando as atribuições de cada simulador.

O fluxo de projeto tem uma estrutura *top-down* – onde analisa-se a técnica de reconstrução não-uniforme considerando apenas os erros de deriva de *clock* para diferentes características do filtro – etapa feita inteiramente no MATLAB®. O código então é traduzido para operar com a aritmética de vírgula fixa – etapa feita através

do MATLAB® e *Simulink*. O código em vírgula fixa é então traduzido para linguagem VHDL que de fato pode ser usada para alocação de portas lógicas – etapa entre realizada através doo *Simulink* e *ModelSim*. O código validado é então passado para análise dentro dos simuladores da *Cadence Design Systems*. O código é analisado sob o *Cadence Incisive Enterprise Simulator* para verificar a compatibilidade de resultados entre o *ModelSim* e o simulador de linguagem de descrição de *hardware* da Cadence. Após a verificação, o código VHDL é usado para criação do desenho das portas lógicas através da *Cadence Genus*$^{TM}$ *Synthesis Solution*, o circuito digital então pode ter as suas características de atraso, potência e área estimadas. O simulador final – *Cadence Innovus Implementation System* faz o roteamento de potência e lógico das portas lógicas com trilhas de diferentes metais para análise completa do circuito digital – verificando as regras de desenho e levando em conta as características parasitas que cada um dos elementos reais traz para os resultados. O circuito então é submetido ao *Cadence Incisive Enterprise Simulator* para validação funcional e extração dos dados de operação do circuito mais próximo do real que as ferramentas podem representar.

## 3.1 NMSE (*Normalised Mean Square Error*)

A métrica de qualidade utilizada em todos os ambientes de simulação é a NMSE. Esta métrica considera as amplitudes dos sinais envolvidos com e sem a influência dos erros de deriva de relógio.

Para o cálculo do referido parâmetro a seguinte equação é utilizada:

$$NMSE = 10 \cdot log_{10} \left( \frac{\sum_n \left( y[n] - y'[n] \right)^2}{\sum_n y[n]^2} \right) \tag{51}$$

Onde (52) ,

$$\sum_n y[n]^2 \tag{52}$$

é construída a partir da amostragem referente ao sinal sem erro.

$$\sum_n \left( y[n] - y'[n] \right)^2 \tag{53}$$

(53) corresponde a amplitude do sinal correspondente ao sinal amostrado com erro. A relação sinal-distorção (SDR) apresentada no texto é o inverso do NMSE:

$$SDR = \frac{1}{NMSE} \tag{54}$$

### 3.1.1 Janelamento e impacto na reconstrução do sinal

A técnica de janelamento foi recatada na seção 2.3 da revisão da literatura. Todavia quando o período do sinal amostrado não tem um número inteiro de períodos, a truncagem levará a reconstrução imperfeita do sinal – dado que existirá uma descontinuidade entre o final de uma aquisição e o início da próxima. Corroborou-se que a janela que apresenta o maior desempenho quando aplicado junto ao filtro é a janela retangular.

## 3.2 Metodologia de teste do filtro digital no MATLAB

Os filtros discretos são construídos a partir da fundamentação teórica encontrada na seção 2.1.2. Erros de deriva de relógio são artificialmente induzidos no sinal criado para testar o funcionamento dos filtros segundo a teoria. A amostragem não uniforme é realizada a partir do sinal proveniente. Para cada subportadora são calculados os coeficientes que irão ser gravados no circuito digital segundo a equação 21. A truncagem dos coeficientes é realizada após isso. Todas as janelas listadas na fundamentação teórica são testadas para verificação de qual imbui o maior desempenho na operação dos filtros discretos. O sinal de cada subportadora é convoluído no tempo com os filtros digitais. O sinal pode ser reconstruído a partir dessas configurações e então comparado com o original a partir da métrica da SDR.

A *toolbox fixed-point converter* é utilizada para conversão da simulação já desenvolvida para o teste do método do janelamento, indução de erros e teste de desempenho de diferentes fatores de ordem e magnitude de erro para uma simulação análoga executada com a aritmética de vírgula fixa. Vale notar que como o sinal compreende amplitudes negativas, um bit do comprimento de palavra é reservado para indicação de uma variável *signed* na aritmética de vírgula fixa. Utilizando esta *toolbox*, funções que realizavam cálculos em vírgula fixa para o *comprimento de palavra* entre 4 e 16 bits foi criado. Os diferentes *comprimentos de palavra* foram testados e concluiu-se que o *comprimento de palavra* que garante um desempenho sem perdas significativas teria 8 bits na parte inteira e 8 bits na parte fracionária.

## 3.3 Metodologia de teste do filtro digital no *Simulink* e *ModelSim*

Os testes realizados para validação funcional e de desempenho dos filtros em *hardware* foram simulados a partir do *Simulink* – uma ferramenta adicional do MATLAB utilizada para compilação de partes de código em blocos – e *ModelSim* ferramenta da *Mentor Graphics* utilizada para monitoramento dos diferentes tipos de variáveis presentes no circuito lógico construído pelo código VHDL.

A ferramenta *Simulink* possui *toolboxes* específicas para conversão do código escrito através do MATLAB para linguagem VHDL. A *toolbox* utilizada para operação foi a *DSP System Toolbox HDL Support*. A topologia do circuito dentro do ambiente é apresentado na figura 8.

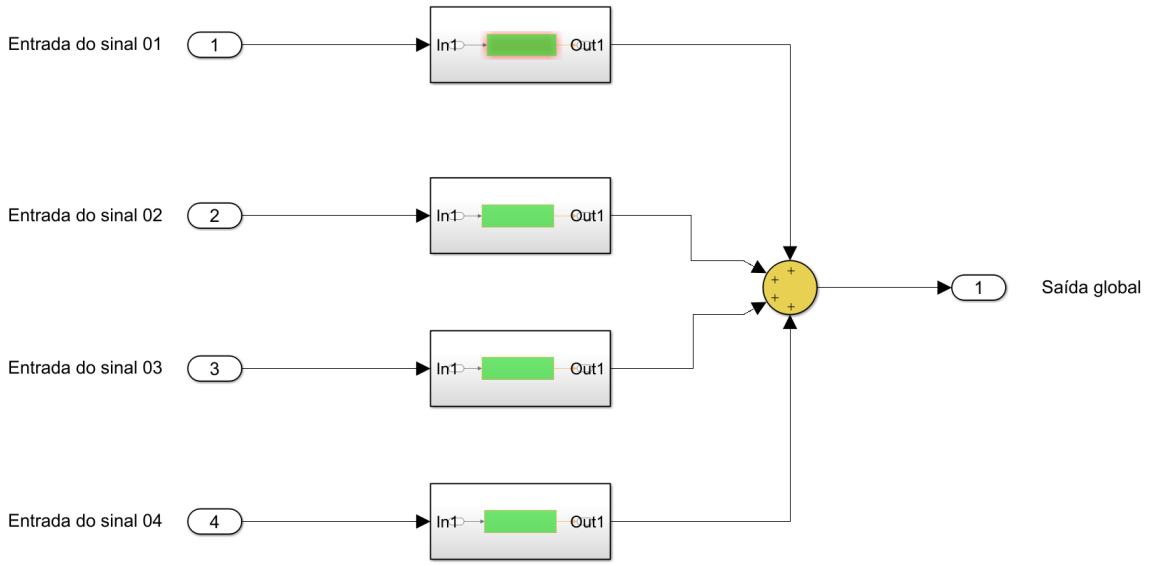


Figura 8: Estrutura de dados de cada filtro individual implementado no *Simulink*.

Cada bloco de filtragem possui um componente *Discrete FIR filter* – no qual pode-se atribuir os valores dos coeficientes calculados a partir da equação 21. Os valores específicos de cada coeficiente são inseridos em cada um dos filtros distintos de acordo com os valores obtidos na etapa anterior.

O *Simulink* também utiliza os recursos de avaliação de consumo lógico do simulador *Xilinx Zynq Support from HDL Coder* da *Xilinx*. Os relatórios desse consumo são listados no apêndice A.

A co-simulação entre os dois simuladores funciona da seguinte forma: utilizando a *toolbox DSP System Toolbox HDL Support* o código em vírgula fixa sintetizado pelo *fixed-point converter* do MATLAB é convertido para linguagem VHDL respeitando características de comprimento de palavra e reserva do primeiro bit para sinalização de uma variável *signed*. Uma *testbench* customizada para o código gerado específico é também criada para validação do código pela co-simulação. O *Simulink* estabelece uma conexão com o simulador *ModelSim*. Este recebe os arquivos VHDL e a *testbench* criada para co-simulação. O *Simulink* testa o filtro digital a partir da aritmética de vírgula fixa implementada pelo sistema do MATLAB, enquanto o *ModelSim* testa os mesmos parâmetros com os arquivos VHDL e a *testbench*. Nota-se que o *ModelSim* deve ser configurado para exportar os dados em forma decimal dado que no teste dos sinais são realizadas apenas operações binárias – como pode ser visto na figura 9.

Aplica-se a mesma entrada nos dois simuladores e se constatadas saída idênticas

Figura 9: Forma de simulação implementada através do simulador *ModelSim* para o teste do filtro digital.

– valida-se o funcionamento do filtro digital em linguagem de descrição de *hardware*.

## 3.4 Metodologia de teste do filtro digital no *Cadence Genus*$^{TM}$ *Synthesis Solution*

Com o filtro digital validado em termos funcionais pelos simuladores anteriores, concentra-se na etapa de alocação das portas lógicas da tecnologia CMOS 130 nm para construção do circuito digital. A síntese lógica executada por este simulador produz dois tipos de arquivos:

- Uma *netlist* Verilog que representa o circuito lógico gerado pelo *software* a partir da tecnologia CMOS 130 nm.
- Um arquivo SDF (*Standard Delay Format*): que contêm todos os atrasos de propagação da *netlist*.

O simulador analisa características de consumo de potência, área e os atrasos não cobertos nas analises dos outros simuladores.

Após a síntese o circuito é analisado com o simulador de HDL da própria *Cadence Design Systems*: o *Cadence Incisive Enterprise Simulator*. Esta verificação se faz necessária para assegurar que o circuito digital gerado ainda é capaz de compensar os erros de relógio sem grandes perdas de desempenho dado que demais atrasos são inseridos na operação do circuito. Este simulador funciona de forma análoga ao ModelSim – logo a plataforma de validação desenvolvida na segunda etapa do projeto pode ser reaproveitada para verificação dos resultados exibidos por esse simulador. Os relatórios entregues pelo simulador estão dispostos nos apêndices B, C e D.

## 3.5 Metodologia de teste do filtro digital no *Cadence Innovus Implementation System*

A síntese final é realizada pela ferramenta *Cadence Innovus Implementation System*. Este simulador faz o roteamento das trilhas e vias presentes no circuito digital para que seja possível gerar o desenho do *layout* completo para que seja enviado para fabricação física. O simulador, além de fazer o papel de *place  route* do circuito final – também verifica *Design Rule Checks* (DRCs) para certificar-se que o circuito seja veemenetemente fabricável.

A verificação de atrasos das trilhas e portas lógicas, dissipação de potência de elementos parasitas e área total são informados em forma de relatório ao final da síntese.

Ao final desta etapa, verifica-se o funcionamento final do circuito digital após o desenho do *layout* com o simulador *Cadence Incisive Enterprise Simulator* novamente. Após a extração de dados, verifica-se a SDR do sinal reconstruído com o circuito digital implementado. Os relatórios entregues pelo simulador estão apresentados nos apêndices E e F.

## 3.6 Confecção dos sinais

### 3.6.1 Confecção e teste do sinal de múltiplos tons

#### 3.6.1.1 Síntese do sinal de múltiplos tons

O primeiro sinal é um sinal de múltiplos tons descrito pela equação 55:

$$f(t) = sin(2\pi fc\,t + 2\pi\phi_1 - \pi) + cos(2\pi fc2\,t + 2\pi\phi_2 - \pi) + cos(2\pi fc3\,t + 2\pi\phi_3 - \pi) \quad (55)$$

Sendo $fc$, $fc2$, $fc3$, as diferentes frequências do sinal de múltiplos tons. Essas frequências são normalizadas em relação a frequência de amostragem efetiva; são múltiplas de 2, 3 e 4 – respectivamente. Como são usados 4 conversores, a frequência de amostragem efetiva é 4 vezes 150 MHz – frequência de cada conversor analógico digital individual. As variáveis $\phi_1, \phi_2\ e\ \phi_3$ são fases aleatórias compreendidas entre 0 e 1.

#### 3.6.1.2 Teste do sinal de múltiplos tons

A reconstrução do sinal a partir da amostragem discreta inerentemente leva desvios do sinal original em pontos onde a derivada temporal do sinal é expressiva – essa é significativamente maior quando existe a deriva de *clock* que objetiva-se retificar. Os erros e sugestões de correção são encontrados na seção 2.2.1. No código de simulação do comportamento do circuito os erros de

*offset* e deriva de *clock* são manufaturados para testar o funcionamento do filtro *FIR* para todos os ambientes de simulação. Os parâmetros ordem e tamanho da palavra binária são simulados para diferentes frequências e magnitudes de erro para o teste de desempenho dos filtros digitais projetados a partir do *Simulink*, *MATLAB* e *ModelSim* para o teste na linguagem VHDL. Após a verificação dos valores que mantêm o melhor compromisso entre desempenho e consumo – o sinal é usado para o teste dos filtros nos simuladores de componentes reais da *Cadence Design Systems*.

# 4 Resultados

## 4.1 Avaliação do filtro digital a partir do MATLAB

A etapa de avaliação dos filtros digitais a partir do MATLAB avalia sob quais valores de comprimento de palavra, janela e ordem garante-se o melhor desempenho para um consumo aceitável de recursos.

### 4.1.1 SDR em função da ordem do filtro digital

A figura a seguir mostra a relação de sinal-distorção em função da ordem do filtro digital para diferentes magnitudes de deriva de *clock*.

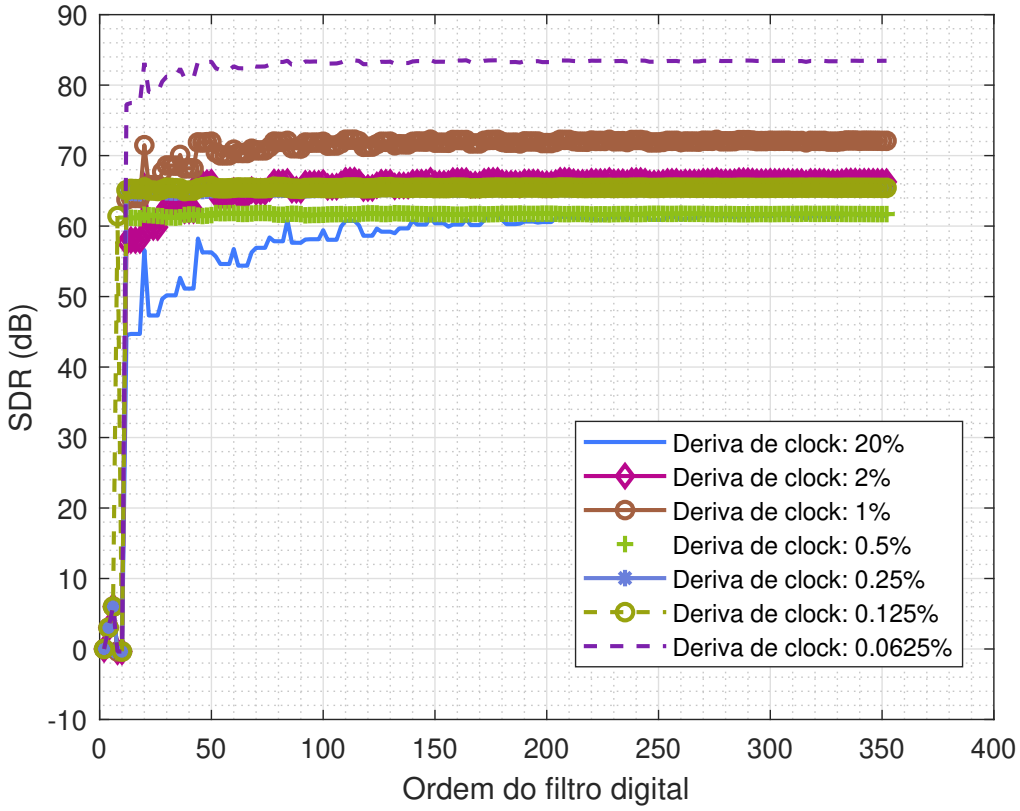


Figura 10: SDR em função da ordem do filtro digital para diferentes magnitudes de deriva de *clock*.

Os erros de deriva de relógio encontrados em dispositivos reais são de no máximo 0.005% do valor ideal. Todos os testes executados a partir desta etapa seguem com o erro de relógio estabelecido para este valor para a validação executada pelos demais simuladores. No entanto, garante-se a operação em saturação de qualidade para valores de ordem superiores a 50. Logo, a ordem utilizada para as simulações posteriores para o filtro digital será 50.

### 4.1.2 SDR em função da frequência para diferentes comprimentos de palavra

Testa-se o valor da SDR para frequências que se aproximam do limite de *Nyquist*. A SDR é analisada também para diferentes comprimentos de palavra que compreendem o valor inteiro e fracionário que será usado para os cálculos em vírgula fixa:

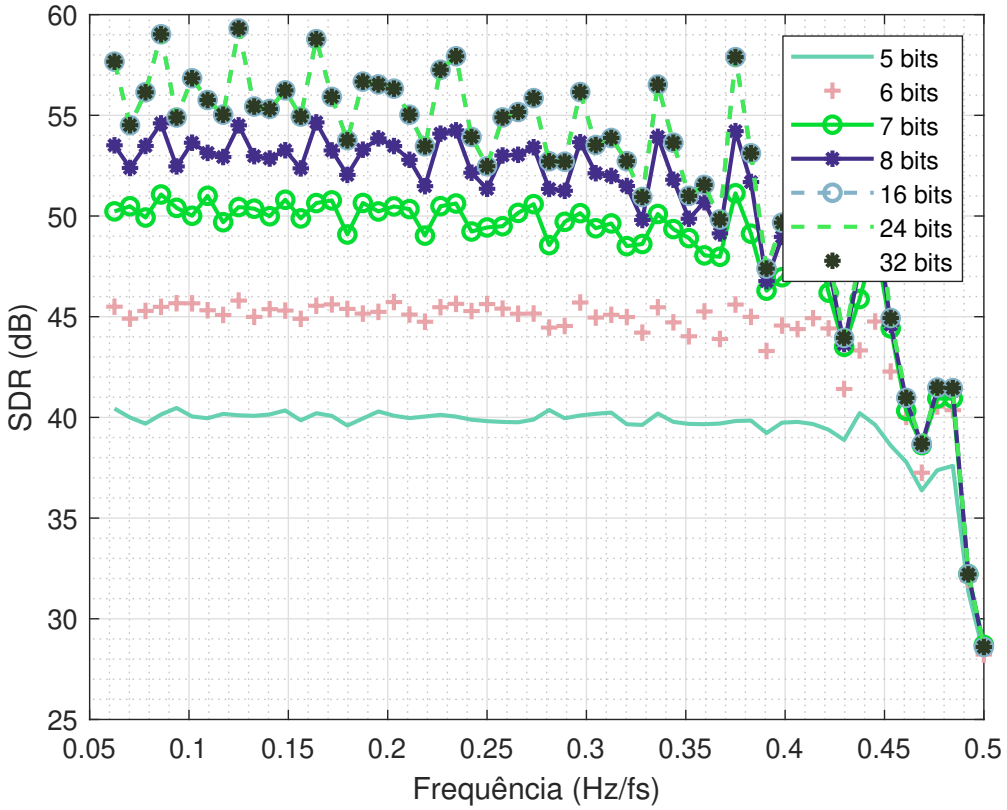


Figura 11: SDR em função da frequência pela frequência de amostragem efetiva para diferentes comprimentos de palavra.

A partir do gráfico anterior, observa-se que a SDR satura para valores iguais ou superiores a 16 bits. Define-se este valor para as simulações de vírgula fixa e VHDL executados nos simuladores posteriores.

### 4.1.3 Avaliação do janelamento

Todas as janelas dispostas na revisão bibliográfica foram testadas para avaliação do SDR a partir de funções internas ao MATLAB, as diferentes janelas foram multiplicadas com filtro projetado produzindo as diferentes características:

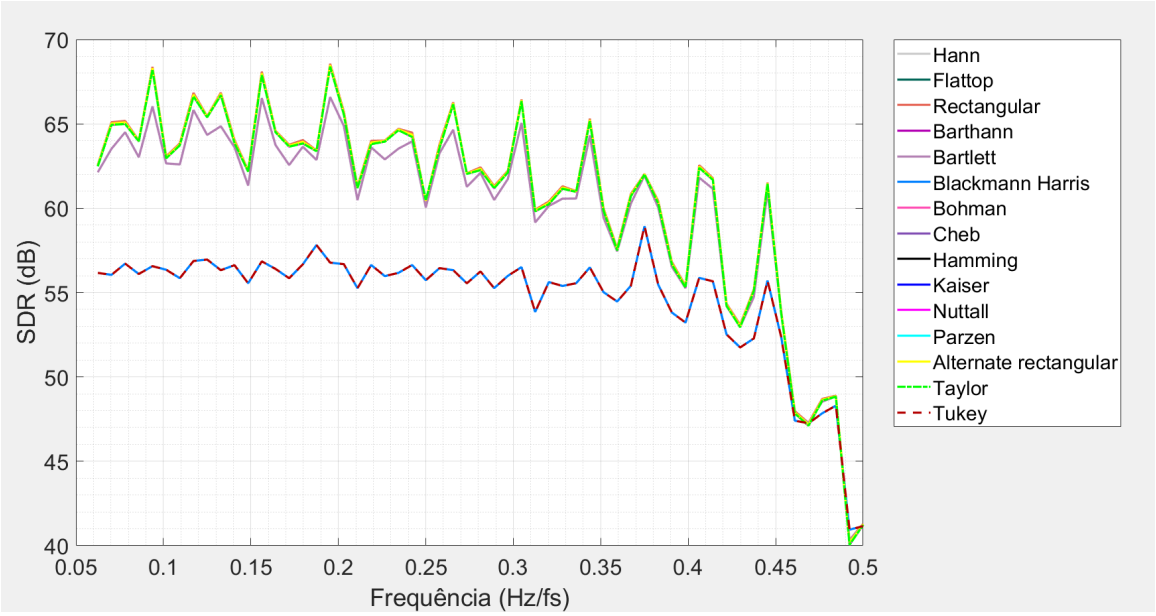


Figura 12: SDR em função da frequência pela frequência de amostragem efetiva para diferentes janelas.

A partir da figura, nota-se que diferentes janelas produzem a mesma relação sinal-distorção. No entanto, dentre as presentes a mais simples é a janela retangular – a qual foi escolhida para realização das simulações subsequentes.

## 4.2 Avaliação do filtro digital a partir do *Simulink* e *ModelSim*

A etapa da avaliação do algoritmo nos simuladores *Simulink* e *ModelSim* são pertinentes a conversão do código em vírgula fixa desenvolvido no MATLAB para a linguagem VHDL e teste do mesmo sob os dois simuladores. Após a criação do código, este é testado para verificação funcional. A figura abaixo representa um diagrama de como a simulação é realizada:

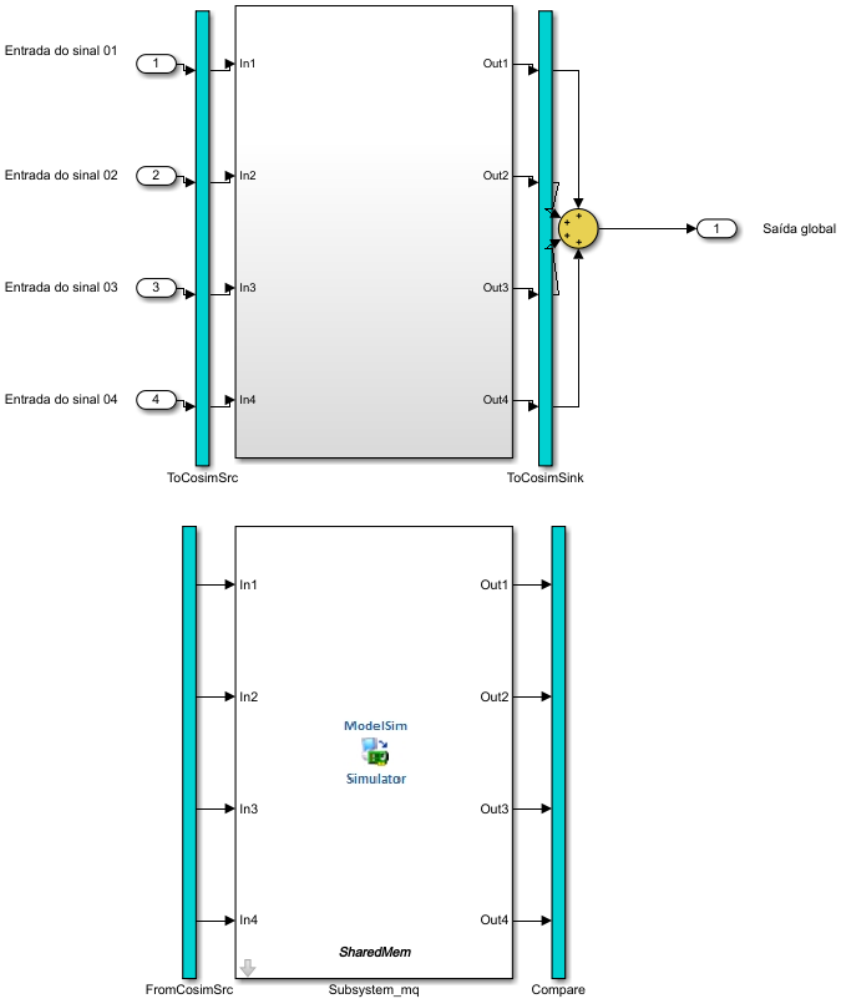


Figura 13: Representação do processo de co-simulação criado para validação do código VHDL sintetizado no *Simulink*.

O mesmo sinal de entrada é submetido ao *Simulink* e ao *ModelSim*. Analisando os cálculos em vírgula fixa feitos sob os diferentes simuladores produz-se o sinal de saída compensando a deriva de relógio com o mesmo SDR previsto na análise feita somente com o MATLAB:

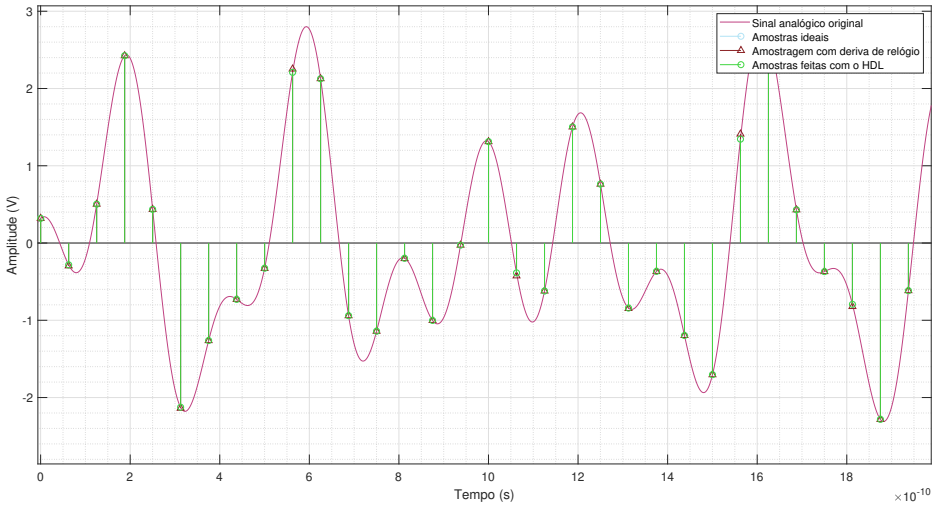


Figura 14: Resultado da compensação feita através da linguagem de descrição de *hardware*.

Os dados de consumo lógico estimado pelo pacote *Xilinx Zynq Support from HDL Coder* do *Simulink* estão listados no apêndice A. O relatório mostra a quantidade de portas lógicas e LUTs necessárias para implementação do circuito digital completo em um dispositivo ASIC ou uma FPGA. O SDR obtido nesta etapa foi idêntico ao feito com a análise do MATLAB.

## 4.3 Avaliação do filtro digital a partir do *Cadence Genus$^{TM}$ Synthesis Solution*

Esta etapa de simulação cria o circuito digital construído a partir das portas lógicas da tecnologia CMOS 130 nm. A síntese realizada pelo simulador avalia o código VHDL e verifica quais são as células lógicas mais eficientes em termos de área para implementação do circuito lógico. O circuito com as células da tecnologia CMOS 130 nm usadas para o circuito podem ser vistas na figura 15.

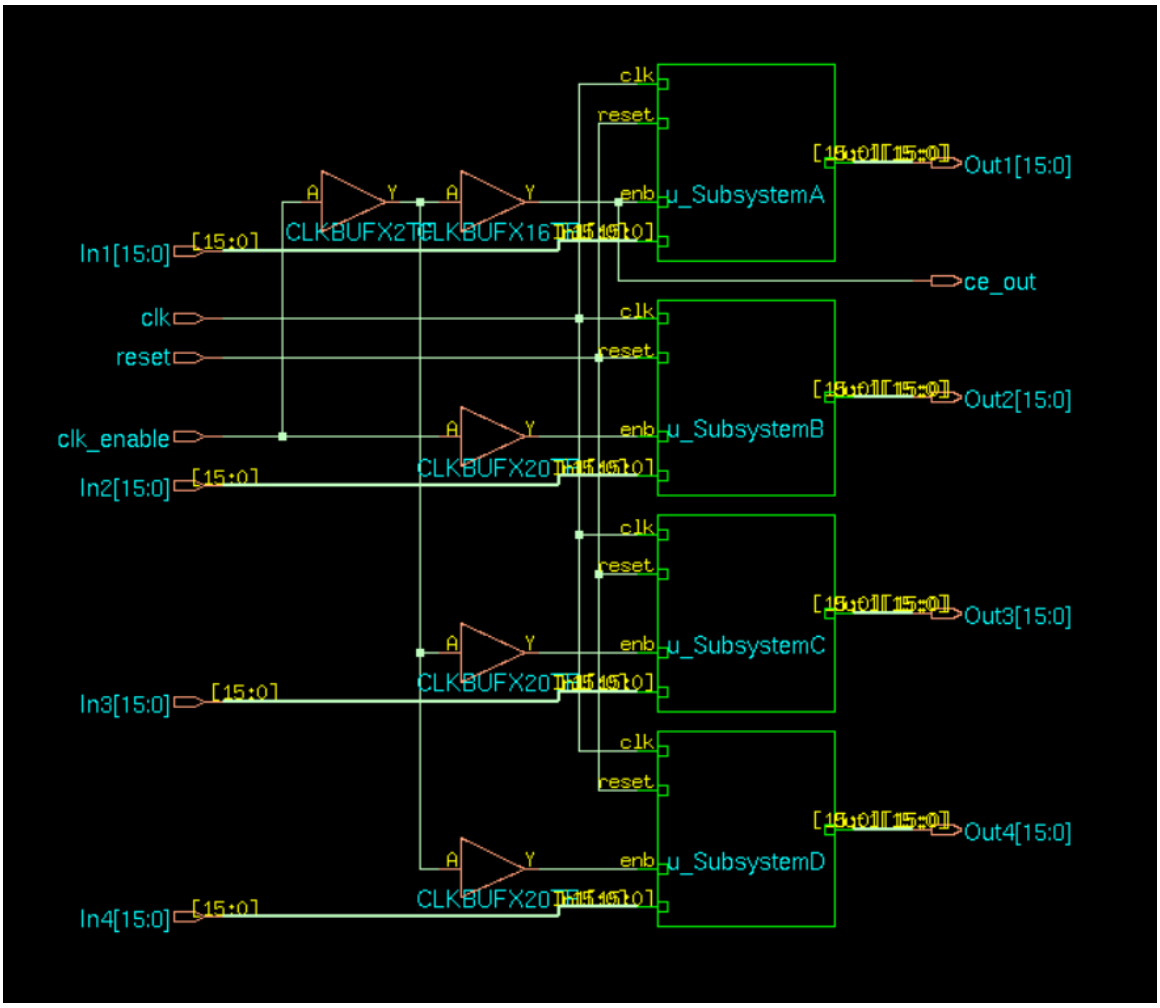


Figura 15: Esquemático RTL criado pelo Genus.

São utilizados quatro sub-sistemas para cada um dos filtros digitais projetados nas etapas anteriores. A figura 16 mostra a parte interna do sub-sistema com um componente filtro discreto descrito na linguagem VHDL com todas as declarações de sinais de entrada e saída.

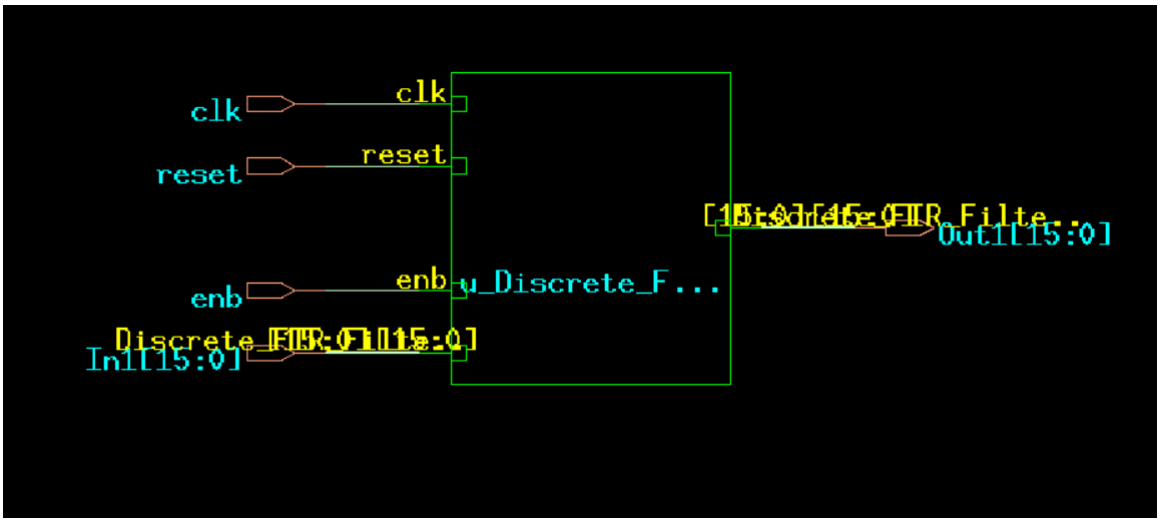


Figura 16: Componente "subsistema"do circuito geral contendo os circuitos do filtro discreto em esquemático RTL.

O simulador utiliza 644 elementos lógicos para implementação de um único filtro digital. A figura 17 apresenta o diagrama com as ligações virtuais feitas pelo simulador.

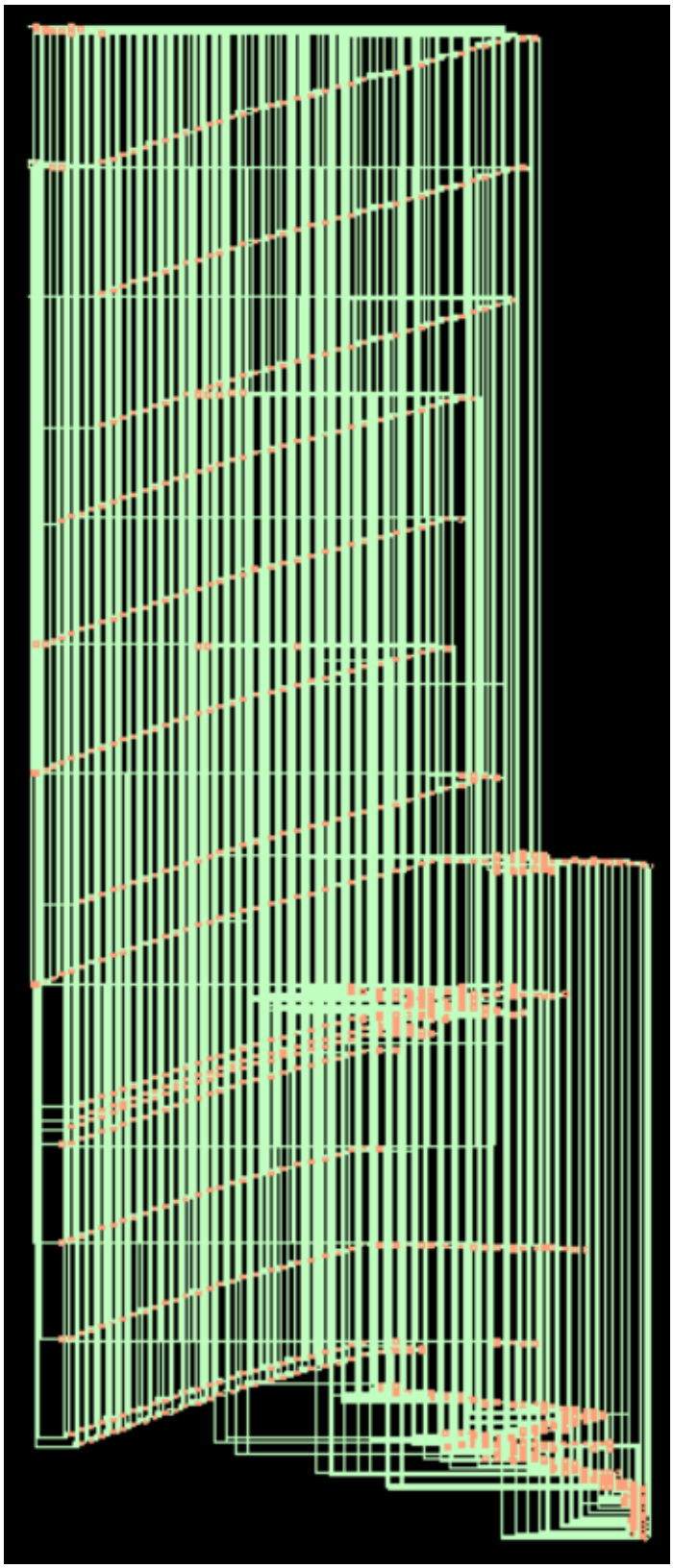

Figura 17: Figura gerada pelos Genus mostrando a quantidade de portas lógicas necessárias para implementação de um filtro digital.

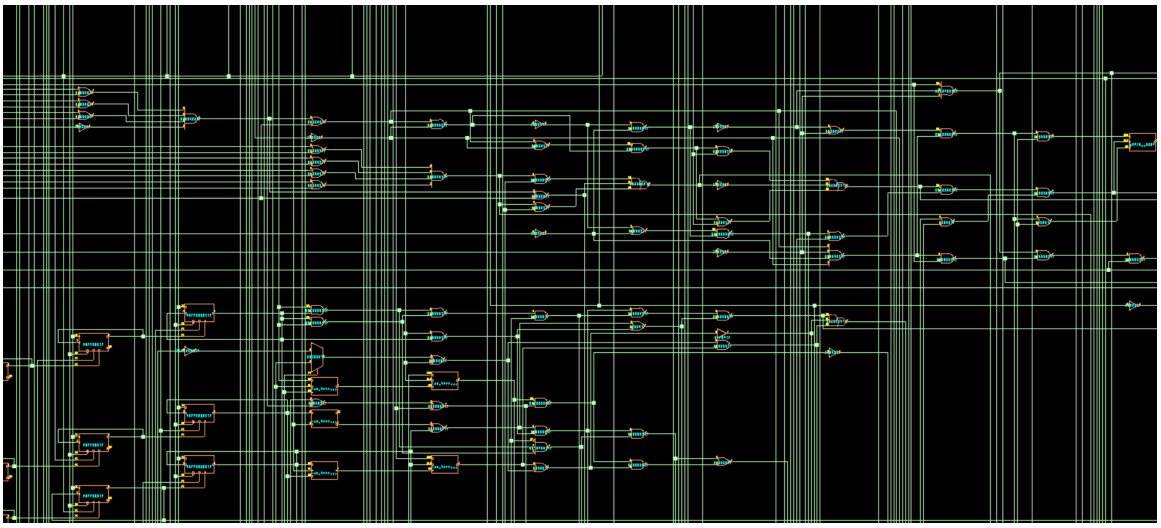

Figura 18: Figura gerada pelos Genus mostrando a quantidade de portas lógicas necessárias para implementação de um filtro digital. (Zoom)

Em detalhe, a figura 18 apresenta alguns registradores do circuito geral da figura 17. Nesta etapa de simulação, apenas as portas lógicas da tecnologia são consideradas – características de indutâncias, capacitâncias e resistências parasitas não são consideradas para os relatórios de consumo de potência, atraso e ocupação de área. Nos apêndices B, C e D são encontrados os relatórios dos respectivos dados.

### 4.3.1 Características de síntese

As características de síntese obtidas do *Cadence Genus*$^{TM}$ *Synthesis Solution* são apresentados na tabela 1. Nota-se que diferente da seção subsequente, as características apresentadas na tabela 1 representam o consumo do circuito digital global.

| Área ($mm^2$) | Potência total (mW) | Atraso (ns) | Portas lógicas |
|---|---|---|---|
| 0,093 | 16,8 | 1,512 | 3544 |

Tabela 1: Características de síntese de uma porta lógica.

Os resultados obtidos através dos relatórios fornecidos são semelhantes aos encontrados no estado da arte e também são compatíveis com as simulações obtidas nas etapas anteriores. No entanto, devida a introdução de elementos parasitas, o SDR avaliado a partir da *Cadence Incisive Enterprise Simulator* decresceu do valor máximo de 66.38 dB para 62.14 dB.

## 4.4 Avaliação do filtro digital a partir do *Cadence Innovus Implementation System*

Os resultados obtidos para o *Cadence Innovus Implementation System* foram obtidos para o filtro discreto representado na figura 17. A partir deste simulador, são extraídos relatórios contendo dados referentes ao consumo de área total, o uso da potência por cada tipo de circuito do sistema, perdas obtidas nas trilhas e caractísticas de atraso de propagação de cada elemento do circuito. Os relatórios são listados nos apêndices E, F e G.

O resultado do circuito digital desenhado com as dimensões totais é exibido na figura 19

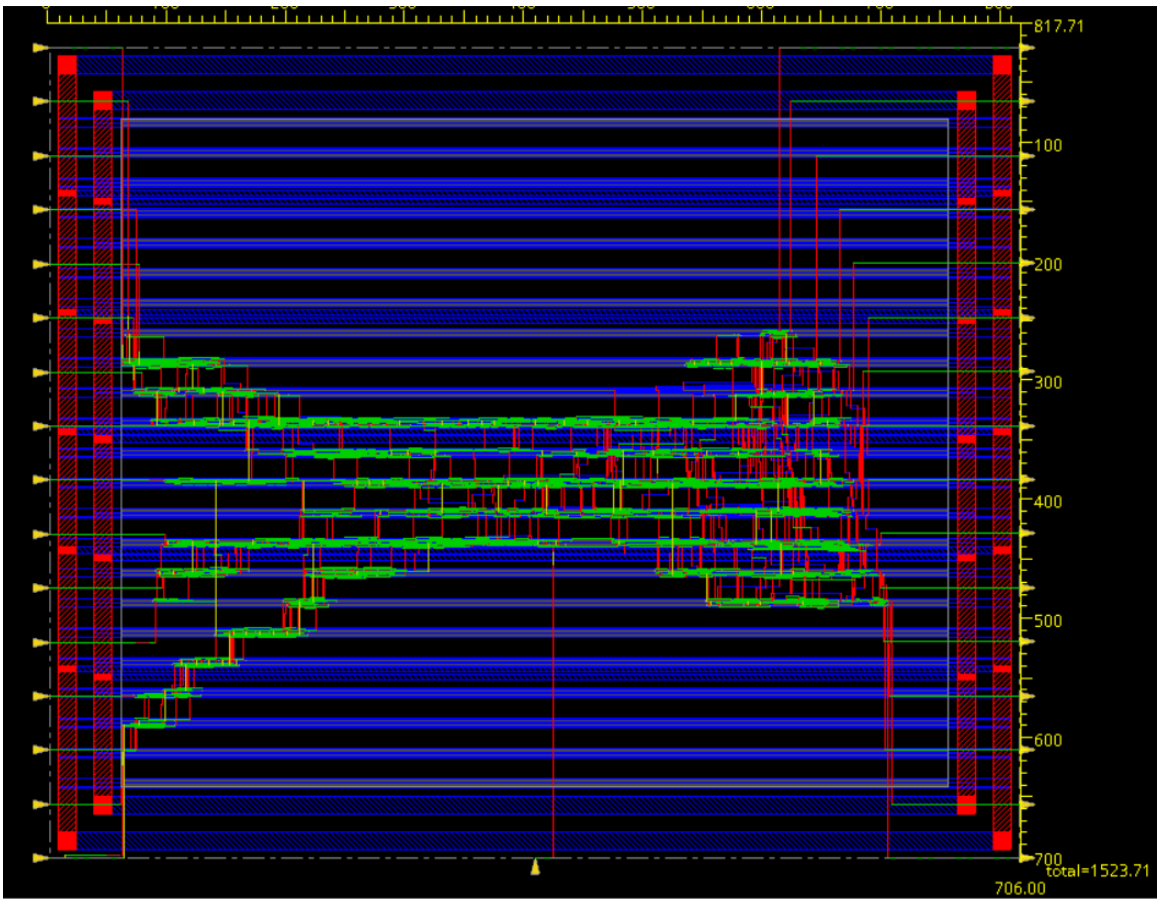


Figura 19: *Layout* do circuito digital com dimensões.

Os componentes individuais organizados pelo simulador, embora passem por etapas de otimização de posicionamento e roteamento – não fazem a ocupação de menor espaço que respeita as regras de desenho. O registrador em detalhe nas figuras 20 e 21 fica distante dos outros componentes e compromete a utilização eficiente da área

total – característica que permite a otimização do circuito. As figuras 22 e 23 são exemplos de utilização de área com eficiência, onde vários componentes são agrupados para ocupar o menor espaço sem comprometer as regras de desenho.

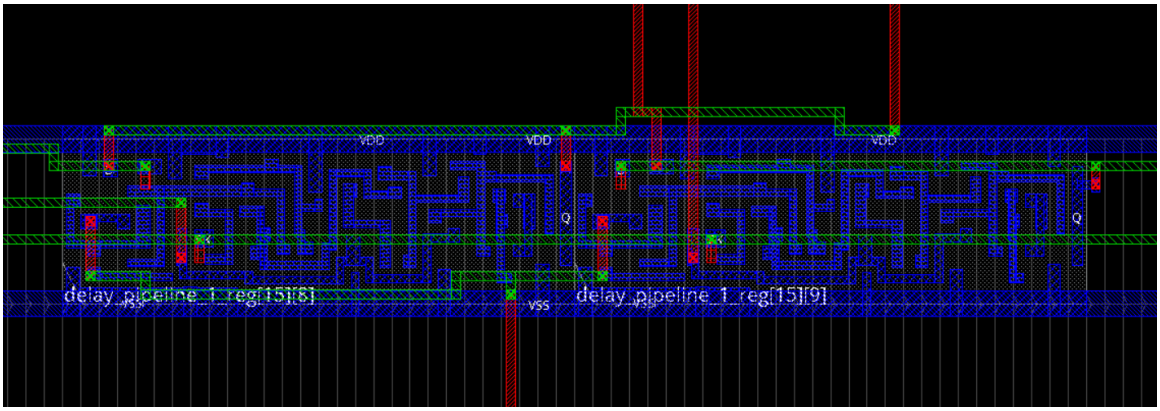


Figura 20: *Layout* do registrador, com um registrador em detalhe.

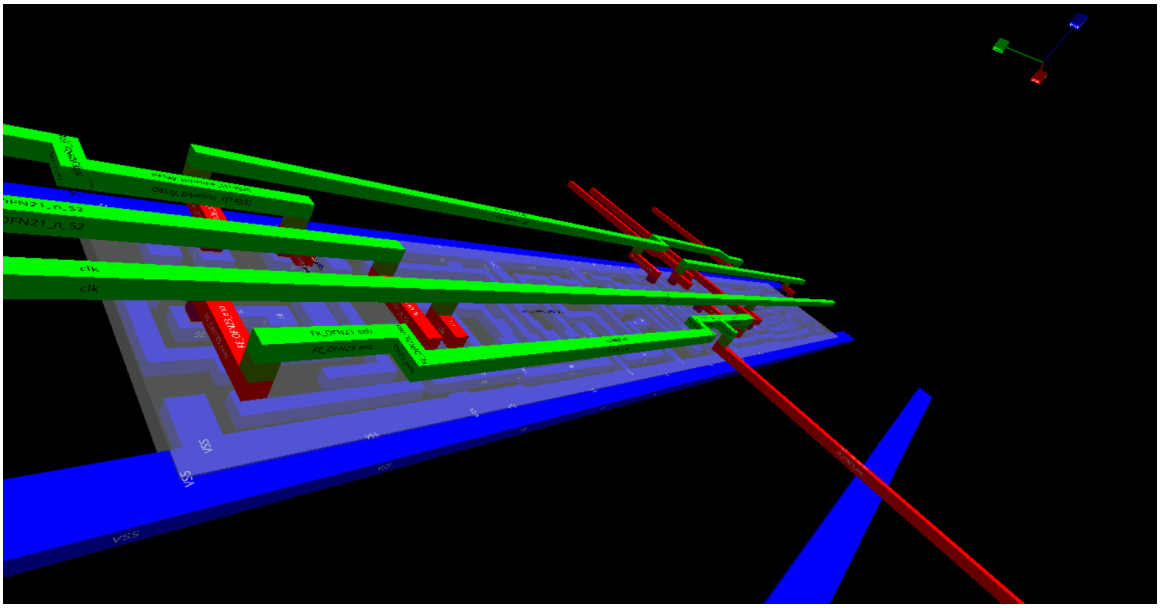

Figura 21: *Layout* do registrador, com um registrador em detalhe. (3D)

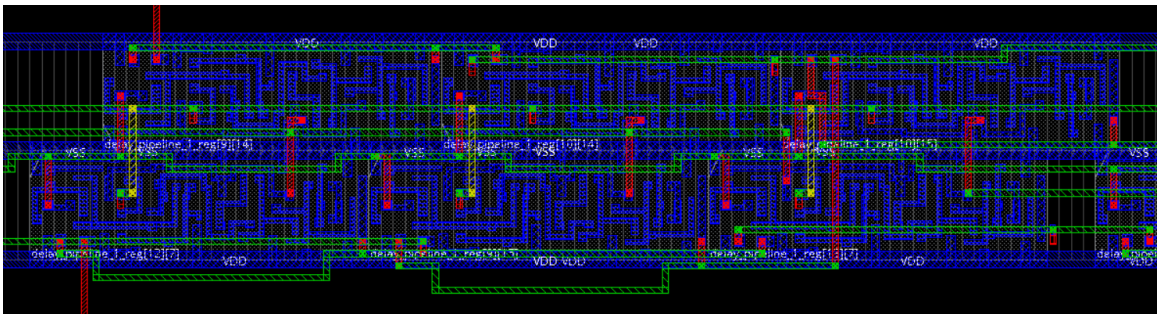


Figura 22: Representação de registradores posicionados para a ocupação de área com eficiência.

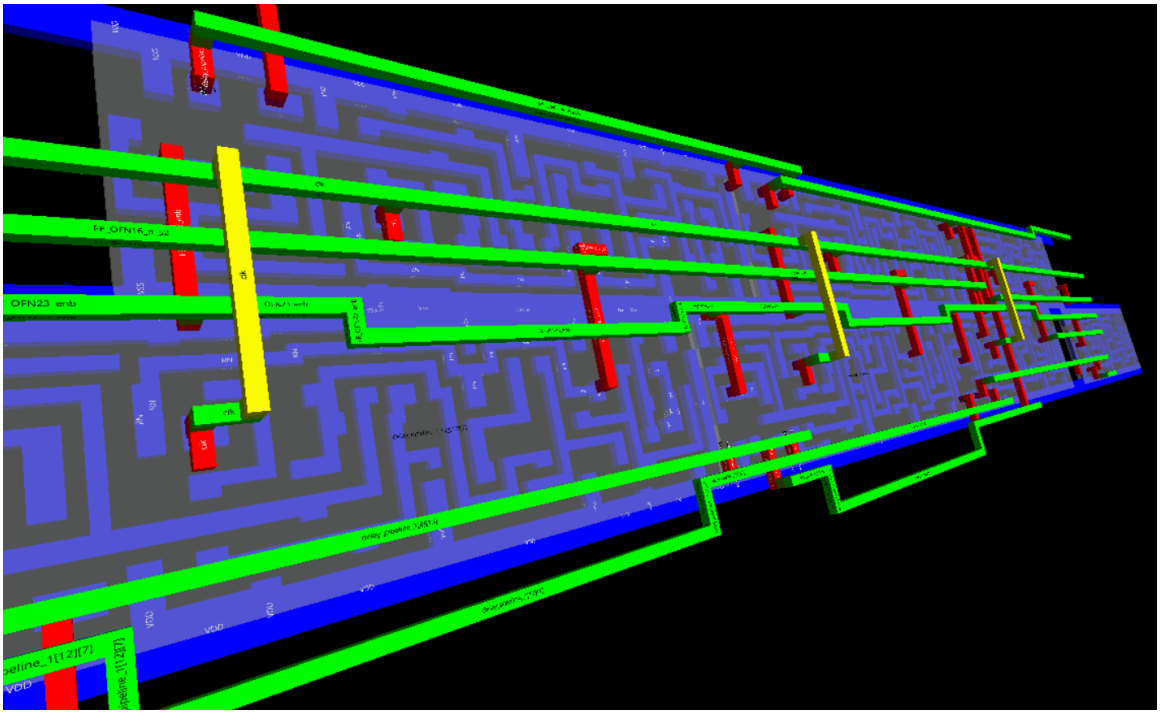

Figura 23: Representação de registradores posicionados para a ocupação de área com eficiência. (3D)

Os componentes do circuito são roteados a partir de quatro diferentes tipos de metais destinados especificamente para este fim. Os metais escolhidos pelo simulador não incluem a camada de silício policristalino, mas incluem os metais mais baixos dispostos pela tecnologia que são nomeados pela ferramenta como: M1, M2, M3 e M4.

O operador da ferramenta pode colocar elementos chamados de *fillers* nos espaços não ocupados para prevenção de erros de fabricação na síntese do circuito real pela fundidora.

### 4.4.1 Características de síntese

Os dados produzidos pelos relatórios dispostos pelo *Cadence Innovus Implementation System* são referentes a apenas um filtro discreto. A tabela 2 apresenta as características notáveis da síntese digital. Os apêndices E e F dispõem de informações referentes aos subsistemas e elementos específicos do sistema.

| Área ($mm^2$) | Potência total (mW) | Atraso (ns) | Portas lógicas |
|---|---|---|---|
| 0,022 | 5,787 | 4,060 | 644 |

Tabela 2

A inserção de não idealidades altera significativamente as características de síntese emitidas pelo simulador. Embora sejam testadas partes do circuito diferentes, como o consumo de potência é linear – apontado pelo simulador *Cadence Genus Synthesis Solution* – as características de área e potência deveriam ser próximas as relatadas pelo simulador anterior divididas por quatro, dado que são utilizados quatro filtros. As não idealidades aumentam o uso dos recursos de área e potência e também aumenta o atraso o atraso em torno de dez vezes mais do que foi estimado pelo simulador anterior.

### 4.4.2 Resultados pós-*layout*

O arquivo verilog e .sdf produzidos pelo *Cadence Innovus Implementation System* podem ser utilizados juntamente com o *Cadence Incisive Enterprise Solution* para o teste dos filtros implementados a partir do *layout* considerando os elementos parasitas presentes na simulação do dispositivo físico.

De forma análoga a feita para as outras etapas, os arquivos são processados no *Cadence Incisive Enterprise Solution* para que a saída seja extraída em forma de números decimais e entçao comparados com a amostragem ideal. O resultado obtido para reconstrução do sinal é verificada na figura 24.

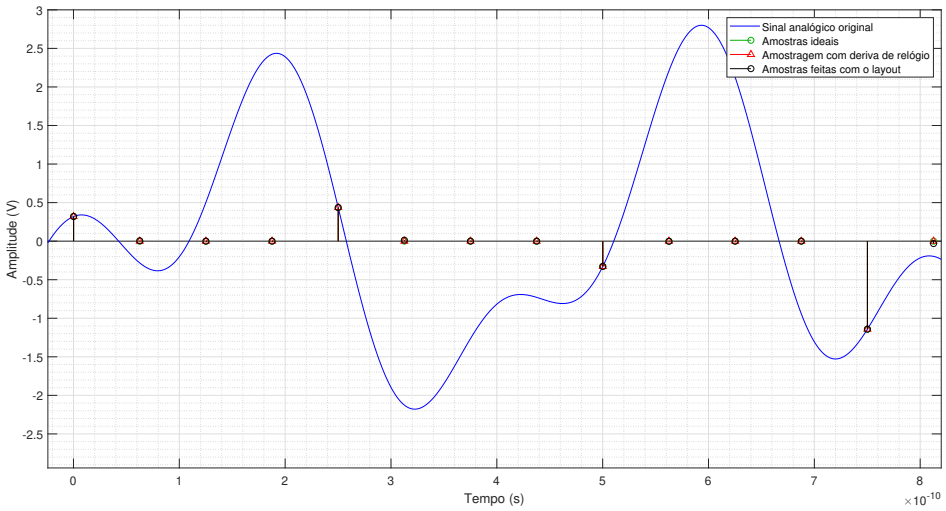


Figura 24: Resultado da amostragem do filtro a partir dos resultados fornecidos pelo *Cadence Innovus Implementation System*.

A figura mostra o resultado de apenas um dos filtros implementados funcionando – dado que a implementação em *layout* foi executada para apenas um dos quatro filtros. Como a sequência de dados é *upsampled*, um dos filtros é responsável por um quarto das amostras totais, dado que as demais são nulas. A operação dos quatro filtros juntos corrigiria todas as amostras com erro de deriva presentes no sinal. A SDR obtida para este caso foi de 55,32 dB.

# 5 Conclusão

Os filtros projetados a partir da bibliografia estudada foram sintetizados e validados com sucesso seguindo a análise *top-down* proposta nos objetivos através de cada um dos simuladores listados.

Conforme o projeto transitava do nível de maior abstração para componentes com valores mais próximos aos reais, decrescia o desempenho do circuito digital através da análise da SDR. O efeito de perda de qualidade é explicado pela inclusão de não-idealidades em cada simulador. Conforme mais variáveis representando variáveis físicas não ideais eram incluídas na análise do circuito, mais decrescia a qualidade do sinal reconstruído. Os elementos parasitas que tiveram maior influência na reconstrução do sinal foram as resistências das trilhas de condução de sinal analisadas a partir do *Cadence Innovus Implementation System*. O simulador usa camadas de metal mais baixas para que o circuito seja sintetizado, juntamente com o tamanho mínimo de trilhas para o roteamento. Otimizações substituindo os metais mais baixos por mais altos poderiam ser implementadas em trabalhos futuros para o aumento da qualidade do sinal reconstruído. Otimizações envolvendo áreas das trilhas maiores, no entanto, comprometeriam significativamente a ocupação da área total do circuito, envolvendo custos mais altos de fabricação.

O projeto desenvolvido deixa clara a relação de compromisso entre o desempenho e a utilização de recursos. Otimizações nas características do circuito devem ser consideradas utilizando as premissas da aplicação do circuito digital para que custos de fabricação ou potências maiores não fiquem acima dos valores limite impostos. Exemplos incluem a utilização do circuito digital para correção de erros de relógio de ADCs que operem com frequências maiores do que a testada.

O SDR máximo previsto para operação do circuito através da plataforma de mais alta abstração foi de 66.38 dB. A implementação do circuito considerando características parasitas e atrasos fornece uma relação sinal-distorção de 55,32 dB – o processo de implementação decresce a relação em 11.6 dB. O SDR necessário para construção de um ADC de 9 bits é de 55.94 dB, logo com o desempenho observado é possível a construção de um ADC de 8 bits.

No entanto, o projeto ainda possui características que podem ser otimizadas sem o compromisso descrito. A área de ocupação dos registradores pode ser otimizada em diferentes iterações do *Cadence Innovus Implementation System* para ocupação da menor área sem que as características mais altas de indutância e/ou capacitância afetem o desempenho da reconstrução de maneira significativa. Estudos específicos podem ser desenvolvidos para a análise de deriva de relógio em função das variáveis do circuito – desta forma a escolha das ordens do filtro de compensação seriam as mais eficientes para cada aplicação, dado que ordens menores se traduzem para consumo de área, características de atraso e consumo de potência menores. Dessa forma, a

partir da implementação das otimizações listadas, a relação sinal-distorção poderia atingir o nível necessário para construção de um ADC de 9 bits.

# Referências

[1] Y. C. Eldar and A. V. Oppenheim, "Filterbank reconstruction of bandlimited signals from nonuniform and generalized samples," *IEEE Transactions on Signal Processing*, vol. 48, no. 10, pp. 2864–2875, Oct 2000.

[2] B. Murmann, *Boris Murmann: ADC Survey.* Disponível em: https://web.stanford.edu/ murmann/adcsurvey.html: [online] Web.stanford.edu, 2019.

[3] A. Oppenheim, *Discrete-Time Signal Processing*, ser. Pearson education signal processing series. Pearson Education, 1999. [Online]. Available: https://books.google.com.br/books?id=geTn5W47KEsC

[4] S. Maymon and A. V. Oppenheim, "Sinc interpolation of nonuniform samples," *IEEE Transactions on Signal Processing*, vol. 59, no. 10, pp. 4745–4758, Oct 2011.

[5] H. Le Duc, D. M. Nguyen, C. Jabbour, P. Desgreys, O. Jamin, and V. Tam Nguyen, "Fully digital feedforward background calibration of clock skews for subsampling tiadcs using the polyphase decomposition," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, no. 6, pp. 1515–1528, June 2017.

[6] X. Liu, H. Xu, Y. Wang, L. Nan, G. Liu, and Q. Tian, "Statistics-based correction method for sample-and-hold mismatch in 2-channel tiadcs," 07 2018, pp. 1–4.

[7] A. Oppenheim, A. Willsky, and S. Nawab, *Signals and Systems*, ser. Prentice-Hall signal processing series. Prentice Hall, 1997. [Online]. Available: https://books.google.com.br/books?id=LwQqAQAAMAAJ

[8] M. El-Chammas and B. Murmann, *Background Calibration of Time-Interleaved Data Converters*, ser. Analog Circuits and Signal Processing. Springer New York, 2011. [Online]. Available: https://books.google.com.br/books?id=A-7ycbEqLtkC

[9] H. Veendrick, *Nanometer CMOS ICs: From Basics to ASICs.* Springer International Publishing, 2017. [Online]. Available: https://books.google.com.br/books?id=Lv_EDgAAQBAJ

# Apêndice A
# Dados de consumo lógico

| Slice Logic Utilization | Used |
|---|---|
| Number of Slice Registers | 1,999 |
| Number used as Flip Flops | 1,664 |
| Number used as Latches | 0 |
| Number used as Latch-thrus | 0 |
| Number used as AND/OR logics | 335 |
| Number of Slice LUTs | 1,733 |
| Number used as logic | 1,234 |
| Number using O6 output only | 734 |
| Number using O5 output only | 8 |
| Number using O5 and O6 | 492 |
| Number used as ROM | 0 |
| Number used as Memory | 415 |
| Number used as Dual Port RAM | 0 |
| Number used as Single Port RAM | 0 |
| Number used as Shift Register | 415 |
| Number using O6 output only | 415 |
| Number using O5 output only | 0 |
| Number using O5 and O6 | 0 |
| Number used exclusively as route-thrus | 84 |
| Number with same-slice register load | 84 |
| Number with same-slice carry load | 0 |
| Number with other load | 0 |
| Number of occupied Slices | 616 |
| Number of LUT Flip Flop pairs used | 2,318 |
| Number with an unused Flip Flop | 413 |
| Number with an unused LUT | 585 |
| Number of fully used LUT-FF pairs | 1,32 |
| Number of unique control sets | 2 |
| Number of slice register sites lost to control set restrictions | 9 |
| Number of bonded IOBs | 132 |
| Number of RAMB36E1/FIFO36E1s | 0 |
| Number of RAMB18E1/FIFO18E1s | 0 |
| Number of BUFG/BUFGCTRLs | 1 |
| Number used as BUFGs | 1 |
| Number used as BUFGCTRLs | 0 |
| Number of IDELAYE2/IDELAYE2_FINEDELAYs | 0 |
| Number of ILOGICE2/ILOGICE3/ISERDESE2s | 0 |
| Number of ODELAYE2/ODELAYE2_FINEDELAYs | 0 |
| Number of OLOGICE2/OLOGICE3/OSERDESE2s | 0 |
| Number of PHASER_IN/PHASER_IN_PHYs | 0 |
| Number of PHASER_OUT/PHASER_OUT_PHYs | 0 |
| Number of BSCANs | 0 |
| Number of BUFHCEs | 0 |
| Number of BUFRs | 0 |

| | |
|---|---|
| Number of CAPTUREs | 0 |
| Number of DNA_PORTs | 0 |
| Number of DSP48E1s | 37 |
| Number of EFUSE_USRs | 0 |
| Number of FRAME_ECCs | 0 |
| Number of IBUFDS_GTE2s | 0 |
| Number of ICAPs | 0 |
| Number of IDELAYCTRLs | 0 |
| Number of IN_FIFOs | 0 |
| Number of MMCME2_ADVs | 0 |
| Number of OUT_FIFOs | 0 |
| Number of PCIE_2_1s | 0 |
| Number of PHASER_REFs | 0 |
| Number of PHY_CONTROLs | 0 |
| Number of PLLE2_ADVs | 0 |
| Number of STARTUPs | 0 |
| Number of XADCs | 0 |
| Average Fanout of Non-Clock Nets | 2.26 |

# Apêndice B
# Dados de consumo de potência – Cadence $Genus^{TM}$ Synthesis Solution

Instance: /Subsystem
Power Unit: W
PDB Frames: /stim#0/frame#0

| Category | Leakage | Internal | Switching | Total | Row% |
|---|---|---|---|---|---|
| memory | 0.00000e+00 | 0.00000e+00 | 0.00000e+00 | 0.00000e+00 | 0.00% |
| register | 2.02710e-06 | 1.43000e-02 | 1.37060e-04 | 1.44000e-02 | 85.71% |
| latch | 0.00000e+00 | 0.00000e+00 | 0.00000e+00 | 0.00000e+00 | 0.00% |
| logic | 3.46260e-07 | 7.57770e-04 | 1.82780e-04 | 9.40900e-04 | 5.60% |
| bbox | 0.00000e+00 | 0.00000e+00 | 0.00000e+00 | 0.00000e+00 | 0.00% |
| clock | 0.00000e+00 | 0.00000e+00 | 1.50000e-03 | 1.50000e-03 | 8.93% |
| pad | 0.00000e+00 | 0.00000e+00 | 0.00000e+00 | 0.00000e+00 | 0.00% |
| pm | 0.00000e+00 | 0.00000e+00 | 0.00000e+00 | 0.00000e+00 | 0.00% |
| Subtotal | 2.3733e-06 | 1.51080e-02 | 1.84779e-03 | 1.68000e-02 | 100.00% |
| Percentage | 0.01% | 89.07% | 10.91% | 100.00% | 100.00% |

# Apêndice C
# Dados de atraso – Cadence *Genus*$^{TM}$ Synthesis Solution

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 19.10-p002_1
  Module:                 Subsystem
  Operating conditions:   ff_typical_min_1p32v_0c (balanced_tree)
  Wireload mode:          segmented
  Area mode:              timing library
============================================================


Path 1: MET (49 ps) Late External Delay Assertion at pin Out2[5]
          Group: myclk
     Startpoint: (R) u_SubsystemB/u_Discrete_FIR_Filter1/delay_pipeline1reg[9][7]/CK
          Clock: (R) myclk
       Endpoint: (F) Out2[5]
          Clock: (R) myclk

                     Capture       Launch
        Clock Edge:+    1660            0
       Src Latency:+       0            0
       Net Latency:+       0 (I)        0 (I)
           Arrival:=    1660            0

      Output Delay:-     100
     Required Time:=    1560
      Launch Clock:-       0
         Data Path:-    1511
             Slack:=      49

Exceptions/Constraints:
  output_delay             100             ou_del_92_1

#-------------------------------------------------------------------------------
Arc   Edge    Cell      Fanout Load Trans Delay Arrival Instance
#
#-------------------------------------------------------------------------------
  R     (arrival)     2688    -     0     -       0    (-,-)
```

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| CK->Q | R | DFFRHQX2TF | 4 | 9.4 | 56 | 114 | 114 | (-,-) |
| A->Y | F | INVX2TF | 2 | 2.1 | 20 | 18 | 132 | (-,-) |
| AN->Y | F | NAND2BX1TF | 1 | 4.4 | 45 | 80 | 212 | (-,-) |
| B->Y | R | NAND2X2TF | 2 | 8.5 | 57 | 49 | 261 | (-,-) |
| B->Y | R | XOR2X2TF | 1 | 4.0 | 109 | 70 | 332 | (-,-) |
| A->Y | R | XOR2X2TF | 3 | 9.7 | 138 | 74 | 405 | (-,-) |
| B->Y | R | XOR2X1TF | 1 | 6.8 | 157 | 101 | 506 | (-,-) |
| A->Y | R | XOR2X4TF | 2 | 13.4 | 112 | 71 | 577 | (-,-) |
| B0->Y | F | OAI21X2TF | 1 | 6.6 | 45 | 40 | 616 | (-,-) |
| B0->Y | R | OAI2BB1X4TF | 3 | 16.4 | 58 | 48 | 664 | (-,-) |
| B->Y | R | XOR2X4TF | 3 | 20.6 | 128 | 79 | 743 | (-,-) |
| A->Y | R | XNOR2X4TF | 1 | 7.1 | 96 | 62 | 804 | (-,-) |
| A->Y | R | XNOR2X4TF | 1 | 14.7 | 112 | 62 | 867 | (-,-) |
| B->S | F | ADDFHX2TF | 2 | 9.1 | 44 | 118 | 984 | (-,-) |
| A->Y | F | XOR2X4TF | 3 | 16.6 | 56 | 56 | 1041 | (-,-) |
| A->Y | R | INVX3TF | 1 | 4.4 | 27 | 29 | 1070 | (-,-) |
| B->Y | F | NAND2BX2TF | 1 | 4.2 | 29 | 23 | 1093 | (-,-) |
| A->Y | R | NAND2X2TF | 1 | 6.6 | 49 | 35 | 1128 | (-,-) |
| B0->Y | F | OAI2BB1X4TF | 3 | 12.5 | 34 | 28 | 1156 | (-,-) |
| A1->Y | R | AOI21X2TF | 1 | 5.0 | 68 | 67 | 1223 | (-,-) |
| A0->Y | F | OAI21X2TF | 2 | 7.1 | 38 | 35 | 1258 | (-,-) |
| A1->Y | R | AOI21X2TF | 1 | 6.6 | 78 | 74 | 1332 | (-,-) |
| B0->Y | F | OAI2BB1X4TF | 5 | 22.1 | 44 | 38 | 1370 | (-,-) |
| A0->Y | R | AOI21X2TF | 2 | 4.6 | 66 | 60 | 1430 | (-,-) |
| A0->Y | F | OAI21X1TF | 1 | 2.1 | 29 | 33 | 1463 | (-,-) |
| A->Y | F | XOR2XLTF | 1 | 0.0 | 43 | 48 | 1511 | (-,-) |

# Apêndice D
# Dados de consumo de área – Cadence *Genus*$^{TM}$ Synthesis Solution

```
=============================================================
  Generated by:           Genus(TM) Synthesis Solution 19.10-p002_1
  Module:                 Subsystem
  Operating conditions:   ff_typical_min_1p32v_0c (balanced_tree)
  Wireload mode:          segmented
  Area mode:              timing library
=============================================================

Module          Cell Count  Cell Area  Net Area   Total Area  Wireload
-------------------------------------------------------------------------------
Subsystem             3544  92897.280     0.000    92897.280    <none> (D)
SubsystemB             886  23224.320     0.000    23224.320    <none> (D)
Discrete_FIR_Filter1   886  23224.320     0.000    23224.320    <none> (D)
SubsystemC             886  23224.320     0.000    23224.320    <none> (D)
Discrete_FIR_Filter2   886  23224.320     0.000    23224.320    <none> (D)
SubsystemD             886  23224.320     0.000    23224.320    <none> (D)
Discrete_FIR_Filter3   886  23224.320     0.000    23224.320    <none> (D)
SubsystemA             886  23224.320     0.000    23224.320    <none> (D)
Discrete_FIR_Filter    886  23224.320     0.000    23224.320    <none> (D)

 (D) = wireload is default in technology library
```

# Apêndice E
# Dados de consumo de potência – Cadence Innovus Implementation System

```
*------------------------------------------------------------------------------
*       Innovus 19.10-p002_1 (64bit) 04/19/2019 15:18 (Linux 2.6.32-431.11.2.el6.
*
*
*       Date & Time:    2019-Nov-22 11:35:45 (2019-Nov-22 13:35:45 GMT)
*
*------------------------------------------------------------------------------
*
*       Design: Discrete_FIR_Filter
*
*       Liberty Libraries used:
*       default_emulate_view: sc9tap_bicmos8hp_base_rvt_ff_typical_min_1p32v_m40c.lib
*
*       Parasitic Files used:
*
*       Power View : default_emulate_view
*
*       User-Defined Activity : N.A.
*
*       Activity File: N.A.
*
*       Hierarchical Global Activity: N.A.
*
*       Global Activity: N.A.
*
*       Sequential Element Activity: 0.200000
*
*       Primary Input Activity: 0.200000
*
*       Default icg ratio: N.A.
*
*       Global Comb ClockGate Ratio: N.A.
*
*       Power Units = 1mW
*
*       Time Units = 1e-09 secs
```

```
*
*       report_power
*
--------------------------------------------------------------------------------
*



Total Power
--------------------------------------------------------------------------------
Total Internal Power:        5.14670728            88.9334%
Total Switching Power:       0.64036766            11.0653%
Total Leakage Power:         0.00007392             0.0013%
Total Power:                 5.78714888
--------------------------------------------------------------------------------


Group          Internal   Switching     Leakage       Total  Percentage
               Power      Power         Power         Power  (%)
------------------------------------------------------------------------------------
Sequential        4.825      0.1568   6.373e-05       4.982  86.08
Macro             0           0           0           0  0
IO                0           0           0           0  0
Combinational     0.3218      0.4836   1.019e-05      0.8054  13.92
CLK(Combinational)0           0           0           0  0
Clock (Sequential)0           0           0           0  0
------------------------------------------------------------------------------------
Total             5.147      0.6404   7.392e-05       5.787  100
------------------------------------------------------------------------------------


Rail Voltage   Internal   Switching     Leakage       Total  Percentage
               Power      Power         Power         Power  (%)
------------------------------------------------------------------------------------
VDD  1.32      5.147      0.6404   7.392e-05       5.787     100
```

# Apêndice F - Dados de atraso – Cadence Innovus Implementation System

```
#====================================================================
#  Generated by:      Cadence Innovus 19.10-p002_1
#  OS:                Linux x86_64(Host ID galileu)
#  Design:            Discrete_FIR_Filter
#  Command:           report_timing
#====================================================================
Path 1: MET Late External Delay Assertion
Endpoint:   Discrete_FIR_Filter_out[15]   (^) checked with  leading edge of
'myclk'
Beginpoint: delay_pipeline_1_reg[21][0]/Q (v) triggered by  leading edge of
'myclk'
Path Groups: {myclk}
Analysis View: default_emulate_view
Other End Arrival Time          0.000
- External Delay                0.100
+ Phase Shift                   4.160
= Required Time                 4.060
- Arrival Time                  3.418
= Slack Time                    0.642
     Clock Rise Edge                     0.000
     + Clock Network Latency (Prop)      0.052
     = Beginpoint Arrival Time           0.052
     +-------------------------------------------------------------------------------
     |              Arc              |    Cell     | Delay | Arrival | Required |
     |                               |             |       |  Time   |   Time   |
     |-------------------------------------------+-----------------------+----------
     | CK ^                          |             |       |   0.052 |    0.694 |
     | CK ^ -> Q v                   | SDFFRX2TF   | 0.259 |   0.312 |    0.954 |
     | C v -> Y ^                    | NOR4XLTF    | 0.191 |   0.503 |    1.145 |
     | AN ^ -> Y ^                   | NOR4BX1TF   | 0.147 |   0.650 |    1.292 |
     | AN ^ -> Y ^                   | NOR2BX1TF   | 0.145 |   0.795 |    1.437 |
     | AON ^ -> Y ^                  | AOI2BB2X1TF | 0.130 |   0.925 |    1.567 |
     | A ^ -> CO ^                   | CMPR32X2TF  | 0.193 |   1.118 |    1.760 |
     | C ^ -> CO ^                   | CMPR32X2TF  | 0.117 |   1.235 |    1.877 |
     | C ^ -> CO ^                   | CMPR32X2TF  | 0.128 |   1.363 |    2.005 |
     | C ^ -> CO ^                   | CMPR32X2TF  | 0.112 |   1.475 |    2.117 |
```

| | | | | |
|---|---|---|---|---|
| C ^ -> CO ^ | CMPR32X2TF | 0.114 | 1.589 | 2.231 |
| C ^ -> CO ^ | CMPR32X2TF | 0.102 | 1.691 | 2.333 |
| A ^ -> Y v | INVX1TF | 0.027 | 1.718 | 2.360 |
| C v -> CO v | CMPR32X2TF | 0.121 | 1.839 | 2.481 |
| B0 v -> Y ^ | OAI2BB2XLTF | 0.259 | 2.098 | 2.740 |
| C ^ -> S v | CMPR32X2TF | 0.195 | 2.293 | 2.935 |
| A v -> CO v | CMPR32X2TF | 0.178 | 2.471 | 3.113 |
| C v -> CO v | CMPR32X2TF | 0.126 | 2.597 | 3.239 |
| C v -> CO v | CMPR32X2TF | 0.120 | 2.717 | 3.359 |
| C v -> CO v | CMPR32X2TF | 0.126 | 2.843 | 3.485 |
| C v -> CO v | CMPR32X2TF | 0.122 | 2.965 | 3.607 |
| C v -> CO v | CMPR32X2TF | 0.131 | 3.095 | 3.737 |
| A0 v -> Y ^ | AOI22X1TF | 0.083 | 3.178 | 3.820 |
| A1N ^ -> Y ^ | OAI2BB2X1TF | 0.238 | 3.416 | 4.058 |
| Discrete_FIR_Filter_out[15] ^ | | 0.002 | 3.418 | 4.060 |