UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS

ROGÉRIO DE OLIVEIRA CALSOLARI

Estudo e Avaliação da utilização de circuitos digitais na implementação de circuitos analógicos

São Carlos
2024

ROGÉRIO DE OLIVEIRA CALSOLARI

Estudo e Avaliação da utilização de circuitos digitais para implementação de circuitos analógicos

Monografia apresentada ao Curso de Engenharia Elétrica, da Escola de Engenharia de São Carlos da Universidade de São Paulo, como parte dos requisitos para obtenção do Título de Engenheiro Eletricista

Orientador: Prof. Dr. Maximiliam Luppe

VERSÃO CORRIGIDA

São Carlos

2024

Ficha catalográfica elaborada pela Biblioteca Prof. Dr. Sérgio Rodrigues Fontes da EESC/USP com os dados inseridos pelo(a) autor(a).

C141e

Calsolari, Rogério de Oliveira
Estudo e Avaliação da utilização de circuitos digitais para implementação de circuitos analógicos / Rogério de Oliveira Calsolari; orientador Maximiliam Luppe . São Carlos, 2024.

Monografia (Graduação em Engenharia Elétrica com ênfase em Eletrônica) -- Escola de Engenharia de São Carlos da Universidade de São Paulo, 2024.

1. Spice. 2. Standard Logic Cells. 3. Openlane. 4. Conversor-digital-analógico. 5. Comparadores. I. Título.

Eduardo Graziosi Silva - CRB - 8/8907

FOLHA DE APROVAÇÃO

Nome: Rogério de Oliveira Calsolari

Título: “Estudo e Avaliação da utilização de circuitos digitais na implementação de circuitos analógicos”

Trabalho de Conclusão de Curso defendido e aprovado em_09_/_12_/_2024_,

com NOTA_7,0_(sete , zero), pela Comissão Julgadora:

Prof. Dr. Maximiliam Luppe - Orientador SEL/EESC/USP

Prof. Dr. João Navarro Soares Júnior - SEL/EESC/USP

Prof. Associado João Paulo Pereira do Carmo - SEL/EESC/USP

Coordenador da CoC-Engenharia Elétrica - EESC/USP:
Professor Associado José Carlos de Melo Vieira Júnior

AGRADECIMENTOS

Ao meu orientador Prof. Dr. Maximiliam Luppe por me auxiliar no desenvolvimento do projeto e me despertar uma área da qual eu não conhecia.

À minha família por todo apoio em minha vida.

Ao Departamento de Engenharia Elétrica e de Computação (SEL) da EESC-USP e toda a comunidade USP pela excelência do ensino e pela oportunidade de cursar um dos melhores cursos do país.

RESUMO

CALSOLARI, R. O. **Estudo e Avaliação da utilização de circuitos digitais (standard logic cells) na implementação de circuitos analógicos.** 2024. Monografia (Trabalho de Conclusão de Curso) – Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2024.

Esta monografia descreve a utilização das atuais ferramentas, de código aberto, para o desenvolvimento de circuitos integrados - *Openlane*, *Xschem*, *Ngspice* - em conjunto com a análise de um conversor digital-analógico e circuitos comparadores implementados apenas por células lógicas. O projeto foi desenvolvido no *Windows* com a utilização do *Linux* por meio do *WSL*. As ferramentas foram instaladas nesse ambiente, pela biblioteca *Institute for Integrated Circuits* da Universidade *Johannes Kepler de Linz*, e os circuitos implementados para sua avaliação. Os circuitos propostos de comparadores e conversor digital-analógico foram implementados e testados para diferentes tensões de alimentação, e sua análise foi discutida no projeto.

Palavras-chave: *Spice*. Digital-to-Analog Converter. Standard logic cells. Comparators. DAC. *Openlane*.

ABSTRACT

CALSOLARI, R. O. **Study and evaluation of the use of digital circuits (standard logic cells) in the implementation of analog circuits.** 2024. Monografia (Trabalho de Conclusão de Curso) – Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2024.

This monograph describes the use of current open-source tools to development of integrated circuits - *Openlane*, *Xschem*, *Ngspice* - with the circuit analysis of comparators and digital-to-analog converter, implemented with only logic cells. The project was developed on *Windows* with *Linux* environment through WSL. The tools have been installed in this environment, through the library of Institute for Integrated Circuits from *Johannes Kepler University of Linz*, and the circuits implemented for its evaluation. The proposed comparator and digital-to-analog converter circuits were implemented and tested for different supply voltages, and their analysis was discussed in the project.

Keywords: *Spice*. Digital-to-Analog Converter. Standard logic cells. Comparators. DAC. *Openlane*

LISTA DE ILUSTRAÇÕES

Figura 1 – Publicações do *TCAS* sobre Circuitos analógicos implemetados por circuitos digitais ao longo dos anos 20
Figura 2 – Topologia proposta do circuito completo 21
Figura 3 – Topologia proposta do conversor digital-analógico 22
Figura 5 – Tabela verdade da porta *NAND* 23
Figura 4 – Topologia proposta do comparador de portas NAND 23
Figura 6 – Topologia proposta do comparador de portas AND-OR 24
Figura 7 – Topologia proposta do comparador utilizando MUX 24
Figura 8 – Fluxo de trabalho do *software* 26
Figura 9 – Comparador NAND implementado no software Xschem 29
Figura 10 – Formas de onda do comparador NAND para tensão $V_{in}M$= 0.6V 30
Figura 11 – Formas de onda do comparador NAND para tensão $V_{in}M$= 1.2V 31
Figura 12 – Comparador AND-OR implementado no software Xschem 31
Figura 13 – Formas de onda do comparador AND-OR para uma tensão de referência = 0.6V 32
Figura 14 – Formas de onda do comparador AND-OR com a tensão de referência = 0.9V 33
Figura 15 – Comparador por meio de multiplex 34
Figura 16 – Formas de onda do comparador Multiplex com uma tensão de referência = 0.6V 34
Figura 17 – Formas de onda de saída dos três comparadores com uma tensão de referência = 0.6V e 0.9V 35
Figura 18 – Circuito do conversor digital-analógico 36
Figura 19 – Variação dos 32 níveis de tensão de entrada e sua saída 37
Figura 20 – Senóide produzida pelos 5 bits do conversor e a melhor senóide próxima a esses valores 37
Figura 21 – Senóide produzida pelos 5 bits do conversor com um capacitor de 10pF em sua saída e a melhor senóide próxima a esses valores 38
Figura 22 – Conjunto das formas de onda da FFT do conversor com e sem capacitor em sua saída 39

LISTA DE TABELAS

Tabela 1 – Tensão de saída do conversor para entrada de 5 bits 40

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
ADC	Analog-to-digital converter
CI	Circuito integrado
CMOS	*Complementary Metal-Oxide-Semiconductor*
DAC	Digital-to-analog converter
EDA	Electronic Design Automation
EESC	Escola de Engenharia de São Carlos
FFT	Transformada rápida de *Fourier*
GTKWave	*Software* para visualização de formas de onda
HDL	Linguagem de Descrição de *Hardware*
IoT	*Internet of things*
LED	Diodo emissor de luz
LSB	*Bit* menos significativo
Magic	Software do circuito integrado e sua manipulação em nanoescala
MSB	*Bit* mais significativo
MUX	Circuito multiplexador
ngspice	*Software* de simulação e análise do comportamento de circuitos elétricos
Open source	Código aberto - Aplicações que têm seu código aberto e disponível para qualquer pessoa que pretenda modificar e até redistribuir o software
PDF	Portable Document Format
PLL	*phase-locked loop*
PMOS	P-type metal oxide semiconductor
RF	Rádio-frequência
RTL2GDS	*Register Transfer Level to Graphic Design System*
TCC	Trabalho de Conclusão de Curso

TCAS	IEEE Transactions on Circuits and Systems
USP	Universidade de São Paulo
USPSC	Campus USP de São Carlos
Vcm	Input common mode voltage
Xschem	Editor esquemático para designs de circuitos personalizados

SUMÁRIO

1 INTRODUÇÃO 19

2 REVISÃO BIBLIOGRÁFICA 21
2.1 Conversor Digital-Analógico 21
2.2 Circuitos comparadores 22
2.2.1 Circuito comparador de portas NAND 23
2.2.2 Circuito comparador de portas AND-OR 24
2.2.3 Circuito comparador MUX 24

3 MATERIAIS E MÉTODOS 25
3.1 Utilização do ambiente *WSL* do Windows 25
3.2 Processo de *design - PDK* a 130nm da indústria de semicondutores 25
3.3 Utilização do *Xschem* 27

4 DESENVOLVIMENTO 29
4.1 Análise dos circuitos e Resultados 29
4.1.1 Implementação do circuito de portas *NAND* 29
4.1.2 Comparador de portas AND-OR 31
4.1.3 Comparador utilizando multiplexador 33
4.1.4 Saída dos comparadores 34
4.1.5 Conversor digital-analógico 36
4.1.5.1 Teste do conversor para uma senóide 36
4.1.5.2 Transformada de Fourier das Senóides 38
4.2 Circuitos propostos 40
4.3 Avaliação das ferramentas utilizadas no desenvolvimento 42
4.3.1 Openlane 42
4.3.2 WSL 43
4.3.3 Xchem e Ngspice 44

5 CONCLUSÃO 47

REFERÊNCIAS 49

APÊNDICE 51
5.1 Circuito comparador de portas *NAND* 55
5.2 Circuito comparador de portas *AND-OR* 55
5.3 Comparador MUX 56
5.4 Circuito do conversor digital-analógico 56

ANEXO 59

1 INTRODUÇÃO

A crescente demanda pela internet das coisas (IoT) tem exigido, nos últimos anos, a utilização de circuitos integrados (CI's) nos mais variados objetos, desde os mais simples, como uma lâmpada LED, aos mais complexos como televisores, relógios, computadores de bordo de automóveis, celulares (Smartphones). Este cenário, ainda, promete se tornar cada vez maior ao longo dos anos, estima-se que em 2025 a quantidade de dispositivos eletrônicos conectados possa chegar a 75 bilhões (TANWEER, 2018) para uso em diversas aplicações "inteligentes" como nas redes de iluminação municipal, agropecuária, comércio, saúde, hotelaria e indústria 4.0, graças ao potencial de conectividade e velocidade das redes 5G (Quinta geração da tecnologia de aparelhos telefônicos – celulares).

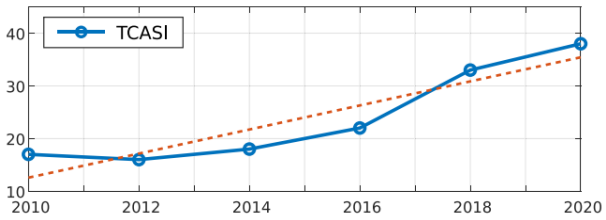
Um dos desafios da última década tem sido a dificuldade de se implementar circuitos analógicos que acompanhem essa tecnologia, uma vez que estes não possuem a vantagem de utilizar o ganho de escala geométrico (geometrical scaling) dos dispositivos *complemetary metal oxide semiconductor* - CMOS (CROVETTI, 2013), além de enfrentar desafios de design pelas características dos transistores em nanoescala (KINGET, 2015) e da oscilação de sinal para fontes de alimentação abaixo de 1V (PAN; SUN; XU, 2017).

Essas desvantagens, frente à tecnologia CMOS, refletem o impacto negativo em termos da área (tamanho do circuito), desempenho (como tempo de resposta - *delay*), eficiência energética (maior consumo de energia e consequentemente maiores baterias), além da dificuldade do *design* de células analógicas. Cumpre ressaltar que os CI's analógicos são caracterizados por uma limitada habilidade de reconfiguração e portabilidade para novas tecnologias se comparados aos CI's digitais, além de aumentar o custo para seu *design*, otimização a nivel de transístor, simulação, e prototipagem. (CROVETTI et al., 2019)

Uma alternativa que tem sido estudada nos últimos anos é a implementação de funções analógicas por meio de circuitos puramente digitais, com a finalidade de se beneficiar da vantagem de ganho de escala, além da facilidade de design pela tecnologia CMOS (TOLEDO et al., 2021). Pela figura 1 é possível observar a crescente demanda por pesquisas nessas áreas nos últimos anos através das publicações no IEEE Transactions on Circuits and Systems - *TCAS*.

Ao longo da última década, em razão dos parágrafos expostos, diversos circuitos analógicos foram estudados e implementados, como sensores de temperatura (ANAND; MAKINWA; HANUMOLU, 2016), amplificadores operacionais (TOLEDO et al., 2020), *buck converters* (KIM et al., 2015), osciladores (AIELLO et al., 2019), conversores digitais-analógicos (DACs) (AIELLO; CROVETTI; ALIOTO, 2019), *standard flash ADCs* (conversor analógico-digital) (WEAVER; HERSHBERG; MOON, 2014), *phase-locked loops* (PLLs) (DENG et al., 2015), comparadores de tensão (AIELLO; CROVETTI; ALIOTO, 2018) e (ZOU; NAKATAKE, 2020), e também em transmissores (PARK; WENTZLOFF, 2011) e receptores (OPTEYNDE, 2010) de rádio-frequência (RF).

Figura 1 – Publicações do *TCAS* sobre Circuitos analógicos implemetados por circuitos digitais ao longo dos anos



Fonte:(TOLEDO et al., 2021)

Com a crescente demanda para implementação desses circuitos, diversas ferramentas utilizadas para sua construção foram elaboradas, dentre elas, destacam-se o *xschem* e o *OpenLane*, as quais serão utilizadas para o desenvolvimento do projeto, e consequentemente sua avaliação. Estes *softwares*, por serem de código aberto - *Open source*, apresentam algumas vantagens aos desenvolvedores de circuitos integrados como:

- a adaptação do software às necessidades específicas do projeto e a modificação e melhoria das funcionalidades existentes;
- a capacidade de personalizar o software, inserindo ou retirando elementos para otimizar seu uso;
- a redução do custo do projeto para seu desenvolvimento;
- o fomento a um ambiente colaborativo com uma comunidade de desenvolvedores e usuários que auxiliam na identificação e correção de falhas, além do aprimoramento do *software*;
- a facilidade de integração entre diferentes ferramentas e fluxos de projeto

Assim, a presente monografia tem por objetivo realizar o estudo e análise sobre o funcionamento de um circuito puramente digital, composto por três comparadores e um conversor digital-analógico, em ambiente de simulação, e também a análise das ferramentas de código aberto para o desenvolvimento deste circuito, como o *ngspice*, o *xschem* e o *OpenLane*.

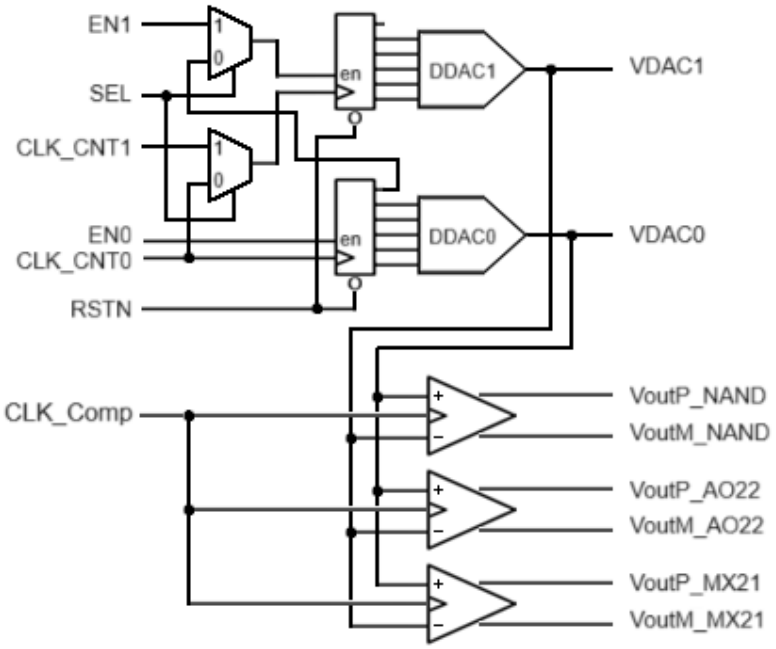
Desta forma, o trabalho está dividido em 5 (cinco) capítulos a saber: Introdução, compreende este capítulo, para contextualizar o tema, apresentar o problema e a organização do trabalho no decorrer do texto; Revisão Bibliográfica, capítulo em que os circuitos propostos e as ferramentas são apresentadas; Materiais e métodos, capítulo com a finalidade de detalhar os métodos utilizados no desenvolvimento do projeto; Desenvolvimento e análise de resultados, capítulo principal em que os circuitos são implementados e seus resultados são discutidos e; Conclusão, capítulo para sintetizar as principais descobertas do estudo e relacionar os resultados com os objetivos iniciais.

2 REVISÃO BIBLIOGRÁFICA

Como apresentado na introdução, a evolução tecnológica e a demanda por circuitos integrados em diversos dispositivos têm impactado na implementação de circuitos analógicos, pois estes não se beneficiam da tecnologia *CMOS*. Assim, uma forma de se contornar esse problema é a utilização de circuitos puramente digitais.

Para se realizar o estudo e o desenvolvimento dessa técnica, serão estudados três topologias de comparadores, baseados no trabalho de (SALA et al., 2023), e de um conversor digital para analógico (DAC), baseado no trabalho de (YANG et al., 2016). Estes circuitos foram submetidos para implementação na iniciativa TinyTapeout, website. O DAC implementado foi ampliado de 4 (quatro) para 5 (cinco) *bits*, e duas variações de comparadores foram implementadas, a partir do comparador baseado apenas em portas *NAND* de 2 (duas) entradas, proposto por (SALA et al., 2023), conforme apresentado na figura 2. Os contadores e multiplexadores da figura servem para controlar os conversores, os quais geram os valores de referência e de teste para os comparadores.

Figura 2 – Topologia proposta do circuito completo



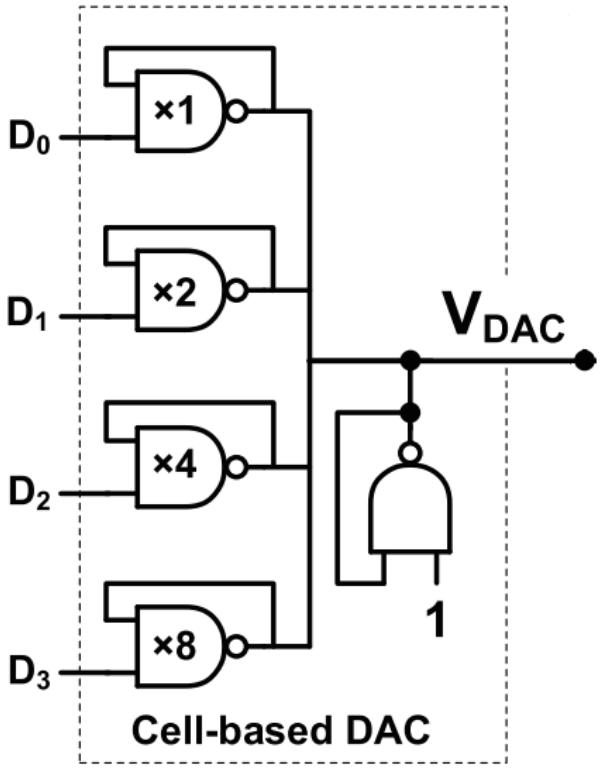
Fonte:(YANG et al., 2016) e (Maximiliam Luppe, 2024, Github)

2.1 Conversor Digital-Analógico

Com o objetivo de se testar os comparadores, os conversores digital-analógico são implementados com células-padrão para gerar os sinais rampa-analógicos em suas saídas, que se conectam às entradas dos comparadores.

O coversor digital-analógico pode ser visualizado na figura 3 a qual é baseada no trabalho de (YANG et al., 2016)

Figura 3 – Topologia proposta do conversor digital-analógico

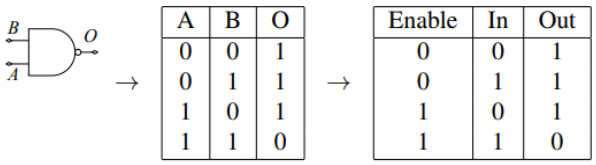


Fonte:(YANG et al., 2016)

Este conversor, de acordo com a figura, é composto de 4 (quatro) *bits*, contudo, sua implementação foi feita com 1 (um) *bit* a mais, possuindo duas portas *NAND x8* em série para fornecer o *bit* mais signiticativo - D4.

2.2 Circuitos comparadores

Os circuitos comparadores são essenciais para os conversores analógico-digital e digital-analógico. São eles os responsáveis pela tradução dos sinais do mundo analógico para o digital e viceversa (WEAVER; HERSHBERG; MOON, 2011). O primeiro circuito comparador é o de portas *NAND* conforme proposto por (SALA et al., 2023) e seu funcionamento é descrito a seguir. Os outros dois circuitos comparadores foram propostos pelo orientador, com a topologia parecida ao de portas *NAND*, mas composto de portas *AND-OR* e de um multiplexador - *MUX*.



A	B	O
0	0	1
0	1	1
1	0	1
1	1	0

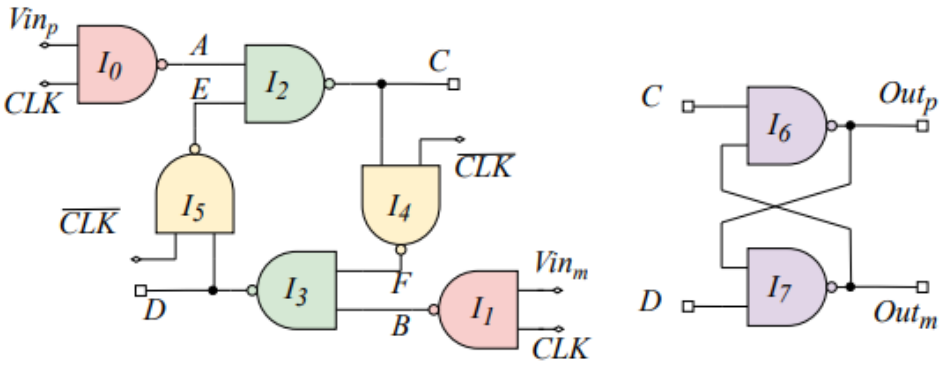
Enable	In	Out
0	0	1
0	1	1
1	0	1
1	1	0

Figura 5 – Tabela verdade da porta *NAND*

2.2.1 Circuito comparador de portas NAND

O comparador de portas *NAND* da figura 4 é composto por oito portas *NAND* de duas entradas, de tal forma que o sinal de entrada $V_{in}P$ e $V_{in}M$ é aplicado em uma das entradas das portas *NAND* I_0 e I_1, respectivamente. As portas I_2, I_3, I_4 e I_5 configuram a retroalimentação positiva a qual permite que o circuito seja carregado na fase de descida do clock, enquanto que na sua fase positiva o sinal diferencial aplicado na entrada é regenerado graças à retroalimentação. A porta *NAND* pode ser compreendida como um sinal de liga-desliga (*Enable*), do seguinte modo: quando o sinal está em nível lógico baixo (próximo ao *ground* - GND), a saída da porta está em nível lógico alto (próximo a tensão V_{DD}) e não se altera independente da variação do sinal de entrada. Por outro lado, quando o sinal *Enable* está em nível lógico alto (V_{DD}), o circuito se comporta como um inversor do sinal de entrada, como a figura 5 mostra sua tabela verdade.

Figura 4 – Topologia proposta do comparador de portas NAND



Fonte:(SALA et al., 2023)

A análise do circuito pode ser dividida em duas etapas:

1. CLK = 0, as portas I_0 e I_1 são pré-carregadas para o nível lógico um (V_{DD}), assim as portas I_2, I_3, I_4 e I_5 configuram a retroalimentação positiva que contém o dado armazenado.

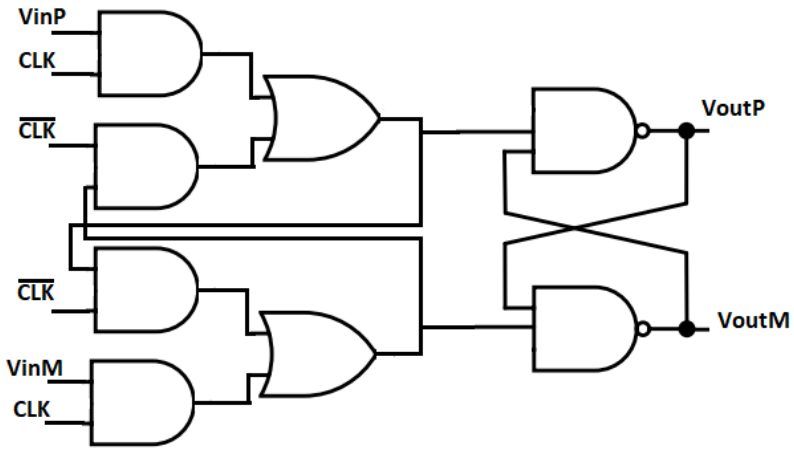
2. CLK = 1, as portas I_0 e I_1 se comportam como duas portas inversoras e amplificam o sinal de entrada $V_{in}P$ e $V_{in}M$ para a entrada das portas I_2 e I_3. As portas I_4 e I_5 estão desabilitadas pois utilizam o sinal de *Clock* invertido nas suas entradas.

Por fim, um circuito *latch* de portas *NAND* compostos pelas portas I_6 e I_7 são utilizadas para se armazenar os dados quando o CLK está em nível lógico zero.

2.2.2 Circuito comparador de portas AND-OR

De forma semelhante ao do circuito *NAND*, o circuito da figura 6 foi proposto para ter seu modo de funcionamento analisado. As 6 (seis) portas *NAND* foram substituídas por 2 (duas) portas *AND-OR*.

Figura 6 – Topologia proposta do comparador de portas AND-OR

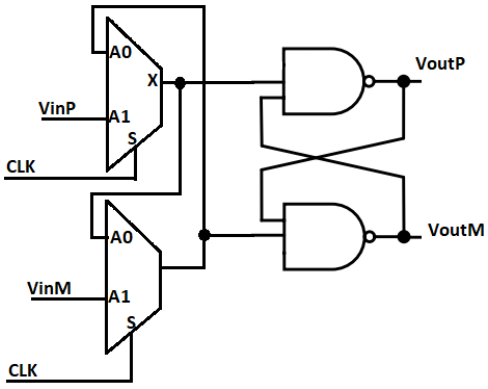


Fonte: Autor

2.2.3 Circuito comparador MUX

Por fim, o último circuito comparador é apresentado conforme a figura 7, o qual as portas lógicas são substituídas por dois multiplexadores, e o *latch* é mantido em suas saídas para guardar os sinais a serem analisados.

Figura 7 – Topologia proposta do comparador utilizando MUX



Fonte: Autor

3 MATERIAIS E MÉTODOS

3.1 Utilização do ambiente *WSL* do Windows

Como parte complementar a esta monografia, o trabalho foi todo desenvolvido no sistema operacional *Windows*, entretanto, os recursos foram utilizados através de uma distribuição do *Linux* - *Ubuntu* - a qual foi instalada por meio do ambiente *Windows Subsystem for Linux* - *WSL*.

Assim, com a distribuição do *Linux* instalada, instalou-se a biblioteca *Github*: Open Source integrated circuits multitool, a qual compreende, além dos recursos do *OpenLane*, uma série de ferramentas para análise e elaboração dos circuitos integrados:

- *Xschem*, compreende um editor esquemático para designs personalizados de circuitos *ASICs*, e *back-ends netlist* para VHDL, *Spice* e *Verilog*.
- Ngspice, simulador de circuitos eletrônicos de código aberto.
- GTKWave, *software* para visualização das formas de onda digitais e analógicas.
- Magic, *software* do circuito integrado e sua manipulação em nanoescala.

O passo-a-passo da instalação do Linux, por meio do WSL, e do *Openlane* está explicado no Apêndice desse trabalho.

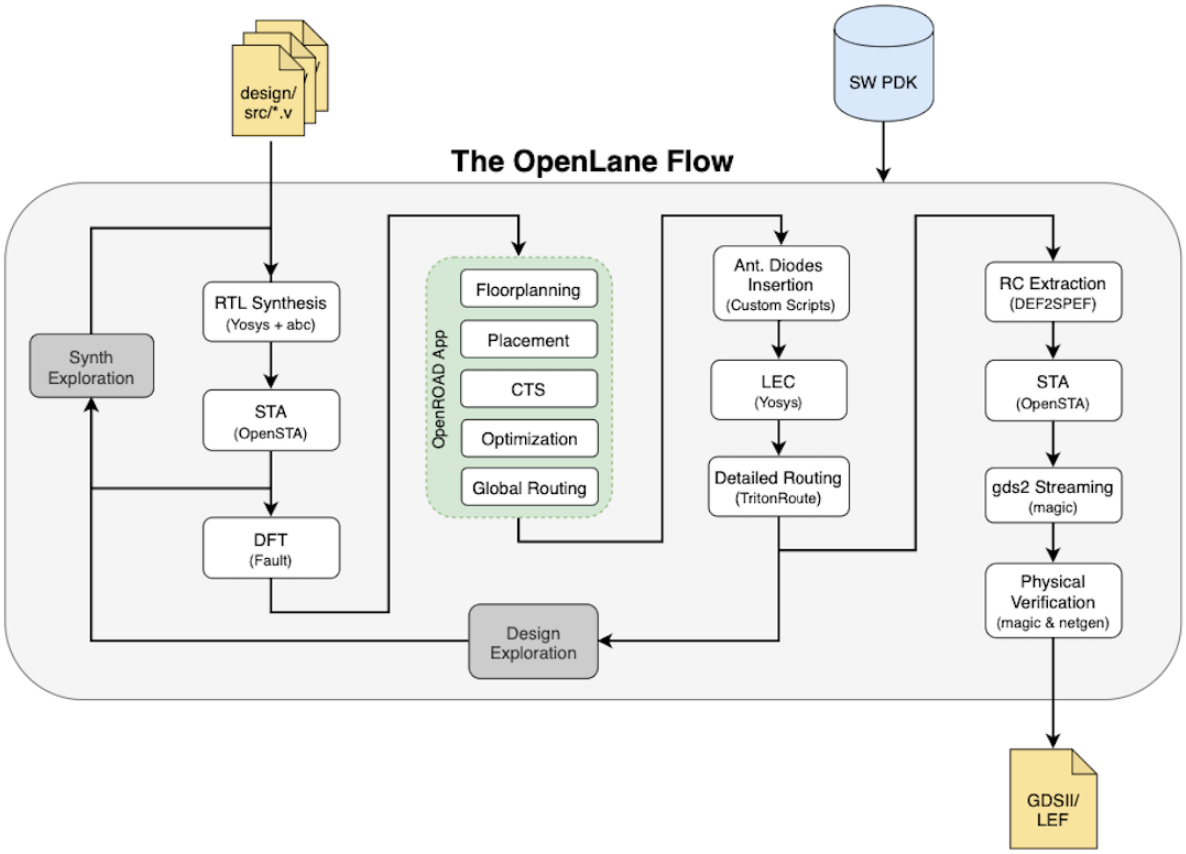
3.2 Processo de *design - PDK* a 130nm da indústria de semicondutores

Como motivação deste trabalho, em 2020, as empresas *Google*, *Skywater* e *Efabless* anunciaram uma parceria de um projeto *OpenSource*, com a finalidade de tornar acessível ao público a produção em cadeia de circuitos integrados de aplicações específicas (*ASICs*) personalizados, através da tecnologia *CMOS* de 130 nm da *Skywater*. A *Efabless*, então, através do projeto de código aberto *OpenLane* (SHALAN; EDWARDS, 2020), disponibilizou um fluxo (*Register Transfer Level to Graphic Design System - RTL2GDS*) que consiste em uma série de ferramentas na qual o usuário, por meio de uma linguagem de descrição de hardware - *HDL*, obtém ao final do fluxo o arquivo necessário utilizado na indústria para produção dos circuitos integrados, como no processo Sky130 PDK da *Skywater*. O fluxo de trabalho do *OpenLane* pode ser visualizado na figura 8.

O OpenLane é uma infraestrutura de automação para o fluxo de projeto de circuitos integrados digitais, desde o RTL (Register-Transfer Level) até o GDSII (formato final do layout). O caminho típico do OpenLane pode ser resumido nas seguintes etapas principais: Etapas do fluxo OpenLane:

1) Síntese

Figura 8 – Fluxo de trabalho do *software*



Fonte:(SHALAN; EDWARDS, 2020)

- Conversão do RTL (geralmente em Verilog) para netlist em nível de porta lógica
- Otimização lógica e mapeamento para células da biblioteca

2) Implementação Física

- Floorplanning: definição da área do chip e posicionamento de macroblocos
- Posicionamento: alocação das células no layout
- Roteamento global: planejamento geral das interconexões
- Roteamento detalhado: conexões finais entre células

3) Avaliação e Verificação

- Análise de timing
- Verificação de regras de projeto (DRC)
- Verificação de layout vs. esquemático (LVS)

4) Geração do GDSII

- Conversão do layout final para o formato GDSII

Do exposto, ao entrar com o código Verilog do projeto, conforme o Anexo, o *Openlane* gera o arquivo GDSII utilizado pelas indústrias para a produção do circuito.

3.3 Utilização do *Xschem*

Com o objetivo de simular o circuito proposto, o *software* foi utilizado para esquematizar o circuito, de maneira que este utiliza o software *Ngspice* para sua simulação. Convém destacar que ambas as ferramentas são gratuitas e de fácil utilização pelo usuário.

Foi utilizado o *Xschem* em detrimento do *LTspice*, em seu processo de escolha, pelo simples fato da biblioteca do *PDK Skywater 130nm* já estar toda implementada no *software*, em conjunto com o *Ngspice*. Assim, a elaboração do circuito esquemático e as simulações foram facilitadas e mais rápidas do que seria se todas as portas lógicas utilizadas fossem importadas para o *LTspice*, uma vez que ainda haveria a necessidade de se implementar as portas lógicas através dos seus NFET e PFET.

4 DESENVOLVIMENTO

O presente capítulo tem por objetivo demonstrar como o projeto foi executado, além de uma seção a respeito dos testes realizados e discussão de seus resultados.

As topologias indicadas nas figuras 4, 6, 7, 3 foram implementadas no ambiente *Xschem* para esquematização dos circuitos propostos, e simuladas através da linguagem *Spice* com o *Software Ngspice* através dos respectivos arquivos `Verilog` (IEEE. . . , 2024) apresentados no Anexo. Com a utilização da biblioteca *SkyWater 130* (Github: *Skywater*), os modelos em *Spice* dos operadores lógicos utilizados na topologia foram carregados nos respectivos programas para sua análise.

4.1 Análise dos circuitos e Resultados

4.1.1 Implementação do circuito de portas *NAND*

Por meio do *software Xschem* o circuito da figura 4 foi implementado para visualização, como apresentado na figura 9, e simulado para observar as características do circuito.

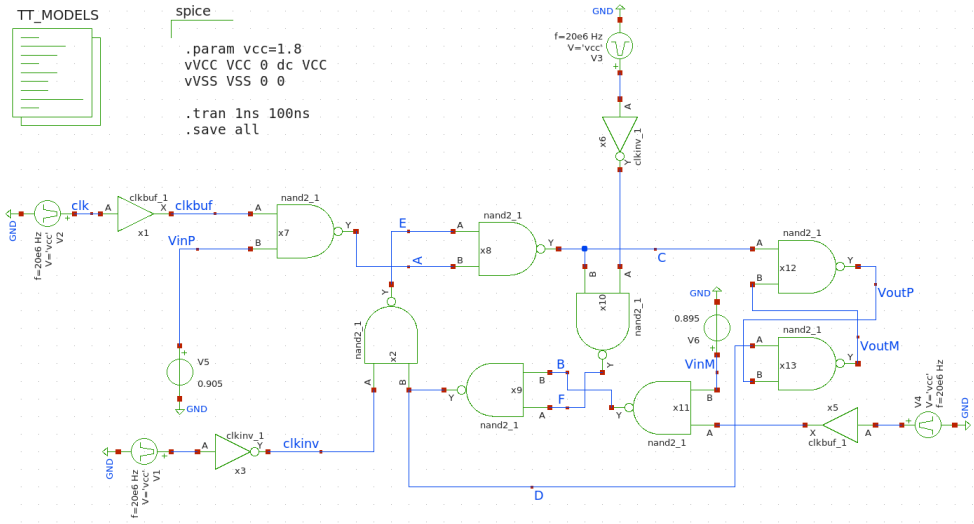


Figura 9 – Comparador NAND implementado no software Xschem

O circuito da figura 9 foi simulado com uma tensão de alimentação V_{DD} = 1.8V, um sinal rampa analógico variando linearmente de 0V a 1.8V para a tensão de entrada $V_{in}P$, em um intervalo de 300ns e uma tensão de entrada fixa (como referência) para diferentes valores 0V, 0.6V, 0.9V e 1.2V, para um *clock* de 40 MHz. As formas de onda das saídas do comparador, $V_{out}P$ e $V_{out}M$, bem como as tensões de entrada, $V_{in}P$ e $V_{in}M$ e o *clock* são apresentadas na figura 10 para a tensão de 0.6V.

Pela análise da figura é possível notar que o circuito, ao comparar as duas tensões de entrada $V_{in}P$ e $V_{in}M$, sendo esta a tensão de referência, determina suas tensões de saída $V_{out}P$ e $V_{out}M$. Quando a tensão de entrada positiva $V_{in}P$ é maior que a tensão de

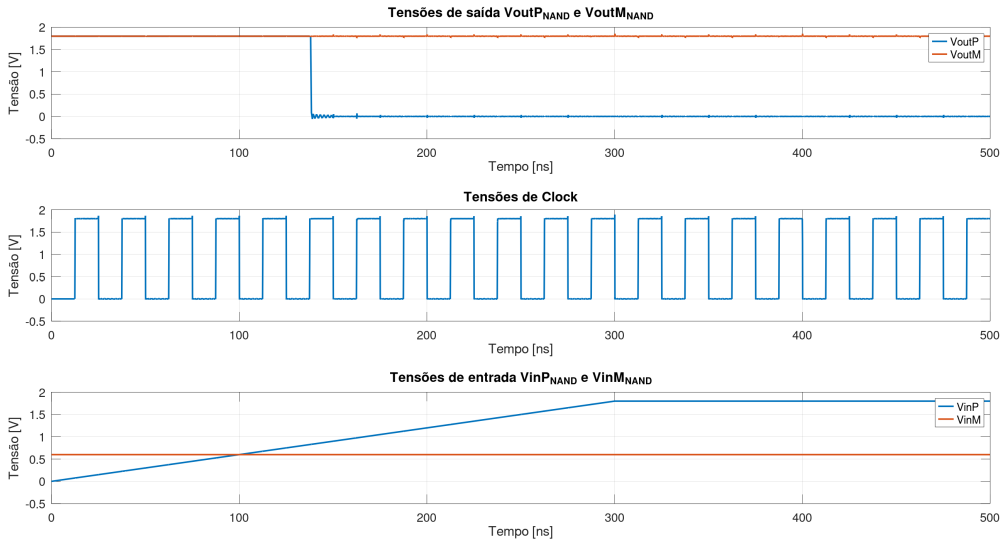


Figura 10 – Formas de onda do comparador NAND para tensão $V_{in}M$= 0.6V

entrada negativa $V_{in}M$, a saída do circuito $V_{out}P$ passa a ser zero, ao passo que a saída $V_{out}M$ é V_{DD} = 1.8V. Importante notar que a mudança da tensão de saída do circuito ocorre na transição para a fase de subida do *Clock*. Entretanto, cumpre observar que, para uma alimentação de V_{DD} = 1.8V, as portas *NAND* do circuito apresentam limite de tensão - *threshold voltage* - de 0.8V. Assim, quando a tensão de entrada $V_{in}P$ alcança esse valor, a porta *NAND* passa a ter um sinal lógico alto de entrada, e consequentemente o circuito passa a funcionar a partir desse limite.

Para as outras tensões de referência $V_{in}M$=0V, 0.9V e 1.2V, convém destacar que não houve diferença dos gráficos para as tensões de 0V e 0.9V, de modo que o sinal mudou de acordo com o limite de tensão = 0.8V para a tensão de entrada de 0V (na subida do *clock*), e conforme a fase de subida do *clock* para uma tensão $V_{in}P$ maior que $V_{in}M$ para a tensão de referência a 0.9V.

Contudo, para uma tensão de referência a 1.2V o circuito não apresentou um funcionamento correto, de modo que não houve transição do sinal de saída $V_{out}P$ quando a tensão de entrada $V_{in}P$ passa a tensão de referência $V_{in}M$, conforme demonstrado na figura 11.

O circuito ainda foi testado para diferentes tensões de alimentação V_{DD} menores que 1.8 V para observar o limite ao qual o comparador está sujeito, de modo que a partir de uma tensão abaixo de 1 V, o circuito não apresentou uma resposta esperada para suas saídas.

Este fato pode ser explicado quando a tensão de modo comum não é alta o suficiente para que os transístores PMOS trabalhem na região de corte, ou seja, ela deve ser o suficiente para que os PMOS conectados a entrada não estejam em funcionamento.

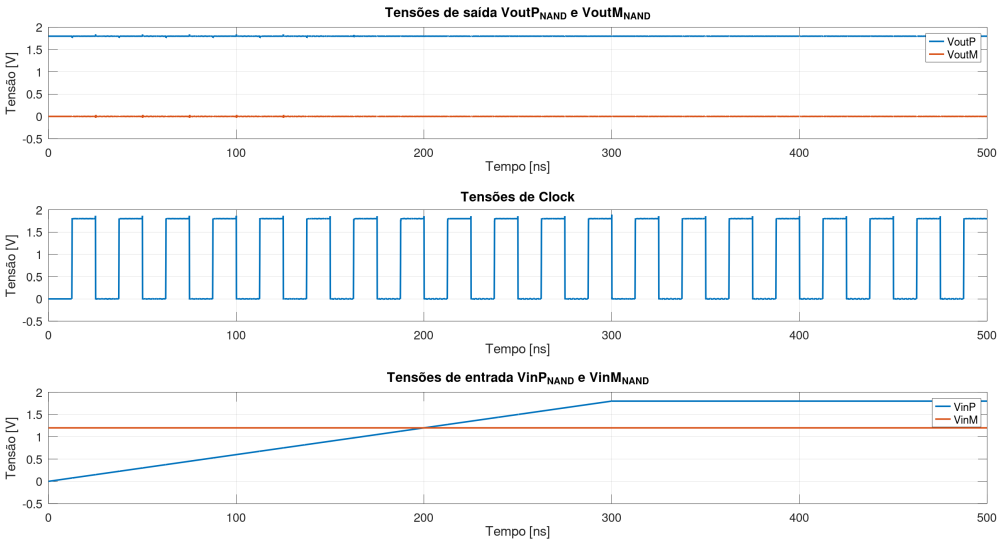


Figura 11 – Formas de onda do comparador NAND para tensão $V_{in}M$= 1.2V

4.1.2 Comparador de portas AND-OR

De maneira análoga, o circuito de portas AND-OR foi implementado pelo *software Xschem* e simulado junto ao *Ngspice* para as mesmas configurações do teste anterior com o circuito *NAND*. O circuito é apresentado na figura 12, e suas formas de onda podem ser visualizadas na figura 13 para uma tensão de referência de 0.6V.

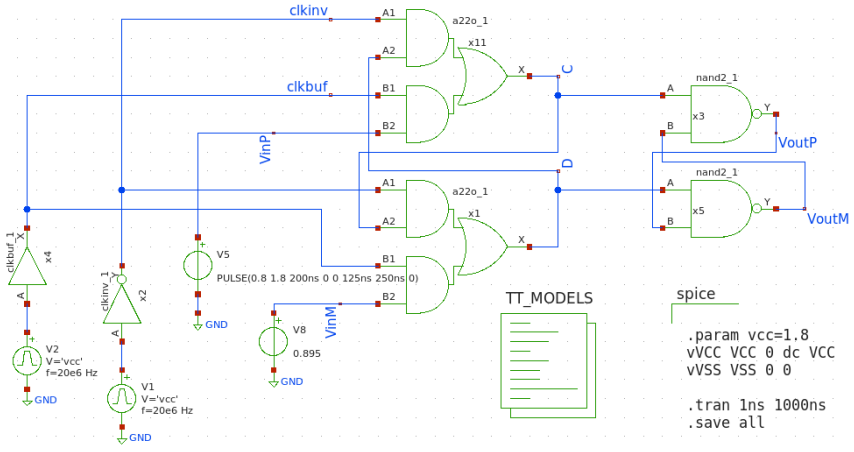


Figura 12 – Comparador AND-OR implementado no software Xschem

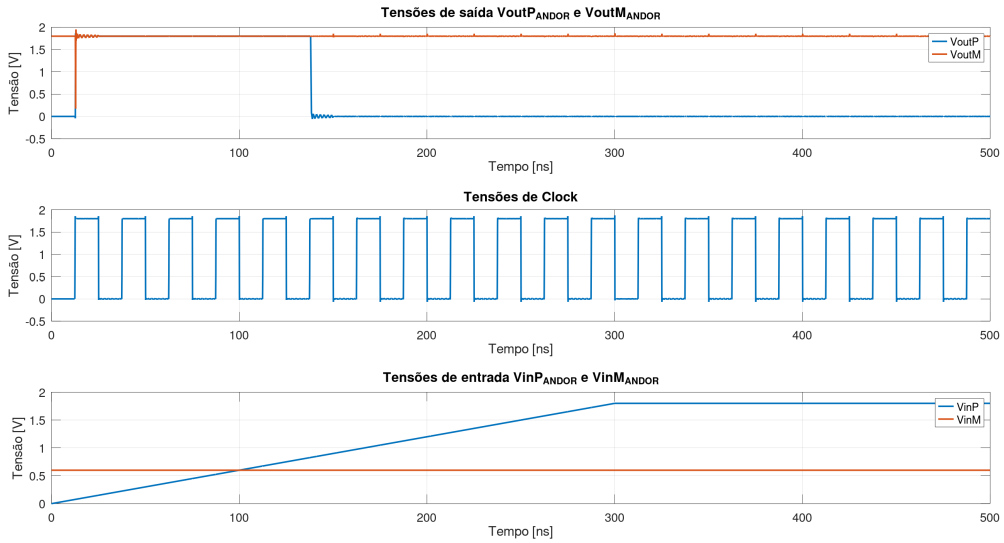


Figura 13 – Formas de onda do comparador AND-OR para uma tensão de referência = 0.6V

A análise da figura 13 mostra que o circuito funcionou de maneira semelhante ao circuito de portas *NAND* para as mesmas configurações, a saber: sinal de entrada rampa analógico variando linearmente de 0V a 1.8V para a tensão de entrada $V_{in}P$, em um intervalo de 300ns e uma tensão de entrada fixa (como referência) para diferentes valores 0V, 0.6V, 0.9V e 1.2V, para um *clock* de 40 MHz.

É possível perceber que a tensão de saída $V_{out}P$ é atualizada na fase de subida do *Clock*, observando sempre a tensão limite de 0.8V para considerar a transição de sinal lógico da porta.

De maneira semelhante ao circuito da figura 9, o circuito apresentou, também, correto funcionamento para sua tensão de referência a 0V. Por outro lado, para as referências de 0.9V e 1.2V o circuito não apresentou correto funcionamento. A explicação dessa observação se deve ao fato da tensão da entrada negativa (referência) ser sempre alta, para estes valores, pois está acima da tensão limite de operação do circuito - *threshold voltage* = 0.8V resultando sempre em um nível lógico alto para a saída **D** da porta AND-OR x_1.

Pela análise da figura 14 é possível observar o comportamento do circuito para a tensão de referência $V_{in}M$= 0.9V a qual não corresponde a comparação entre as tensões como demonstra a tensão de saída $V_{out}P$.

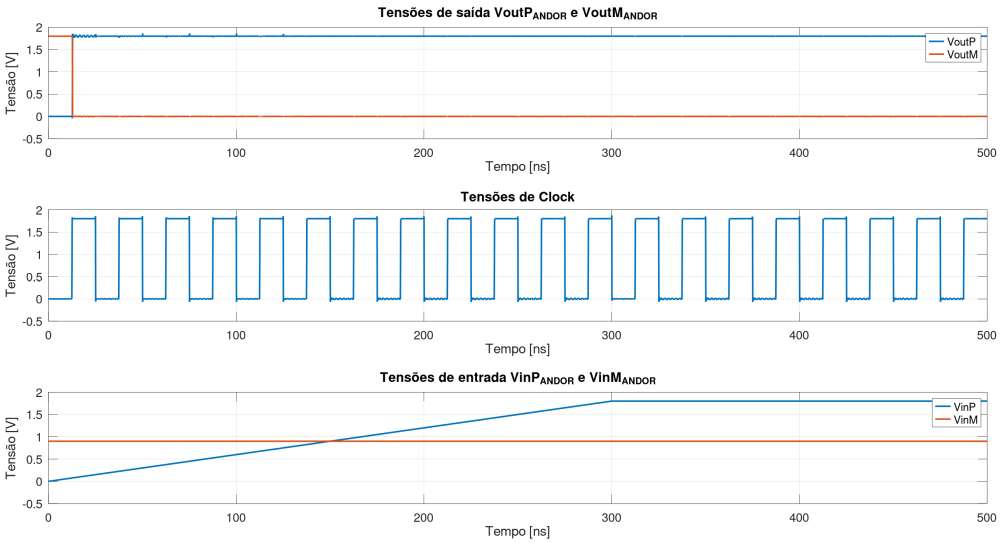


Figura 14 – Formas de onda do comparador AND-OR com a tensão de referência = 0.9V

Ainda, o circuito foi testado para diferentes configurações de alimentação menores que 1.8 V, não apresentando um funcionamento correto para tensões abaixo de 1V, deixando de comparar a tensão de entrada com a referência. Dito de outro modo, o circuito apresentou comportamento semelhante ao comparador de portas *NAND* para diferentes tensões de alimentação. A razão para este fato pode ser explicada pela tensão limite para a transição do nível lógico das portas do circuito, de modo que a porta *AND-OR* manteria o mesmo comportamento para diferentes níveis de tensão.

4.1.3 Comparador utilizando multiplexador

O último circuito comparador corresponde ao multiplex, o qual foi simulado de maneira análoga aos comparadores anteriores, com uma tensão de alimentação V_{DD} = 1.8V, sinal de entrada rampa analógico variando linearmente de 0V a 1.8V para a tensão de entrada $V_{in}P$, em um intervalo de 300ns e uma tensão de entrada fixa (como referência) para diferentes valores 0V, 0.6V, 0.9V e 1.2V, para um *clock* de 40 MHz. O circuito comparador multiplex pode ser visualizado na figura 15 e suas formas de onda na figura 16.

Do mesmo modo que no circuito *AND-OR*, a análise do gráfico da figura 16 mostra que o circuito funcionou corretamente para as configurações de teste, apresentando comportamento semelhante ao do circuito *AND-OR*. Ademais, o circuito foi testado para valores menores que 1.8 V de alimentação, apresentando as mesmas características do circuito anterior.

De modo semelhante ao da figura 12, o circuito MUX, para as tensões de 0.9V e 1.2V de $V_{in}M$, possui tensão da entrada negativa (referência) sempre alta, para estes valores, pois está acima da tensão limite de operação do circuito - *threshold voltage* = 0.8V resultando

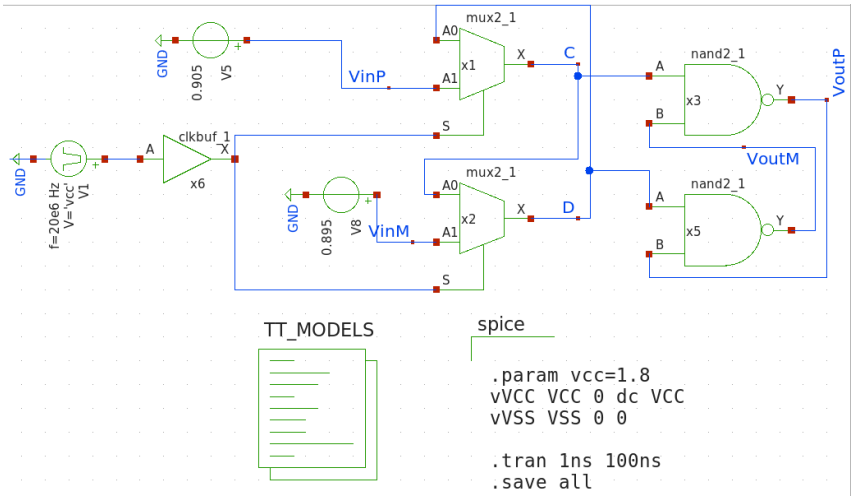


Figura 15 – Comparador por meio de multiplex

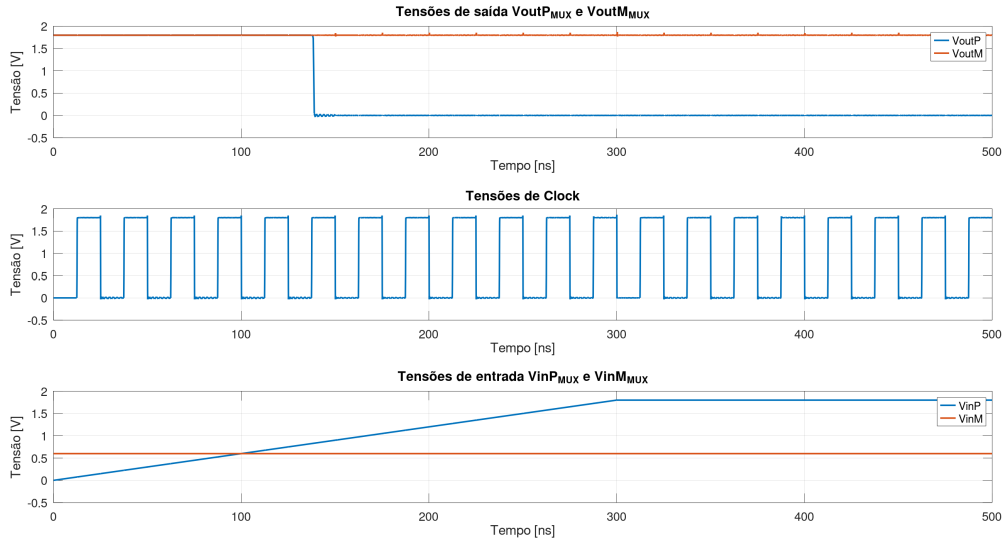


Figura 16 – Formas de onda do comparador Multiplex com uma tensão de referência = 0.6V

sempre em um nível lógico alto para a saída **D** do MUX x_2. De modo que seu funcionamento resta prejudicado.

4.1.4 Saída dos comparadores

Com o objetivo de se resumir os testes realizados, a figura 17 mostra as tensões de saída $V_{out}P$ e $V_{out}M$ para os 3 (três) comparadores do circuito estudado, para duas tensões de referência $V_{in}M$ de 0.6V e 0.9V. É notório que os três comparadores apresentam funcionamento semelhante para uma tensão de referência variando de $V_{in}M$ = 0V até o limite - *threshold* - das portas lógicas = 0.8V.

Assim, para tensões acima desse limite, os comparadores *AND-OR* e MUX apresentam

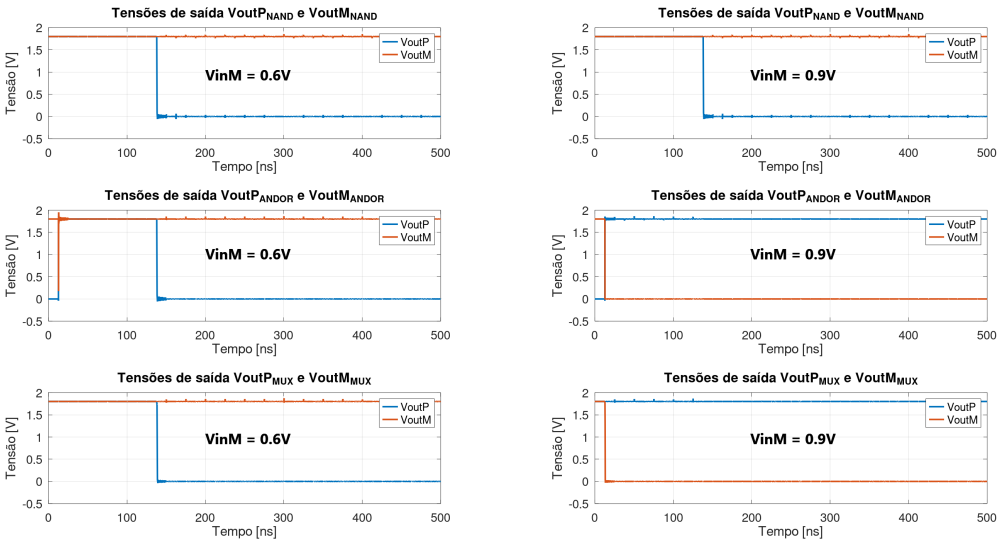


Figura 17 – Formas de onda de saída dos três comparadores com uma tensão de referência = 0.6V e 0.9V

tensão sempre alta para a saída D. Isso pode ser observado na mesma figura 17. Fica claro, assim, que o circuito deve operar com uma tensão de referência abaixo de 0.8V para seu correto funcionamento, com exceção do comparador *NAND* o qual apresenta bons resultados até uma tensão de 1V.

4.1.5 Conversor digital-analógico

O circuito do conversor digital-analógico da figura 3 foi implementado no *Xschem* como mostra a figura 18. O conversor possui 5 *bits* de entrada, perfazendo um total de 32 níveis de tensão. Esse circuito é composto por 7 (sete) portas *NAND* de duas entradas *(x1 a x7)* de diferentes tamanhos, relativamente aos bits mais significativos.

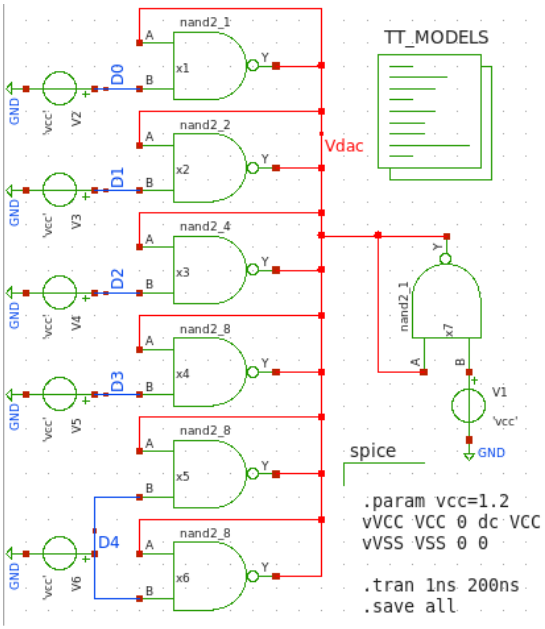


Figura 18 – Circuito do conversor digital-analógico

Com a finalidade de se testar o circuito, a figura 19 exibe, para uma tensão de alimentação do circuito de 1.8 V, os pulsos gerados para os bits, num intervalo de 320 ns, e a forma de onda da tensão de saída *Vdac* do conversor.

É notório perceber que, de acordo com a biblioteca do *PDK Skywater 130nm*, as portas *NAND* têm um tempo de resposta de 1 ns, como se observa nas variações bruscas de tensão nas transições entre os *bits* de entrada.

Ainda pela análise da tabela 1 nota-se uma variação média de 30 mV para cada nível de tensão associado, e uma variação máxima total de 995 mV, sendo a menor tensão de 0.805 V, o suficiente para testar os comparadores sem maiores problemas.

O circuito foi testado para diferentes tensões de alimentação menores que 1.8 V, de maneira que, a partir de uma tensão de 1 V de alimentação, o circuito apresentou um comportamento anormal para sua entrada digital mais significativa *11111*, e uma tensão mínima **média** de 0.5 V.

4.1.5.1 Teste do conversor para uma senóide

Com o objetivo de se testar o conversor, uma senóide de 10,47MHz foi gerada através de seus 5 *bits* de entrada e comparada com a melhor função seno descrita pelos pontos

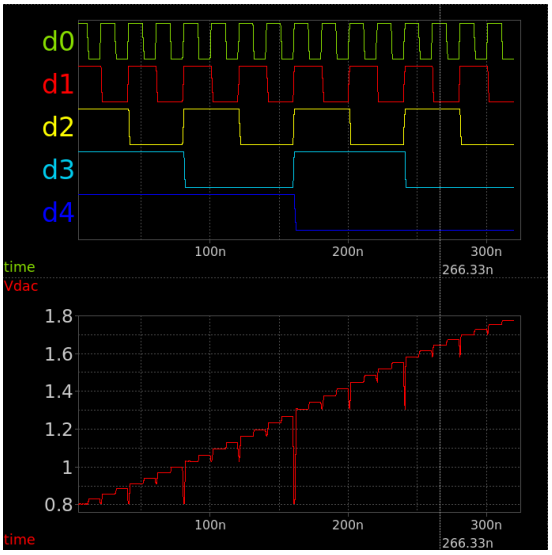


Figura 19 – Variação dos 32 níveis de tensão de entrada e sua saída

da tabela 1. As formas de onda da senóide gerada pelo conversor e da melhor senóide que passam pelos pontos utilizados podem ser visualizadas na figura 20. Pela análise da figura, é de se observar que o conversor, a princípio, produz um bom resultado, contudo, a fim de melhor demonstrar suas características, a função raiz do erro quadrático médio foi calculada para os valores do gráfico.

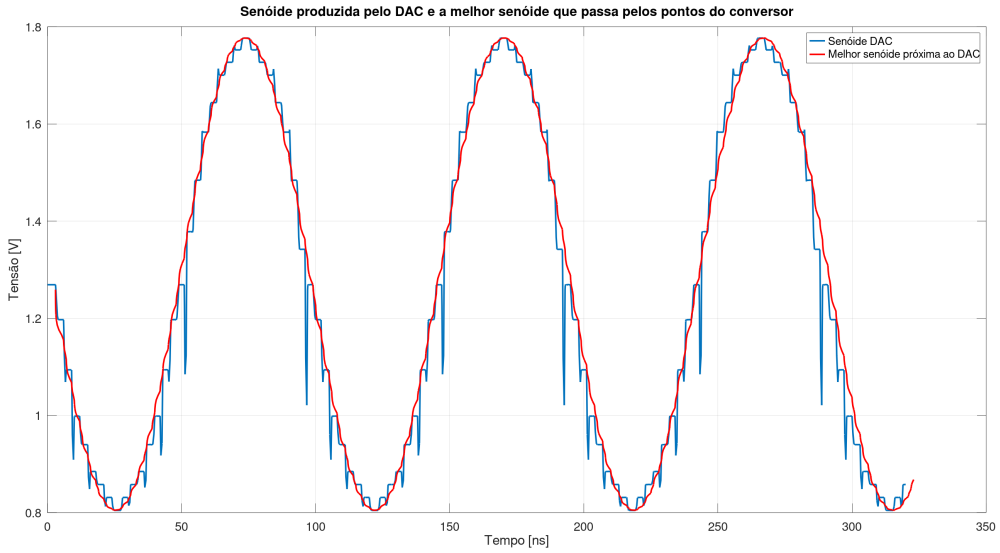


Figura 20 – Senóide produzida pelos 5 bits do conversor e a melhor senóide próxima a esses valores

O erro quadrático médio pode ser calculada pela raiz da média da diferença dos valores observados e esperados elevados ao quadrado conforme a equação 4.1, e o valor obtido para a raiz do erro quadrático médio foi $RMSE = 0.070054$.

$$RMSE = (media((V_{observado} - V_{esperado})^2))^{1/2} \tag{4.1}$$

Com a finalidade de se comparar a senóide na entrada do conversor, o circuito do conversor foi, então, simulado com um capacitor em sua saída para se testar uma carga e as atenuações das transições dos *bits* do conversor. As capacitâncias de 1 pF a 80 pF foram testadas e, conforme a capacitância aumentava, a senóide perdia amplitude e aumentava seu atraso, apesar da curva se tornar menos sensível às transições dos *bits*. Por outro lado, quando a capacitância diminuía o circuito ficava mais sensível às transições dos bits. Assim, a capacitância de 10 pF se tornou um bom valor para balancear essas características e sua senóide pode ser vista na figura 21. É possível observar como o capacitor reduziu significativamente os erros gerados pelas transições dos *bits* das portas NAND.

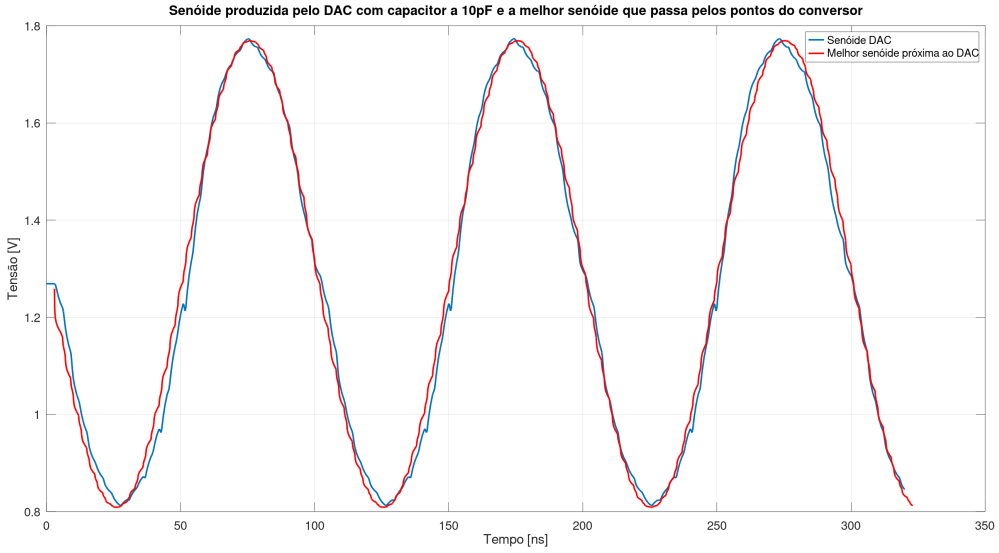


Figura 21 – Senóide produzida pelos 5 bits do conversor com um capacitor de 10pF em sua saída e a melhor senóide próxima a esses valores

O erro quadrático médio também foi calculado para o circuito com capacitor correspondendo a $RMSE = 0.079754$. Apesar do valor ter sido maior que o do conversor sem o capacitor em sua saída (quanto menor o erro, melhor o resultado), o valor é considerado bom, uma vez que o erro quadrático médio depende da ordem de grandeza do que está sendo testado. Como o conversor opera entre 0.8 V a 1.8 V, a ordem de grandeza do erro quadrático médio é 10^{-2} vezes menor.

4.1.5.2 Transformada de Fourier das Senóides

Por fim, as transformadas rápidas de *Fourier* (FFT) foram calculadas para as senóides do circuito conversor com e sem o capacitor em sua saída com o objetivo de se comparar

suas componentes harmônicas. Pela figura 22 é possível observar uma leve atenuação das componentes harmônicas quando o capacitor é adicionado. Um maior conjunto de níveis de tensão (DAC com mais *bits*) seria necessário para uma análise mais detalhada da FFT do sinal.

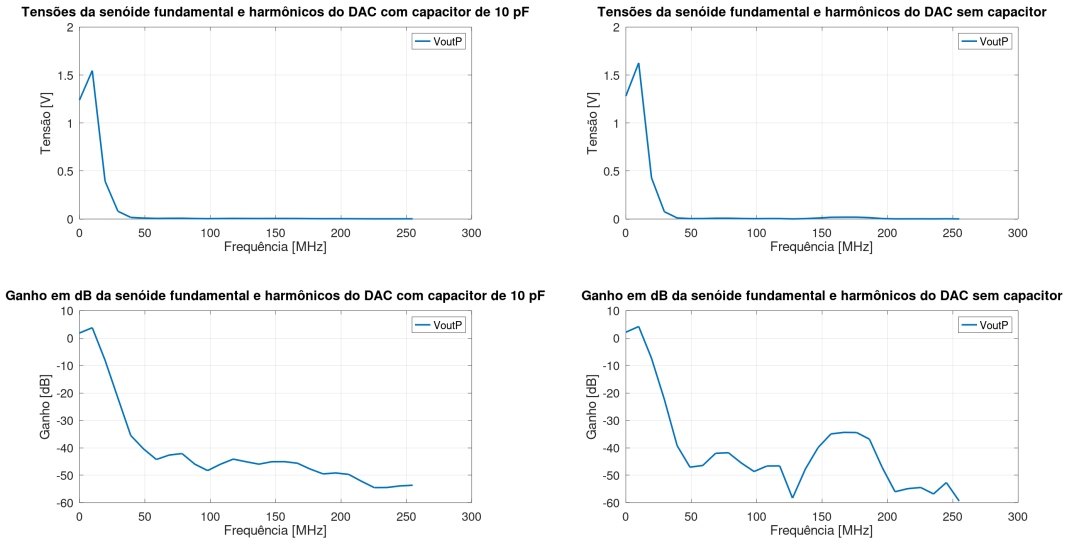


Figura 22 – Conjunto das formas de onda da FFT do conversor com e sem capacitor em sua saída

Tabela 1 – Tensão de saída do conversor para entrada de 5 bits

Entrada Binária (D4D3D2D1D0)	Decimal	Tensão [V]	Variação [mV]
00000	0	1.7767	23.3
00001	1	1.7523	24.4
00010	2	1.7268	25.5
00011	3	1.7003	26.5
00100	4	1.6726	27.7
00101	5	1.6439	28.7
00110	6	1.6140	29.9
00111	7	1.5831	30.9
01000	8	1.5511	32.0
01001	9	1.5181	33.0
01010	10	1.4841	34.0
01011	11	1.4494	34.7
01100	12	1.4141	35.3
01101	13	1.3782	35.9
01110	14	1.3419	36.3
01111	15	1.3055	36.4
10000	16	1.2691	36.4
10001	17	1.2328	36.3
10010	18	1.1971	35.7
10011	19	1.1620	35.1
10100	20	1.1275	34.5
10101	21	1.0940	33.5
10110	22	1.0612	32.8
10111	23	1.0295	31.7
11000	24	0.9987	30.8
11001	25	0.9689	29.8
11010	26	0.9399	29.0
11011	27	0.9121	27.8
11100	28	0.8846	27.5
11101	29	0.8578	26.8
11110	30	0.8312	26.6
11111	31	0.8050	26.2

4.2 Circuitos propostos

O uso de comparadores de tensão dinâmicos puramente digitais no desenvolvimento de circuitos integrados oferece uma série de vantagens e desafios. Este tipo de comparador é crucial em aplicações que exigem alta velocidade e baixo consumo de energia, como conversores analógico-digitais (ADCs).

Benefícios dos Comparadores Dinâmicos

1. Alta Velocidade: Comparadores dinâmicos são capazes de operar em altas velocida-

des devido ao uso de realimentação positiva e sinais de clock, o que os torna ideais para aplicações em ADCs de alta frequência (GUPTA; SINGH; AGARWAL, 2021).

2. Baixo Consumo de Energia: Eles são projetados para consumir menos energia do que seus equivalentes estáticos, uma vez que não permanecem continuamente ativos. Isso é particularmente vantajoso em dispositivos portáteis e sistemas embarcados onde a eficiência energética é crítica.

3. Redução de Ruído: A arquitetura dinâmica ajuda a mitigar problemas como ruído de kickback, que é comum em comparadores estáticos (GUPTA; SINGH; AGARWAL, 2021).

Desafios e Limitações

1. Complexidade do Projeto: O design de comparadores dinâmicos pode ser mais complexo devido à necessidade de gerenciar cuidadosamente o feedback positivo e o clocking para evitar instabilidades.

2. Sensibilidade ao Ruído: Apesar das melhorias, comparadores dinâmicos ainda podem ser sensíveis a ruídos nos sinais de entrada, o que pode levar a disparos falsos se não forem projetados adequadamente.

3. Compromisso entre Resolução e Consumo: Aumentar a resolução frequentemente resulta em maior consumo de energia, exigindo um equilíbrio cuidadoso no design do circuito (GUPTA; SINGH; AGARWAL, 2021).

4.3 Avaliação das ferramentas utilizadas no desenvolvimento

4.3.1 Openlane

O desenvolvimento de circuitos integrados através do OpenLane apresenta uma série de benefícios e desafios que merecem consideração, dentre suas vantagens destacam-se:

1. Automação e simplicidade: O fluxo inteiro é configurado através de um único arquivo de configuração, reduzindo a necessidade de intervenção manual.
2. Código aberto: Sendo uma plataforma de código aberto, o OpenLane não tem custos de licenciamento, tornando-o acessível para pesquisadores, estudantes e pequenas empresas.
3. Redução de tempo e expertise: O OpenLane diminui o tempo e o nível de especialização necessários para obter o layout final em formato GDSII.
4. Flexibilidade e extensibilidade: A infraestrutura permite a customização do fluxo para atender necessidades específicas, através de scripts Python e utilitários.
5. Compatibilidade: O OpenLane 2 mantém alta compatibilidade com designs e arquivos de configuração existentes, facilitando a transição de versões anteriores: Efabless.

Apesar dos benefícios, o OpenLane também apresenta algumas **desvantagens**:

1. Controle limitado: Comparado a ferramentas comerciais, o OpenLane oferece menos controle sobre o fluxo de desenvolvimento.
2. Otimização de tempo: Ferramentas comerciais geralmente proporcionam melhor otimização de tempo no processo de desenvolvimento.
3. Uso de células lógicas: O OpenLane tende a utilizar mais células lógicas no design, o que pode impactar a eficiência do circuito.
4. Consumo de energia: Os designs gerados pelo OpenLane tendem a consumir mais energia, o que pode ser uma preocupação em aplicações de baixo consumo.
5. Complexidade de instalação: Versões anteriores do OpenLane apresentavam um processo de instalação complicado, embora isso tenha sido melhorado nas versões mais recentes: Efabless.

O OpenLane representa um avanço significativo na democratização do desenvolvimento de circuitos integrados. Sua natureza de código aberto e fluxo automatizado o tornam uma ferramenta valiosa para educação, pesquisa e prototipagem rápida. No entanto, para aplicações que exigem otimização extrema de desempenho ou consumo de energia, ferramentas comerciais ainda podem oferecer vantagens.

A evolução contínua do OpenLane, evidenciada pela transição para o OpenLane 2, demonstra o compromisso da comunidade em melhorar a plataforma. Com o tempo, espera-se que muitas das limitações atuais sejam superadas, tornando o OpenLane uma opção cada vez mais atraente para o desenvolvimento de circuitos integrados em diversos contextos.

Do exposto, conclui-se que o OpenLane é uma ferramenta promissora que, apesar de suas limitações, oferece um caminho acessível e flexível para o desenvolvimento de circuitos integrados, contribuindo significativamente para a inovação e educação neste campo da engenharia eletrônica.

4.3.2 WSL

A utilização do Windows Subsystem for Linux (WSL) com Ubuntu para o desenvolvimento de projetos no OpenLane representa uma abordagem inovadora e eficaz para engenheiros, estudantes e pesquisadores que trabalham em ambientes Windows.

Vantagens da integração WSL-OpenLane

1. Flexibilidade de plataforma: O WSL permite aos usuários de Windows acessar ferramentas e fluxos de trabalho baseados em Linux sem a necessidade de dual boot ou máquinas virtuais tradicionais.

2. Desempenho otimizado: A integração direta do WSL com o kernel do Windows resulta em um desempenho superior em comparação com soluções de virtualização convencionais.

3. Ambiente de desenvolvimento unificado: A capacidade de executar o OpenLane em um ambiente Linux, mantendo o acesso às ferramentas e recursos do Windows, cria um fluxo de trabalho mais integrado e eficiente.

4. Facilidade de configuração: A instalação do Ubuntu via WSL e a subsequente configuração do OpenLane são processos relativamente simples, reduzindo barreiras técnicas para iniciantes.

Desafios e considerações

1. Curva de aprendizado: Usuários não familiarizados com sistemas Linux podem enfrentar uma curva de aprendizado inicial ao navegar no ambiente Ubuntu dentro do WSL.

2. Limitações de hardware: Embora o WSL 2 ofereça melhor desempenho, ainda pode haver algumas limitações em comparação com uma instalação nativa de Linux, especialmente em tarefas intensivas de processamento.

3. Compatibilidade de ferramentas: Nem todas as ferramentas Linux são totalmente compatíveis com o WSL, o que pode exigir soluções alternativas ou configurações adicionais em alguns casos.

A utilização do WSL com Ubuntu para executar o OpenLane demonstrou ser uma solução viável e eficaz para o desenvolvimento de circuitos integrados em plataformas Windows. Esta abordagem não apenas amplia o acesso a ferramentas de projeto de circuitos integrados de código aberto, mas também promove a colaboração entre equipes que utilizam diferentes sistemas operacionais. O sucesso desta integração sugere um futuro promissor para o desenvolvimento de hardware em ambientes híbridos, onde as fronteiras entre sistemas operacionais se tornam cada vez mais tênues. A capacidade de aproveitar o melhor dos mundos Windows e Linux abre novas possibilidades para inovação e eficiência no campo da engenharia de circuitos integrados. À medida que o WSL continua a evoluir e o OpenLane se torna mais robusto, espera-se que esta combinação se torne ainda mais poderosa e acessível. Futuros desenvolvimentos podem incluir:

1. Melhor integração gráfica entre WSL e Windows para ferramentas de visualização de layout.

2. Otimizações de desempenho específicas para fluxos de trabalho de EDA (Electronic Design Automation).

3. Maior suporte da comunidade para configurações e soluções de problemas específicos do OpenLane no ambiente WSL.

Esta solução não apenas supera as limitações tradicionais de plataforma, mas também abre novos caminhos para a educação, pesquisa e desenvolvimento profissional no campo do design de circuitos integrados.

4.3.3 Xchem e Ngspice

O XSCHEM é um software de design de sistemas eletrônicos que oferece várias vantagens como:

1. Interface gráfica intuitiva: O *XSCHEM* possui uma interface gráfica simples que facilita a criação de esquemas complexos.

2. Capacidade de lidar com projetos complexos: O software é capaz de gerenciar designs de circuitos integrados (CI) e gerar netlists para simulações em larga escala.

3. Eficiência em grandes projetos: O *XSCHEM* foi projetado para lidar com designs muito grandes de maneira eficiente, podendo manipular mais de dez hierarquias em um único projeto.

4. Recursos avançados de conexão: O software possui um sistema avançado de reconhecimento de conexões de fios, facilitando a construção da conectividade do circuito.

5. Flexibilidade de netlist: Oferece três modos pré-definidos de netlist: SPICE, Verilog e VHDL, permitindo uma ampla gama de aplicações.

6. Destaque de redes: Possui a capacidade de destacar redes e propagar a cor de destaque, melhorando a visibilidade em designs grandes e complexos.

7. Simulação integrada: O *XSCHEM* permite a execução de simulações diretamente do esquemático, sem necessidade de edição manual adicional de arquivos.

Por outro lado, apresenta algumas desvantagens:

1. Curva de aprendizado: O *XSCHEM* não é considerado fácil de usar inicialmente, exigindo um período de aprendizagem.

2. Não é uma ferramenta de layout: Embora seja um editor de esquemáticos poderoso, o *XSCHEM* não é uma ferramenta de layout de circuitos.

3. Limitações de memória: Embora seja eficiente, projetos muito grandes podem exigir sistemas com pelo menos 1GB de RAM.

4. Dependência de bibliotecas limitadas: O *XSCHEM* depende de um número muito limitado de bibliotecas, o que pode restringir algumas funcionalidades.

Em resumo, o *XSCHEM* é uma ferramenta poderosa e eficiente para design de circuitos eletrônicos, especialmente adequada para projetos complexos e hierárquicos. No entanto, sua curva de aprendizado e algumas limitações técnicas podem ser consideradas desvantagens para alguns usuários.

5 CONCLUSÃO

Este trabalho teve como objetivo analisar as ferramentas atuais no contexto do desenvolvimento de circuitos integrados. A pesquisa focou na análise e construção de circuitos analógicos puramente digitais, comparadores e conversores, para compreender como essa nova tecnologia impactará no futuro, em razão da crescente demanda por esses dispositivos.

O circuito proposto, de comparadores dinâmicos puramente digitais, representa uma solução eficiente e eficaz para aplicações que exigem alta velocidade e baixo consumo energético. No entanto, seu uso requer atenção especial ao projeto para mitigar desafios como sensibilidade ao ruído e complexidade do circuito.

As ferramentas de código aberto *Xschem*, *OpenLane* e *NGSpice* têm uma relevância significativa para o meio acadêmico e um impacto considerável na sociedade, especialmente no campo de design de circuitos integrados (CI) e eletrônica. Estas ferramentas permitem que estudantes e pesquisadores tenham acesso a recursos de design de CI de alta qualidade sem custos proibitivos, democratizando o conhecimento na área. A natureza open-source dessas ferramentas facilita a modificação e extensão de suas funcionalidades, permitindo que pesquisadores desenvolvam novas técnicas e metodologias de design.

A metodologia utilizada trouxe benefícios ao desenvolvimento do projeto, principalmente quanto ao uso do *Xschem* por ter a biblioteca do processo de design da *Skywater* já integrado. Por outro lado, apesar do *software* ser intuitivo e possuir manual em seu site, assim como o *NGSpice*, é de se esperar uma certa dificuldade para esquematização e simulação dos circuitos, inicialmente, uma vez que é um *software* novo para o usuário.

Este trabalho conclui que, com técnicas adequadas de design e otimização, os comparadores dinâmicos baseados em células-padrão podem oferecer um desempenho relevante em muitas aplicações modernas, especialmente em sistemas onde a eficiência energética é uma prioridade. A evolução contínua das técnicas de design promete superar as limitações atuais, tornando esses dispositivos ainda mais valiosos no futuro da eletrônica digital. Por fim, as ferramentas utilizadas no desenvolvimento do trabalho representam uma alternativa viável a fim de facilitar a elaboração de circuitos integrados.

REFERÊNCIAS

AIELLO, O.; CROVETTI, P.; ALIOTO, M. Fully synthesizable, rail-to-rail dynamic voltage comparator for operation down to 0.3 v. In: *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*. [S.l.: s.n.], 2018. p. 1–5.

AIELLO, O. et al. A pw-power hz-range oscillator operating with a 0.3–1.8-v unregulated supply. *IEEE Journal of Solid-State Circuits*, v. 54, n. 5, p. 1487–1496, 2019.

AIELLO, O.; CROVETTI, P. S.; ALIOTO, M. Fully synthesizable low-area digital-to-analog converter with graceful degradation and dynamic power-resolution scaling. *IEEE Transactions on Circuits and Systems I: Regular Papers*, v. 66, n. 8, p. 2865–2875, 2019.

ANAND, T.; MAKINWA, K. A. A.; HANUMOLU, P. K. A vco based highly digital temperature sensor with 0.034 ℃/mv supply sensitivity. *IEEE Journal of Solid-State Circuits*, v. 51, n. 11, p. 2651–2663, 2016.

CROVETTI, P. S. A digital-based analog differential circuit. *IEEE Transactions on Circuits and Systems I: Regular Papers*, n. 438, p. 3107–3116, 2013. Doi: 10.1109/TCSI.2013.2255671.

CROVETTI, P. S. et al. breaking the boundaries between analogue and digital. *Electronics Letters*, n. 438, p. 672–673, 2019. Doi: 10.1049/el.2019.1622.

DENG, W. et al. A fully synthesizable all-digital pll with interpolative phase coupled oscillator, current-output dac, and fine-resolution digital varactor using gated edge injection technique. *IEEE Journal of Solid-State Circuits*, v. 50, n. 1, p. 68–80, 2015.

GUPTA, A.; SINGH, A.; AGARWAL, A. A low-power high-resolution dynamic voltage comparator with input signal dependent power down technique. *AEU - International Journal of Electronics and Communications*, v. 134, p. 153682, 2021. ISSN 1434-8411. Disponível em: https://www.sciencedirect.com/science/article/pii/S1434841121000790.

IEEE Standard for SystemVerilog–Unified Hardware Design, Specification, and Verification Language. *IEEE Std 1800-2023 (Revision of IEEE Std 1800-2017)*, p. 1–1354, 2024.

KIM, S. J. et al. A 4-phase 30–70 mhz switching frequency buck converter using a time-based compensator. *IEEE Journal of Solid-State Circuits*, v. 50, n. 12, p. 2814–2824, 2015.

KINGET, P. R. Scaling analog circuits into deep nanoscale cmos: Obstacles and ways to overcome them. *IEEE Custom Integr. Circuits Conf. (CICC)*, p. 1–8, 2015.

OPTEYNDE, F. A maximally-digital radio receiver front-end. In: *2010 IEEE International Solid-State Circuits Conference - (ISSCC)*. [S.l.: s.n.], 2010. p. 450–451.

PAN, D. Z.; SUN, N.; XU, B. A scaling compatible, synthesis friendly vco-based delta–sigma adc design and synthesis methodology. *54th ACM/EDAC/IEEE Design Autom.Conf. (DAC)*, p. 1–6, 2017.

PARK, Y.; WENTZLOFF, D. D. An all-digital 12 pj/pulse ir-uwb transmitter synthesized from a standard cell library. *IEEE Journal of Solid-State Circuits*, v. 46, n. 5, p. 1147–1157, 2011.

SALA, R. D. et al. A novel ultra-low voltage fully synthesizable comparator exploiting nand gates. In: *2023 18th Conference on Ph.D Research in Microelectronics and Electronics (PRIME)*. [S.l.: s.n.], 2023. p. 21–24.

SHALAN, M.; EDWARDS, T. Building openlane: A 130nm openroad-based tapeout- proven flow : Invited paper. In: *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. [S.l.: s.n.], 2020. p. 1–6.

TANWEER, A. A reliable communication framework and its use in internet of things (iot). *International Journal of Scientific Research in Computer Science, Engineering and Information Technology.*, 2018.

TOLEDO, P. et al. Fully digital rail-to-rail ota with sub-1000-m² area, 250-mv minimum supply, and nw power at 150-pf load in 180 nm. *IEEE Solid-State Circuits Letters*, v. 3, p. 474–477, 2020.

TOLEDO, P. et al. Re-thinking analog integrated circuits in digital terms: A new design concept for the iot era. *IEEE Transactions on Circuits and Systems II: Express Briefs*, v. 68, p. 816–822, march 2021. Doi: 10.1109/TCSII.2021.3049680.

WEAVER, S.; HERSHBERG, B.; MOON, U.-K. Digitally synthesized stochastic flash adc using only standard digital cells. In: *2011 Symposium on VLSI Circuits - Digest of Technical Papers*. [S.l.: s.n.], 2011. p. 266–267.

WEAVER, S.; HERSHBERG, B.; MOON, U.-K. Digitally synthesized stochastic flash adc using only standard digital cells. *IEEE Transactions on Circuits and Systems I: Regular Papers*, v. 61, n. 1, p. 84–91, 2014.

YANG, D. et al. A 0.0055mm² 480µw fully synthesizable pll using stochastic tdc in 28nm fdsoi. *IEICE Transactions on Electronics*, E99.C, n. 6, p. 632–640, 2016.

ZOU, X.; NAKATAKE, S. A fully synthesizable, 0.3v, 10nw rail-to-rail dynamic voltage comparator. In: *2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS)*. [S.l.: s.n.], 2020. p. 199–202.

APÊNDICE

TUTORIAL WSL E OPENLANE

Objetivo:

- Habilitar o WSL no Windows 10 e Windows 11;
- Instalar o Ubuntu 22.04.5 LTS através da Loja de Aplicativos Microsoft, pelo prompt de comando ou pelo powershell;
- Atualizar o sistema do Ubuntu;
- Atualizar o Kernel do Linux;
- Instalar as ferramentas pela biblioteca do *Institute for Integrated Circuits* da Universidade *Johannes Kepler* de *Linz*: Openlane/OpenROAD, Icarus Verilog e Verilator, GTKWave, Xschem, ngspice, gaw3, Magic e Netgen.

Requisitos:

- Ter o Windows 10 ou 11 instalado em uma máquina virtual ou dispositivo físico;
- Estar com o Windows devidamente atualizado.

Para começar a instalação, o **primeiro passo** é instalar a distribuição Linux pelo ambiente do WSL, o que pode ser feito de duas formas:

1. Através da Linha de Comando:

```
1) Abra o prompt de comando ou o powershell como Administrador
(clique com o botão direito no ícone para escolher essa opção)
e digite o seguinte comando:

    > wsl --list --online

2) Ele exibirá uma lista com todas as distribuições disponíveis
para instalação. Depois de escolhida a distribuição e versão
basta digitar (neste exemplo utilizaremos o Ubuntu-22.04):

    > wsl --install -d Ubuntu-22.04

3) Espere o Windows terminar a instalação e depois inicie o
Ubuntu através do menu iniciar ou da loja de Aplicativos da Microsoft.

4) Pode ser feita então a atualização do seu sistema Ubuntu
```

através dos comandos na sequência:

```
$ sudo apt update
$ sudo apt full-upgrade -y
```

2. Loja de Aplicativos Microsoft:

1) Entre na Loja de Aplicativos Microsoft através do menu iniciar ou pesquisepor ela na barra de pesquisa do menu para realizar o acesso.

2) Procure pela distribuição Ubuntu 22.04 LTS (ou outra de sua preferência; Debian, etc) e então clique no botão "Adquirir".

3) Espere o Windows terminar a instalação e depois inicie o Ubuntu através do menu iniciar ou da loja de Aplicativos da Microsoft.

4) Pode ser feita então a atualização do seu sistema Ubuntu através dos comandos na sequência:

```
$ sudo apt update
$ sudo apt full-upgrade -y
```

A seguir o **segundo passo** compreende a atualização do WSL e o Kernel do Linux:

1) Fazer logout na seção do Ubuntu:

```
$ exit
```

2) Abrir o poweshell e atualizar o WSL:

```
> wsl --update
```

3) Reiniciar o WSL para as atualizações surtirem efeito:

```
> wsl --shutdown
```

4) Baixar o pacote de atualização do Kernel do Linux através do link fornecido pela Microsoft:

https://wslstorestorage.blob.core.windows.net/wslblob/wsl_update_x64.msi

5) Clicar duas vezes no pacote de atualização do Kernel

```
para instalar.

6) Reiniciar o WSL para as atualizações surtirem efeito:
       > wsl --shutdown
```

Por fim, o **terceiro e último passo** compreende a instalação da biblioteca do *Institute for Integrated Circuits* da *Universidade Johannes Kepler de Linz*:

```
1) Iniciar o Ubuntu através do menu iniciar

2) Clonar o repositório para instalar as ferramentas:

      $ git clone https://github.com/iic-jku/osic-multitool.git

3) Entrar no repositório clonado:

      $ cd osic-multitool/

4) Fazer um ajuste no arquivo de instalação para a versão
correta do ngspice:
     (será utilizado o editor nano neste tutorial)

      $ nano iic-osic-setup.sh

5) Procurar a linha "export NGSPICE_VERSION=42" e trocar o "42"
por "43" da seguinte maneira:
      export NGSPICE_VERSION=43

6) Salvar o arquivo com ctrl+x, selecione "y" e confirme
com a tecla "ENTER"

7) Executar o arquivo de instalação através do comando (confirmar
com a senha criada do usuário toda vez que o sistema solicitar):

      $ ./iic-osic-setup.sh

8) Abrir o arquivo .bashrc para editá-lo:
      $ cd ~/
      $ nano .bashrc
```

9) Ir até a última linha do arquivo .bashrc e digitar (para sempre iniciar o ambiente quando iniciar o ubuntu):

```
source ~/iic-init.sh
```

10) Salvar o arquivo com ctrl+x, selecione "y" e confirme com a tecla "ENTER"

11) Fazer logout na seção do Ubuntu:

```
$ exit
```

12) Reiniciar o WSL para as atualizações surtirem efeito:

```
> wsl --shutdown
```

13) Iniciar o Ubuntu através do menu iniciar

13) Mudar para o diretório do Openlane e rodar o make test:

```
$ cd OpenLane/
$ make test
```

Após estas etapas, todas as ferramentas já estarão prontas para uso!

5.1 Circuito comparador de portas *NAND*

```
** sch_path: /home/mangrost/TCC/final/comparador_nand.sch
.lib /home/mangrost/pdk/sky130A/libs.tech/combined/sky130.lib.spice tt
.include /home/mangrost/pdk/sky130A/libs.ref/sky130_fd_sc_hd/spice/sky130_fd_sc_hd.spice
**.subckt comparador_nand
x1 clk VSS VSS VCC VCC clkbuf sky130_fd_sc_hd__clkbuf_1
x3 net1 VSS VSS VCC VCC clkinv sky130_fd_sc_hd__clkinv_1
x5 net4 VSS VSS VCC VCC net5 sky130_fd_sc_hd__clkbuf_1
V2 clk GND pulse 0 'vcc' '0.495/ 40e6 ' '0.01/40e6 ' '0.01/40e6 ' '0.49/40e6 ' '1/40e6 '
V1 net1 GND pulse 0 'vcc' '0.495/ 40e6 ' '0.01/40e6 ' '0.01/40e6 ' '0.49/40e6 ' '1/40e6 '
V3 net2 GND pulse 0 'vcc' '0.495/ 40e6 ' '0.01/40e6 ' '0.01/40e6 ' '0.49/40e6 ' '1/40e6 '
V4 net4 GND pulse 0 'vcc' '0.495/ 40e6 ' '0.01/40e6 ' '0.01/40e6 ' '0.49/40e6 ' '1/40e6 '
x7 clkbuf VinP_NAND VSS VSS VCC VCC A sky130_fd_sc_hd__nand2_1
x2 clkinv D VSS VSS VCC VCC E sky130_fd_sc_hd__nand2_1
x8 E A VSS VSS VCC VCC C sky130_fd_sc_hd__nand2_1
x9 F B VSS VSS VCC VCC D sky130_fd_sc_hd__nand2_1
x10 net3 C VSS VSS VCC VCC F sky130_fd_sc_hd__nand2_1
x11 net5 VinM_NAND VSS VSS VCC VCC B sky130_fd_sc_hd__nand2_1
x12 C VoutM_NAND VSS VSS VCC VCC VoutP_NAND sky130_fd_sc_hd__nand2_1
x13 D VoutP_NAND VSS VSS VCC VCC VoutM_NAND sky130_fd_sc_hd__nand2_1
V5 VinP_NAND GND PULSE(0 1.8 0 300ns 0 0 0 0)
V6 VinM_NAND GND 1.2
x6 net2 VSS VSS VCC VCC net3 sky130_fd_sc_hd__clkinv_1
**** begin user architecture code
.param vcc=1.8
vVCC VCC 0 dc VCC
vVSS VSS 0 0
.tran 1ns 500ns
.save all
**** end user architecture code
**.ends
.GLOBAL GND
.end
```

5.2 Circuito comparador de portas *AND-OR*

```
** sch_path: /home/mangrost/TCC/Circuitos/comparador_andor.sch
.lib /home/mangrost/pdk/sky130A/libs.tech/combined/sky130.lib.spice tt
.include /home/mangrost/pdk/sky130A/libs.ref/sky130_fd_sc_hd/spice/sky130_fd_sc_hd.spice
**.subckt comparador_andor
x11 clkinv D clkbuf VinP VSS VSS VCC VCC C sky130_fd_sc_hd__a22o_1
x1 clkinv C clkbuf VinM VSS VSS VCC VCC D sky130_fd_sc_hd__a22o_1
x2 net1 VSS VSS VCC VCC clkinv sky130_fd_sc_hd__clkinv_1
V1 net1 GND pulse 0 'vcc' '0.495/ 20e6 ' '0.01/20e6 ' '0.01/20e6 ' '0.49/20e6 ' '1/20e6 '
x4 net2 VSS VSS VCC VCC clkbuf sky130_fd_sc_hd__clkbuf_1
V2 net2 GND pulse 0 'vcc' '0.495/ 20e6 ' '0.01/20e6 ' '0.01/20e6 ' '0.49/20e6 ' '1/20e6 '
V5 VinP GND 0.805
V8 VinM GND 1.005
x3 C VoutM VSS VSS VCC VCC VoutP sky130_fd_sc_hd__nand2_1
x5 D VoutP VSS VSS VCC VCC VoutM sky130_fd_sc_hd__nand2_1
**** begin user architecture code
.param vcc=1.2
vVCC VCC 0 dc VCC
vVSS VSS 0 0
.tran 1ns 500ns
.save all
```

```
**** end user architecture code
**.ends
.GLOBAL GND
.end
```

5.3 Comparador MUX

```
 ** sch_path: /home/mangrost/TCC/final/comparador_mux.sch
.lib /home/mangrost/pdk/sky130A/libs.tech/combined/sky130.lib.spice tt
.include /home/mangrost/pdk/sky130A/libs.ref/sky130_fd_sc_hd/spice/sky130_fd_sc_hd.spice
**.subckt comparador_mux
x1 D VinP_MUX clkbuf VSS VSS VCC VCC C sky130_fd_sc_hd__mux2_1
x2 C VinM_MUX clkbuf VSS VSS VCC VCC D sky130_fd_sc_hd__mux2_1
V5 VinP_MUX GND PULSE(0 1.8 0 300ns 0 0 0 0)
V8 VinM_MUX GND 0.6
x3 C VoutM_MUX VSS VSS VCC VCC VoutP_MUX sky130_fd_sc_hd__nand2_1
x5 D VoutP_MUX VSS VSS VCC VCC VoutM_MUX sky130_fd_sc_hd__nand2_1
x6 net1 VSS VSS VCC VCC clkbuf sky130_fd_sc_hd__clkbuf_1
V1 net1 GND pulse 0 'vcc' '0.495/ 40e6 ' '0.01/40e6 ' '0.01/40e6 ' '0.49/40e6 ' '1/40e6 '
**** begin user architecture code
.param vcc=1.8
vVCC VCC 0 dc VCC
vVSS VSS 0 0
.tran 1ns 500ns
.save all
**** end user architecture code
**.ends
.GLOBAL GND
.end
```

5.4 Circuito do conversor digital-analógico

```
** sch_path: /home/mangrost/TCC/Circuitos/DAC.sch
.lib /home/mangrost/pdk/sky130A/libs.tech/combined/sky130.lib.spice tt
.include /home/mangrost/pdk/sky130A/libs.ref/sky130_fd_sc_hd/spice/sky130_fd_sc_hd.spice
**.subckt DAC
x1 Vdac D0 VSS VSS VCC VCC Vdac sky130_fd_sc_hd__nand2_1
x2 Vdac D1 VSS VSS VCC VCC Vdac sky130_fd_sc_hd__nand2_2
x3 Vdac D2 VSS VSS VCC VCC Vdac sky130_fd_sc_hd__nand2_4
x4 Vdac D3 VSS VSS VCC VCC Vdac sky130_fd_sc_hd__nand2_8
x5 Vdac D4 VSS VSS VCC VCC Vdac sky130_fd_sc_hd__nand2_8
x6 Vdac D4 VSS VSS VCC VCC Vdac sky130_fd_sc_hd__nand2_8
x7 Vdac net5 VSS VSS VCC VCC Vdac sky130_fd_sc_hd__nand2_1
V1 net5 GND 'vcc'
V2 D0 GND 'vcc'
V3 D1 GND 'vcc'
V4 D2 GND 'vcc'
V5 D3 GND 'vcc'
V6 D4 GND 'vcc'
x8 clk VSS VSS VCC VCC clkbuf sky130_fd_sc_hd__clkbuf_1
x9 net1 VSS VSS VCC VCC clkinv sky130_fd_sc_hd__clkinv_1
x10 net4 VSS VSS VCC VCC teste sky130_fd_sc_hd__clkbuf_1
V7 clk GND pulse 0 'vcc' '0.495/ 20e6 ' '0.01/20e6 ' '0.01/20e6 ' '0.49/20e6 ' '1/20e6 '
V8 net1 GND pulse 0 'vcc' '0.495/ 20e6 ' '0.01/20e6 ' '0.01/20e6 ' '0.49/20e6 ' '1/20e6 '
V9 net2 GND pulse 0 'vcc' '0.495/ 20e6 ' '0.01/20e6 ' '0.01/20e6 ' '0.49/20e6 ' '1/20e6 '
V10 net4 GND pulse 0 'vcc' '0.495/ 20e6 ' '0.01/20e6 ' '0.01/20e6 ' '0.49/20e6 ' '1/20e6 '
x11 clkbuf Vdac VSS VSS VCC VCC A sky130_fd_sc_hd__nand2_1
x12 clkinv D VSS VSS VCC VCC E sky130_fd_sc_hd__nand2_1
```

```
x13 E A VSS VSS VCC VCC C sky130_fd_sc_hd__nand2_1
x14 F B VSS VSS VCC VCC D sky130_fd_sc_hd__nand2_1
x15 net3 C VSS VSS VCC VCC F sky130_fd_sc_hd__nand2_1
x16 teste VinM VSS VSS VCC VCC B sky130_fd_sc_hd__nand2_1
x17 C VoutM VSS VSS VCC VCC VoutP sky130_fd_sc_hd__nand2_1
x18 D VoutP VSS VSS VCC VCC VoutM sky130_fd_sc_hd__nand2_1
V12 VinM GND 0.595
x19 net2 VSS VSS VCC VCC net3 sky130_fd_sc_hd__clkinv_1
**** begin user architecture code
.param vcc=1.8
vVCC VCC 0 dc VCC
vVSS VSS 0 0
.tran 1ns 300ns
.save all
**** end user architecture code
**.ends
.GLOBAL GND
.end
```

ANEXO

Código *verilog* dos circuitos apresentados na figura 4 e os três circuitos comparadores, a saber: portas NAND, portas AND OR e MUX. Além do código *Verilog* do circuito apresentado na figura 2 (DAC).

Listing 5.1 – Código Verilog do circuito completo

```
module Digital_DAC ( D, Vdac );

        input [4:0] D;
        output Vdac;

        sky130_fd_sc_hd__nand2_1 ix1 (.Y (Vdac), .A (Vdac), .B (D[0])) ;
        sky130_fd_sc_hd__nand2_2 ix2 (.Y (Vdac), .A (Vdac), .B (D[1])) ;
        sky130_fd_sc_hd__nand2_4 ix4 (.Y (Vdac), .A (Vdac), .B (D[2])) ;
        sky130_fd_sc_hd__nand2_8 ix8 (.Y (Vdac), .A (Vdac), .B (D[3])) ;
        sky130_fd_sc_hd__nand2_8 ix16a (.Y (Vdac), .A (Vdac), .B (D[4])) ;
        sky130_fd_sc_hd__nand2_8 ix16b (.Y (Vdac), .A (Vdac), .B (D[4])) ;

        sky130_fd_sc_hd__nand2_1 ix0 (.Y (Vdac), .A (1'b1), .B (Vdac)) ;

endmodule

module NAND_Comparator_NAND02 ( CLK, VinP, VinM, OutP, OutM ) ;

    input CLK ;
    input VinP ;
    input VinM ;
    output OutP ;
    output OutM ;

    wire C, D, A, B, E, F, CLKn, CLKb;

    sky130_fd_sc_hd__clkbuf_1 ix01 (.X(CLKb), .A(CLK));
    sky130_fd_sc_hd__clkinv_1 ix02 (.Y(CLKn), .A(CLK));

    sky130_fd_sc_hd__nand2_1 ix25 (.Y (OutM), .A (D), .B (OutP)) ;
    sky130_fd_sc_hd__nand2_1 ix31 (.Y (OutP), .A (C), .B (OutM)) ;

    sky130_fd_sc_hd__nand2_1 ix108 (.Y (A), .A (CLKb), .B (VinP)) ;
    sky130_fd_sc_hd__nand2_1 ix21 (.Y (C), .A (E), .B (A)) ;
    sky130_fd_sc_hd__nand2_1 ix11 (.Y (F), .A (CLKn), .B (C)) ;
    sky130_fd_sc_hd__nand2_1 ix5 (.Y (B), .A (CLKb), .B (VinM)) ;
    sky130_fd_sc_hd__nand2_1 ix112 (.Y (D), .A (F), .B (B)) ;
    sky130_fd_sc_hd__nand2_1 ix110 (.Y (E), .A (CLKn), .B (D)) ;

```

```
endmodule

module NAND_Comparator_AO22 ( CLK, VinP, VinM, OutP, OutM ) ;

        input CLK ;
        input VinP ;
        input VinM ;
        output OutP ;
        output OutM ;

        wire C, D, CLKn, CLKb;

        sky130_fd_sc_hd__clkbuf_1 ix01 (.X(CLKb), .A(CLK));
        sky130_fd_sc_hd__clkinv_1 ix02 (.Y(CLKn), .A(CLK));

        sky130_fd_sc_hd__nand2_1 ix25 (.Y (OutM), .A (D), .B (OutP)) ;
        sky130_fd_sc_hd__nand2_1 ix31 (.Y (OutP), .A (C), .B (OutM)) ;

        sky130_fd_sc_hd__a22o_1 ix21 (.X (C), .B1 (CLKb), .B2 (VinP), .A1 (
           CLKn), .A2 (D)) ;
        sky130_fd_sc_hd__a22o_1 ix108 (.X (D), .B1 (CLKb), .B2 (VinM), .A1
           (CLKn), .A2 (C)) ;

endmodule

module NAND_Comparator_MUX21_NI ( CLK, VinP, VinM, OutP, OutM ) ;

        input CLK ;
        input VinP ;
        input VinM ;
        output OutP ;
        output OutM ;

        wire C, D, CLKb;

        sky130_fd_sc_hd__clkbuf_1 ix01 (.X(CLKb), .A(CLK));


        sky130_fd_sc_hd__nand2_1 ix25 (.Y (OutM), .A (D), .B (OutP)) ;
        sky130_fd_sc_hd__nand2_1 ix31 (.Y (OutP), .A (C), .B (OutM)) ;

        sky130_fd_sc_hd__mux2_1 ix21 (.X (C), .A0 (D), .A1 (VinP), .S (CLKb
           )) ;
        sky130_fd_sc_hd__mux2_1 ix107 (.X (D), .A0 (C), .A1 (VinM), .S (
           CLKb)) ;

endmodule
```

```
83
84  module counter #(parameter SIZE=5) ( CLK, RSTN, EN, Q, COUT ) ;
85
86          input CLK;
87          input RSTN;
88          input EN;
89          output reg [SIZE-1:0] Q;
90          output COUT;
91
92          always @(posedge(CLK) or negedge(RSTN))
93                  begin
94                          if (~RSTN)
95                                  Q <= 0;
96                          else
97                                  if (EN)
98                                          Q <= Q + 1;
99                  end
100
101         assign COUT = (&Q) & EN;
102
103 endmodule
104
105 module Digital_Analog ( CLK_CNT0, CLK_CNT1, CLK_COMP, RSTN, EN0, EN1, SEL,
        VinP, VinM, VoutP_NAND, VoutM_NAND, VoutP_AO22, VoutM_AO22, VoutP_MX21,
        VoutM_MX21 ) ;
106
107         input CLK_CNT0, CLK_CNT1, CLK_COMP, RSTN, EN0, EN1, SEL;
108         output VinP, VinM;
109         output VoutP_NAND, VoutM_NAND;
110         output VoutP_AO22, VoutM_AO22;
111         output VoutP_MX21, VoutM_MX21;
112
113         wire [4:0] D0;
114         wire [4:0] D1;
115         wire COUT0;
116
117         counter #(5) u0 (
118                 .CLK(CLK_CNT0),
119                 .RSTN(RSTN),
120                 .EN(EN0),
121                 .Q(D0),
122                 .COUT(COUT0)
123         );
124
125         counter #(5) u1 (
126                 .CLK(SEL ? CLK_CNT1 : CLK_CNT0),
127                 .RSTN(RSTN),
```

```
128                 .EN(SEL ? EN1 : COUT0),
129                 .Q(D1),
130                 /* verilator lint_off PINCONNECTEMPTY */
131                 .COUT()
132                 /* verilator lint_on PINCONNECTEMPTY */
133         );
134
135         (* keep_hierarchy = "yes" *) Digital_DAC DDAC0 (
136                 .D(D0),
137                 .Vdac(VinP)
138         );
139
140         (* keep_hierarchy = "yes" *) Digital_DAC DDAC1 (
141                 .D(D1),
142                 .Vdac(VinM)
143         );
144
145         (* keep_hierarchy = "yes" *) NAND_Comparator_NAND02 Comp1a (
146                 .CLK(CLK_COMP),
147                 .VinP(VinP),
148                 .VinM(VinM),
149                 .OutP(VoutP_NAND),
150                 .OutM(VoutM_NAND)
151         ) ;
152
153         (* keep_hierarchy = "yes" *) NAND_Comparator_AO22 Comp2 (
154                 .CLK(CLK_COMP),
155                 .VinP(VinP),
156                 .VinM(VinM),
157                 .OutP(VoutP_AO22),
158                 .OutM(VoutM_AO22)
159         ) ;
160
161         (* keep_hierarchy = "yes" *) NAND_Comparator_MUX21_NI Comp5 (
162                 .CLK(CLK_COMP),
163                 .VinP(VinP),
164                 .VinM(VinM),
165                 .OutP(VoutP_MX21),
166                 .OutM(VoutM_MX21)
167         ) ;
168
169 endmodule
```