CEFET/RJ
DIRETORIA DE PESQUISA E PÓS-GRADUAÇÃO

FERRAMENTA BASEADA EM ALGORITMO EVOLUCIONÁRIO PARA PROJETO DE REGULADORES DE TENSÃO CMOS

Igor Meirelles Martins

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, Centro Federal de Educação Tecnológica Celso Suckow da Fonseca CEFET/RJ, como parte dos requisitos necessários à obtenção do grau de mestre.

Orientador:
Luis Fabián Olivera Mederos

Rio de Janeiro,

10 de Outubro de 2025

FERRAMENTA BASEADA EM ALGORITMO EVOLUCIONÁRIO PARA PROJETO DE REGULADORES DE TENSÃO CMOS

Dissertação de Mestrado em Engenharia Elétrica, Centro Federal de Educação Tecnológica Celso Suckow da Fonseca, CEFET/RJ.

Igor Meirelles Martins

Aprovada por:

Documento assinado digitalmente
gov.br
LUIS FABIAN OLIVERA MEDEROS
Data: 18/11/2025 21:42:23-0300
Verifique em https://validar.iti.gov.br

Prof. Luis Fabián Olivera Mederos, D.Sc. (orientador)

Documento assinado digitalmente
gov.br
LUCIANA FALETTI ALMEIDA
Data: 19/11/2025 14:19:29-0300
Verifique em https://validar.iti.gov.br

Profa. Luciana Faletti Almeida, D.Sc.

Documento assinado digitalmente
gov.br
GABRIEL MATOS ARAUJO
Data: 27/11/2025 16:59:47-0300
Verifique em https://validar.iti.gov.br

Prof. Gabriel Matos Araujo, D.Sc.

Documento assinado digitalmente
gov.br
FERNANDA DUARTE VILELA REIS DE OLIVEIRA
Data: 19/11/2025 21:39:29-0300
Verifique em https://validar.iti.gov.br

Profa. Fernanda Duarte Vilela Reis de Oliveira, D.Sc.

Rio de Janeiro,

10 de Outubro de 2025

Ficha catalográfica elaborada pela Biblioteca Central do CEFET/RJ

M386 Martins, Igor Meirelles
Ferramenta baseada em algoritmo evolucionário para projeto de reguladores de tensão CMOS / Igor Meirelles Martins. — 2025.
55f. : il. color. , enc.

Dissertação (Mestrado) Centro Federal de Educação Tecnológica Celso Suckow da Fonseca, 2025.
Bibliografia : f. 53-55
Orientador: Luis Fabián Olivera Mederos

1. Algoritmos Evolucionário. 2. Semicondutores complementares de óxido metálico. 3. Circuitos eletrônicos. 4. Energia – Consumo. I. Mederos, Luis Fabián Olivera. (Orient.). II. Título.

CDD 006.31

Elaborada pela bibliotecária Tania Mello – CRB/7 nº 5507/04

AGRADECIMENTOS

Agradeço primeiramente a Deus pela sabedoria e determinação durante esta jornada.

Agradeço ao meu pai, minha mãe, irmã e namorada, Wander, Janilze, Érika e Beatriz pelo apoio e auxílio sempre que necessário.

Agradeço a minha prima Lilian e sua família por me acolherem em sua casa.

Agradeço profundamente ao meu professor e orientador Luis Fabián Olivera Mederos pelo aprendizado, pela dedicação e por ter me guiado nestes últimos anos de pesquisa, ampliando consideravelmente o conhecimento adquirido até o momento.

Agradeço à professora Luciana Faletti Almeida pelo apoio na pesquisa.

Agradeço aos meus amigos, que sempre estiveram ao meu lado em momentos desafiadores.

Agradeço ao CEFET, à UFRJ e aos professores pela dedicação e pelo aprendizado.

Agradeço ao PPEEL/CEFET pelo suporte na pesquisa de mestrado.

Por fim, agradeço aos órgãos de fomento Capes, CNPQ e FAPERJ, responsável por fomentar esta pesquisa.

RESUMO

FERRAMENTA BASEADA EM ALGORITMO EVOLUCIONÁRIO PARA PROJETO DE REGULADORES DE TENSÃO CMOS

Igor Meirelles Martins

Orientadores:
Luis Fabián Olivera Mederos

Resumo da Dissertação submetida ao Programa de Pós-graduação em Engenharia Elétrica do Centro Federal de Educação Tecnológica Celso Suckow da Fonseca CEFET/RJ como parte dos requisitos necessários à obtenção do grau de mestre.

Neste trabalho de dissertação de mestrado apresenta-se o desenvolvimento de uma ferramenta auxiliada por computador que utiliza um algoritmo genético básico para projetar reguladores lineares de baixa queda de tensão (LDO). O desempenho ideal desses circuitos geralmente depende de muitos parâmetros de projeto, como corrente de polarização, capacitância de carga máxima, entre outros. Neste contexto, um fluxo Python-NGSpice é proposto a fim de alcançar LDOs de alta eficiência, mediante a realização das etapas de projeto e verificação através de simulações com variações Monte Carlo, a nível esquemático e pós-leiaute. Os circuitos otimizados neste trabalho foram implementados em processos CMOS de 130 nm (180 nm). Em relação ao desempenho dos circuitos desenvolvidos e verificado através de extensas simulações em Ngspice, foi mostrado consumo de corrente de 5 nA (50 nA), tensão de regulação de 1 V e capacitância de saída de 10 fF (60 fF). Um protótipo implementado no processo CMOS de 130 nm da Skywater foi enviado para fabricação através do programa Universalização de Circuitos Integrados das Sociedade de Circuitos e Sistemas (UNIC-CASS) [1], com simulações mostrando consumo de 465 nA, tensão de regulação de 1 V, e capacitância de saída de 41 fF.

Palavras-Chave:
Algoritmo Evolucionário, CMOS, Regulador LDO, Ultra Baixa Potência.

Rio de Janeiro,

10 de Outubro de 2025

ABSTRACT

AN EVOLUTIONARY ALGORITHM BASED TOOL FOR CMOS VOLTAGE REGULATOR DESIGNS

Igor Meirelles Martins

Advisors:
Luis Fabián Olivera Mederos

Abstract of master's dissertation submitted to Programa de Pós-graduação em Engenharial Elétrica - Centro Federal de Educação Tecnológica Celso Suckow da Fonseca CEFET/RJ as partial fulfillment of the requirements for the degree of master.

In this Master's qualifying exam, an evolutionary algorithm (EA)-based approach for the design of low-power low-drop-out (LDO) voltage regulators is carried out. The optimal performance of LDO regulators usually depends on many design parameters, such as quiescent current, maximum output current, maximum load capacitance, among others. Therefore, an EA flow that combines Python and NGSpice tools is proposed in this work to achieve high-performance designs, through the completion of the project steps and verifications through simulations with Monte Carlo variations, being on schematic and layout level. The CMOS nodes optimized by the tool were 130 and (180 nm), with a current consumption of 5 nA (50 nA), regulation voltage of 1 V and output capacitance of 10 fF (60 fF). A design example carried out in the SkyWater 130 nm CMOS process and verified by post-layout simulation, whose results showed a maximum load current of 10 μA and load capacitance of 5 pF. The circuit was sent to be fabricated through the UNIC-CASS IEEE program [1], with simulations showing 465 nA of consumption, regulation voltage of 1 V and output capacitance of 41 fF.

Key-words:
CMOS, evolutionary algorithm, Internet-of-Things, LDO voltage regulator, low-power design.

Rio de Janeiro,

10 de Outubro de 2025

Sumário

I Introdução **1**

I.1 Motivação 1

I.2 Objetivo e Contribuições 2

I.3 Metodologia da Dissertação 3

I.4 Estrutura da Dissertação 4

II Ferramenta Auxiliada por Computador Baseada em Algoritmo Genético **5**

II.1 Abordagem do Desenvolvimento da Ferramenta Computacional 5

II.2 Testes e Simulações de Avaliação do Desempenho do LDO 6

II.3 Variabilidade Monte Carlo nas Simulações 11

II.4 Algoritmo Evolucionário Para Otimização do Circuito LDO 13

II.4.1 Inicialização da População Inicial 13

II.4.2 Avaliação e Melhorias Computacionais 15

II.4.3 Seleção de Indivíduos 16

II.4.4 Cruzamento 17

II.4.5 Mutação 18

II.4.6 Substituição da População 19

II.4.7 Critérios de Parada 20

II.4.8 Implementação do Algoritmo Genético 20

III Desempenho do Algoritmo Proposto **23**

III.1 Descrição de Execuções 23

III.1.1 Proposta de FoM Alternativa 23

III.1.2 Parâmetros do Circuito 24

III.1.3 Parâmetros Utilizados na Otimização e Simulações Elétricas 24

III.1.4 Parâmetros do Algoritmo Genético 25

III.2 Resultados de Otimização 26

III.3 Custo Computacional 30

III.4 Comparação dos Resultados em Relação à Literatura 32

IV Simulações de Protótipo Enviado para Fabricação **33**

IV.1 Resultados da Otimização 33

IV.2 Simulações a Nível de Esquemático 35

IV.3 Projeto de Leiaute 39

IV.4 Simulações a Nível de Leiaute 40

IV.5 Envio à Fabricação do Circuito Integrado 45

IV.6 Desafios de Medição 47

IV.6.1 Testes de Bancada Propostos 49

V Conclusões e Trabalhos Futuros **51**

V.1 Conclusões 51

V.2 Trabalhos Futuros 52

Referências Bibliográficas 53

Lista de Figuras

I.1	Diagrama da fonte de energia.	2
I.2	Circuito do regulador de tensão LDO.	3
II.1	Diagrama ilustrativo do fluxo proposto.	6
II.2	Teste de regulação de tensão (teste A).	7
II.3	Fluxograma do algoritmo para obtenção da regulação.	7
II.4	Fluxograma para obtenção da máxima corrente de carga.	8
II.5	Teste de corrente máxima (Teste B).	8
II.6	Teste transiente para variação de tensão de entrada (Teste C).	9
II.7	Fluxograma para obtenção do somatório da distância euclidiana.	9
II.8	Teste transiente para variação de corrente de saída (Teste D).	10
II.9	Fluxograma para obtenção da margem de fase.	10
II.10	Circuito em malha aberta para teste de margem de fase.	11
II.11	Circuito para o teste de extração de capacitância de saída.	11
II.12	Descrição de Monte Carlo no algoritmo.	12
II.13	Fluxograma ilustrativo proposto.	14
II.14	Comparação de inicialização linear/logarítmica.	15
II.15	Intervalo de mutação.	19
II.16	Variações do parâmetro K_p em 2 dimensões.	19
II.17	Representação das funções de testes utilizadas.	21
III.1	Tensão de saída em função da tensão de entrada (Teste A).	27
III.2	Tensão de saída em função da corrente de carga (Teste B).	27
III.3	Resposta dinâmica à variação da tensão de entrada (Teste C).	28
III.4	Vista detalhada da saída de tensão do Teste C.	28
III.5	Resposta dinâmica à variação de carga (Teste D).	29
III.6	Vista detalhada da saída de tensão do Teste D.	29
III.7	Resposta em frequência do circuito em malha aberta (Teste E).	30
III.8	Progresso do algoritmo durante a otimização.	31

IV.1 Progresso do algoritmo durante a otimização anterior às modificações. 34
IV.2 Histograma para 50 rodadas de Monte Carlo. 35
IV.3 Teste A em *corners* a nível de esquemático. 36
IV.4 Teste B em *corners* a nível de esquemático. 36
IV.5 Teste C em *corners* a nível de esquemático. 37
IV.6 Vista detalhada da saída de tensão em *corners* do Teste C a nível de esquemático. 37
IV.7 Teste D em *corners* a nível de esquemático. 38
IV.8 Vista detalhada da saída de tensão do Teste D em *corners* a nível de esquemático. 38
IV.9 Teste E em *corners* a nível de esquemático. 39
IV.10 Implementação de leiaute do circuito otimizado com a ferramenta. 40
IV.11 Tensão de saída em função da tensão de entrada em *corners* (Teste A). 41
IV.12 Tensão de saída em função da corrente de carga em *corners* (Teste B). 42
IV.13 Resposta dinâmica à variação de tensão de entrada em *corners* (Teste C). 42
IV.14 Vista detalhada da saída de tensão em *corners* (Teste C). 43
IV.15 Resposta dinâmica à variação de carga em *corners* (Teste D). 43
IV.16 Vista detalhada da saída de tensão do Teste D em *corners*. 44
IV.17 Teste E em *corners* a nível de leiaute. 44
IV.18 Leiaute de implementação do circuito de todo o time. 45
IV.19 Projeto final enviado à UNIC-CASS. 46
IV.20 Amostras de *chips* recebidos. 46
IV.21 Placa de desenvolvimento para o projeto. Fonte: [28]. 47
IV.22 Placa desenvolvida pelo autor desta dissertação para a medição do *chip*. 48
IV.23 Bancada de teste utilizada. 49
IV.24 Proposta de realização do Teste A. 50
IV.25 Proposta de realização do Teste C. 50
IV.26 Proposta de realização do Teste B e D. 50

Lista de Tabelas

II.1 Resultados para 128 rodadas da otimização em funções de teste. 22

III.1 Parâmetros fixos durante a otimização 25
III.2 Intervalos do espaço de busca passado ao AE 25
III.3 Parâmetros do algoritmo genético 25
III.4 Resultados de rodadas da otimização 26
III.5 Valores obtidos através das simulações 26
III.6 Valores encontrados pelo AE 26
III.7 Descrição das máquinas utilizados 30
III.8 Tempo de otimização para 10 rodadas em cada máquina 31
III.9 Comparação dos resultados obtidos com a literatura 32

IV.1 Valores dos intervalos encontrados pelo AG que foi enviado para fabricação 34
IV.2 Resultado de simulação para os *corners* de fabricação 35
IV.3 Resultado de simulação para os *corners* de fabricação pós-leiaute 40
IV.4 Comparação de resultados pré-leiaute com pós-leiaute 41
IV.5 Equipamentos de bancada utilizados 48

Capítulo I Introdução

I.1 Motivação

As fontes de alimentação em sistemas em *chip* (SoCs) precisam de reguladores lineares de baixa queda de tensão (LDO, sigla em inglês, de *low drop-out*) para fornecer tensões de alimentação estáveis [2, 3, 4, 5, 6]. O crescimento contínuo da Internet das Coisas (IoT) e aplicações industriais de colheita de energia fazem com que o consumo de energia seja um dos maiores desafios nos últimos anos. Como, na maior parte do tempo, os dispositivos IoT operam em modo repouso (do inglês *standby*), as correntes de polarização dos circuitos acabam sendo cruciais em aplicações de projetos de ultra baixa potência (ULP, sigla em inglês *Ultra-low-power*) [2, 4, 6]. Uma forma clássica de avaliar e comparar o desempenho do circuito é através da métrica conhecida como Figura de Mérito (FoM), que possui uma definição particular para cada circuito, considerando as principais variáveis do projeto. Como reportado em [2, 5], a FoM para LDOs inclui a razão entre a corrente de polarização e a corrente máxima de carga, estabilidade transiente a variações, entre outros parâmetros. Com isso, otimizando o compromisso entre os parâmetros, podem ser alcançadas soluções eficientes em termos de consumo de energia.

Na Figura I.1 um diagrama de fonte de energia regulada para IoT é mostrado. Esta fonte captura energia através de recursos alternativos, sejam vibrações, que são captadas através de um piezoelétrico, ou térmica, através de um gerador termoelétrico, por exemplo. A tensão gerada por estes geradores sofre flutuações inesperadas de acordo com as situações em que são operados, tendo saída de tensões indefinidas, e em certos casos, abaixo da tensão necessária. Desta forma, o diagrama proposto na Figura I.1 consegue automaticamente selecionar entre obter a energia diretamente da fonte ou passar por um estágio de amplificação. Independentemente da forma com que a tensão será tratada, antes de ser entregue à carga, ela precisa ser regulada por um regulador linear LDO, bloco cinza, que possua alto desempenho, baixo consumo e forte regulação em situações transitórias. O dimensionamento destes circuitos é bem desafiador, pois envolvem várias variáveis.

Ferramentas de projetos assistidas por computadores (CAD, traduzido do inglês *Computer-Aided Design*) têm sido estudadas na literatura [7, 8, 9] como uma solução eficiente para a redução do número de simulações e do tempo de desenvolvimento de projeto, diminuindo assim o tempo

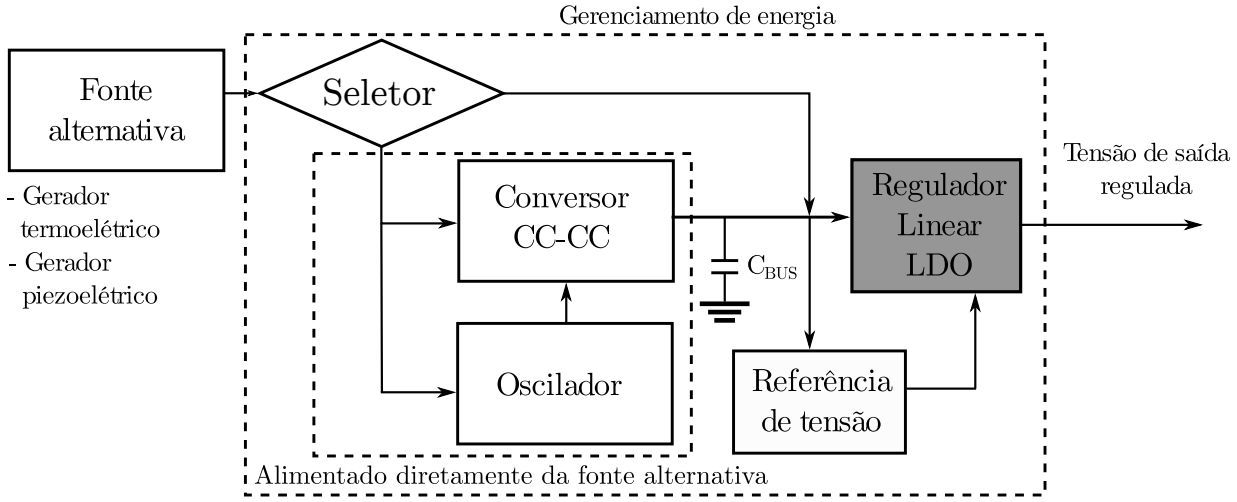


Figura I.1: Diagrama da fonte de energia.

de mercado. Para otimizar o consumo de energia, sem perder o foco nos outros parâmetros do LDO, é proposto o uso da FoM como função custo a ser passada a um algoritmo evolucionário (AE). A ferramenta CAD proposta dimensiona automaticamente e avalia, através de simulações, o desempenho do circuito, aproveitando que a definição da FoM considera os parâmetros mais importantes para o dimensionamento do mesmo.

I.2 Objetivo e Contribuições

Ao invés de obter um regulador através de técnicas como polarização adaptativa e polarização dinâmica (do inglês *adaptive biasing* ou *dynamic biasing*) [4], que resultam em circuitos finais mais complexos, neste trabalho usou-se uma versão simplificada do circuito, mostrada na Fig. I.2 [4]. Dessa forma, com a ferramenta computacional uma vez implementada, o esforço de obter o projeto do circuito LDO para outras especificações e tecnologias CMOS é diminuído, já que não é necessário realizar modelagens dos transistores, como feito em [10], destacando a adaptabilidade da ferramenta desenvolvida para diversos casos de projeto e tecnologias CMOS. Neste trabalho é descrita uma ferramenta computacional focada na otimização do circuito da Fig. I.2.

Com isso, deseja-se obter as dimensões físicas, tais como comprimento e largura do canal dos transistores M_{N1}, $M_{N2,a,b,c}$, M_{P1}, M_{P2} e $M_{P3,a,b}$; as tensões V_{SET} e V_B; a corrente de polarização I_B; e o capacitor C_m. Os demais parâmetros foram mantidos fixos durante a otimização.

Espera-se obter um circuito funcional com alto desempenho em menos tempo que um projetista levaria para entender os detalhes da tecnologia CMOS, entender o circuito, realizar análises dos circuitos e reunir tudo isso em um projeto funcional.

Em 2021, parte deste trabalho foi apresentado no congresso internacional IBERCHIP, no Peru [11] e em 2025 outro artigo foi apresentado no congresso *IEEE Latin American Conference on*

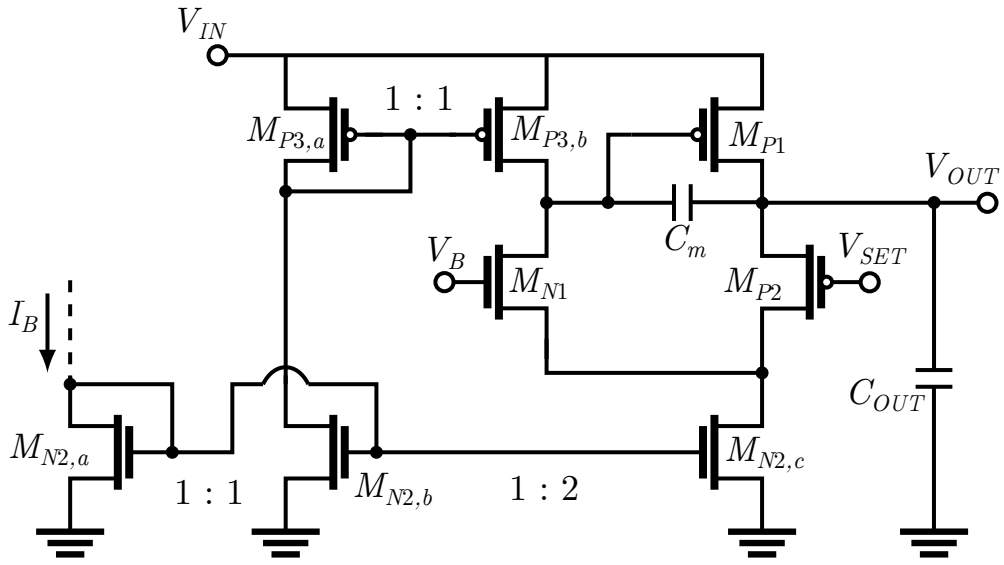


Figura I.2: Circuito do regulador de tensão LDO.

Internet of Things em Fortaleza [12]. Além do apresentado em [11], neste trabalho de dissertação de mestrado foram desenvolvidos novos testes, tais como teste de margem de fase aprimorado e simulação com variação Monte Carlo. Também foram propostos algumas modificações na métrica FoM e a substituição da Evolução Diferencial por um algoritmo genético básico. Ao final, o circuito também foi submetido a simulações pós-leiaute e, por fim, enviado para fabricação no processo *sky130*.

I.3 Metodologia da Dissertação

Este trabalho de dissertação de mestrado aborda o tema da seguinte forma:

- Revisão bibliográfica: através de pesquisas e revisões de trabalhos da literatura que abordam o tema de ULP, reguladores de tensão LDO e algoritmos genéticos. Com enfoque em topologias de circuitos LDO bem como métricas de avaliação aplicadas ao circuito, buscando maneiras de aplicá-las aos algoritmos genéticos.

- Desenvolvimento da Ferramenta Computacional: o fluxo de trabalho proposto entre o algoritmo genético e um simulador de circuitos. Bem como, para avaliar o circuito, testes foram realizados, e com os resultados destes testes calcula-se uma métrica encontrada na literatura.

- Simulações e resultados: todos os parâmetros utilizados para executar as simulações são descritos.

- Comparação de resultados com a literatura: os circuitos obtidos são comparados com circuitos presentes na literatura.

- Divulgação de resultados na literatura: o trabalho desenvolvido nesta dissertação foi apresentado no congresso *IEEE Latin American Conference on Internet of Things* 2025, Fortaleza, Ceará [12].

I.4 Estrutura da Dissertação

A organização deste trabalho é dada pela seguinte estruturação: O Capítulo II descreve a abordagem dada com o AE ao problema, descrevendo testes e métodos utilizados na avaliação do circuito e propondo uma integração entre a linguagem de programação Python e o simulador de circuitos NGSpice, bem como outras ferramentas utilizadas, como bibliotecas externas, utilizadas pelo Python. No Capítulo III são mostrados modificações na métrica do LDO para melhor aproveitamento da integração com o algoritmo genético, os parâmetros passados ao simulador de circuitos e ao algoritmo genético. Também são descritos os resultados obtidos através da otimização e também a comparação com a literatura. O Capítulo IV discorre sobre as simulações antes do leiaute, a implementação do circuito em um leiaute, simulações pós-leiaute, e o envio para fabricação através do programa UNIC-CASS [1]. Por fim, o Capítulo V apresenta as considerações finais e dá a direção de trabalhos futuros.

Capítulo II Ferramenta Auxiliada por Computador Baseada em Algoritmo Genético

Descreve-se neste capítulo a abordagem proposta da ferramenta computacional, bem como a metodologia utilizada para seu desenvolvimento, como a integração de outras ferramentas computacionais em um fluxo de trabalho; desenvolvimento de algoritmos e métodos para avaliação do circuito; e, por fim, a modelagem de um algoritmo genético, que realizará a otimização.

II.1 Abordagem do Desenvolvimento da Ferramenta Computacional

Para a implementação da ferramenta, foram utilizadas em conjunto a linguagem de programação Python (versão 3.10) e o simulador de circuitos NGSpice, ambos programas de código aberto. Algumas bibliotecas externas foram utilizadas no Python, já que este não possuía todas as funções necessárias para a execução do programa, sendo estas: **Numpy**, para utilidades de matemática e vetores; **Matplotlib**, para plotagem de gráficos; **Sys** e **Os**, para comunicação entre o Python e o sistema operacional; **Multiprocessing**, para paralelismo computacional; e a biblioteca **Time**, para utilidades de tempo, como espera de execução e registro de datas e horários. Para ler os dados do simulador de circuitos no Python, foi utilizada uma rotina das ferramentas da CPPSim [13]. Com ela é possível ler o arquivo *.raw*[1] com as informações necessárias para a avaliação. Esta integração foi focada em minimizar a função custo, e assim obter um resultado bem próximo do ótimo global. O fluxograma que contém a lógica geral da otimização, juntamente com a integração entre o Python e o NGSpice, é mostrado na Fig. II.1, em que a ferramenta implementada em Python cria a netlist[2] a partir de uma estrutura padrão de circuito e a executa no NGSpice usando o modo em lotes (traduzido do inglês *batch mode*) no *shell* do sistema operacional. O NGSpice salva os dados da simulação em um arquivo *.raw*, e estes são importados de volta através da rotina da CPPSim para o cálculo da função custo no Python. Como o programa trabalha com arquivos salvos no armazenamento da máquina, os arquivos são apagados ao fim de cada rodada para evitar conflitos de arquivos.

[1] *.raw*: Arquivo de saída do simulador de circuitos que contém o vetor com o resultado da simulação executada.
[2] Netlist: é uma descrição do circuito na forma de texto, com funcionalidade semelhante a um esquemático em simuladores de circuitos.

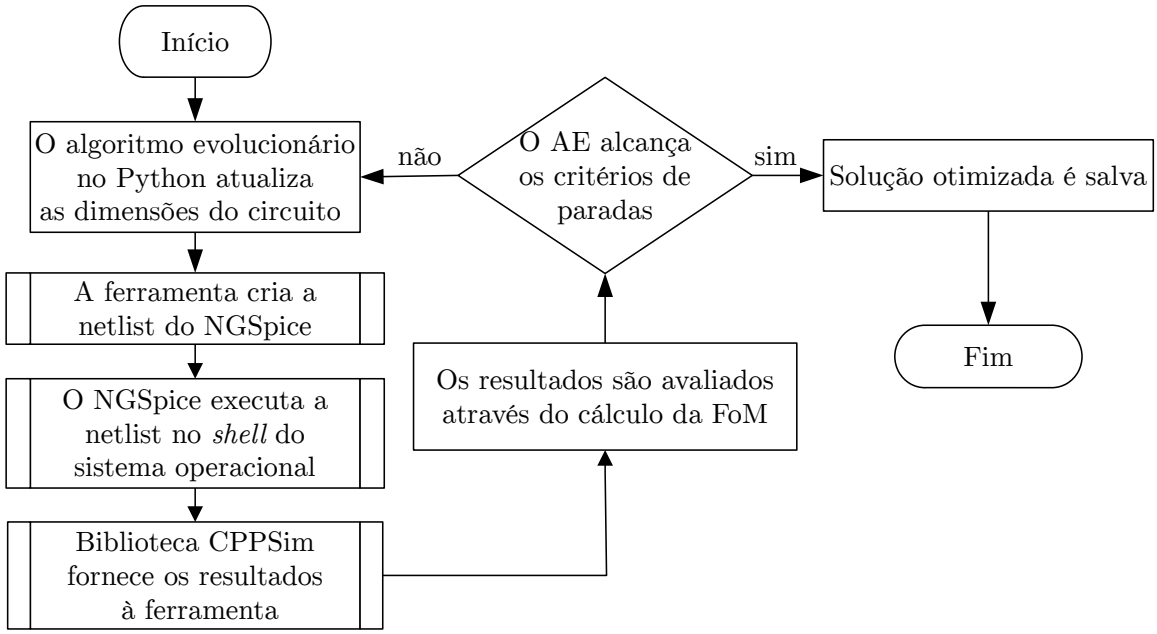


Figura II.1: Diagrama ilustrativo do fluxo proposto.

II.2 Testes e Simulações de Avaliação do Desempenho do LDO

Além das simulações, rotinas foram implementadas no Python para processar os valores brutos das simulações em valores que serão utilizados nos cálculos dos parâmetros utilizados na função custo. Estas rotinas realizam as operações necessárias para obter os valores finais dos testes e consequentemente calcular o valor da FoM. Os testes propostos para avaliar o desempenho do circuito e depois serem utilizados para calcular a função custo são descritos da seguinte forma:

- **Teste A**, $V_{OUT\text{nom}}$: a tensão de entrada é elevada até o máximo permitido pela tecnologia CMOS (V_{INmax}), medindo a tensão de saída. Como mostrado na Figura II.2, a tensão de saída é comparada com o valor desejado para a regulação (V_{OBJ}) a partir do momento em que a tensão de entrada é maior que V_{OBJ}. Neste teste a distância euclidiana do valor da tensão de saída com a tensão V_{OBJ} é calculada e depois dividida pelo número de pontos vindos do simulador. Isto tem o efeito de minimizar a área mostrada em cinza na Figura II.2, representada pela variável V_{LR}. Um fluxograma desta função é mostrado na Figura II.3, onde os vetores X_A e Y_A representam a tensão de entrada e saída respectivamente, sendo os valores dos eixos do gráfico na Figura II.2. Esta função também retorna uma aproximação do *dropout* (V_{do}). Já a distância euclidiana é mostrada em outro fluxograma na Figura II.7, retornando o valor V_{LR}.

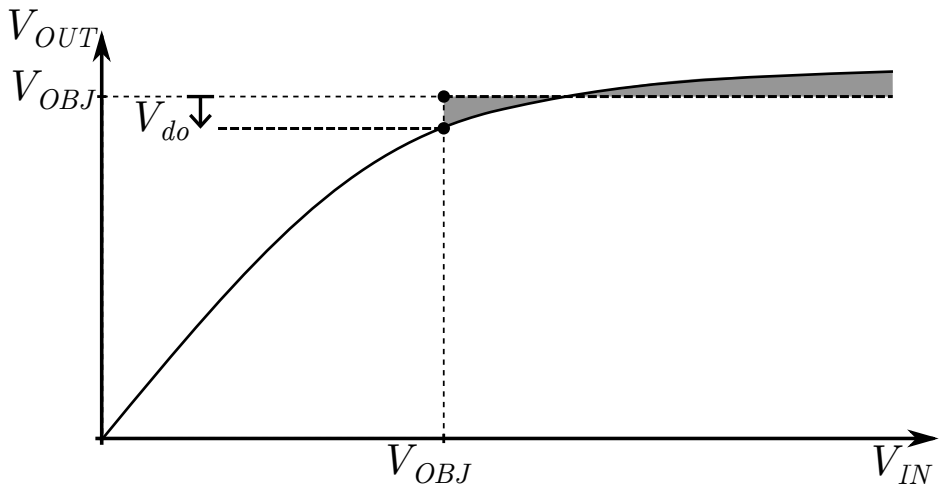


Figura II.2: Teste de regulação de tensão (teste A).

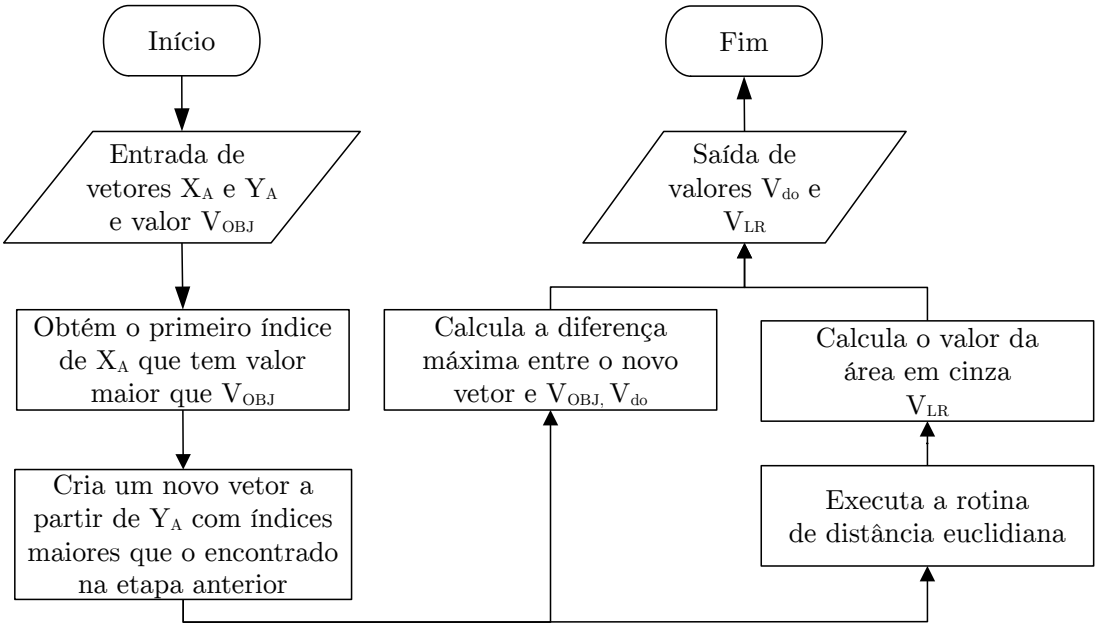


Figura II.3: Fluxograma do algoritmo para obtenção da regulação.

- **Teste B**, I_{MAX}: monitora-se V_{OUT} enquanto a corrente de carga no nó de saída é aumentada (ver Fig. II.5), e retorna o valor da corrente quando a tensão fica abaixo de um limite (lim_{B}) pré-determinado ($V_{OUT}@I_{MAX}$). Este teste foi depreciado devido à influência negativa observada na convergência, com este teste piorando o resultado dos demais, sendo executado apenas ao final da otimização, o que também poupa esforço computacional. O resultado apenas explica o desvio de V_{OUT} durante variações de correntes de saída no Teste D. O fluxograma da Figura II.4 foi implementado e é utilizado para realizar a obtenção do valor I_{MAX}, onde o vetor X_B representa a corrente de saída e Y_B representa o valor da tensão no nó de saída.

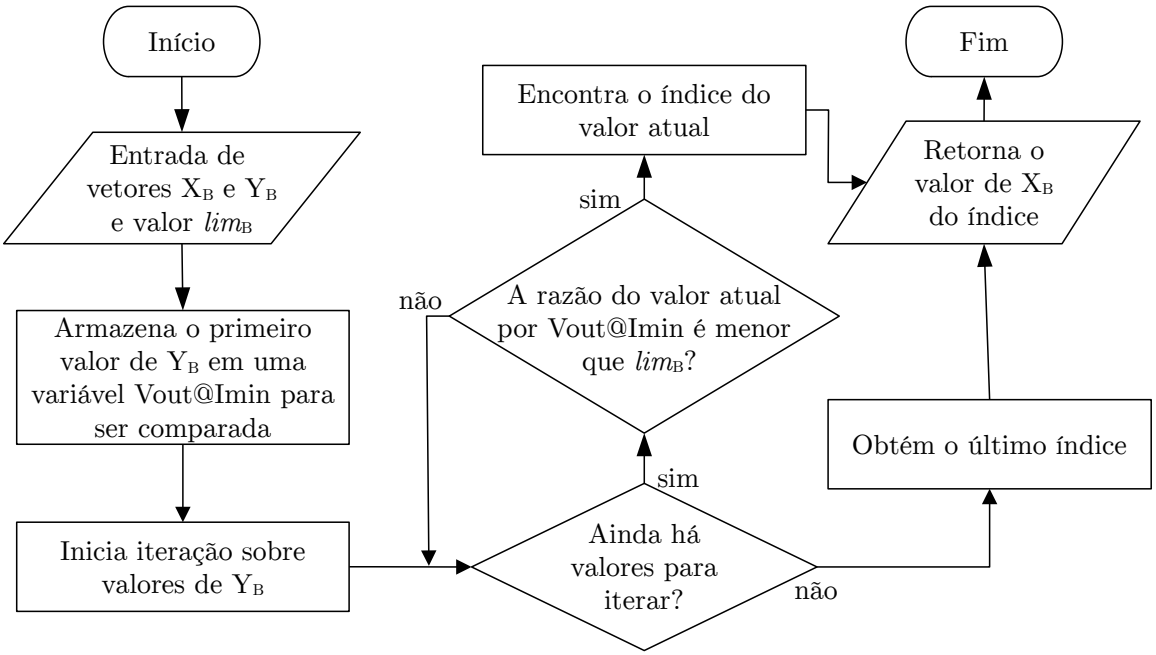


Figura II.4: Fluxograma para obtenção da máxima corrente de carga.

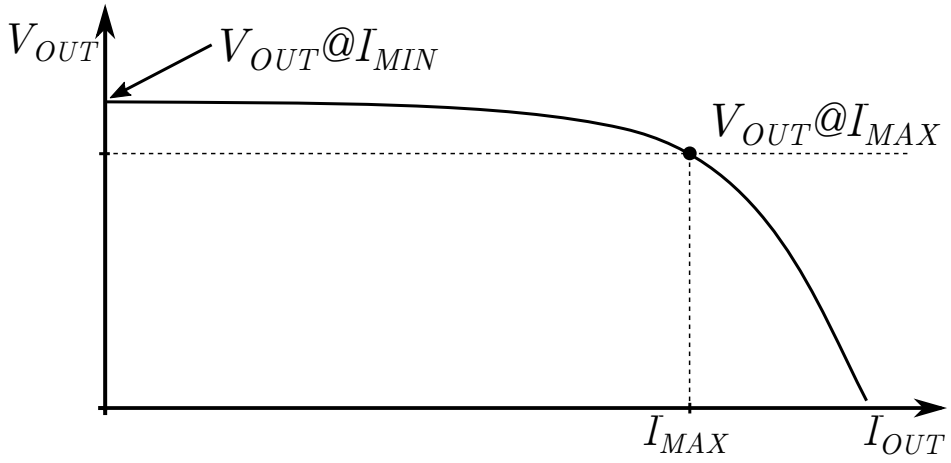


Figura II.5: Teste de corrente máxima (Teste B).

- **Teste C**, $\Delta V_{OUT,C}$: observa-se a tensão de saída enquanto um degrau de tensão é aplicado no nó de entrada, como pode ser visto na Fig. II.6. O pulso aplicado tem tensão mínima maior que $V_{OBJTest}$ e máxima de $V_{IN\max}$. Assim como o Teste A, este teste é calculado somando a distância euclidiana do valor instantâneo com a tensão desejada em que o circuito regule, V_{OBJ}, e depois dividida pelo número de pontos vindos do simulador. Também minimizando a área mostrada em cinza na Fig. II.6 através do algoritmo mostrado no fluxograma da Figura II.7. No entanto, este teste foi depreciado devido ao custo computacional de uma simulação transiente quando comparado à pequena melhora observada na convergência.

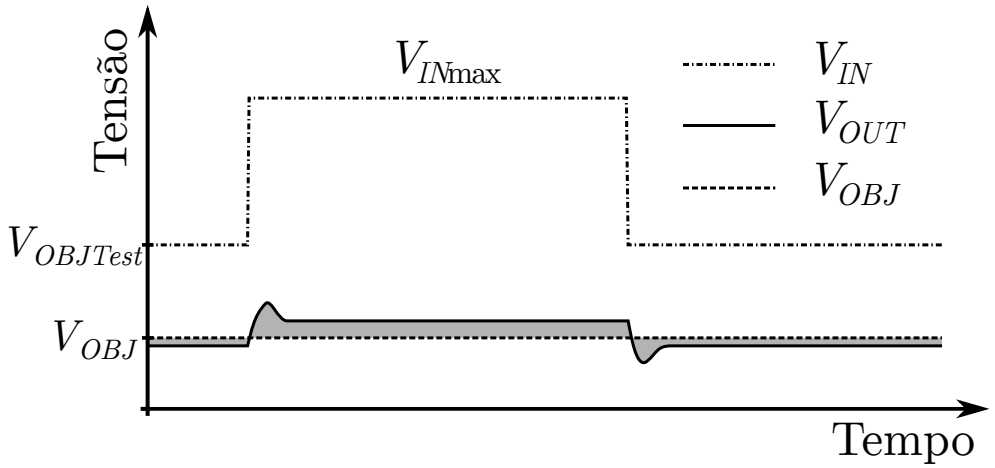


Figura II.6: Teste transiente para variação de tensão de entrada (Teste C).

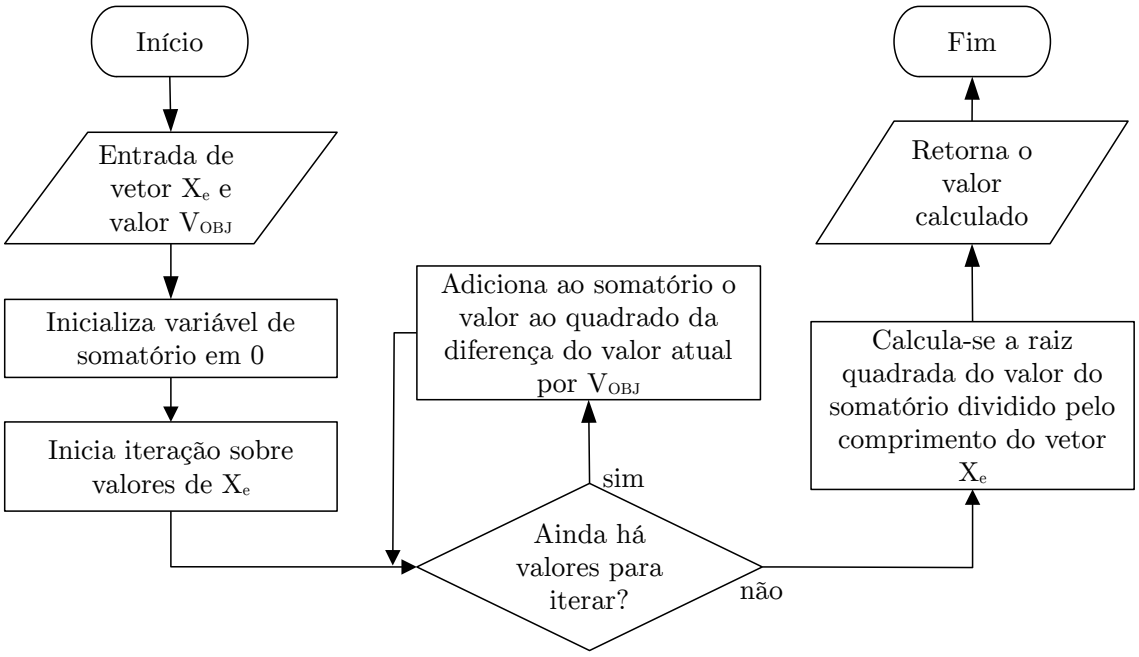


Figura II.7: Fluxograma para obtenção do somatório da distância euclidiana.

- **Teste D**, ΔV_I: é similar ao Teste C, mas variando a corrente de saída ao invés da tensão de entrada. A tensão no nó de saída também é monitorada, como pode ser visto na Fig. II.8, em que I_{MAX} é um valor pré definido. O resultado deste teste (ΔV_I) é calculado de modo análogo ao Teste C.

- **Teste E**, margem de fase: este teste é realizado no circuito em malha aberta como mostrado na Fig. II.10. O bloco STB, de [14], realiza a abertura da malha e os testes necessários para a análise. O resultado do simulador é obtido na forma de um vetor de números complexos, que são transformados em módulo através da Equação (II.2) e em fase através da rotina *angle*

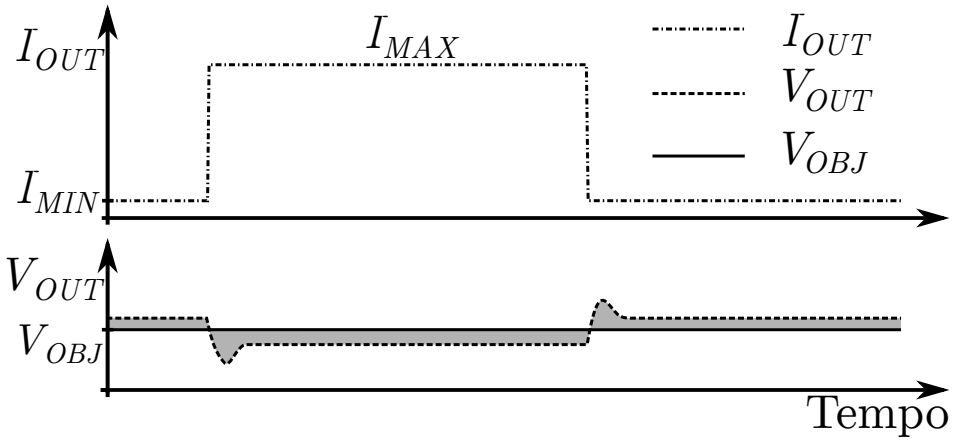


Figura II.8: Teste transiente para variação de corrente de saída (Teste D).

da biblioteca *numpy*. Caso o valor da margem de fase encontrado seja menor que 45º, a Equação (II.1) é utilizada; caso contrário, o valor 1 é retornado. O fluxograma do algoritmo utilizado neste teste é mostrado na Fig. II.9.

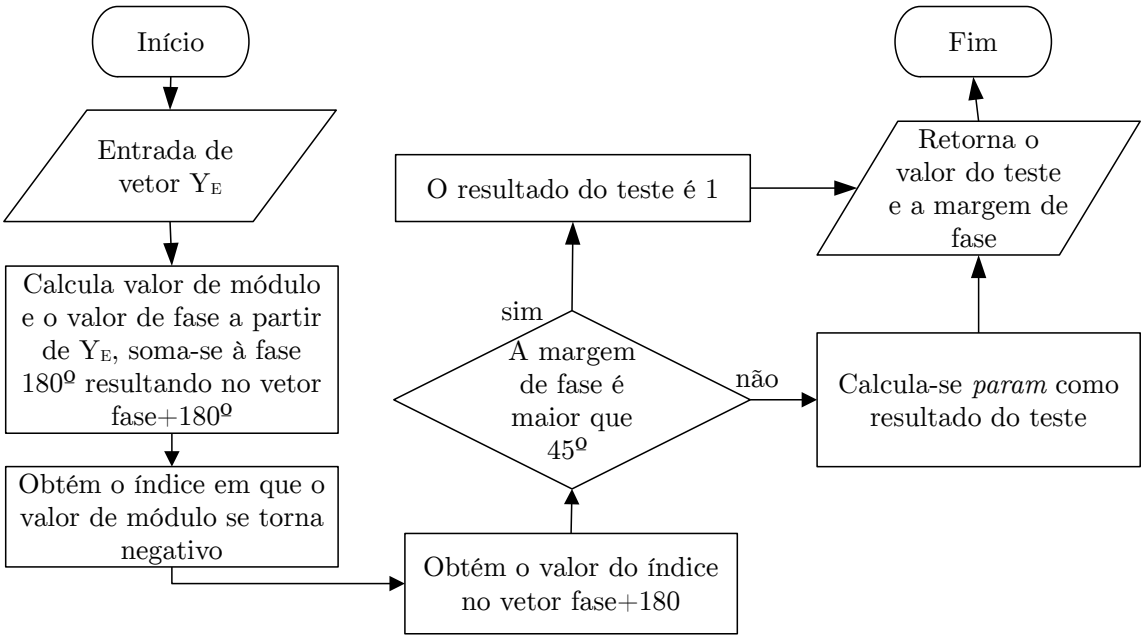


Figura II.9: Fluxograma para obtenção da margem de fase.

$$param = \left(1 + \frac{45 - \text{margem de fase}}{K_{ph}}\right)^2 \tag{II.1}$$

$$\text{módulo}_Y = 20 \times \log_{10} Y_E \tag{II.2}$$

- **Teste F**, C_{TOT}: capacitância total do circuito medida no nó de saída que inclui a capacitância do circuito somada à capacitância conectada ao nó de saída. Este teste é realizado com apenas os transistores M_{P1} e M_{P2} a fim de poupar esforço computacional, já que são os únicos

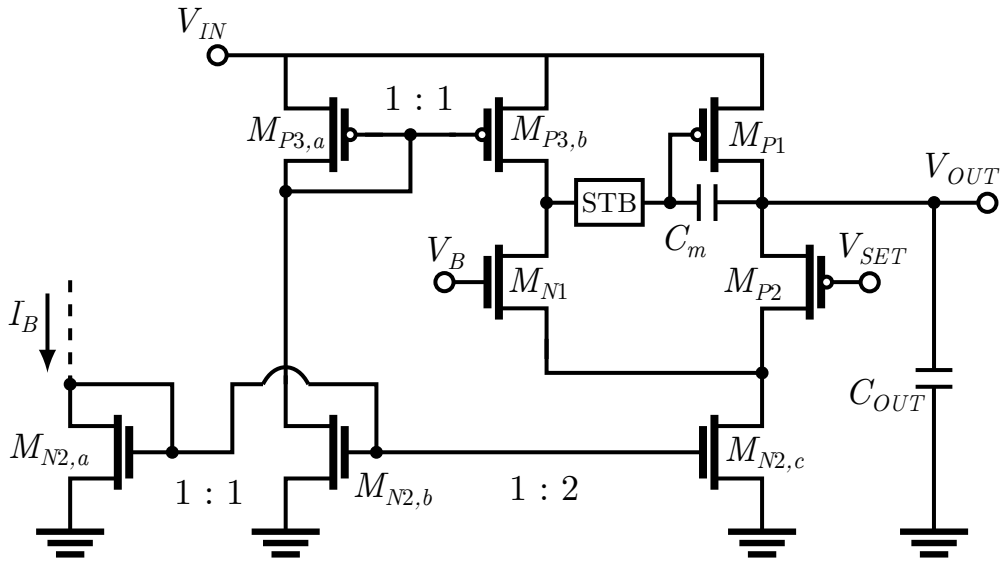


Figura II.10: Circuito em malha aberta para teste de margem de fase.

transistores conectados ao nó V_{OUT} como mostrado na Figura II.11. Uma fonte de corrente *ac* é conectada ao nó de saída e a Equação (II.3) é aplicada à tensão complexa ($V_{OUT_{(f_0)}}$) medida durante o teste a fim de se obter C_{TOT}, em que f_0 é a frequência em que o teste é executado, e Img$\{x_c\}$ representa o cálculo da parte imaginária do número complexo x_c.

$$C_{TOT} = \frac{\text{Img}\left\{\frac{1}{V_{OUT_{(f_0)}}}\right\}}{2 \times \pi \times f_0} \tag{II.3}$$

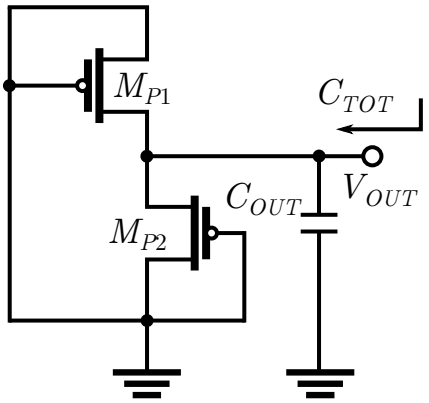


Figura II.11: Circuito para o teste de extração de capacitância de saída.

II.3 Variabilidade Monte Carlo nas Simulações

Dispositivos semicondutores estão sujeitos a variações durante sua fabricação que não podem ser controladas, incluindo variações aleatórias, sistêmicas e de processo [15], com estas duas últimas abordadas no Capítulo IV. Para lidar com as variações aleatórias e garantir que o circuito obtido ao

final da otimização seja funcional, variações Monte Carlo são empregadas durante a simulação do circuito no NGSpice, com os blocos de processo pré-definidos na Figura II.1 mostrando as etapas em que esta técnica é empregada na otimização. Essas simulações utilizam parâmetros dos modelos de fabricação, fornecidos pelo fabricante (*foundries*) do *chip*, que internamente aplicam as variações. A implementação utilizada não realiza todas as simulações internamente no NGSpice, como em soluções comerciais, e por isso, o NGSpice precisa simular o mesmo circuito várias vezes. Cada uma dessas simulações do mesmo circuito resulta em valores diferentes, e devem ser levados em conta durante a otimização, pois possuem em si uma representação de como um possível dispositivo em um *chip* pode se comportar. O diagrama da Figura II.12 mostra como as variações Monte Carlo do NGSpice são levadas em consideração, em que as *netlists* com Monte Carlo contêm apenas a configuração dos parâmetros referentes às variações adicionadas.

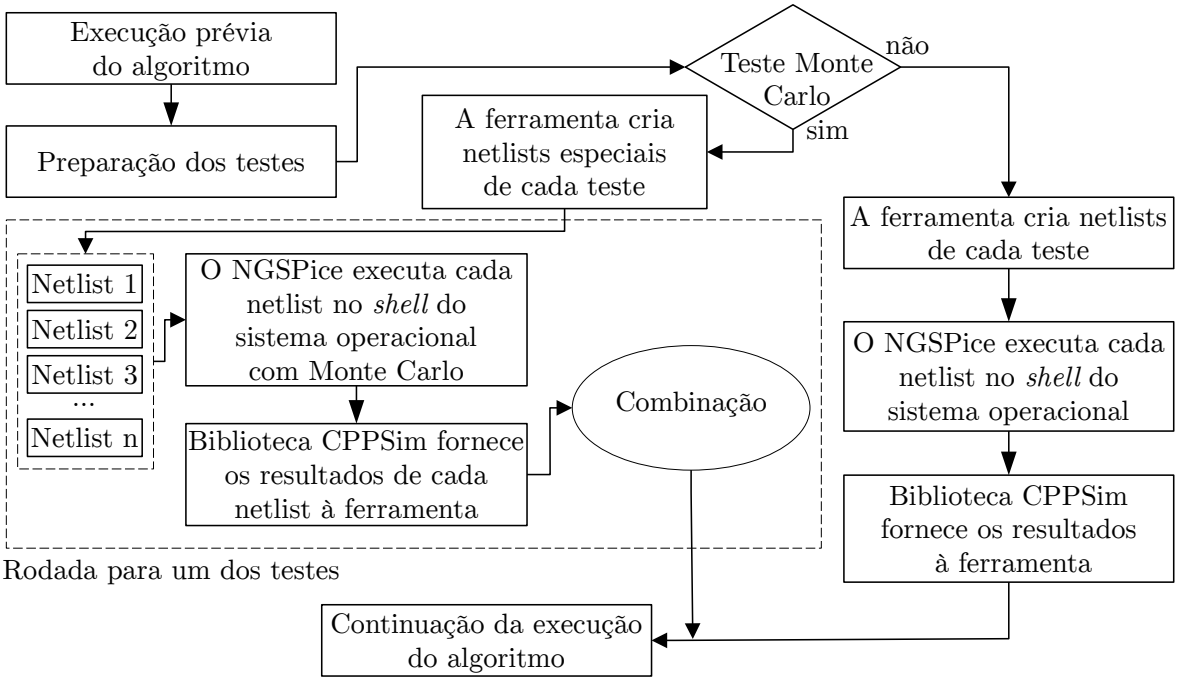


Figura II.12: Descrição de Monte Carlo no algoritmo.

Os testes A, D e E, apenas, são executados com variações Monte Carlo, com o objetivo de simular os efeitos das variações de fabricação nos testes que desempenham um papel mais relevante. Cada circuito é simulado mais de uma vez no NGSpice, resultando em vários resultados com casos de variabilidade diferentes, mas esses valores obtidos devem ser combinados em um único valor por teste, para simplificar a representação de uma possível solução no AE. Cada teste pode ter uma forma de combinação diferente e é descrito a seguir:

- **Teste A**: Para o valor de V_{LR} e V_{do} é calculada a média e o desvio padrão. Como se deseja minimizar este valor, quanto menor, melhor, adiciona-se à média o valor do desvio padrão multiplicado por 3, resultando no valor final do Teste A.

- **Teste D**: É obtido de forma análoga ao Teste A, somando-se à média o valor do desvio padrão multiplicado por 3.
- **Teste E**: Devido à importância do teste de margem de fase, apenas o pior valor encontrado em todas as rodadas do Monte Carlo é considerado, garantindo que somente o pior valor das variações seja levado em conta.

De posse dos valores dos testes com Monte Carlo, o AE pode, juntamente com os testes que não têm as variações ativadas, computar o valor final da função custo e prosseguir com a otimização. Durante a otimização, o AE não possui controle sobre as variações Monte Carlo, sendo estas totalmente independentes e executadas pelo NGSpice, com as únicas implementações no Python sendo: a execução do mesmo teste mais de uma vez, com os mesmos parâmetros e valores; e as rotinas de combinação descritas anteriormente.

II.4 Algoritmo Evolucionário Para Otimização do Circuito LDO

Em trabalhos passados [11, 16] foi utilizado um outro tipo de algoritmo evolucionário chamado de Evolução Diferencial (ED), bastante utilizado no início do desenvolvimento desta ferramenta, aproveitando-se de uma função já implementada no *tensorflow* [17], pois na época o direcionamento maior era voltado para: testes a serem executados; métodos de integração com o simulador de circuitos; e o próprio circuito em si. No entanto a ED do *tensorflow* possuía algumas limitações, como não limitar o espaço de busca; não salvar o progresso ao longo das iterações; e não possuir limite de melhora intergeracional. Com os testes, métodos e circuito já razoavelmente estabelecidos, um Algoritmo Genético (AG) básico [18] foi implementado do zero em Python, e o fluxograma mostrado na Fig. II.13 detalha como os testes discutidos anteriormente, bem como os parâmetros do circuito, são processados pelo algoritmo.

II.4.1 Inicialização da População Inicial

O problema abordado possui como solução um conjunto de valores específicos para cada parâmetro, em que cada solução é chamada de indivíduo, ou cromossomo, e cada parâmetro da solução é chamado de gene; esses parâmetros são codificados por valores reais neste trabalho. Os AGs funcionam com uma população de indivíduos, realizando operações entre eles e criando novas gerações da população, mas a população deve ser inicializada antes de ser otimizada, e essa operação deve levar em conta o número de variáveis do problema; o espaço de busca para as variáveis do problema; e o tamanho da população. Uma das formas de inicializar a população inicial é por meio de uma distribuição aleatória no espaço de busca, que o cobre de maneira linear, sorteando valores para cada gene de cada indivíduo dentro do respectivo domínio, ou de forma logarítmica, que cobre o

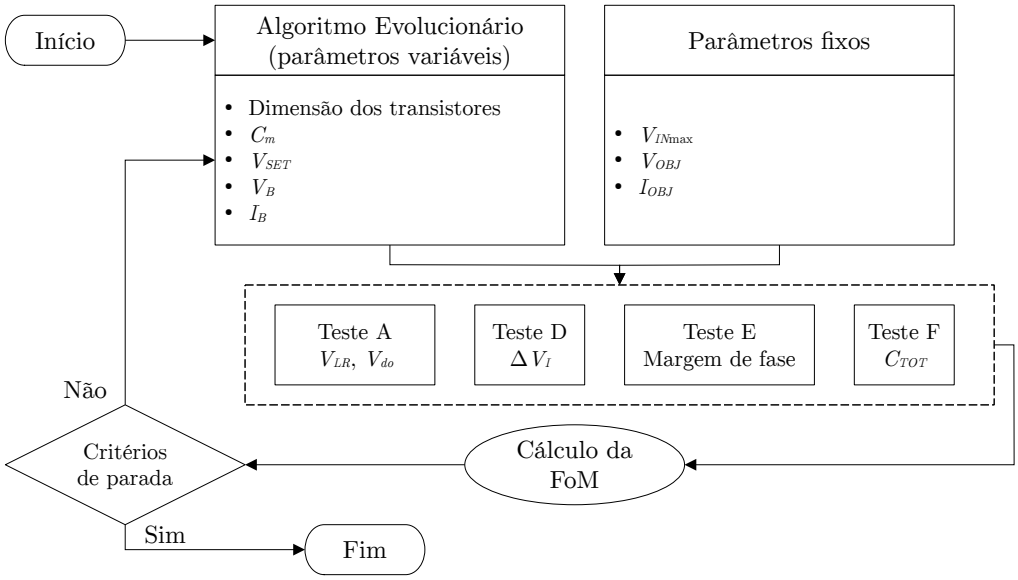


Figura II.13: Fluxograma ilustrativo proposto.

espaço de busca de modo a distribuir os genes com a mesma chance entre 1 e 10 e entre 11 e 100. Este último foi implementado pois as dimensões dos transistores possuem respostas logarítmicas, ficando a cargo do usuário escolher qual deseja utilizar. A forma com que os genes são gerados de modo logarítmico é dada por:

$$C_a = 10^{\log_{10}(lim_{min})+v_a(\log_{10}(lim_{max})-\log_{10}(lim_{min}))}, \tag{II.4}$$

em que C_a é o valor do gene, lim_{min} é o valor mínimo do intervalo de busca para o respectivo gene, v_a é um valor aleatório gerado entre 0 e 1 e lim_{max} é o valor máximo do domínio do gene.

No entanto, não foi observada diferença significativa no resultado final quando se compara a inicialização linear com a logarítmica, mas com os valores iniciais destoando dos lineares. Por isso, a otimização foi inicializada de forma linear. Como pode ser observado na Figura II.14, algumas das rodadas logarítmicas foram inicializadas com o valor da função custo muito acima dos demais, e com uma outra rodada ficando presa em um mínimo local até o fim da otimização.

Com isso busca-se uma solução para o circuito mostrado na Fig. I.2 [19, 4]. Os genes de cada indivíduo são representados pela tensão de ajuste V_{SET}, tensão de polarização V_B, corrente de polarização I_B, capacitor de *miller* C_m, e as dimensões físicas de todos os transistores contidos no circuito.

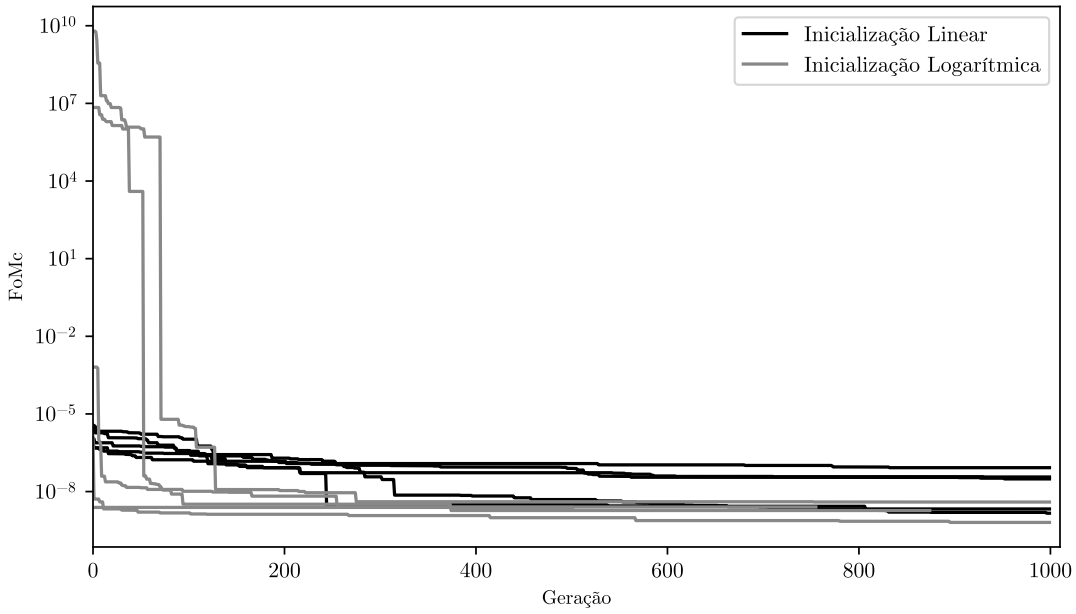


Figura II.14: Comparação de inicialização linear/logarítmica.

II.4.2 Avaliação e Melhorias Computacionais

Após ser criada, toda a população é avaliada através da função de avaliação, em que nesta etapa o usuário tem algumas configurações a serem escolhidas. Primeiramente, o código permite escolher entre executar indivíduo por indivíduo ou todos os indivíduos em paralelo. Segundo, também é possível escolher entre executar todos os testes de avaliação por simulações em paralelo ou individualmente, dependendo do custo computacional desejado. O resultado dessa avaliação resulta no valor de aptidão de cada indivíduo, que é um valor que mensura o quão bem a possível solução se posiciona em relação às outras.

Ao escolher a avaliação individual, cada indivíduo é simulado no NGSpice de maneira independente, o que permite uma verificação dos resultados de simulação mais fácil, visando entender o que pode estar acontecendo com um circuito em específico. Essa execução é utilizada, por exemplo, no indivíduo já otimizado, após a finalização da otimização, para obter os valores dos testes. Caso se opte pela execução em paralelo, uma *netlist* contendo todos os indivíduos da população é criada para cada teste. Onde cada circuito possui conexões compartilhadas com os outros em nós que não interferem nos circuitos adjacentes, possuindo apenas componentes específicos para cada um que não poderiam ser compartilhados. Essa execução da população como um todo pelo NGSpice é feita com o intuito de economizar recursos computacionais.

Outra opção é executar os testes em paralelo, ou seja, alocar cada processo a um *thread* no computador, que permite que o teste seja executado ao mesmo tempo que outros caso o processador possua mais de um núcleo de processamento. Essa opção é vantajosa em máquinas com muitos

núcleos de processamento disponíveis, mas não é recomendada caso o usuário utilize a máquina para outras atividades enquanto a otimização é executada, para o caso em que o número de *threads* exceda o número de núcleos disponíveis, pois é perceptível a diminuição da responsividade da máquina. Caso o usuário opte por não utilizar o processamento em paralelo, cada teste é executado em sequência, o que reduz a carga de trabalho no processador, permitindo que outras atividades possam ser realizadas na mesma máquina ao mesmo tempo em que se executa o AG, ao custo de um tempo significativamente maior na otimização.

Como as máquinas disponíveis puderam ser alocados para a tarefa exclusivamente, tanto a opção de executar a população como um todo, quanto a opção de executar os testes em paralelo foram utilizadas. O número de rodadas Monte Carlo é adicionada ao número de testes, por exemplo, dos 4 testes utilizados, se 3 possuem 10 variações Monte Carlo cada, o total resultará em 31 *threads*.

Um outro tipo de melhoria computacional foi utilizada para a avaliação da população, essa sem a opção de ser ativada/desativada. A primeira vez que a população é avaliada, todos os indivíduos são avaliados; posteriormente, apenas os descendentes são avaliados. Como foi utilizado um cruzamento que resulta em um número menor de indivíduos novos em relação à população inicial, apenas metade ou 1/4 da população é simulada. Isso reduz o custo computacional em detrimento de que população como um todo não sofra mutação, mas apenas os descendentes.

A função de avaliação utilizada no AG será chamada de Figura de Mérito, FoM, do inglês *Figure of Merit*, ao longo da dissertação, pois é referenciada com este nome na literatura, em que um valor menor é considerado com melhor aptidão, e com isso o AG desenvolvido deve minimizá-la. A função de avaliação utilizada foi baseada nos trabalhos [4, 20] definida como

$$\text{FoM} = \frac{C_{TOT}\Delta V_{OUT} I_Q}{\Delta I_{LOAD}^2} \quad . \tag{II.5}$$

Onde a variável C_{TOT} é a capacitância total, que inclui a capacitância do circuito somada à capacitância conectada à saída; ΔV_{OUT} é o resultado do Teste D; I_Q é a corrente de polarização, sendo três vezes I_B; e ΔI_{LOAD} é a variação de corrente aplicada ao Teste D. Algumas modificações foram feitas na Equação (II.5) para considerar os testes A; D; e E, já que estas métricas são importantes para encaminhar melhor o AG para o resultado ótimo. A versão adaptada da FoM é abordada na seção III.1.1, mostrada na Equação (III.4).

II.4.3 Seleção de Indivíduos

O operador de seleção em AGs é encarregado de selecionar indivíduos já existentes para um determinado processo, seja ele o de cruzamento ou o de substituição de indivíduos. As duas formas de seleção foram implementadas durante o desenvolvimento das rotinas do AG, ficando a cargo

do usuário escolher qual delas será utilizada durante a otimização, escolhendo entre os modos de torneio ou roleta.

Modo Torneio

No modo torneio, a população é dividida em dois grupos de tamanhos iguais, ganhadores e perdedores, em que dois participantes do torneio são escolhidos aleatoriamente na população; o que tiver a melhor aptidão é alocado no grupo dos ganhadores e o pior no grupo dos perdedores.

Modo Roleta

No modo roleta, a escolha dos indivíduos possui relação com sua aptidão, em que quanto melhor, maior é a chance de ser escolhido. Como a FoM possui melhor aptidão em menores valores, o valor desta função é invertido e então é calculado o quão bom o indivíduo é em relação aos outros de modo normalizado, para que a soma desses valores para todos os indivíduos seja igual a 1. Esses valores normalizados geram intervalos correspondentes à aptidão do indivíduo, em que o mais apto possui o maior intervalo e o menos apto um intervalo menor, com isso é possível sortear um número aleatório entre 0 e 1 de modo que caia em um dos intervalos, sendo este intervalo respectivo ao indivíduo que será escolhido. Após serem selecionados, estes são removidos da lista para não serem selecionados novamente.

II.4.4 Cruzamento

O operador de cruzamento em AGs é encarregado de criar novos indivíduos a partir dos já existentes, no caso deste trabalho, foram implementados dois métodos diferentes que produzem um novo indivíduo a partir de outros dois. A escolha do método de cruzamento depende do tipo de seleção escolhida.

Cruzamento para o Modo Torneio

Os novos indivíduos são criados apenas a partir do subgrupo dos ganhadores, que são submetidos à operação de cruzamento de dois a dois até se esgotar o subgrupo. Durante o cruzamento, cada genoma possui um valor percentual definido pelo usuário representando a chance de passar pela operação ($P_{\text{cruzamento}}$); se ele for escolhido, é feita a média dos valores dos dois indivíduos selecionados; se não for escolhido, o valor é mantido inalterado do primeiro indivíduo. Essa operação gera 1/4 da população inicial de novos indivíduos; por exemplo, se a população inicial for de 100, haverá 25 descendentes, contribuindo para a melhoria de esforço computacional mencionado na seção II.4.2.

Cruzamento para o Modo Roleta

Os indivíduos selecionados pelo modo roleta têm a chance reduzida de cruzar um indivíduo com boa aptidão com outro de menor aptidão. Com os dois indivíduos selecionados, o mesmo método de cruzamento para indivíduos selecionados pelo modo torneio é aplicado, com um valor definido pelo usuário determinando se haverá ou não cruzamento de um determinado genoma, efetuando a média dos valores caso haja. Mas, diferentemente do modo torneio, o gene que não passou pela operação de cruzamento pode ser herdado tanto do primeiro quanto do segundo, com 50% de chance. Esta operação gera 1/2 da população inicial de novos indivíduos, gerando 50 descendentes para uma população de 100.

II.4.5 Mutação

A técnica de mutação contribui para que o AG não fique preso em mínimos locais, fazendo com que um gene selecionado seja alterado para um valor que não possui relação com a geração passada, diferentemente do cruzamento, aumentando a variabilidade genética. Durante o desenvolvimento da ferramenta foram implementadas duas formas de realizar a mutação, uma que gera um valor aleatório no espaço de busca e outra que altera um valor em uma faixa menor em torno do valor que irá sofrer a mutação. Para definir a probabilidade de um gene sofrer mutação, utiliza-se a Equação (II.6). Para que haja probabilidade de ocorrer mutação em MUT_n genomas, em que Dim_P é o número de genomas por indivíduo, que também representa a dimensão do problema.

$$P_{\text{mutação}} = \frac{MUT_n}{Dim_P} \tag{II.6}$$

O valor aleatório gerado no domínio do gene é distribuído de forma uniforme, não logarítmica, fazendo com que cada valor no intervalo tenha probabilidade igual de ser sorteado. O outro método, proposto por [21], sorteia um valor em um novo intervalo calculado a partir de uma fração do intervalo de busca; a Figura II.15 mostra como seria para uma dimensão, e o valor é obtido através de

$$C_{a(i+1)} = C_{a(i)} + \frac{v_b \times R \times (lim_{min} - lim_{max})}{2^{K_p v_a}}, \tag{II.7}$$

com $C_{a(i)}$ e $C_{a(i+1)}$ sendo o valor do gene antes e depois da mutação; v_b é um valor aleatório entre -1 e 1; R é um valor entre 0 e 1 definido pelo usuário que multiplica o limite que define a fração do intervalo de busca, mantido em 0,3 durante a otimização; K_p é outro valor definido pelo usuário que quanto maior, menor é a fração do intervalo de busca na Figura II.15, mantido em 4 durante a otimização; e por fim v_a que é um valor aleatório entre 0 e 1. A Figura II.15 mostra uma variação do valor do parâmetro K_p para um espaço de busca de 2 dimensões com seu valor acrescentado de

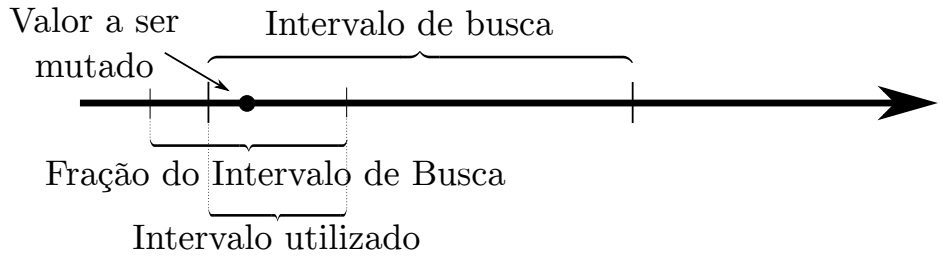


Figura II.15: Intervalo de mutação.

10 em 10 (de (a) para (b), (c) e (d)), aplicado a indivíduos com valor na origem. Após a operação descrita na Equação (II.7), o valor é colocado no limite original da variável, fazendo com que, se o valor saia do intervalo de busca, o valor de limite do intervalo extrapolado seja utilizado; com isso, a Figura II.16 mostra o intervalo que o novo valor pode ser sorteado no intervalo utilizado. A função de mutação também realiza o truncamento para 9 dígitos do valor, com o objetivo de evitar valores que seriam truncados pelo simulador de circuitos. Durante a otimização foi utilizado o método da fração do intervalo.

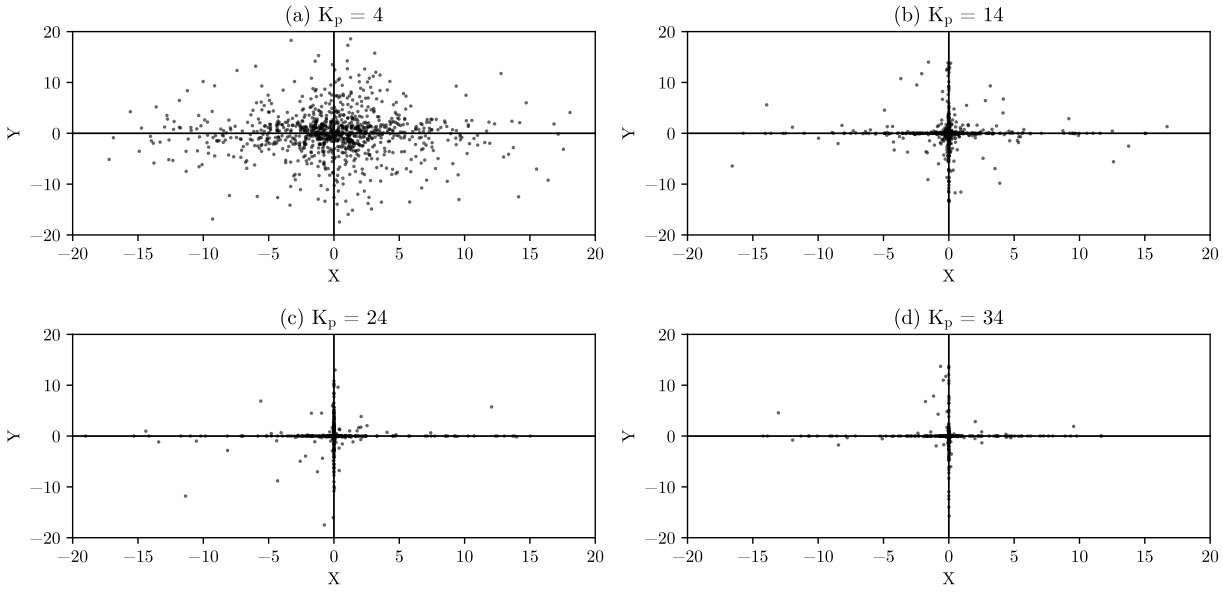


Figura II.16: Variações do parâmetro K_p em 2 dimensões.

II.4.6 Substituição da População

Após o processo de cruzamento, novos indivíduos são gerados, e, como foi utilizado um número fixo de indivíduos na população por geração, alguns indivíduos precisam ser eliminados da população e substituídos pelos novos. Caso o processo de seleção utilizado seja o torneio, que divide a população em 2 grupos, com apenas o grupo dos ganhadores gerando descendentes, a nova população de descendentes gerada tem tamanho 1/4 do tamanho da população e substitui indivíduos apenas no subgrupo dos perdedores. Caso o método utilizado seja o da roleta, que gera metade

do tamanho da população de novos descendentes, qualquer indivíduo pode ser substituído na população geral a depender do método utilizado.

Foram implementadas duas formas de substituição que podem ser selecionadas pelo usuário na programação. O mais simples substitui o final da lista de indivíduos da população pelos descendentes, que, quando gerados pelo método de torneio, são obtidos de maneira aleatória do cruzamento, e a lista também não é organizada, o que acaba tendo o efeito de substituir aleatoriamente os indivíduos. Já com os descendentes originados do modo roleta, os piores seriam substituídos, já que neste método a população é organizada do melhor para o pior. A outra forma de substituição utiliza um método similar ao da roleta para selecionar indivíduos que serão substituídos, mas, desta vez, quanto pior a avaliação do indivíduo, maior é a chance de ser substituído. No entanto, os 10% melhores são sempre mantidos, com os demais podendo substituir qualquer indivíduo no restante da população, sendo este o método utilizado. Ambos os métodos são implementados com elitismo, de modo a preservar os melhores indivíduos.

II.4.7 Critérios de Parada

Para determinar quando o algoritmo deve parar, foram utilizados conjuntamente dois critérios, em que o primeiro critério de parada atingido é suficiente para encerrar a otimização.

O primeiro, um limite de gerações, Gen_{max}, interrompe a otimização quando o número máximo de iterações é atingido. E o segundo, um limite máximo de melhoramento intergeracional, que avalia como o melhor indivíduo da população evolui ao longo das gerações. Esse critério é obtido calculando a razão entre o melhor indivíduo da geração atual e o melhor da geração passada. Se este valor for menor que Gen_{tol} por um determinado número de gerações, Gen_{th}, a otimização é encerrada. Isso evita que a otimização fique presa em um mínimo local, desperdiçando recursos computacionais.

II.4.8 Implementação do Algoritmo Genético

O AG descrito foi implementado em uma biblioteca separada em Python, de modo que pode ser facilmente utilizado em outros projetos. Durante seu desenvolvimento, o AG foi testado em duas funções, sendo utilizada inicialmente uma função análoga a uma parábola, mas de dimensão n, que possui apenas um mínimo local/global em (0,0, ..., 0), e, posteriormente, a função Schaffer F6. No entanto, o problema possui dimensão maior que 2, e por isso foi utilizado o método de envolvimento ponderado (do inglês *weighted wrap*) que faz com que a função bidimensional seja calculada para n dimensões [22], que também possui um mínimo global em (0,0, ..., 0), mas, diferentemente da

primeira função utilizada, esta possui muitos mínimos locais, sendo dada para duas dimensões por:

$$f(x,y) = 0.5 + \frac{\sin^2\left(\sqrt{x^2+y^2}\right) - 0.5}{\left[1 + 0.001\left(x^2+y^2\right)\right]^2}, \tag{II.8}$$

com x e y sendo limitados no algoritmo entre -100 e 100. Essas duas funções podem ser vistas na Figura II.17 para o paraboloide e a função Schaffer F6 respectivamente.

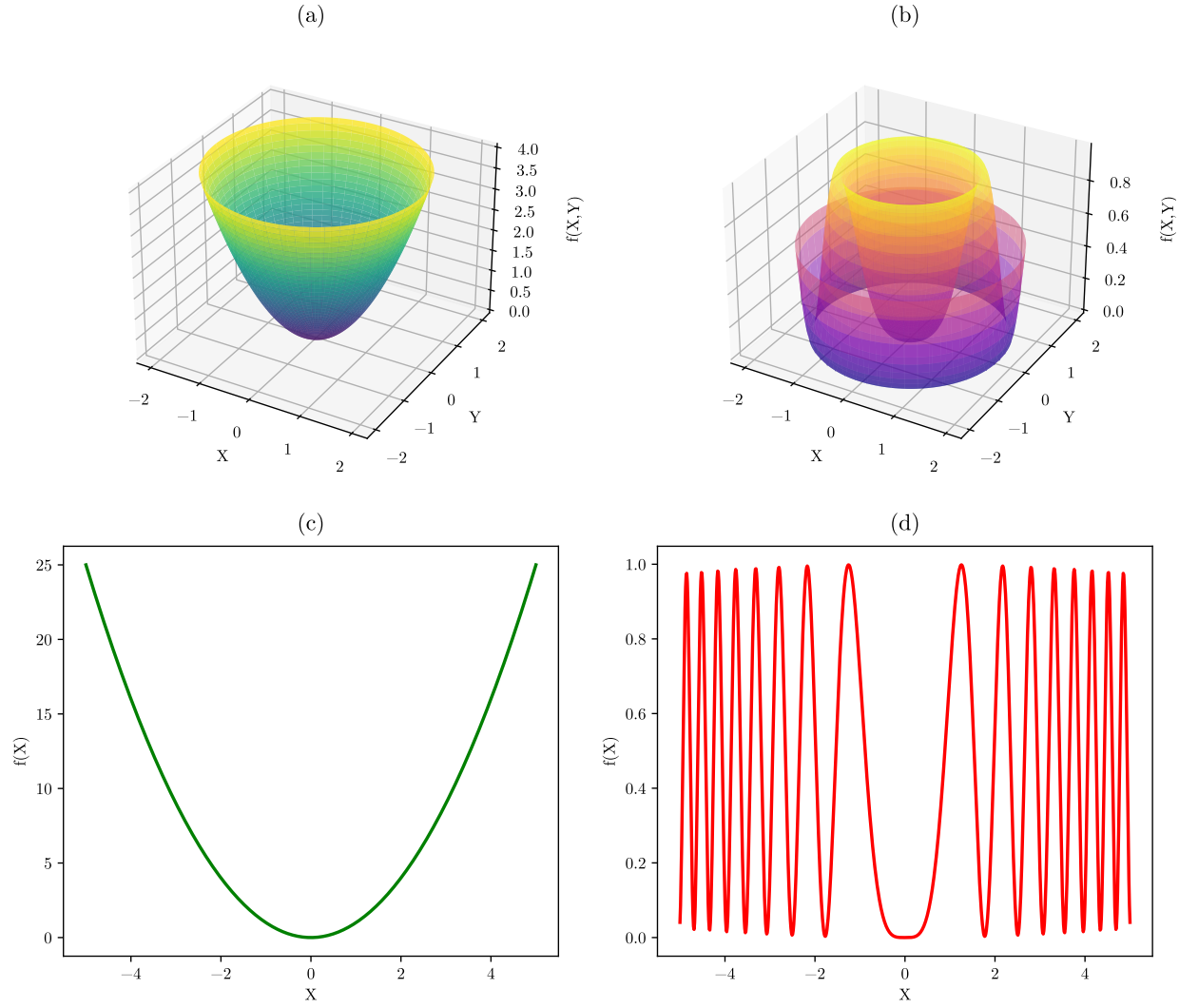


Figura II.17: Representação das funções de testes utilizadas.

Já em (c) e (d) as funções vistas são para apenas uma dimensão. Em (a) e (b) foram utilizadas coordenadas polares com raio máximo de 2 para evitar distorções devido à resolução limitada da figura 3D.

As funções passaram pelo processo de otimização, mas com os critérios de parada alterados, já que ambas possuem mínimos globais conhecidos. O critério de melhoria intergeracional foi alterado para encerrar a otimização quando o módulo de todas as variáveis for menor que 10^{-3}. O número máximo de gerações foi de 10.000 para as rodadas com 2 dimensões e de 100.000 para as rodadas

com 14 dimensões. A Tabela II.1 mostra a média de gerações para alcançar o ótimo global e também a porcentagem de vezes em que o algoritmo conseguiu alcançar o resultado, a taxa de ótimo global, com a otimização sendo executada 128 vezes para cada função.

Tabela II.1: Resultados para 128 rodadas da otimização em funções de teste.

	F6 2D	Parábola 2D	F6 14D	Parábola 14D
Média (Iterações)	2071	477	38543	18286
Desvio Padrão (Iterações)	1872	1056	16774	4855
Taxa de ótimo global (%)	97,66	100	21,88	100

Capítulo III Desempenho do Algoritmo Proposto

Neste capítulo são definidos os parâmetros usados nas execuções do algoritmo e suas respectivas funções, assim como a descrição dos resultados obtidos através das simulações elétricas. O tempo necessário para as execuções das otimizações também é discutido, em que três máquinas distintas foram utilizadas para a execução das otimizações. O circuito otimizado nas duas tecnologias CMOS obteve resultados razoáveis quando comparados aos da literatura, mas sem a utilização de outras técnicas no circuito.

III.1 Descrição de Execuções

A ferramenta foi utilizada para projetar o circuito LDO proposto em processos CMOS de 130 nm e 180 nm, sendo a primeira uma tecnologia de código aberto do fabricante Skywater e a segunda da UMC (*United Microelectronics Corporation*). Para realizar a otimização foi proposta uma função custo alternativa baseada na FoM da literatura. Os parâmetros utilizados no circuito, e para a otimização do circuito são descritos nesta seção.

III.1.1 Proposta de FoM Alternativa

A FoM da Equação (II.5) contempla apenas os valores de capacitância total, corrente de polarização, e variação de corrente de saída, correspondentes aos testes realizados. No entanto, a FoM original da Equação (II.5) não cobre os testes que contemplam a regulação de tensão; margem de fase; e área utilizada do circuito. Para considerar essas métricas na função custo do algoritmo, as seguintes alterações foram propostas:

- **Regulação de tensão**: os valores de *dropout* (V_{do}) e a distância euclidiana V_{LR} do Teste A são aplicados

$$LR = A_c \cdot ((1 + V_{do}) \cdot (1 + V_{LR}))^2 \ , \tag{III.1}$$

  em que o termo A_c é uma constante usada para manter o termo LR adimensional.

- **Margem de fase**: define-se um valor mínimo desejável da margem de fase, e caso o valor encontrado durante a simulação seja maior que o valor mínimo, retorna 1. Caso contrário a operação da Equação (III.2) é retornada. Desta forma, como pode ser visto na Equação (III.2)

phlim é o limite inferior pré-determinado, *ph* é o valor da margem de fase medida da simulação e K_{ph} é uma constante para tornar o termo adimensional, fixada em 1 grau durante as otimizações.

$$PH_{teste} = \left(1 + \frac{|phlim - ph|}{K_{ph}}\right) \tag{III.2}$$

- **Área**: estima-se a área dos transistores calculando a área de gate através do produto dos valores de W e L de cada transistor, essas áreas são somadas, e, por fim, divide-se este valor pela área mínima, sendo o produto de W_{min} por L_{min}, com o objetivo de normalizar o termo.

$$A_n = \frac{\sum_1^n W_n \times L_n}{W_{min} \times L_{min}} \tag{III.3}$$

Desta forma, com as ideias propostas nas Equações (III.1), (III.3), (III.2), a nova FoM resulta

$$\mathrm{FoM_c} = \sqrt{A_n} \cdot \frac{LR \cdot PH_{teste}}{2} \cdot \frac{C_{TOT}\Delta V_I I_Q}{I_{OBJ}^2}. \tag{III.4}$$

III.1.2 Parâmetros do Circuito

Em relação aos componentes do circuito mencionados na Seção I.2, o capacitor C_{OUT} foi mantido em 5 pF durante a otimização para todas as tecnologias.

Para os testes que analisam a resposta transiente dos circuitos, a variação foi feita da seguinte forma: (i) para o Teste C, a tensão de entrada foi elevada de 1 V, por ser a tensão desejada, até o limite máximo do processo CMOS, com tempo de subida e descida de 0.1 μs. Para facilitar a visualização, um atraso de 10 μs foi usado; (ii) para o Teste D, a corrente foi elevada de 1 nA até um valor definido como I_{OBJ}, com tempo de subida e descida também de 0.1 μs, e atraso inicial de 20 μs. Os tempos de subida e descida dos testes foram os mesmos utilizados em [20]. (iii) No teste de margem de fase (Teste E) a frequência analisada compreende o intervalo de 0.1 Hz até 10 MHz, pois durante o desenvolvimento percebeu-se que esta janela contempla toda a parte relevante da resposta em frequência.

III.1.3 Parâmetros Utilizados na Otimização e Simulações Elétricas

Os parâmetros fixos utilizados durante a otimização são mostrados na Tabela III.1. Já o espaço de busca passado ao AE é mostrado na Tabela III.2.

Nos testes que não envolvem corrente de saída nas simulações (Testes A, C), foi conectada uma carga de forma que o comportamento não fosse avaliado a vazio. Para o Teste A, foi conectado um resistor de 2 kΩ, o qual consome 500 μA quando a tensão de saída atinge 1 V, evitando, assim, que

Tabela III.1: Parâmetros fixos durante a otimização

Parâmetros	130 nm	180 nm
Capacitor de saída (C_{OUT})	5 pF	
$V_{IN\max}$	1.8 V	
$V_{OBJTest}$	1.6 V	
V_{OBJ}	1 V	
I_{OBJ}	10 μA	500 μA

Tabela III.2: Intervalos do espaço de busca passado ao AE

Parâmetros	130 nm	180 nm
I_B (nA)	de 5 a 500	de 50 a 500
C_m (pF)	de 0,1 a 5	
V_{SET} (V)	de 0 a 1,8	
V_B (V)	de 0 a 1,8	
Comprimento do transistor (μm)	de 0,35 a 5	de 0,3 a 60
Largura do transistor (μm)	de 0,55 a 40	de 0,18 a 40

a tensão deste teste fique negativa, caso uma fonte de corrente ideal fosse conectada à V_{OUT}. Para o Teste C, uma fonte ideal de corrente foi conectada ao nó de saída, consumindo 500 μA para a otimização de ambas as tecnologias CMOS.

III.1.4 Parâmetros do Algoritmo Genético

O algoritmo genético implementado em Python tem como entradas a função peso e a população inicial que já foram mencionadas nas seções I.1 e II.4. Além destes, existem outros parâmetros que alteram as propriedades do algoritmo que foram abordados na Seção II.4, sendo: o fator $P_{\text{cruzamento}}$ que controla se há ou não cruzamento em um genoma; o fator $P_{\text{mutação}}$ que é obtido através da Equação (II.6); e, por fim, os valores relacionados aos critérios de parada, que são mostrados na Tabela III.3.

Tabela III.3: Parâmetros do algoritmo genético

Parâmetros	
Dimensão do espaço de busca (Dim_P)	14
$P_{\text{cruzamento}}$	0,7
MUT_n	3
$P_{\text{mutação}}$	0,21
Gen_{max}	1000
Gen_{tol}	10^{-4}
Gen_{th}	500

III.2 Resultados de Otimização

A otimização foi rodada 30 vezes para cada tecnologia CMOS devido à natureza aleatória do método, e a FoMc para os 5 melhores resultados, assim como o pior resultado, para cada tecnologia são mostradas na Tabela III.4. Os valores obtidos dos testes para o melhor resultado são mostrados na Tabela III.5, e seus respectivos parâmetros encontrados pelo AE podem ser vistos na Tabela III.6.

Tabela III.4: Resultados de rodadas da otimização

Posição	130 nm	180 nm
1	$1,142 \times 10^{-10}$	$3,279 \times 10^{-13}$
2	$1,149 \times 10^{-10}$	$3,347 \times 10^{-13}$
3	$1,159 \times 10^{-10}$	$3,930 \times 10^{-13}$
4	$1,207 \times 10^{-10}$	$4,485 \times 10^{-13}$
5	$1,409 \times 10^{-10}$	$4,945 \times 10^{-13}$
30	$3,065 \times 10^{-9}$	$1,595 \times 10^{-11}$

Tabela III.5: Valores obtidos através das simulações

Parâmetro	130 nm	180 nm
V_{LR} (mV)	20	22
V_{do} (mV)	51	70
ΔV_I (mV)	161	69
C_{tot} (pF)	5,01	5,06
Margem de Fase (graus)	145	58

Tabela III.6: Valores encontrados pelo AE

Parâmetros	130 nm	180 nm
Largura do transistor Mp1 (μm)	14,25	45,01
Largura do transistor Mp2 (μm)	1,17	0,31
Largura do transistor Mp3 (μm)	0,63	0,30
Largura do transistor Mn1 (μm)	0,83	0,40
Largura do transistor Mn2 (μm)	3,27	0,54
Comprimento do transistor Mp1 (μm)	0,60	0,18
Comprimento do transistor Mp2 (μm)	1,43	3,81
Comprimento do transistor Mp3 (μm)	0,68	4,62
Comprimento do transistor Mn1 (μm)	1,70	1,65
Comprimento do transistor Mn2 (μm)	0,39	0,18
I_B (nA)	5	50
C_m (fF)	131	156
V_{SET} (V)	0,80	0,39
V_B (V)	1,52	1,32

A Figura III.1 mostra o resultado das simulações para regulação de tensão dos circuitos obtidos com o algoritmo. Como pode ser observado, nas duas tecnologias CMOS podemos obter um correto

funcionamento próximo de 1 V de tensão de saída depois que o circuito começa a regular com uma tensão de entrada de aproximadamente 1,1 V para ambas as tecnologias, no entanto, devido à falta de um circuito de realimentação para controlar esta tensão, percebe-se uma pequena variação da tensão de saída em relação à tensão de entrada.

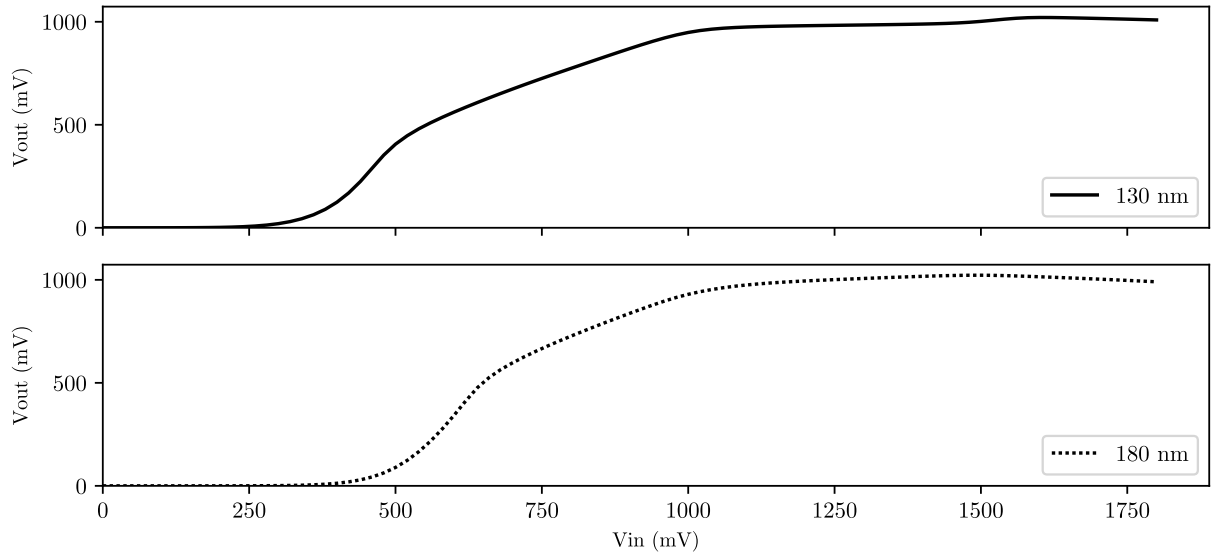


Figura III.1: Tensão de saída em função da tensão de entrada (Teste A).

Na Figura III.2 é mostrada a tensão de saída em função da corrente de carga. É importante observar que, também, devido à falta de um circuito de realimentação, observa-se uma pequena variação da tensão de saída para a tecnologia de 130 nm. Esta tecnologia, também, entrega 1 μA de corrente máxima de saída, o que é 50 vezes maior que a corrente consumida.

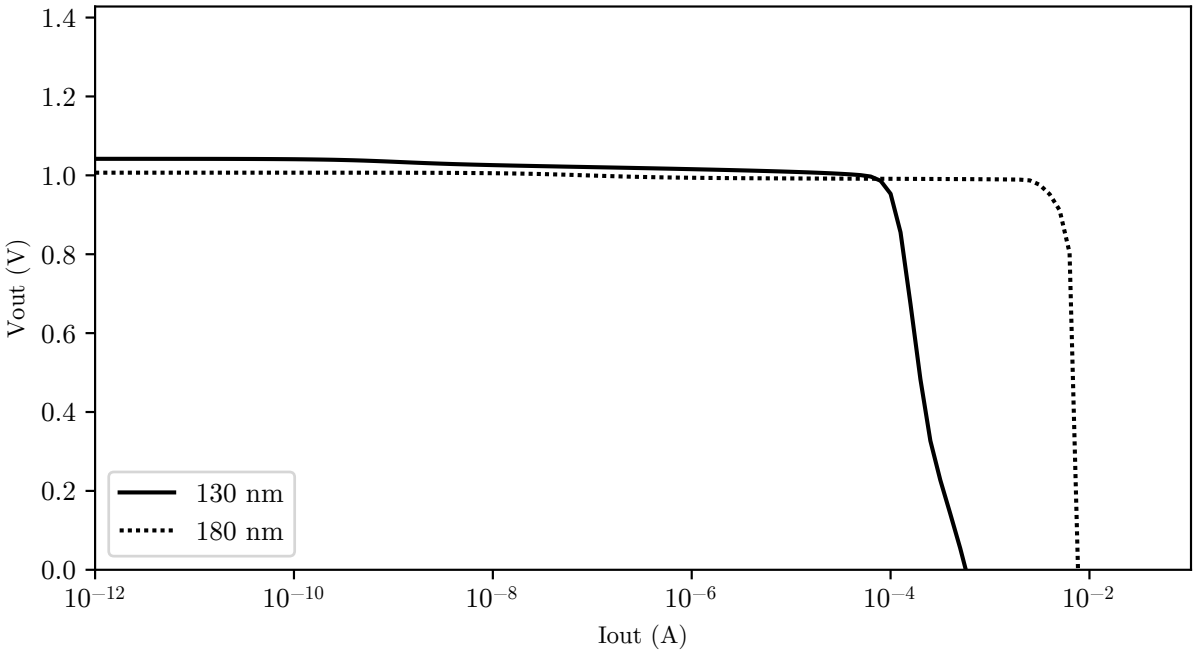


Figura III.2: Tensão de saída em função da corrente de carga (Teste B).

Como pode ser visto na Figura III.3 a tensão de saída no tempo em função de um degrau de tensão de entrada. Com isso observa-se que os circuitos conseguiram retornar a tensão nominal de saída pouco tempo depois da perturbação. As partes com variação de tensão durante a subida e descida da tensão de entrada são mostradas de modo mais detalhado nas Figuras III.4 (a) e (b), respectivamente.

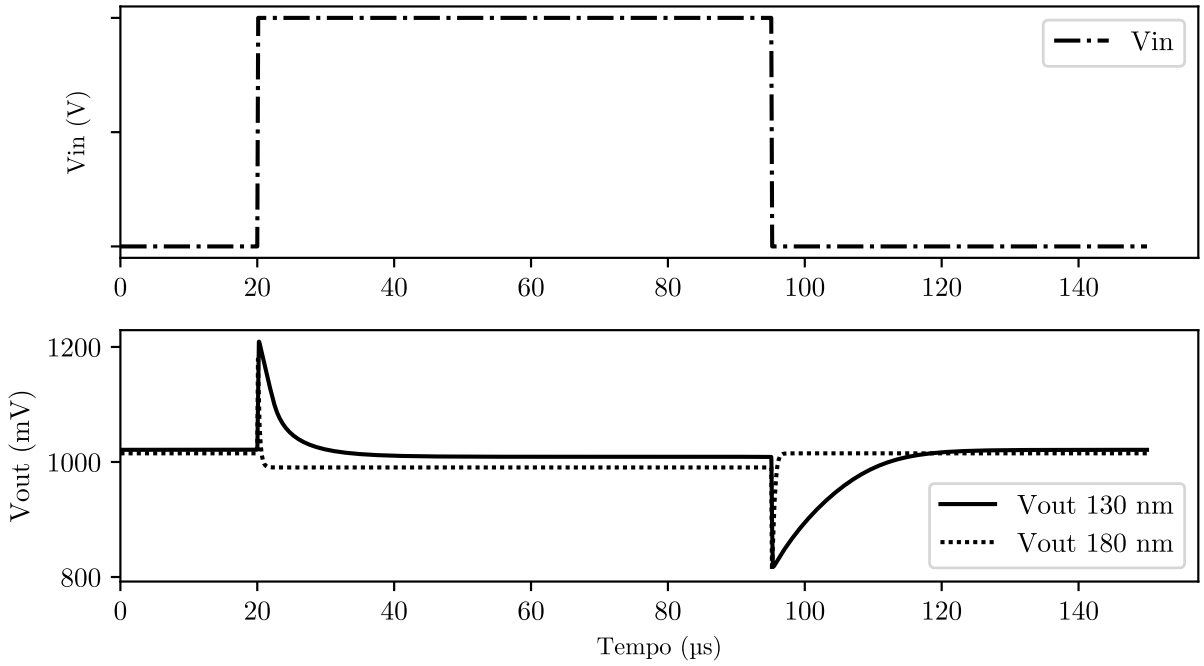


Figura III.3: Resposta dinâmica à variação da tensão de entrada (Teste C).

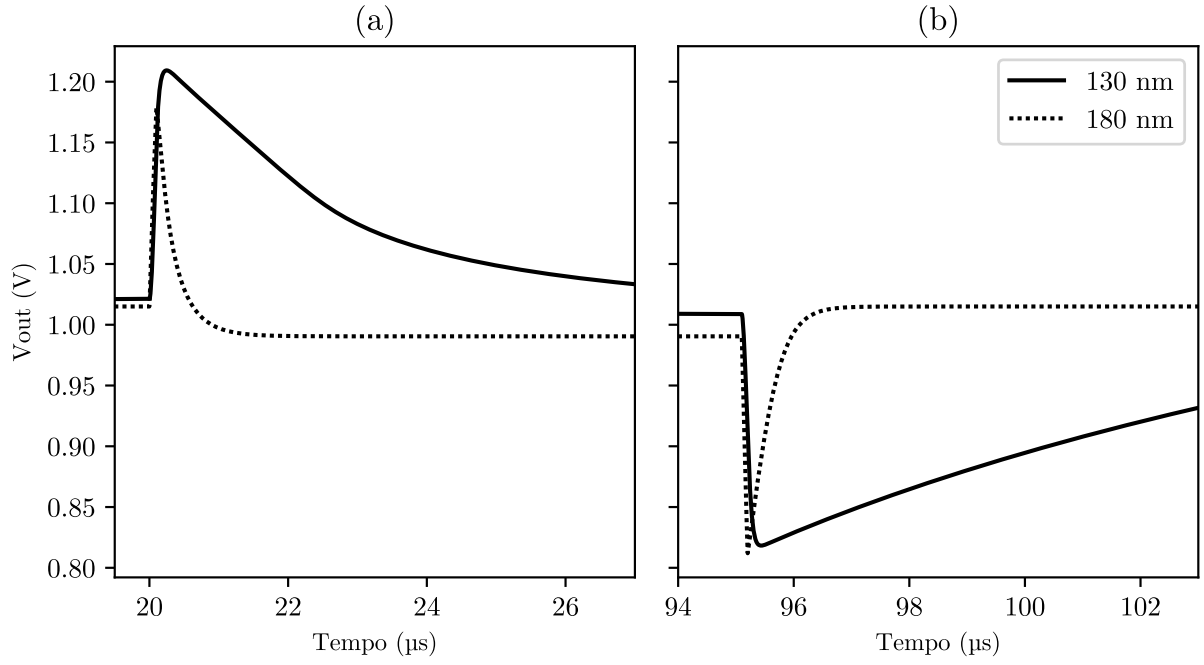


Figura III.4: Vista detalhada da saída de tensão do Teste C.

Na Figura III.5 é mostrada a tensão de saída no tempo em função de um degrau de corrente de saída extraída por uma carga. Observa-se que a tensão de saída não teve resposta rápida, especialmente para a tecnologia CMOS de 130 nm, demorando aproximadamente 330 μs para retornar ao valor inicial, com pico de aproximadamente 1,25 V. O gráfico é mostrado de forma mais detalhada nas partes em que há variação de subida de corrente de saída na Figura III.4 (a) e descida em (b).

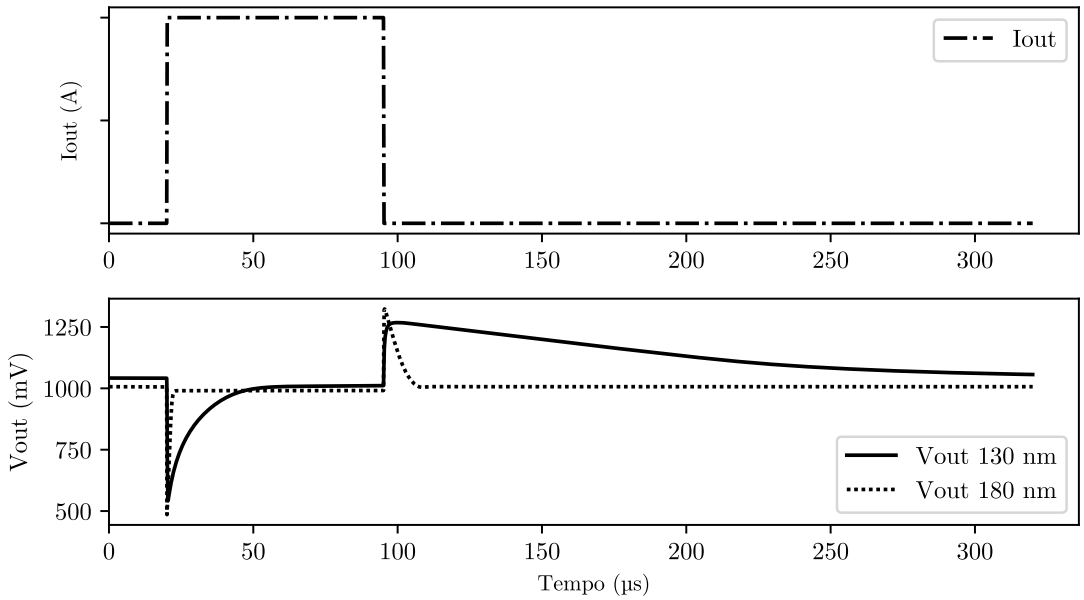


Figura III.5: Resposta dinâmica à variação de carga (Teste D).

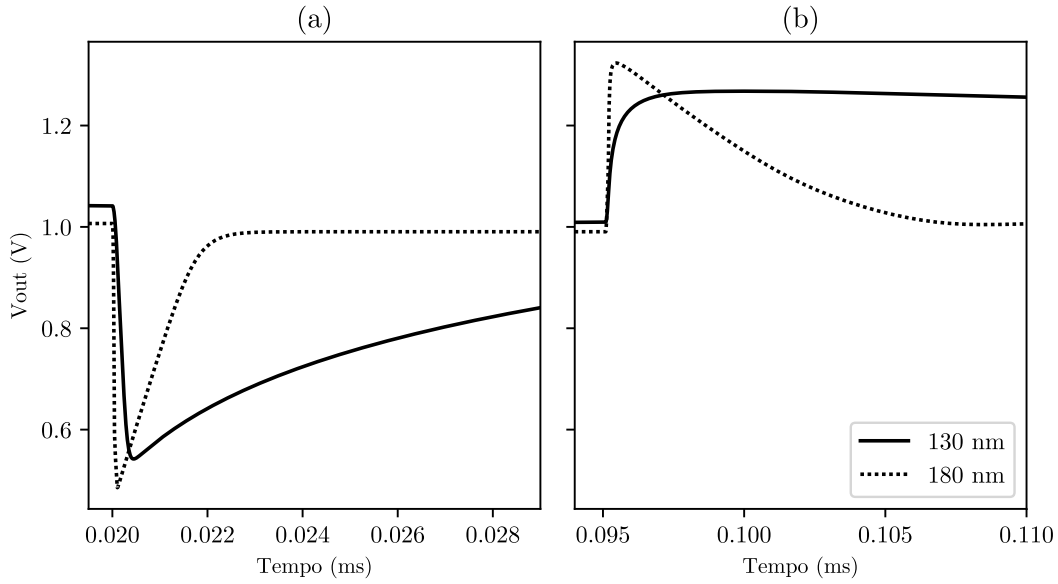


Figura III.6: Vista detalhada da saída de tensão do Teste D.

Na Figura III.7 é mostrada a amplitude e a margem de fase do Teste E para todas as tecnologias CMOS, com a resposta do circuito mostrada na Fig. II.10.

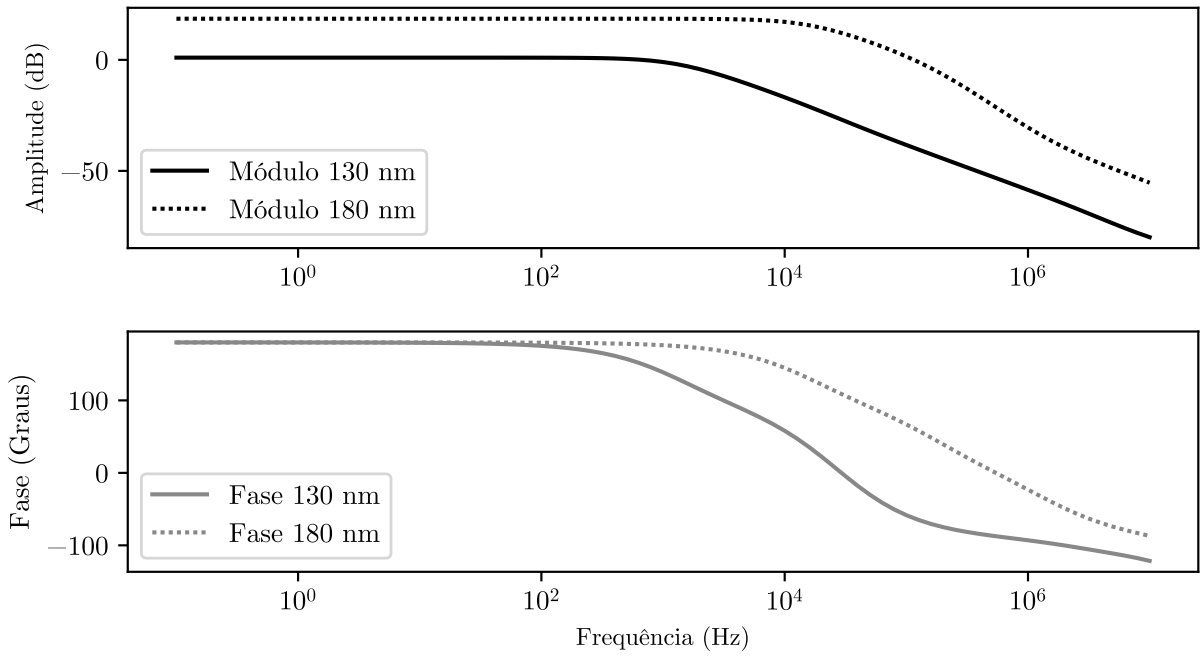


Figura III.7: Resposta em frequência do circuito em malha aberta (Teste E).

III.3 Custo Computacional

A otimização foi executada 10 vezes em três máquinas diferentes, com os detalhes de cada uma mostradas na Tabela III.7. O limite máximo de gerações (Gen_{tol}) foi atingido para todas as rodadas, com o tempo médio, tempo total e desvio padrão das rodadas sendo mostrados na Tabela III.8, com os processadores das respectivas máquinas sendo lançados no terceiro trimestre de 2022, quarto trimestre de 2023 e segundo trimestre de 2020, respectivamente, para as máquinas 1, 2 e 3. As simulações foram rodadas exclusivamente no CPU utilizando paralelismo das simulações Monte Carlo. O gráfico ilustrando o progresso do algoritmo ao longo da otimização pode ser visto na Figura III.8, em que as linhas em cinza claro representam todas as rodadas exceto pela rodada do melhor indivíduo, mostrada em cinza escuro, e em preto a média da FoMc ao longo da otimização.

Tabela III.7: Descrição das máquinas utilizados

Nº	Tipo de uso	Processador	*Threads*	RAM	Sistema Operacional
1	Estação de trabalho	AMD Ryzen 9 7950X	32	64 GB	Ubuntu 22.04.5
2	Servidor	Intel Core i7 14700K	28	64 GB	Ubuntu Server 24.04.3
3	Estação de trabalho	Intel Core i7 10700F	16	128 GB	Rocky Linux 8.8

Tabela III.8: Tempo de otimização para 10 rodadas em cada máquina

Tecnologia	130 nm			180 nm		
Nº Máquina	1	2	3	1	2	3
Média (minutos)	77	132	527	14	19	56
Tempo total (minutos)	777	1322	5226	143	192	562
Desvio padrão (segundos)	39	121	241	31	73	280

Observa-se na Tabela III.8 que quanto mais *threads* disponíveis há nas máquinas, mais rápido é a execução do algoritmo, O Servidor 2, com o processador mais novo e com arquitetura híbrida de 8 núcleos de desempenho, cada um com 2 *threads*, e 12 núcleos de eficiência, cada um com apenas 1 *thread*, fica com o segundo menor tempo de execução. Ficando atrás da Estação de Trabalho (ET) 1, que possui 16 núcleos e 32 *threads*, mas é um ano mais velho. Devido às melhorias descritas na Subseção II.4.2, caso fosse possível executar a otimização em uma máquina com mais *threads*, as rodadas poderiam ser executadas com mais testes Monte Carlo, sem aumentar muito o tempo de otimização, mas, em contrapartida, estes processadores apresentam custo muito elevado. Com isso, o fluxo proposto consegue atingir um resultado relativamente bom em algumas horas de otimização em uma máquina equipada com processadores mais acessíveis.

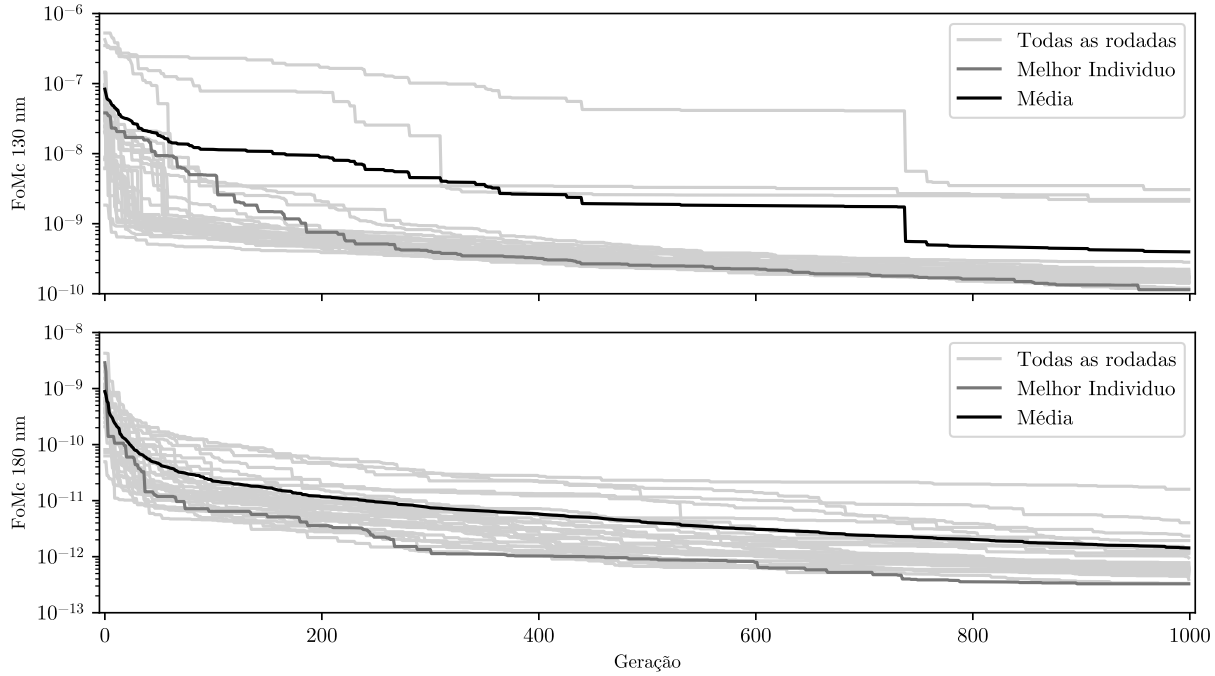


Figura III.8: Progresso do algoritmo durante a otimização.

III.4 Comparação dos Resultados em Relação à Literatura

A FoM foi recalculada utilizando a expressão mostrada na Equação (II.5), que leva apenas em consideração os Testes D e F. Como pode ser apreciado na Tabela III.9, dos projetos de circuito LDO obtidos com a ferramenta desenvolvida neste trabalho, apenas a tecnologia CMOS de 180 nm consegue alcançar uma FoM de ordem semelhante aos relatados na literatura, ainda que maior. Isto deve-se ao foco em baixa corrente de consumo e ausência de um circuito de controle da tensão de saída. O que impacta diretamente ΔV_I, que tem seu valor ao quadrado no cálculo da FoM. Além disso, o circuito não possui técnicas associadas ao circuito LDO puro, como corrente de polarização dinâmica, dois loops de controle e circuito de assistência push-pull, como relatados, respectivamente, em [4, 23, 24]. A corrente máxima de saída também é menor devido ao circuito não possuir nenhuma outra técnica de estabilização; no entanto, devido às aplicações de ultra baixo consumo não consumirem excessiva corrente, acaba não sendo uma desvantagem do projeto e, devido a isso, o circuito produzido é mais simples do que os da literatura e com um consumo menor no caso do processo CMOS de 130 nm.

Tabela III.9: Comparação dos resultados obtidos com a literatura

	Este trabalho*	[4]	[23]	[24]
Tecnologia (nm)	130 (180)	65	350	350
V_{OUT} (V)	1 (1)	0,8	1,2-4,8	1,0
I_Q (nA)	5 (50)	30	90	1200
I_{MAX} (mA)	0,001 (3,16)	10	20	100
C_{OUT} (pF)	5,01 (5,06)	10	1000	100
FoM (fs)	182364 (849)	1,59	191	3,25
ΔV_{OUT} (mV)	728 (839)	487	850	270

* Obtido através de simulações.

Capítulo IV Simulações de Protótipo Enviado para Fabricação

O circuito otimizado no processo CMOS de 130 nm da Skywater foi enviado para fabricação através do programa *Universalization of IC Design from CASS* [1] da CASS/IEEE, e por isso este processo CMOS foi mais desenvolvido em relação ao de 180 nm. O projeto, como um todo, foi realizado utilizando um fluxo de desenvolvimento analógico com algumas ferramentas do fluxo digital OpenLane [25], que reúne ferramentas e automações de código aberto para o desenvolvimento de CIs. Serão abordados neste capítulo os resultados da otimização do circuito que foi enviado para fabricação, simulações pós-leiaute e os procedimentos realizados para preparar testes experimentais do circuito fabricado.

IV.1 Resultados da Otimização

Devido ao prazo estabelecido para envio do circuito à fabricação, não houve tempo hábil para que fosse verificado todo o algoritmo, e, antes da realização das 30 rodadas mencionadas na seção III.2, posteriormente à fabricação, houveram algumas alterações na implementação para melhora de desempenho. Dentre estas alterações, foram realizadas algumas melhorias e correções na mutação dos indivíduos, na rotina que avalia a margem de fase, como também, a forma que alguns testes eram computados para avaliar a função custo. Os resultados das otimizações mostrados na Seção III.2 já contêm as novas alterações, mas, os resultados do circuito enviado para fabricação, o qual foi otimizado com código antigo, são mostrados ao longo desta seção.

Os parâmetros encontrados pelo AG antes das alterações podem ser vistos na Tabela IV.1; percebe-se que este circuito utiliza uma área maior, bem como uma corrente de polarização aproximadamente duas ordens de grandeza maior comparado com o circuito otimizado pelo algoritmo com as novas alterações. Os resultados do teste deste circuito são mostrados na Tabela IV.2 na seção IV.2, na coluna TT, e juntamente com os gráficos, também da resposta TT.

A Figura IV.3, curva TT, mostra o resultado da simulação da regulação de tensão do circuito, em que o circuito começa a regular com uma tensão de entrada de aproximadamente 1,4 V. Na Figura IV.4-TT é mostrada a tensão de saída em função da corrente de carga, com o circuito regulando satisfatoriamente até uma corrente de carga máxima de aproximadamente 500 μA, maior que a corrente encontrada com a otimização corrigida. A Figura IV.5-TT mostra o resultado da

Tabela IV.1: Valores dos intervalos encontrados pelo AG que foi enviado para fabricação

Parâmetros	130 nm	Comparação com a Tabela III.6 (%)
Largura do transistor Mp1 (μm)	25,83	-81,3
Largura do transistor Mp2 (μm)	12,13	-936,8
Largura do transistor Mp3 (μm)	1,78	-182,5
Largura do transistor Mn1 (μm)	10,38	-1150,6
Largura do transistor Mn2 (μm)	10,19	-211,6
Comprimento do transistor Mp1 (μm)	0,37	38,3
Comprimento do transistor Mp2(μm)	1,27	11,2
Comprimento do transistor Mp3 (μm)	3,88	-470,6
Comprimento do transistor Mn1 (μm)	1,42	16,5
Comprimento do transistor Mn2 (μm)	0,65	-66,7
I_B (nA)	465	-9200
C_m (fF)	127	3,1
V_{SET} (V)	0,49	38,8
V_B (V)	1,54	-1,3

simulação para o Teste C. Com isso, observa-se que o circuito possui resposta mais rápida do que o encontrado pela otimização corrigida, na Figura III.3. A vista detalhada é mostrada nas Figuras IV.6 (a)-TT e (b)-TT, respectivamente. A Figura IV.7 mostra o resultado da simulação para o Teste D. Quando comparado ao gráfico da Figura III.5, da otimização corrigida, observa-se que a tensão de saída também teve resposta rápida, assim como no Teste C, voltando à tensão nominal rapidamente. O detalhamento dos transientes é mostrado na Figura IV.8 (a)-TT para subida de corrente e descida em (b)-TT. E, por fim, na Figura IV.9-TT é mostrada a amplitude e a margem de fase do Teste E.

A otimização que resultou no circuito enviado para fabricação foi executada em 50 minutos na ET 1, equipada com o processador AMD Ryzen 9 7950X, O gráfico ilustrando o progresso do algoritmo ao longo da otimização pode ser visto na Figura IV.1.

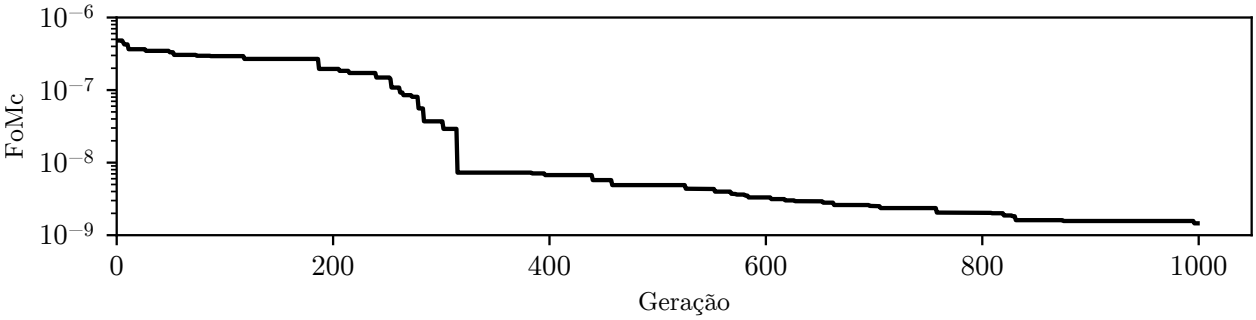


Figura IV.1: Progresso do algoritmo durante a otimização anterior às modificações.

IV.2 Simulações a Nível de Esquemático

Antes de prosseguir com a implementação do leiaute, é preciso saber o comportamento do circuito em relação às variações de fabricação para evitar trabalho desnecessário em implementar o leiaute de um circuito que não possui um bom funcionamento. Para isso, o circuito otimizado foi simulado novamente nos testes de regulação linear de tensão, teste transiente de corrente e teste de margem de fase com variação Monte Carlo em 50 rodadas, que resultaram nos histogramas vistos na Figura IV.2, respectivamente em, (a), (b) e (c) e apresentam variações comparadas com os valores da Tabela IV.2 de 39%, 74% e 94% respectivamente.

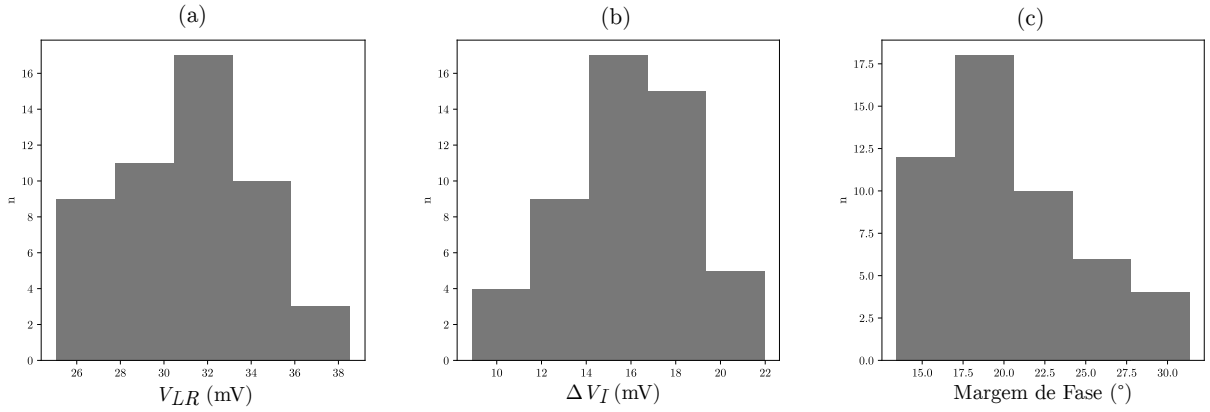


Figura IV.2: Histograma para 50 rodadas de Monte Carlo.

Além das simulações Monte Carlo, simulações nos *corners*[1] de fabricação também são igualmente importantes e são mostradas na Tabela IV.2, em que o circuito se mostrou instável apenas no *corner* SF, com o valor da margem de fase em negrito. O *corner* FS entregou corrente em um valor menor que a corrente que o circuito consome e, por isso, também foi considerado como não funcional, e também é mostrado em vermelho.

Tabela IV.2: Resultado de simulação para os *corners* de fabricação

Parâmetro	*Corners*				
	TT	FF	FS	SF	SS
V_{LR} (mV)	30,4	59,4	59,3	80,4	41
V_{do} (mV)	51,4	130	157	93,2	47,8
Corrente Máxima (µA)	501	630	**2**	794	1000
Transiente de tensão (mV)	15,3	23,1	24,3	92,3	46
ΔV_I (mV)	16,3	31,5	19,8	91,8	45,2
C_{TOT} (pF)	5,041	5,04	5,037	5,045	5,044
Margem de Fase (º)	16	66	95	4	3

[1] *Corner*: termo utilizado para se referir aos piores casos das variações durante a fabricação que afetam o *wafer* da mesma forma.

Na Figura IV.3 observa-se que houve uma variação entre os *corners* na região em que os circuitos começam a regular, e também uma leve diferença entre a tensão de saída, mas não muito significativa.

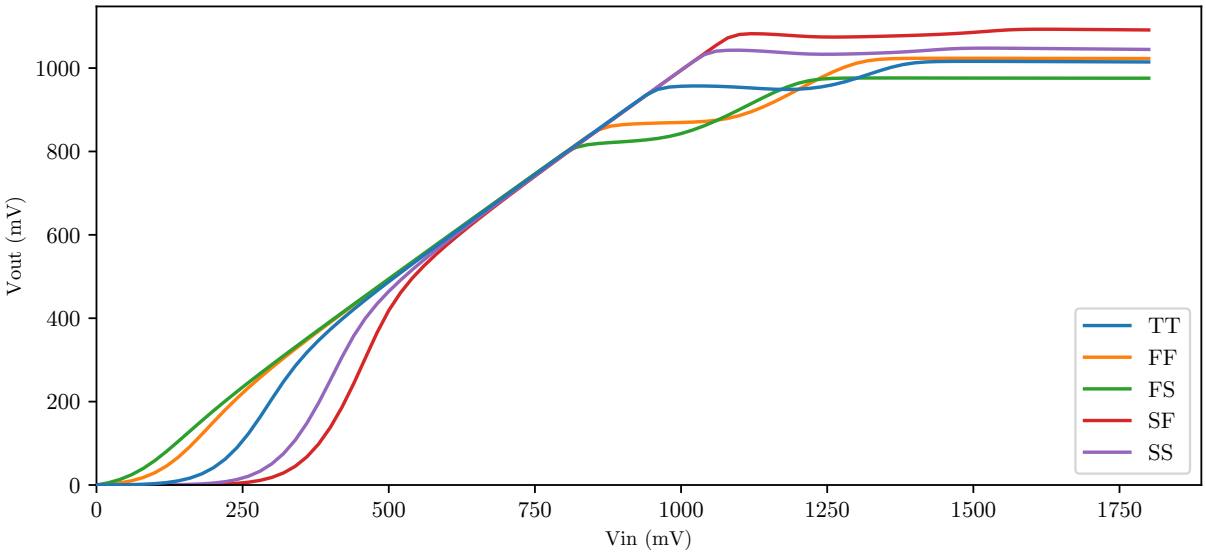


Figura IV.3: Teste A em *corners* a nível de esquemático.

Na Figura IV.4 observa-se que, exceto pelo *corner* FS, não houve variação na ordem de grandeza da corrente entre os *corners*, com os circuitos entregando acima de 500 μA de corrente máxima; no entanto, observa-se que, caso fosse permitida uma variação maior da tensão de saída, não haveria mudança significativa na corrente entregue.

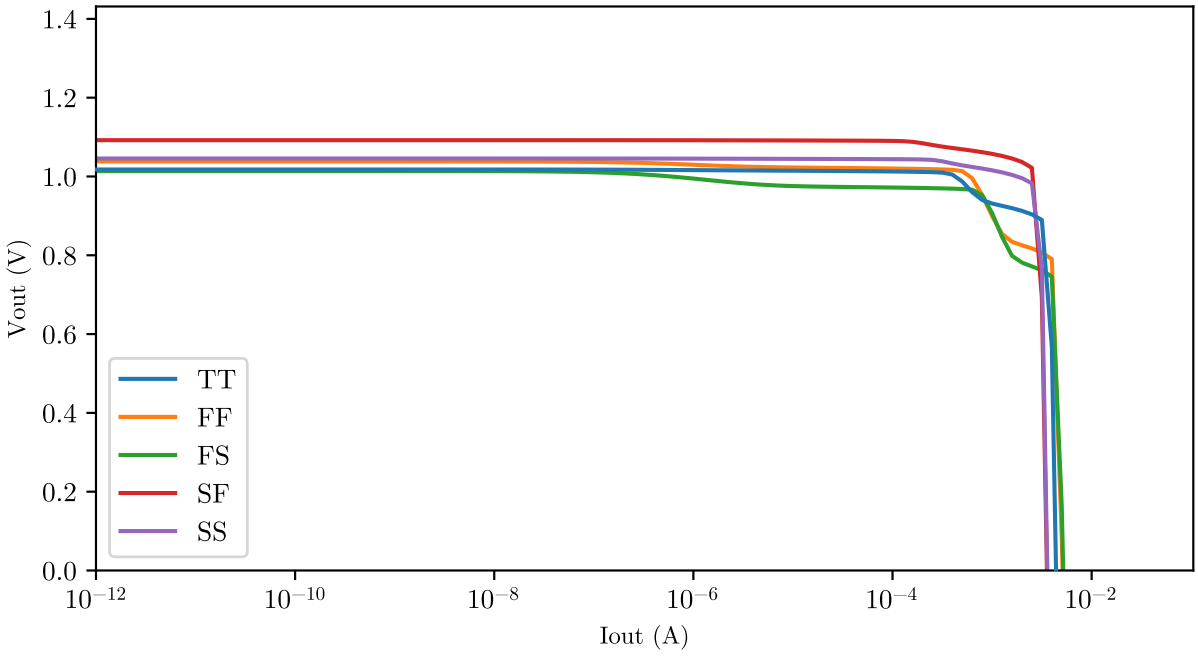


Figura IV.4: Teste B em *corners* a nível de esquemático.

Para o teste de variação de tensão de entrada observa-se na Figura IV.13 que, além da variação da tensão de regulação, não houve diferença significativa na resposta transiente, como detalhada na Figura IV.6.

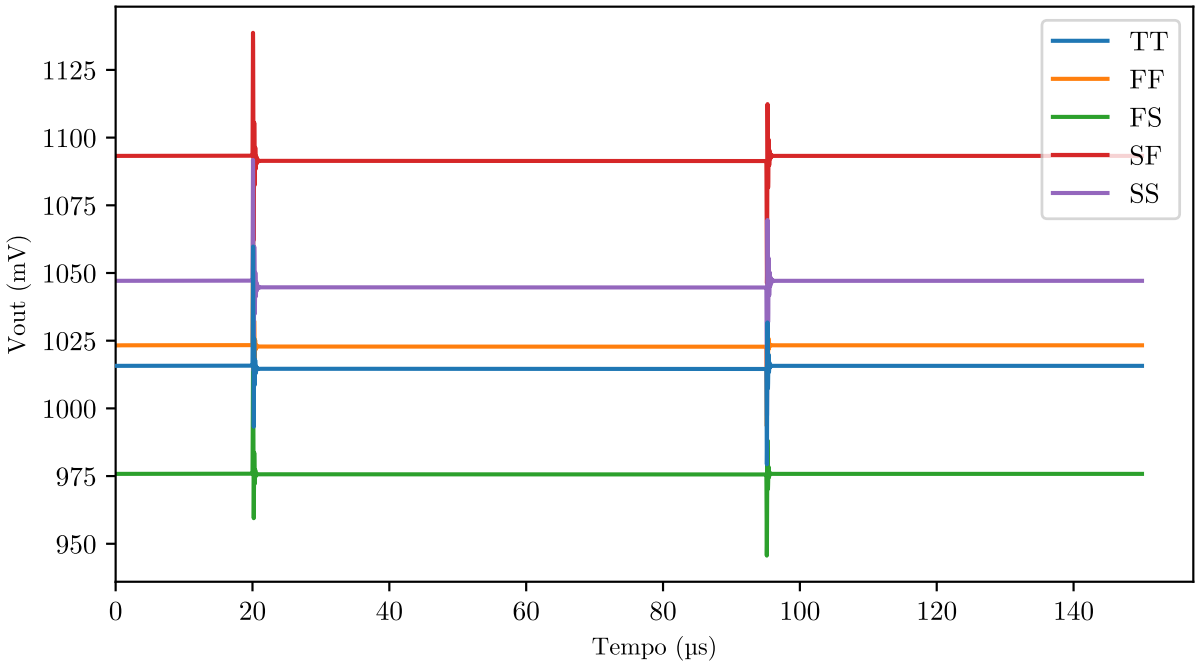


Figura IV.5: Teste C em *corners* a nível de esquemático.

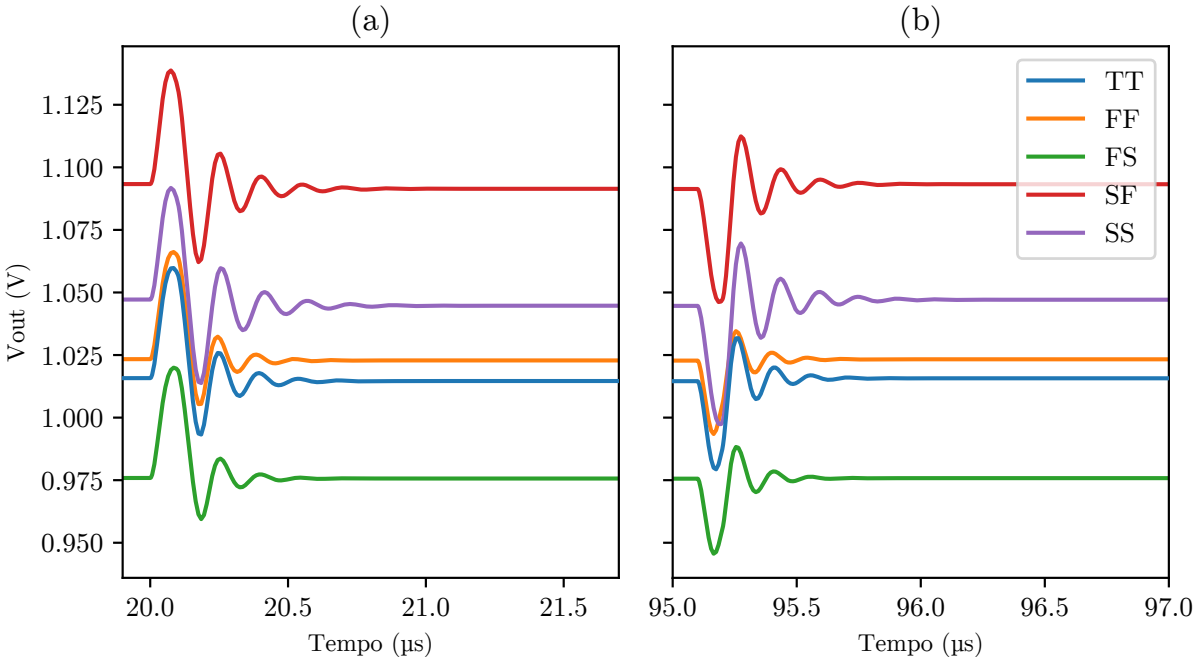


Figura IV.6: Vista detalhada da saída de tensão em *corners* do Teste C a nível de esquemático.

Para o teste de variação de corrente de entrada na Figura IV.15 observa-se nos *corners* SF e SS uma oscilação de maior período em relação aos outros *corners* no momento em que a corrente volta ao patamar inicial, que é vista na Figura II.8, e com mais detalhes na Figura IV.8.

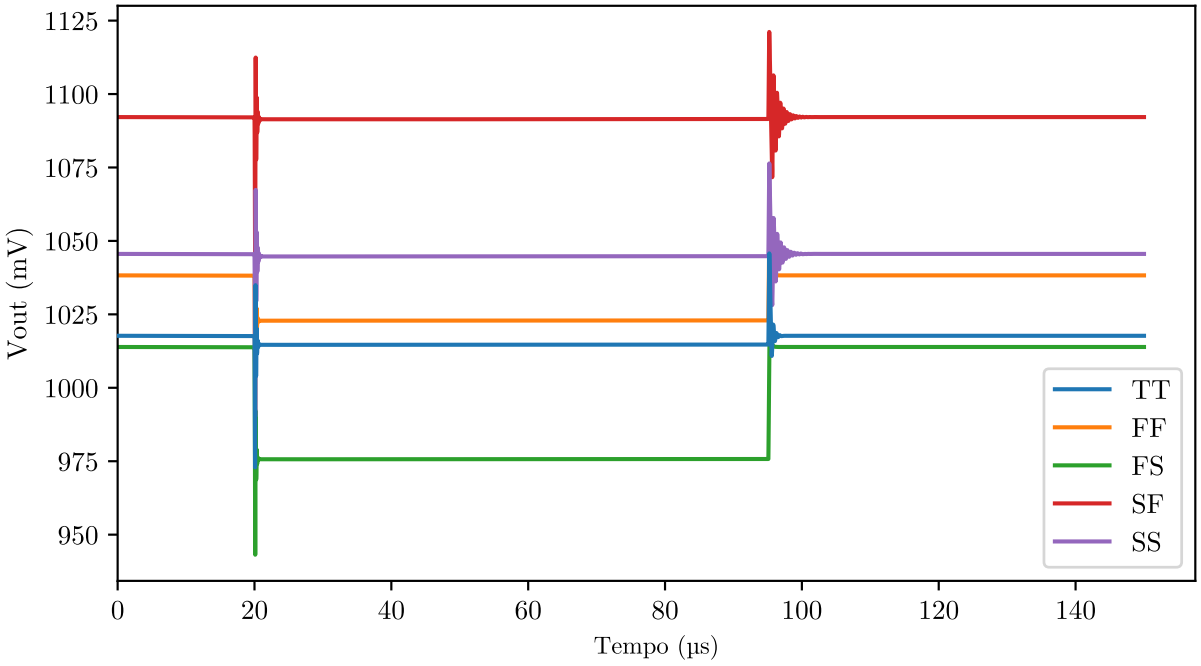


Figura IV.7: Teste D em *corners* a nível de esquemático.

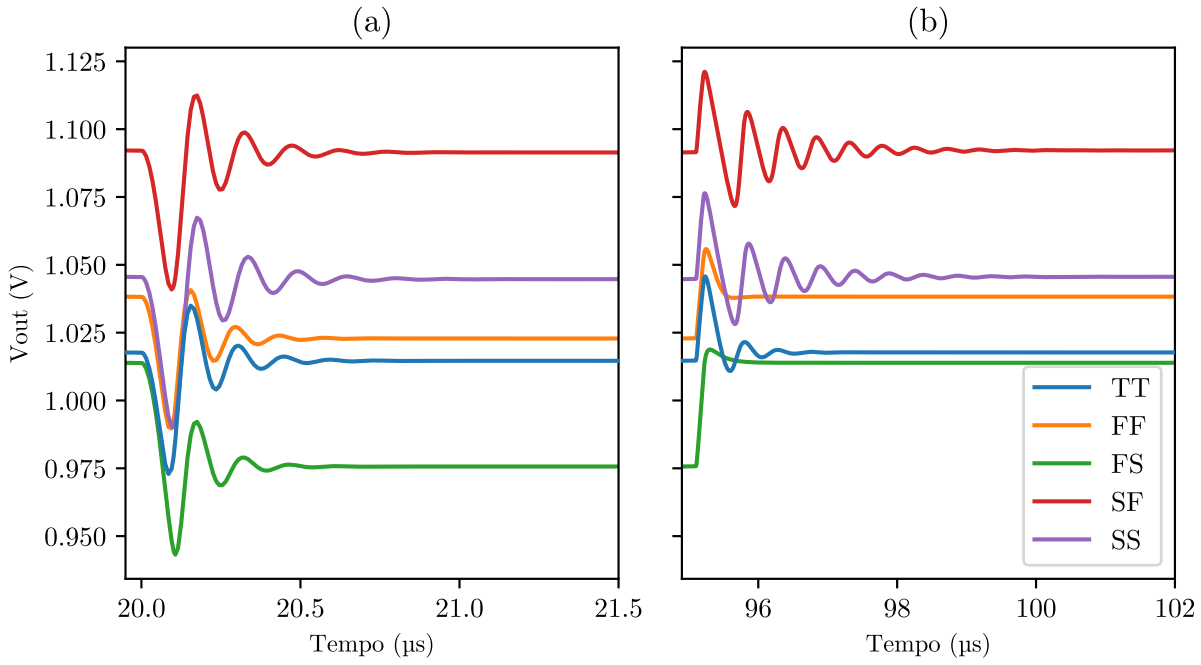


Figura IV.8: Vista detalhada da saída de tensão do Teste D em *corners* a nível de esquemático.

Para o teste de margem de fase mostrado na Figura IV.9 é possível ver uma diferença no ganho em baixas frequências e uma variação na fase em altas frequências.

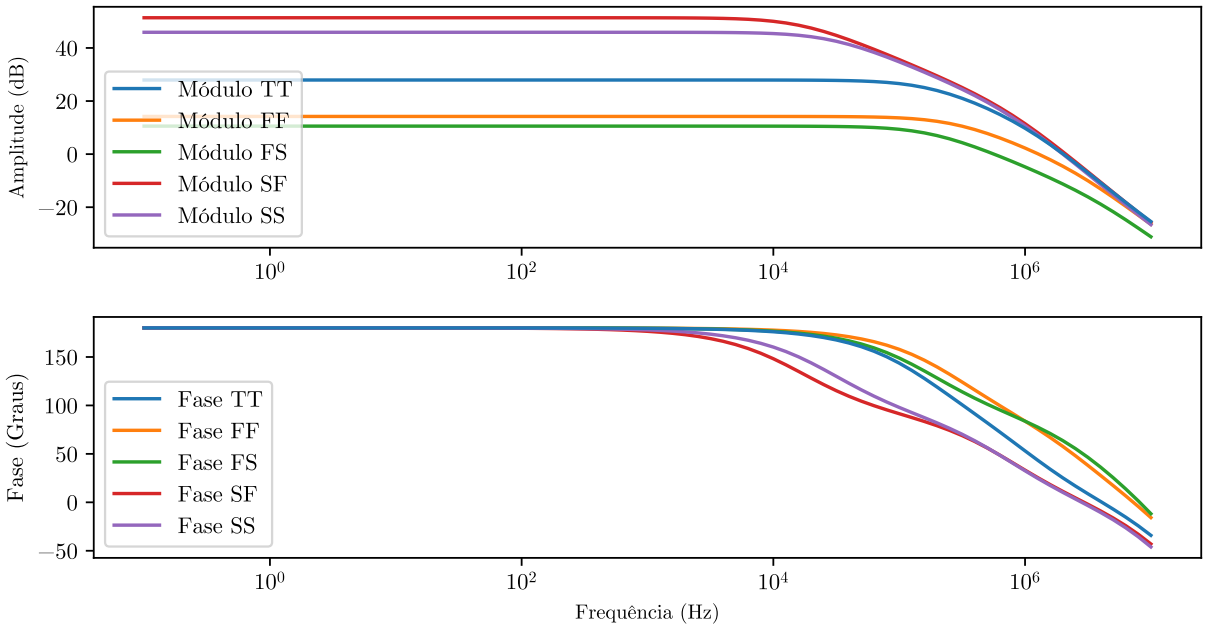


Figura IV.9: Teste E em *corners* a nível de esquemático.

IV.3 Projeto de Leiaute

Com o objetivo de enviar o circuito para fabricação, é importante que seu comportamento seja bem conhecido, o que simulações apenas com esquemáticos não garantem, já que não levam em consideração a disposição física e construção dos transistores, trilhas, vias e demais componentes passivos. Através do leiaute é possível implementar o circuito da forma com que ele seria fabricado fisicamente e, obter tanto uma *netlist* que leve em conta parâmetros parasitas que não foram considerados pelo esquemático, quanto arquivos que serão enviados para a fábrica, chamados de GDS (*Graphic Design System*).

O leiaute foi implementado utilizando o programa de código aberto *Magic*, disponível em [26], e este pode ser visto na Fig IV.10, que possui dimensões de 33 μm x 22 μm, ocupando uma área total de $7,26 \times 10^{-4}$ mm^2. Com isso é possível realizar a extração de parâmetros do circuito, como capacitâncias e resistências parasitas, em uma nova *netlist* representando o novo circuito. Este novo circuito é utilizado em simulações análogas às descritas em II.2, mas desta vez apenas para avaliar o desempenho do circuito nessas simulações pós-leiaute. Devido a isso, não é possível implementar a otimização para esta etapa do projeto, já que possui um aumento no custo computacional nas simulações, além de aumentar muito a complexidade da programação, que precisaria levar em consideração alterações na estrutura física do *chip*. Com a simulação pós-leiaute, o circuito final tem a acurácia da simulação aumentada em relação ao circuito que será fabricado.

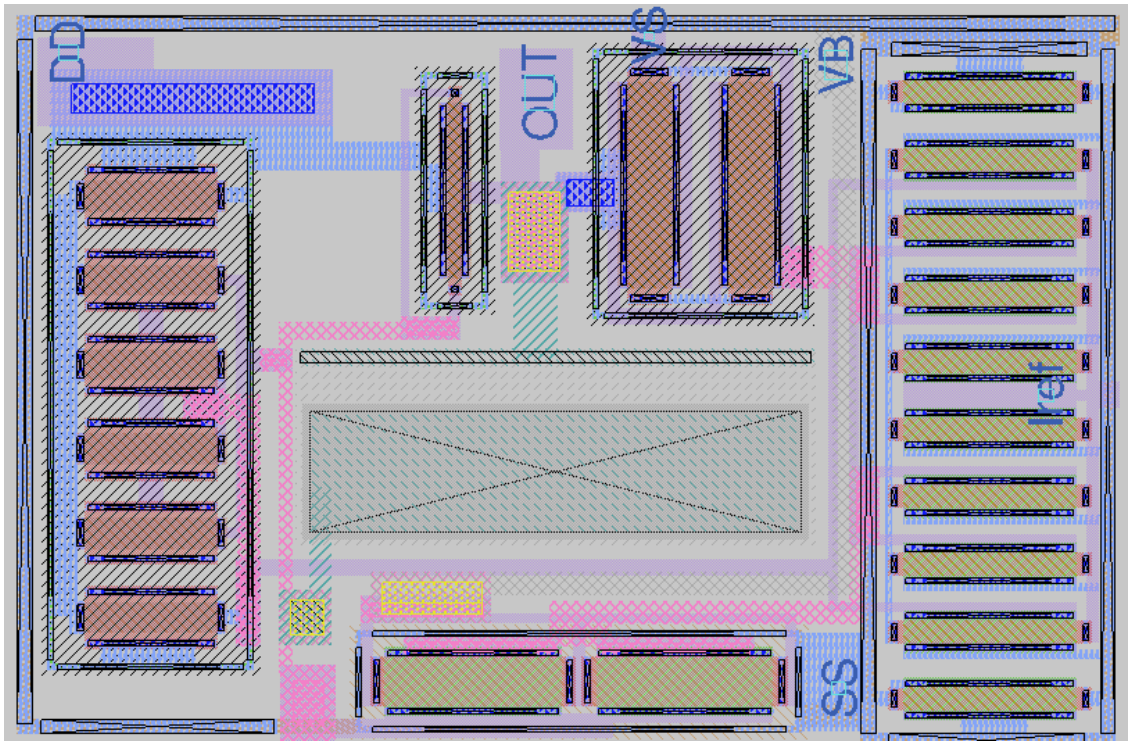


Figura IV.10: Implementação de leiaute do circuito otimizado com a ferramenta.

IV.4 Simulações a Nível de Leiaute

O circuito extraído foi simulado novamente em *corners* de fabricação, e o resultado é mostrado na Tabela IV.3 e pode-se observar uma redução da corrente de saída nos demais *corners* exceto pelo FS, que agora entregam um valor satisfatório quando comparados aos valores da Tabela IV.2. A tabela IV.4 mostra uma comparação percentual entre as Tabelas IV.2 e IV.3, em que percebe-se uma grande alteração nos valores de corrente máxima e transiente de corrente ΔV_I, com os demais valores, como a corrente e o transiente de corrente, não sofrendo grandes alterações, e, sobretudo, com o circuito ainda apresentando desempenho satisfatório.

Tabela IV.3: Resultado de simulação para os *corners* de fabricação pós-leiaute

Parâmetro	*Corners*				
	TT	FF	FS	SF	SS
V_{LR} (mV)	44,8	85,6	95	67,7	30,5
V_{do} (mV)	67,6	154	192	87,5	42
Corrente Máxima (µA)	100	125	158	125	158
Transiente de tensão (mV)	11	18	32	83	42
ΔV_I (mV)	138	123	128	148	155
C_{TOT} (pF)	5,12	5,1	5,08	10,03	5,65
Margem de Fase (^{o})	6	38	74	2	8

Tabela IV.4: Comparação de resultados pré-leiaute com pós-leiaute

Parâmetro	*Corners*				
	TT	FF	FS	SF	SS
Variação V_{LR} (%)	47,4	44,1	60,2	-15,8	-25,6
Variação V_{do}(%)	31,5	18,5	22,3	-6,1	-12,1
Variação Corrente Máxima (%)	-80	-80,2	7800	-84,3	-84,2
Variação Transiente de tensão (%)	-28,1	-22,1	31,7	-10,1	-8,7
Variação ΔV_I (%)	746,6	290,5	546,5	61,2	242,9
Variação C_{TOT} (%)	1,6	1,2	0,9	98,8	12
Variação Margem de Fase (%)	-62,5	-42,4	-22,1	-50	166,7

As variações devido aos *corners* ficam mais evidentes nos gráficos, onde pode-se perceber uma variação na tensão final de regulação. No *corner* TT percebe-se uma elevação na tensão de entrada para o circuito começar a regular, que acaba sendo maior que as demais; no entanto, caso seja considerado que o circuito começaria a regular se a tensão de saída for 90% da tensão de regulação desejada, 0,9 V, pode-se considerar que o circuito começa a regular quando a tensão de entrada é de aproximadamente 1 V, como pode ser visto na Figura IV.11

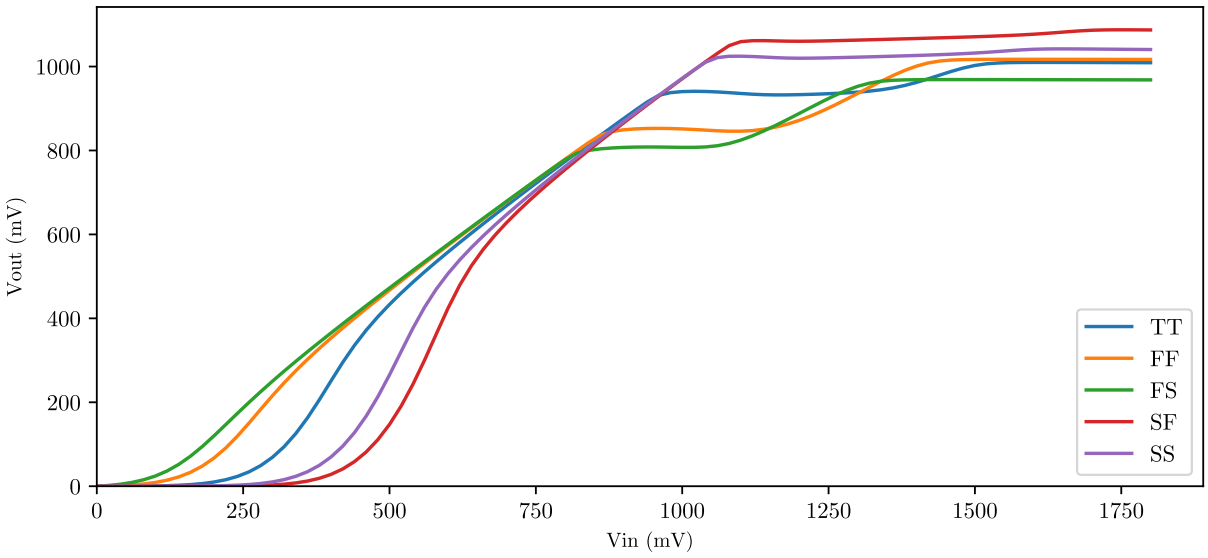


Figura IV.11: Tensão de saída em função da tensão de entrada em *corners* (Teste A).

Na Figura IV.12 observa-se que não houve uma variação significativa entre os *corners*, com os circuitos entregando próximo de 100 μA de corrente máxima.

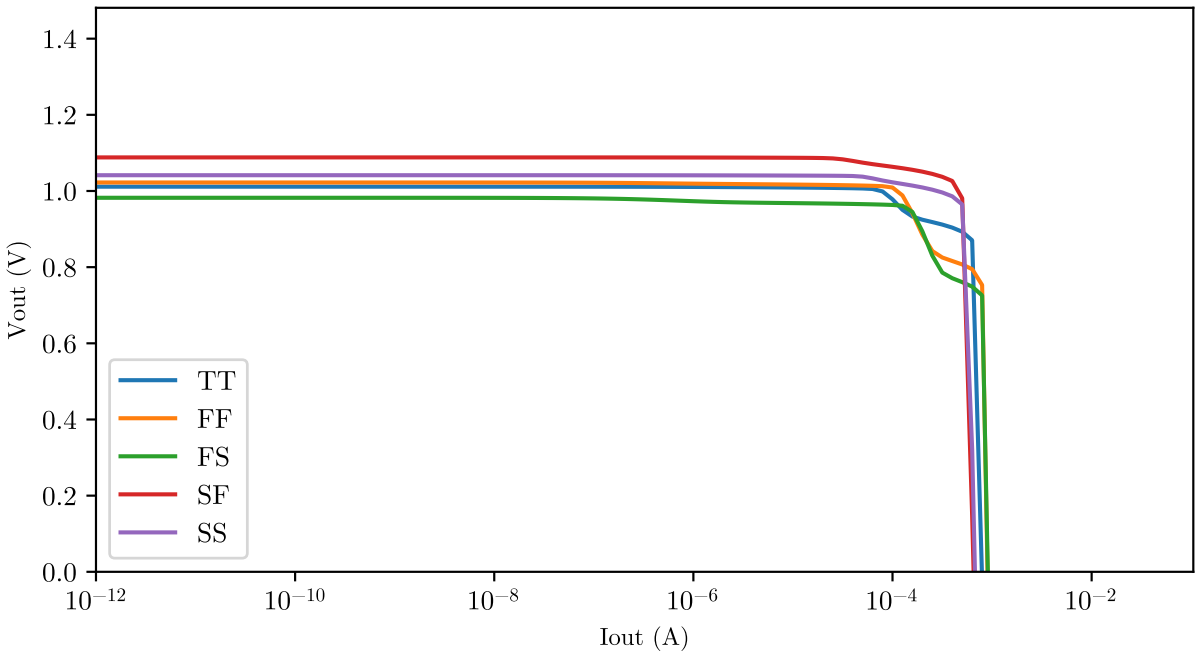


Figura IV.12: Tensão de saída em função da corrente de carga em *corners* (Teste B).

Para o teste de variação de tensão de entrada, observa-se na Figura IV.13 que, além da variação da tensão de regulação, não houve diferença significativa na resposta transiente, como detalhada na Figura IV.14.

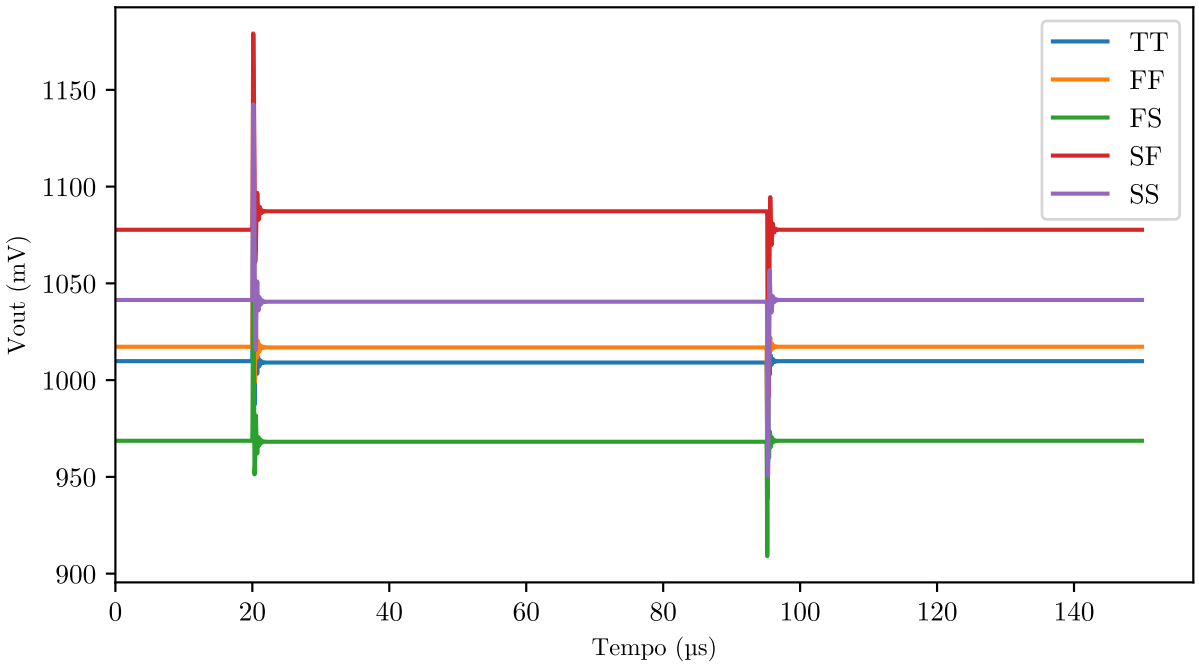


Figura IV.13: Resposta dinâmica à variação de tensão de entrada em *corners* (Teste C).

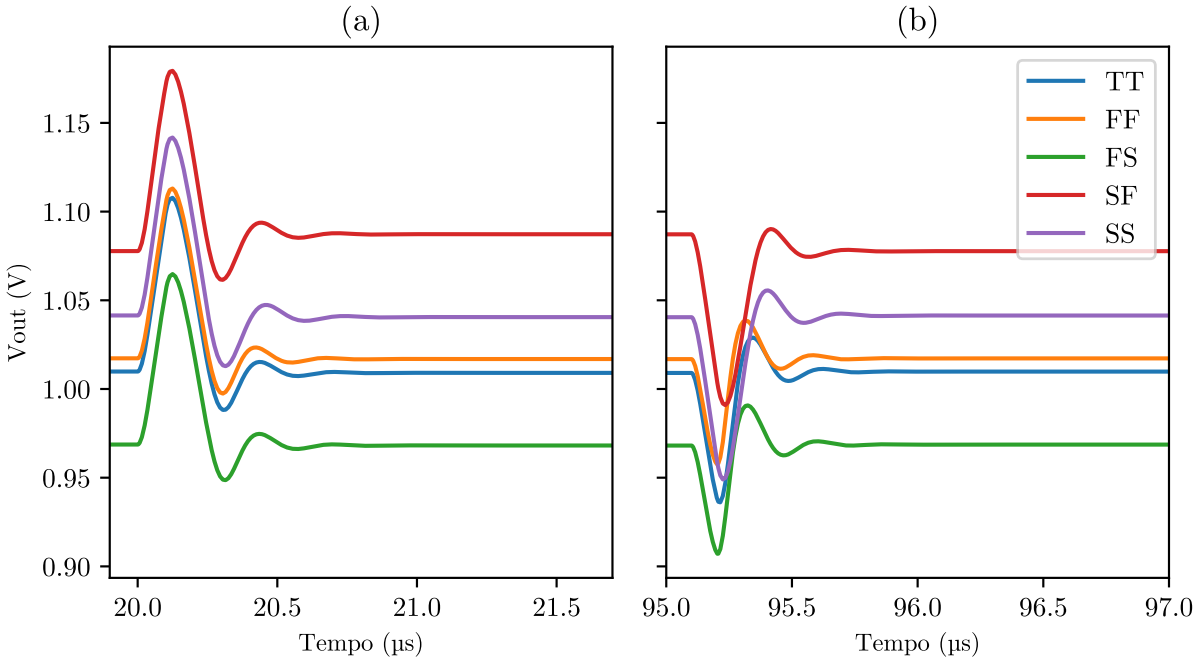


Figura IV.14: Vista detalhada da saída de tensão em *corners* (Teste C).

Para o teste de variação de corrente de entrada na Figura IV.15 observa-se no *corner* TT, SF e SS uma oscilação maior que a vista na simulação a nível de esquemático mostrada na Figura II.8, que é vista com mais detalhes na Figura IV.16.

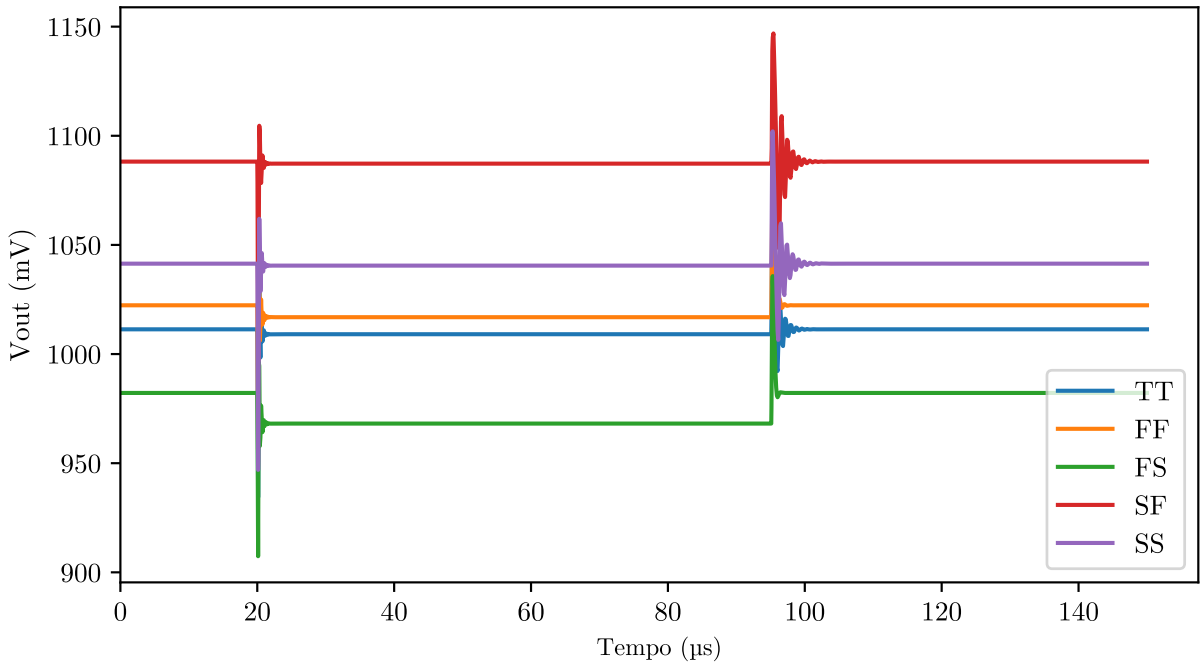


Figura IV.15: Resposta dinâmica à variação de carga em *corners* (Teste D).

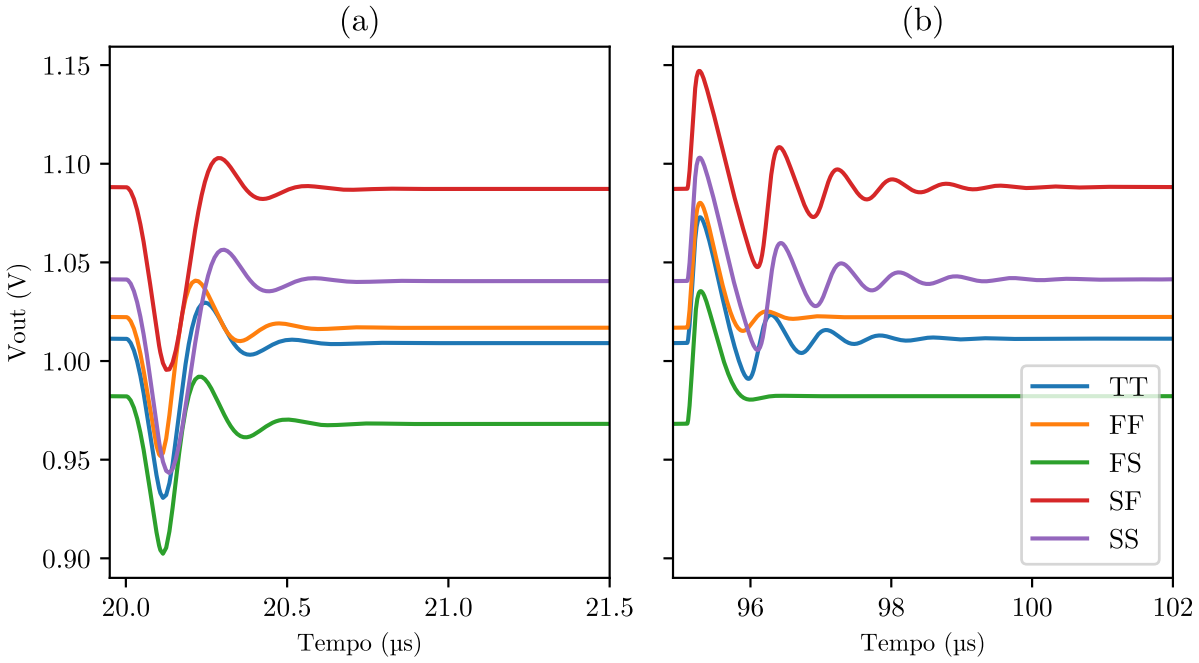


Figura IV.16: Vista detalhada da saída de tensão do Teste D em *corners*.

Para o teste de margem de fase mostrado na Figura IV.17 é possível ver uma diferença no ganho em frequências mais altas, o que explica a variação vista na Tabela IV.4.

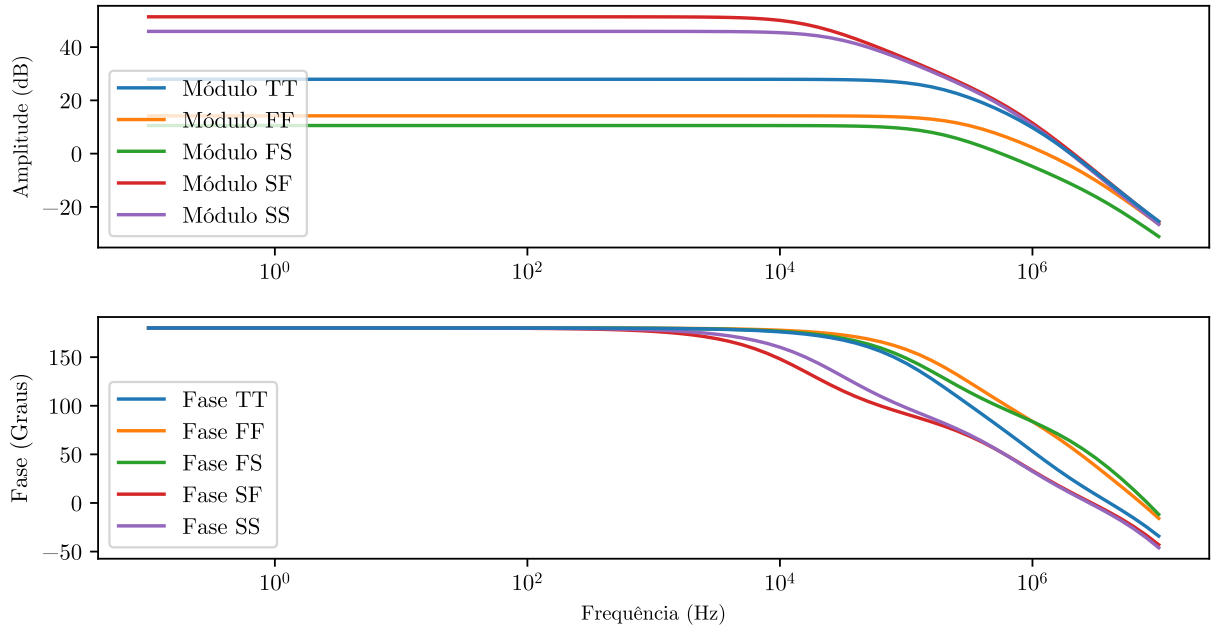


Figura IV.17: Teste E em *corners* a nível de leiaute.

Com os resultados destas simulações a nível de leiaute, é possível observar que o circuito apresenta desempenho satisfatório para ser enviado para fabricação.

IV.5 Envio à Fabricação do Circuito Integrado

A fabricação foi financiada pelo programa UNIC-CASS, que visa tornar mais acessível a tecnologia de *design* para comunidades em países de baixa/média renda [1], como o Brasil, em que o circuito desenvolvido neste trabalho foi submetido a avaliação do programa e aceito. Dentre os pré-requisitos listados estava a implementação do circuito na tecnologia CMOS da Skywater de 130 nm, uma *foundry* estado-unidense, e a formação de um time com pelo menos 3 alunos membros da CASS/IEEE e um professor orientador. No time (GME-Cefet) estavam os alunos, além do autor deste trabalho: Ana Italiano, mestrado, UFRJ; Ítalo Bruni, doutorado, UFRJ; e Nicole Corradini, graduação, CEFET/RJ, que trabalharam em uma referência de tensão de ultrabaixo consumo, uma referência de corrente, e um oscilador em anel, respectivamente. Esses 4 projetos foram consolidados em um leiaute final, mostrado na Figura IV.18.

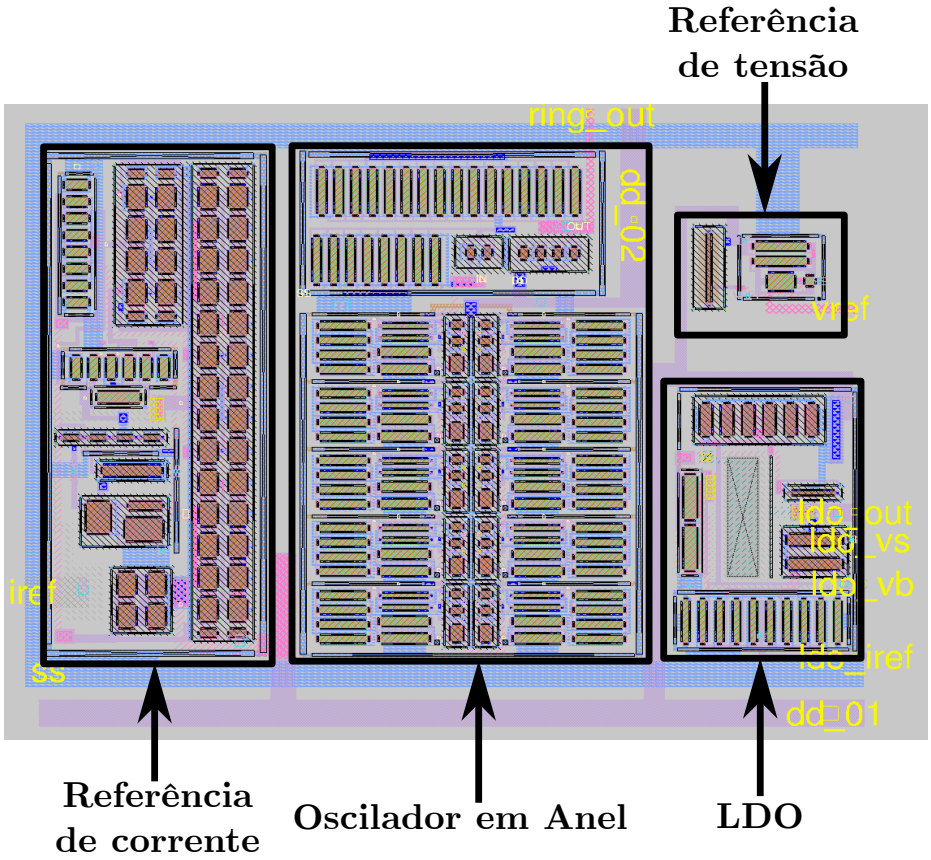


Figura IV.18: Leiaute de implementação do circuito de todo o time.

A metodologia de projeto e fabricação da UNIC-CASS utiliza uma base comum utilizada por todos, o *caravel*, que possui uma área utilizável pelos participantes de 10 mm^2 e possui um SoC com um núcleo RISC-V para controle e interface com circuitos digitais e pinos de acesso aos circuitos [27], que não inclui este projeto, exceto pela conexão com os pinos. O *caravel* em que o circuito foi implementado foi compartilhado por outros dois times de outras instituições e o leiaute finalizado do *chip* é mostrado na Figura IV.19

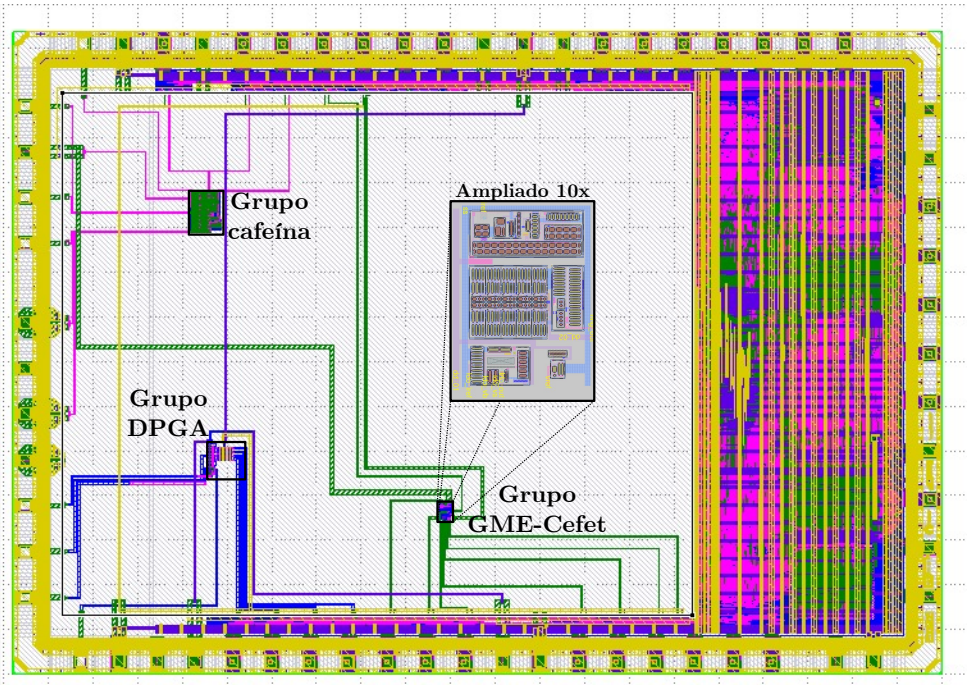


Figura IV.19: Projeto final enviado à UNIC-CASS.

O projeto foi enviado em novembro de 2023 para fabricação e recebido no primeiro semestre de 2025, em 5 amostras, com a Figura IV.20 mostrando alguns dos *chips* recebidos, no encapsulamento QFN64 em que dois dos chips já vieram soldados em placas para o *socket* M.2 chave E.

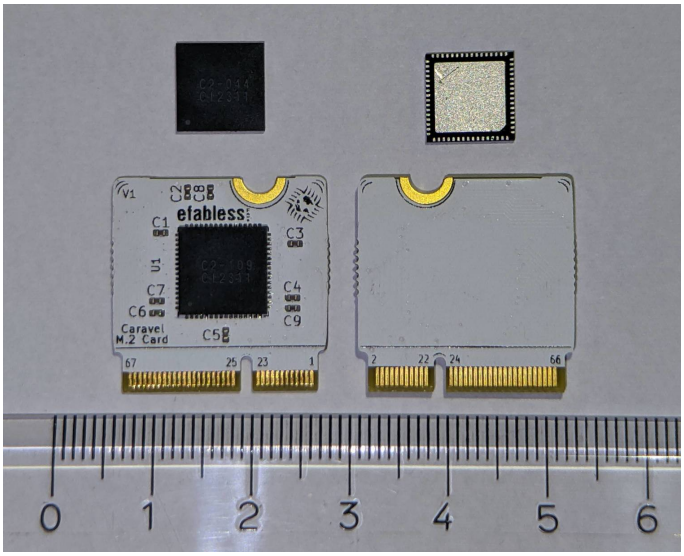


Figura IV.20: Amostras de *chips* recebidos.

IV.6 Desafios de Medição

A placa para o *socket* M.2 é feita para ser testada utilizando uma placa de desenvolvimento, mostrada na Figura IV.21, que contém além do suporte para a placa M.2, um gerador de duas tensões necessárias para alimentar o SoC, de 1,8 V e 3,3 V; um gerador de clock de 10 MHz; um *chip* de memória *eprom* para armazenamento do binário contendo a programação; e uma interface de programação USB que permite a gravação do binário de programação diretamente de um computador conectado a uma porta USB micro-B na placa. A placa também possui *headers* para conexão dos *GPIOs*, clock externo, alimentação externa, entre outros pinos digitais para desenvolvimento e depuração.

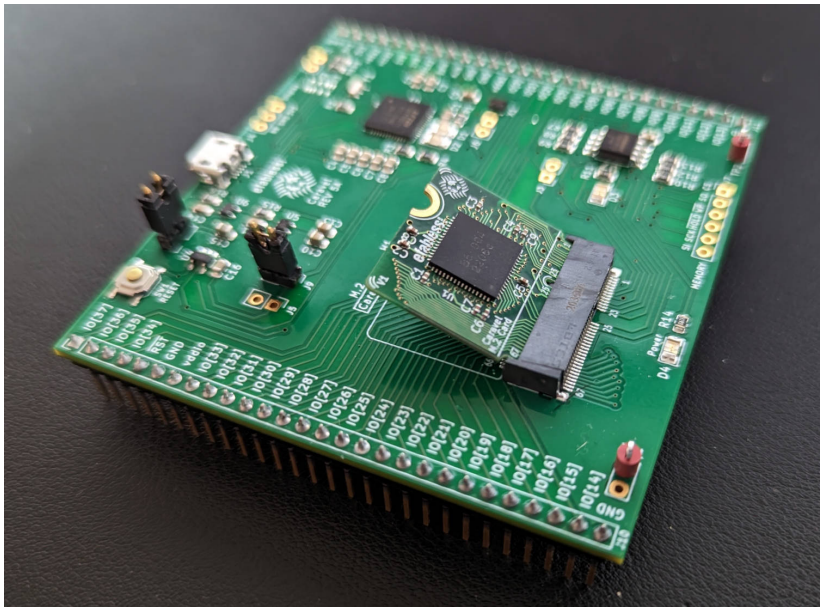


Figura IV.21: Placa de desenvolvimento para o projeto. Fonte: [28].

No entanto, não foi possível receber esta placa a tempo na logística, e, portanto, uma placa de desenvolvimento personalizada foi projetada para que se fosse possível realizar as medições e testes. A placa foi desenvolvida utilizando o programa KiCad, também de código aberto, expondo apenas as conexões importantes para a medição dos circuitos do time GME-CEFET, que incluem a alimentação, *clock*, interface de comunicação com a memória *eprom* e os pinos GPIOs dos circuitos; a placa desenvolvida também conta com trilhas especialmente projetadas para a realização de medições que exigem baixa corrente *leakage*, com trilhas de guarda, espaço para a soldagem direta ou através da entrada M.2, e também com espaço para amplificadores operacionais na configuração *buffer*, já que o instrumento de medição conectado ao nó de saída pode alterar o valor medido, o *buffer* utilizado possui entrada em *gate*, e, portanto, possui consumo muito baixo de corrente. Esta placa pode ser vista na Figura IV.22

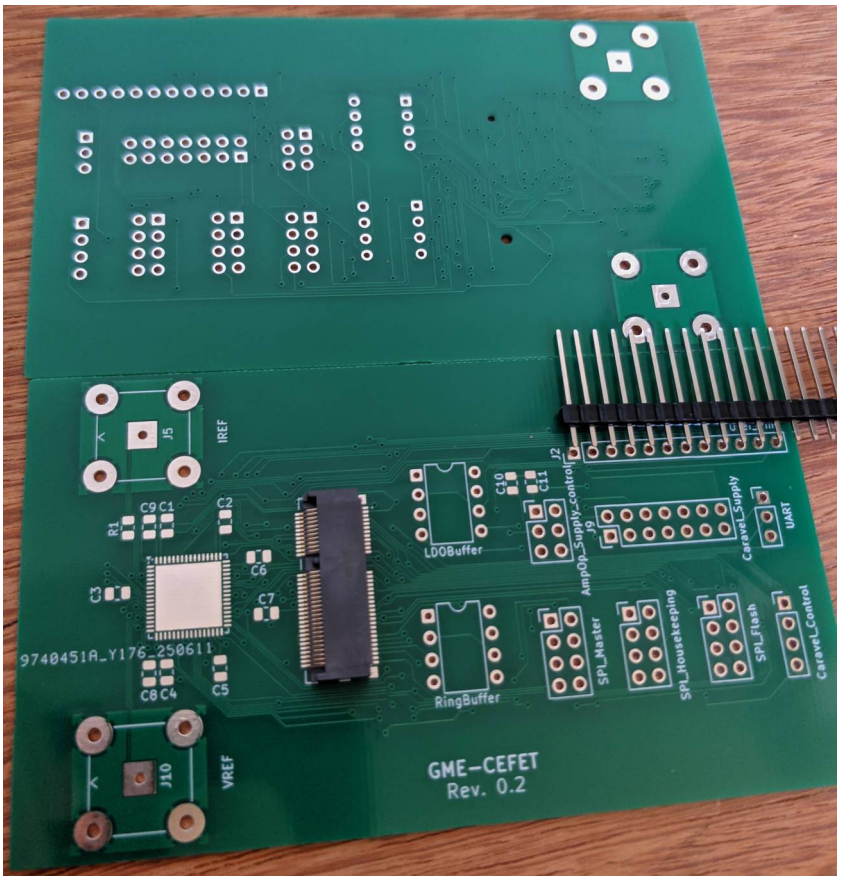


Figura IV.22: Placa desenvolvida pelo autor desta dissertação para a medição do *chip*.

Com a placa em mãos, foi necessário fornecer a alimentação, *clock* e a memória externamente, o que foi feito em uma protoboard com um *Raspberry Pi Pico W*, um microcontrolador que fornece a alimentação de 5 V e 3,3 V para a protoboard; essas tensões alimentam outra placa de regulação de tensão e amplificação, que contém um regulador que gera 1,8 V e amplificadores operacionais, com o amplificador operacional sendo usado como *buffer*. Há também na protoboard a memória eprom, um botão de reset e resistores para proteger as entradas dos circuitos, e para programar a memória foi utilizado um gravador externo. A bancada é mostrada na Figura IV.23 e a lista dos equipamentos que foram utilizados, incluindo dois que não estão na foto, está na Tabela IV.5.

Tabela IV.5: Equipamentos de bancada utilizados

Equipamento	Marca	Modelo	Observações
Osciloscópio	Rigol	MSO5074	Banda de 100 MHz
Fonte de bancada	Longwei	LW-K3010D	Fonte chaveada
Alicate amperímetro	Minipa	ET-3367C	Função de multímetro
Gravador de eprom	Xgecu	T48	-

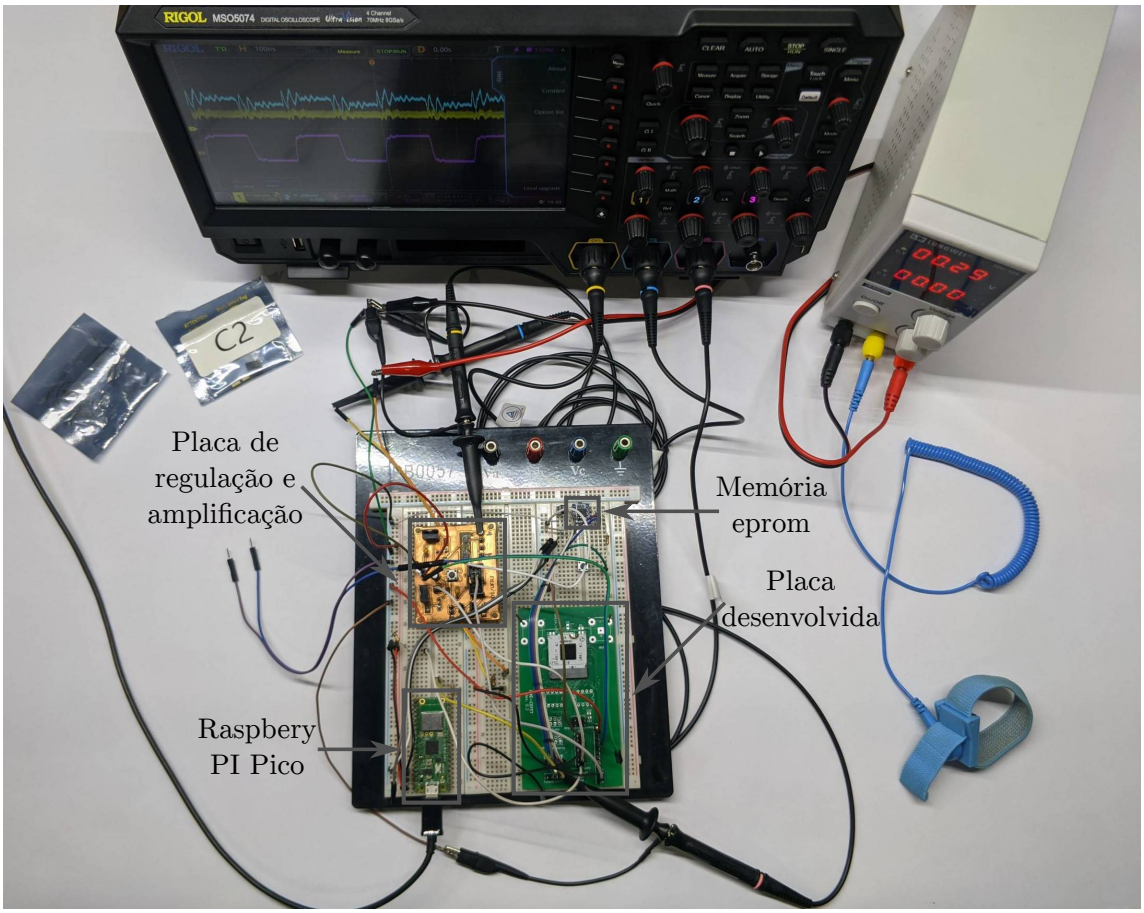


Figura IV.23: Bancada de teste utilizada.

O microcontrolador presente no caravel deve ser programado para permitir o acesso aos circuitos analógicos internos através de configurações nos pinos, no entanto foram feitas várias tentativas nas duas amostras de *chip* soldadas às placas M.2 sem sucesso, e por isso, não houve tempo hábil para realizar medições.

IV.6.1 Testes de Bancada Propostos

Enquanto as placas de circuito impresso eram fabricadas, foi pensada a forma com que os testes seriam realizados na bancada, pois muitas coisas que são fáceis de serem implementadas em um simulador não necessariamente podem ser implementadas na bancada com a mesma facilidade. Para os testes que envolvem a tensão de entrada, Testes A e C, possuem execuções similares ao simulador de circuitos, bastando conectar uma fonte de tensão, ou gerador de sinais, na entrada do circuito e observar a tensão de saída através de um *buffer*, como pode ser visto nas Figuras IV.24 e IV.25, em que o LDO foi simplificado em um bloco com seis nós: o terra, quatro entradas e uma de saída.

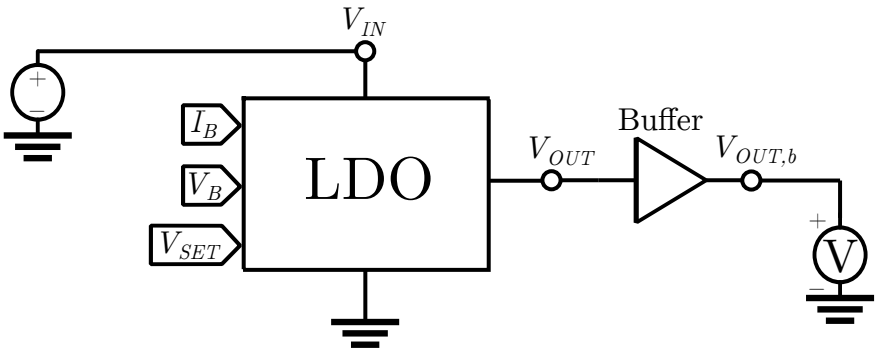


Figura IV.24: Proposta de realização do Teste A.

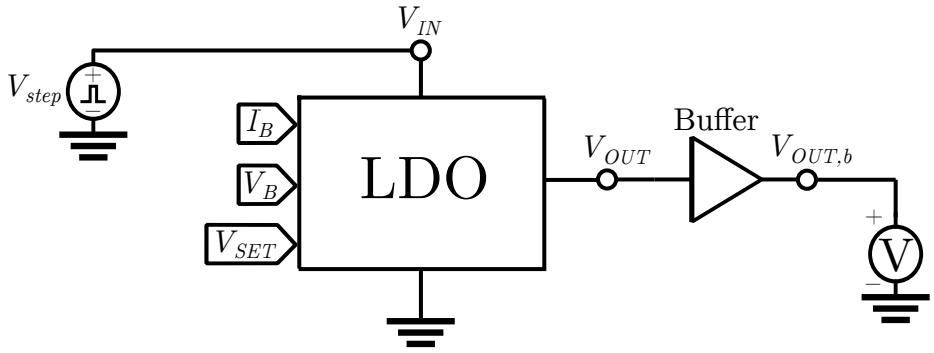


Figura IV.25: Proposta de realização do Teste C.

Para os testes que envolvem a variação da corrente de saída, o circuito mostrado na Figura IV.26 foi elaborado, com o amplificador de transcondutância CA3080 sendo responsável por fazer essa variação. Com isso, o Teste B pode ser realizado variando lentamente a tensão da fonte V_{step}, ou conectando um gerador de sinais para a realização do Teste D.

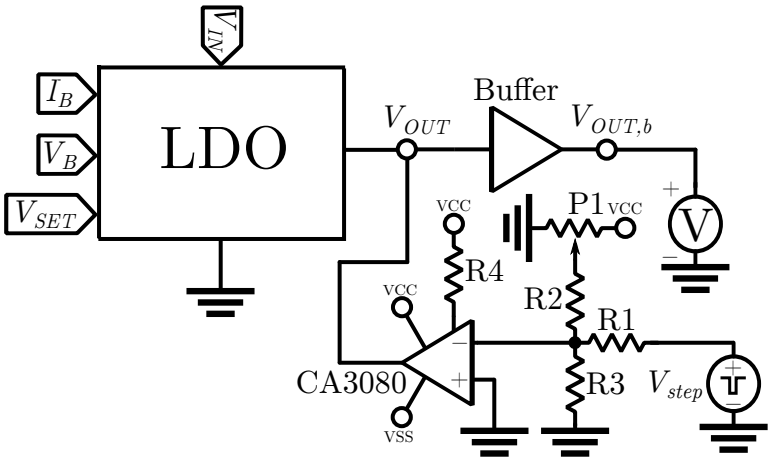


Figura IV.26: Proposta de realização do Teste B e D.

Para o Teste F, de capacitância, foi pensado em conectar todos os pinos ao terra e utilizar um capacímetro conectado ao nó de saída.

Capítulo V Conclusões e Trabalhos Futuros

V.1 Conclusões

Neste trabalho foi desenvolvida uma ferramenta computacional para o projeto de reguladores de tensão de ultra baixo consumo de energia. A ferramenta mostrou desempenho promissor no dimensionamento inicial do circuito LDO, diminuindo assim o tempo normalmente requerido para se obter um circuito funcional. Desta forma, devido ao desempenho computacional que a ferramenta pode alcançar, foi possível levar em conta vários parâmetros de projeto, e assim otimizar a figura de mérito (FoM) do circuito. Por outro lado, a ferramenta foi implementada utilizando uma combinação de softwares de código aberto, tais como NGSpice (simulador de circuitos) e Python (*loop* principal do algoritmo), contribuindo assim com a acessibilidade da comunidade microeletrônica no projeto de circuitos.

Como foi mostrado nos resultados através de simulações elétricas, a ferramenta foi capaz de projetar o circuito LDO em duas tecnologias CMOS diferentes: 130 e 180 nm. Como descrito na Tabela III.9, a ferramenta foi utilizada para projetar dois LDOs, com desempenho satisfatório nos dois processos CMOS, com bom consumo de corrente, sendo ambos adequados para aplicações de ultra baixo consumo de energia.

A placa de teste desenvolvida, abordada na Seção IV.6, será utilizada também pelos outros membros do grupo GME-CEFET para medirem seus respectivos circuitos. Já que a placa foi projetada para ser versátil em relação ao tipo de chip, seja soldado ou na placa M.2, podendo ser inclusive levada a uma estufa, para realizar medições em temperaturas diferentes, já há circuitos que precisam ser testados nestas condições.

De forma a divulgar os resultados desenvolvidos durante esta dissertação, parte deste trabalho foi publicado e apresentado na conferência internacional IBERCHIP/LASCAS 2021 [11], Arequipa, Peru, e posteriormente, resultados adicionais foram publicados em 2025 na conferência internacional *Latin American Conference on Internet of Things*, Fortaleza, CE [12]. Contribuindo assim com a experiência do autor desta dissertação na escrita e apresentação de trabalhos científicos.

V.2 Trabalhos Futuros

Para os próximos passos deste trabalho, planeja-se:

- Adicionar à simulação elétrica o circuito de *feedback* para controlar a tensão de saída como visto na literatura, a fim de se obter circuitos com melhor desempenho.
- Realizar testes nos *chips* recebidos, já que durante as tentativas foi possível verificar que a parte digital do chip funciona, com a programação do microcontrolador sendo executada com sucesso.
- Submeter uma versão estendida do artigo [12] apresentado na conferência em 2025, incluindo resultados experimentais que venham a ser obtidos com o acesso à parte analógica do chip.
- Realizar uma comparação entre a Evolução Diferencial do *tensorflow* e o algoritmo proposto.
- Disponibilizar o código fonte de todas as rotinas implementadas, para que este trabalho possa contribuir para a comunidade *open source* de microeletrônica.

Referências Bibliográficas

[1] UNIC-CASS, "UNIC-CASS: Universalization of ic design from cass," acesso em 2 de Abril de 2025. [Online]. Available: https://ieee-cas.org/universalization-ic-design-cass-unic-cass

[2] S. Y. Peng, L. H. Liu, P. K. Chang, T. Y. Wang, and H. Y. Li, "A Power-Efficient Reconfigurable Output-Capacitor-Less Low-Drop-Out Regulator for Low-Power Analog Sensing Front-End," *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 64, no. 6, pp. 1318–1327, 2017.

[3] M. Cheah, D. Mandal, B. Bakkaloglu, and S. Kiaei, "A 100-ma, 99.11 current efficiency, 2-mvpp ripple digitally controlled ldo with active ripple suppression," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 25, no. 2, pp. 696–704, 2017.

[4] Y. Huang, Y. Lu, F. Maloberti, and R. P. Martins, "Nano-Ampere Low-Dropout Regulator Designs for IoT Devices," *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 65, no. 11, pp. 4017–4026, nov 2018.

[5] X. Wang and P. P. Mercier, "A dynamically high-impedance charge-pump-based ldo with digital-ldo-like properties achieving a sub-4-fs fom," *IEEE J. Solid-State Circuits*, vol. 55, no. 3, pp. 719–730, 2020.

[6] N. Adorni, S. Stanzione, and A. Boni, "A 10-ma ldo with 16-na iq and operating from 800-mv supply," *IEEE J. Solid-State Circuits*, vol. 55, no. 2, pp. 404–413, 2020.

[7] S. DasGupta and P. Mandal, "An automated design approach for cmos ldo regulators," in *2009 Asia and South Pacific Design Automation Conference*, 2009, pp. 510–515.

[8] F. Olivera and A. Petraglia, "A Computer-Aided Approach for Voltage Reference Circuit Design," *Analog Integrated Circuits and Signal Processing*, vol. 89, no. 3, pp. 511–520, 2016.

[9] K. D. Khalil, N. R. Soliman, and H. Omran, "Automation of bandgap voltage reference optimization using vectorized coarse-fine grid search," in *2019 7th International Japan-Africa Conference on Electronics, Communications, and Computations, (JAC-ECC)*, 2019, pp. 54–57.

[10] K. Mohamed, S. Nafea, and H. Omran, "Design automation of low dropout voltage regulators: A general approach," *Electronics*, vol. 12, no. 1, 2023. [Online]. Available: https://www.mdpi.com/2079-9292/12/1/205

[11] I. Meirelles, F. Olivera, and A. Petraglia, "A computer-aided approach for low-power ldo regulator design," in *IBERCHIP Workshop, LASCAS 2021*, 2021, pp. 1–4.

[12] I. Meirelles, L. Almeida, and F. Olivera, "An evolutionary algorithm based approach for low-power ldo voltage regulator designs," in *2025 IEEE Latin Conference on IoT (LCIoT)*, 2025, pp. 207–210.

[13] M. H. Perrott. CppSim and NGSpice data modules for Python. (2020, Aug 18). [Online]. Available: www.cppsim.com/download_hspice_tools.html

[14] M. Tian, V. Visvanathan, J. Hantgan, and K. Kundert, "Striving for small-signal stability," *IEEE Circuits and Devices Magazine*, vol. 17, no. 1, pp. 31–41, 2001.

[15] D. Johns and K. Martin, *Analog Integrated Circuit Design*, 2nd ed. Wiley, 2011.

[16] de SouzaAdemilton Luiz Rodrigues, D. G. Ribeiro, L. de Souza Grigorio, I. M. Martins, and V. S. B. Muniz, "Problema inverso aplicado a equação de antoine para acetona e etanol," *REVISTA CEREUS*, vol. 13, no. 3, pp. 184–195, out. 2021. [Online]. Available: http://www.ojs.unirg.edu.br/index.php/1/article/view/3540

[17] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Aprendizado de máquina em larga escala em sistemas heterogêneos," 2015, software disponível em tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[18] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge: MIT Press, 1996.

[19] P. Hazucha, T. Karnik, B. A. Bloechel, C. Parsons, D. Finan, and S. Borkar, "Area-efficient linear regulator with ultra-fast load regulation," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 4, pp. 933–940, 2005.

[20] H. Valapala and P. M. Furth, "Analysis and design of fully integrated very low quiescent current ldos," in *2012 IEEE 55th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2012, pp. 230–233.

[21] H. Mühlenbein and D. Schlierkamp-Voosen, "Predictive models for the breeder genetic algorithm i. continuous parameter optimization," *Evolutionary Computation*, vol. 1, no. 1, pp. 25–49, 1993.

[22] N. Holtschulte, "Schaffer's f6 benchmark function," https://web.archive.org/web/20220929070539/https://www.cs.unm.edu/~neal.holts/dga/benchmarkFunction/schafferf6.html, 2022, página arquivada; conteúdo indisponível desde 2022.

[23] J. Liu, T. Bryant, N. Maghari, and J. Morroni, "A 90na quiescent current 1.5v–5v 50ma asynchronous folding ldo using dual loop control," in *2016 IEEE Asian Solid-State Circuits Conference (A-SSCC)*, 2016, pp. 221–224.

[24] X. Qu, Z.-K. Zhou, B. Zhang, and Z.-J. Li, "An ultralow-power fast-transient capacitor-free low-dropout regulator with assistant push–pull output stage," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 60, no. 2, pp. 96–100, 2013.

[25] Efabless, "Using openlane to harden your design — caravel harness documentation," https://caravel-harness.readthedocs.io/en/latest/caravel-with-openlane.html, 2021, accesso: 18/09/2025.

[26] gme-cefet, "pmu-circuits," https://github.com/gme-cefet/pmu-circuits, 2025, gitHub repository, branch: main, accessed 2025-10-04.

[27] Efabless, "Efabless Caravel," acesso em 30 de Abril de 2025. [Online]. Available: https://caravel-harness.readthedocs.io/en/latest/

[28] ——, "M2 breakout inserido — imagem da documentação do caravel board," https://github.com/efabless/caravel_board/blob/main/firmware/chipignite/_docs/M2_breakout_inserted.jpg, 2025, acesso: 18/09/2025.