UNIVERSIDADE FEDERAL RURAL DO SEMI-ÀRIDO
DEPARTAMENTO DE ENGENHARIAS
BACHARELADO EM ENGENHARIA ELÉTRICA

HADASSA JULIANA DA COSTA GOMES

SÍNTESE LÓGICA E DESIGN FÍSICO DE UM PROCESSADOR RISC-V

CARAÚBAS
2025

HADASSA JULIANA DA COSTA GOMES

SÍNTESE LÓGICA E DESIGN FÍSICO DE UM PROCESSADOR RISC-V

Monografia apresentada a Universidade Federal Rural do Semi-árido como requisito de obtenção de título de Bacharela em Engenharia Elétrica.

Orientador(a):
Prof. Dr. Francisco de Assis Brito Filho

CARAÚBAS
2025

G633s Gomes, Hadassa Juliana da Costa.
Síntese lógica e design físico de um processador RISC-V / Hadassa Juliana da Costa Gomes. – 2025.
34 f. : il.

Orientador: Francisco de Assis Brito Filho.
Monografia (graduação) – Universidade Federal Rural do Semi-árido, Curso de Engenharia Elétrica, 2025.

1. Design Digital. 2. ASIC. 3. Síntese Lógica. 4. Implementação Física. 5. PDK. I. Brito Filho, Francisco de Assis, orient. II. Título.

Ficha catalográfica elaborada por sistema gerador automáto em conformidade com AACR2 e os dados fornecidos pelo) autor(a).
Biblioteca Campus Caraúbas
Bibliotecária: Sarah Freire Bezerra
CRB: 15/1052

O serviço de Geração Automática de Ficha Catalográfica para Trabalhos de Conclusão de Curso (TCC´s) foi desenvolvido pelo Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo (USP) e gentilmente cedido para o Sistema de Bibliotecas da Universidade Federal Rural do Semi-Árido (SISBI-UFERSA), sendo customizado pela Superintendência de Tecnologia da Informação e Comunicação (SUTIC) sob orientação dos bibliotecários da instituição para ser adaptado às necessidades dos alunos dos Cursos de Graduação e Programas de Pós-Graduação da Universidade.

HADASSA JULIANA DA COSTA GOMES

SÍNTESE LÓGICA E DESIGN FÍSICO DE UM PROCESSADOR RISC-V

Monografia apresentada a Universidade Federal Rural do Semi-árido como requisito de obtenção de título de Bacharela em Engenharia Elétrica.

Defendido em 31 de janeiro de 2025.

BANCA EXAMININADORA:

gov.br Documento assinado digitalmente
FRANCISCO DE ASSIS BRITO FILHO
Data: 09/04/2025 09:39:11-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Francisco de Assis Brito Filho
Presidente

gov.br Documento assinado digitalmente
NAYANA LETICIA DE MORAIS VIANA
Data: 08/04/2025 20:00:03-0300
Verifique em https://validar.iti.gov.br

Ma.Nayana Letícia de Morais Viana
Membro

gov.br Documento assinado digitalmente
LINCOLN ALEXANDRE PAZ SILVA
Data: 08/04/2025 13:50:10-0300
Verifique em https://validar.iti.gov.br

Me. Lincoln Alexandre Paz Silva
Membro

AGRADECIMENTOS

Agradeço, em primeiro lugar, à minha avó Fátima, a quem chamo de mãe, por ter me criado e, desde a infância, sempre me incentivar, acreditando no meu potencial. À minha mãe, Jóedna, às minhas irmãs e à minha sobrinha Laura, pelo apoio e acolhimento incondicional.

Aos meus amigos de Caraúbas, Mara, Cícero e Gesley, por estarem sempre ao meu lado, e aos amigos da residência, que, apesar da distância, continuam presentes. A Bruno e sua família, pelo incentivo e apoio incondicional ao longo dessa jornada.

Expresso também minha gratidão aos amigos que fiz em Campinas. À Sara Brito, por me proporcionar a oportunidade de aprender sobre Physical Design. Ao meu supervisor, Guilherme Godoi, por compartilhar seu conhecimento e contribuir para meu crescimento profissional. A Iago, por me fazer enxergar meu potencial, e a Patrick, Érico e Vini, pelo suporte constante.

Aos que tornaram meus dias mais leves e alegres em Campinas—Júlia, Pedrão, Grazi, Gisa, Bert, Pedrinho, Joseito e Vitória—meu sincero agradecimento.

Por fim, ao professor Francisco Brito, meu orientador, por me apresentar ao universo da microeletrônica e mostrar que este caminho seria possível.

RESUMO

O design digital de circuitos integrados desempenha um papel fundamental na evolução da microeletrônica, permitindo o desenvolvimento de sistemas cada vez mais eficientes e complexos. Dentro desse contexto, este trabalho tem como objetivo a síntese lógica e física de um IP digital, utilizando ferramentas comerciais da Cadence e PDKs (*Process Design Kits*) diversos. O estudo aborda o fluxo de design de ASICs, detalhando suas etapas, desde a especificação inicial até a implementação física, com ênfase na síntese e no *Place & Route* (P&R). Como estudo de caso, o processador PicoRV32 é utilizado para demonstrar o fluxo de implementação. São analisados parâmetros como consumo de potência, desempenho em tempo e área ocupada. Com isso, é feita uma análise comparativa do uso de mais de um nó tecnológico para um mesmo *design*. São utilizados três PDKs: um comercial, um *open source* e um PDK da Cadence. O trabalho visa contribuir para a disseminação do conhecimento sobre o fluxo de design digital de ASICs, proporcionando uma abordagem prática e acessível para estudantes e pesquisadores interessados na área.

Palavras-chave: Design digital; ASIC; Síntese Lógica; Implementação Física; PDK.

ABSTRACT

The digital design of integrated circuits plays a crucial role in the advancement of microelectronics, enabling the development of increasingly efficient and complex systems. In this context, this work aims to perform the logic and physical synthesis of a digital IP using commercial Cadence tools and various Process Design Kits (PDKs). The study explores the ASIC design flow, detailing its stages from initial specification to physical implementation, with a focus on synthesis and *Place & Route* (P&R). As a case study, the PicoRV32 processor is implemented to demonstrate the design flow. Parameters such as power consumption, performance, and area are analyzed. With this, a comparative analysis is made of the use of more than one technology node for the same design. Three PDKs are used: a commercial one, an open-source one, and a Cadence PDK. This work contributes to the dissemination of knowledge on ASIC digital design flow, providing a practical and accessible approach for students and researchers interested in the field.

Key words: Digital design; ASIC; Logic Synthesis; Physical implementation; Open-source PDK.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

A microeletrônica desempenha um papel de grande relevância no desenvolvimento tecnológico do Brasil, sendo essencial para a inovação em setores estratégicos como telecomunicações, automação industrial, saúde e energia. O setor de semicondutores no país está em expansão, com o governo brasileiro incentivando seu crescimento por meio de políticas públicas e parcerias entre universidades, empresas e institutos de pesquisa (DELGADO, 2023). Dentro desse campo, o *design* digital de ASICs (*Application-Specific Integrated Circuits*) se destaca como uma subárea de grande importância, com aplicações práticas em diversas indústrias que demandam soluções personalizadas e de alto desempenho.

O fluxo de *design* digital compreende as etapas de *front-end*, *back-end* e verificação até a fabricação do circuito integrado. Na etapa de *front-end*, são realizadas a especificação do projeto e a criação do *design* em nível lógico, geralmente descrito por meio de linguagens como VHDL ou Verilog. Com os dados gerados nessa etapa, o *back-end* é responsável pela síntese lógica e física, que convertem o *design* lógico em hardware real, seguido pelo *layout* físico do chip. Trata-se de um fluxo processo complexo, que envolve mais detalhes do que os mencionados e demanda o uso de softwares avançados e engenheiros qualificados para otimizar aspectos fundamentais do projeto, como potência, desempenho e área.

No contexto acadêmico de um curso generalista de Engenharia Elétrica, os alunos têm acesso à base de circuitos digitais e eletrônica, bem como a alguns processos relacionados à microeletrônica. No entanto, há uma lacuna significativa em materiais práticos que guiem o aluno especificamente no processo de *design* digital de ASICs, uma vez que esse não é o foco do curso. Essa área, devido à sua especificidade e complexidade, é geralmente aprendida apenas em especializações ou após a inserção no mercado de trabalho. Como consequência, a falta de conhecimento prático nesse campo pode se tornar um obstáculo significativo para estudantes e iniciantes que desejam ingressar em carreiras relacionadas ao *design* digital de ASICs.

Diante desse cenário, este trabalho busca preencher essa lacuna ao oferecer uma abordagem prática para o *design* digital de ASICs, com ênfase nas etapas de síntese lógica e física. Por meio da utilização de ferramentas comerciais amplamente empregadas na indústria e de bibliotecas de desenvolvimento *open source*, como o SkyWater 130nm e o IHP 130 nm, o trabalho pretende proporcionar uma oportunidade de adquirir conhecimento prático na área de *design* digital. Além disso, o estudo visa fomentar avanços nesse campo, colaborando com a preparação para as demandas do mercado e contribuindo para o fortalecimento da microeletrônica no Brasil.

A relevância deste estudo se dá na crescente adoção de ferramentas *open source* no desenvolvimento de circuitos integrados, especialmente em projetos de pesquisa e educação. Ao demonstrar a viabilidade da síntese lógica e física de um IP digital utilizando bibliotecas

open source, este trabalho contribui para a disseminação de conhecimento e para o avanço de metodologias de *design* acessíveis e de baixo custo. Adicionalmente, os resultados obtidos podem servir como referência para futuros projetos que busquem integrar IPs digitais em sistemas mais complexos, utilizando tecnologias abertas e colaborativas.

Este documento está organizado da seguinte forma: no Capítulo 2, é apresentado o referencial teórico necessário para a compreensão do fluxo de síntese lógica e *design* físico, bem como uma visão geral sobre os PDKs utilizados. O Capítulo 3 descreve a metodologia adotada para a implementação do projeto, detalhando o ambiente de desenvolvimento e as ferramentas utilizadas. No Capítulo 4, são apresentados os resultados obtidos durante a síntese e o *design* físico do PicoRV32, juntamente com a análise das métricas de desempenho. Por fim, o Capítulo 5 apresenta as conclusões do trabalho.

1.1 OBJETIVOS

O principal objetivo deste trabalho é realizar a síntese lógica e física de um IP (*Intellectual Property*) digital descrito em Verilog, utilizando ferramentas comerciais da Cadence e PDKs diversos. Para alcançar esse objetivo, foram definidos os seguintes objetivos específicos:

- Compreender e aplicar os conceitos e técnicas da implementação física de circuitos integrados digitais.
- Implementar o processador PicoRV32, abrangendo as seguintes etapas:
  - Síntese lógica: conversão do *design* em uma *netlist*.
  - *Place & Route* (P&R): geração do *layout* físico do circuito.
- Analisar o consumo de potência, a performance e a área utilizada no *design*.
- Analisar a viabilidade do PDK comercial IHP 130 nm em comparação com os outros PDKs e identificar limitações.
- Aprimorar o conhecimento na área de *design* digital e no uso de ferramentas EDA.

2 REFERENCIAL TEÓRICO

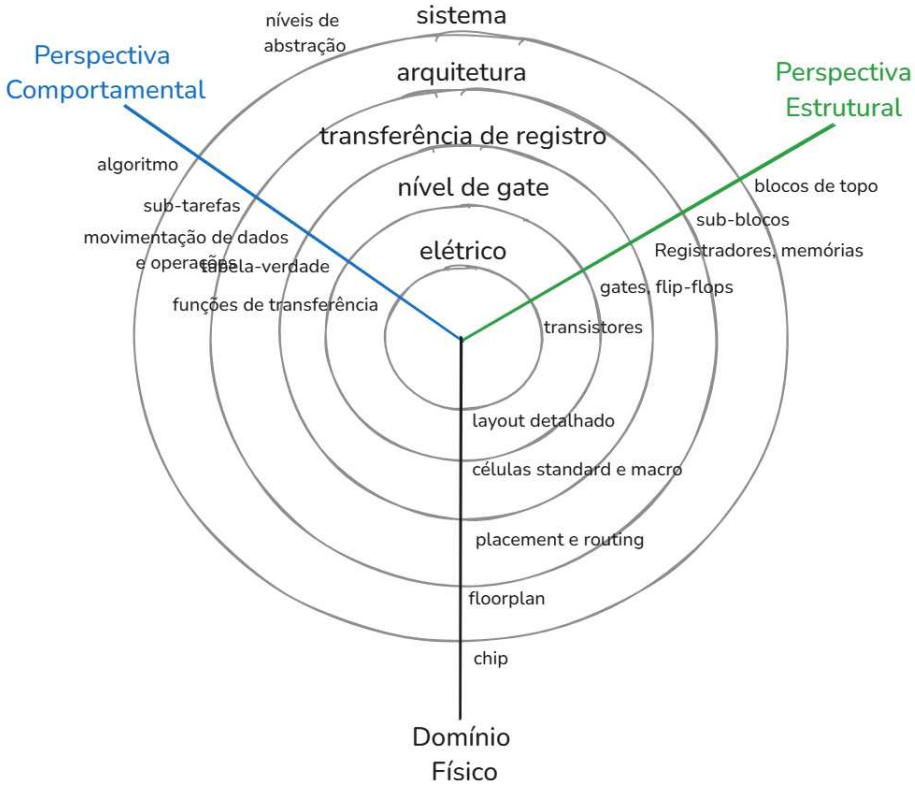
Para a realização deste trabalho, é essencial compreender os princípios do *design* digital e o fluxo de desenvolvimento de circuitos integrados dedicados (ASICs). Este capítulo tem como objetivo apresentar os conceitos essenciais que embasam o projeto e a implementação de sistemas digitais, abrangendo desde a especificação e modelagem funcional até as etapas de síntese lógica e física. Além disso, serão discutidos os recursos e ferramentas utilizados nesse processo, que envolvem desafios como a otimização de área, consumo de energia e desempenho.

2.1 *DESIGN* DIGITAL VLSI

Antes de detalhar o fluxo de *design* de ASICs, é essencial compreender os conceitos básicos que sustentam o *design* digital em VLSI (*Very Large Scale of Integration*), no qual circuitos com milhões ou até bilhões de transistores são integrados em um único chip. Esse processo abrange desde a concepção do sistema, passando pela modelagem funcional, até a implementação física do circuito integrado.

Para ilustrar o fluxo de *design* digital em VLSI, uma ferramenta visual amplamente utilizada é o diagrama em Y, mostrado na Figura 1 (KAESLIN, 2008).

Figura 1 – Diagrama Y do fluxo de *design* digital



Fonte: Autoria Própria

Introduzido pela primeira vez por Gadjski, o diagrama apresenta os três domínios principais do *design* digital (comportamental, estrutural e físico), mostrando como eles se interconectam em diferentes níveis de abstração.

O eixo comportamental ou funcional representa o que um determinado sistema faz. Nele, o circuito é descrito de forma abstrata, sem se preocupar com sua implementação física. Partindo do mais alto nível de abstração, pode-se tomar como exemplo a lógica de um algoritmo, que descreve um comportamento específico cuja funcionalidade é conhecida, mas não há a necessidade de detalhar a forma de sua implementação (KAESLIN, 2008).

O domínio estrutural descreve a estrutura hierárquica do design, ou seja, como são feitas as interconexões dos módulos para corresponder a um determinado comportamento. Para exemplificar, descendo um pouco nos níveis de abstração, no nível lógico, a perspectiva estrutural compreende como é feita a comunicação entre as portas lógicas para moldar determinada funcionalidade (WESTE; HARRIS, 2011).

Por fim, o domínio físico refere-se à implementação física do *design*, que se inicia no domínio comportamental e passa pelo estrutural ao longo do fluxo de *design*. Para cada nível de abstração, o domínio físico explica como construir fisicamente o sistema correspondente (WESTE; HARRIS, 2011). A seção seguinte abordará com mais detalhes como esses domínios estão representados no fluxo de *design* de ASICs.

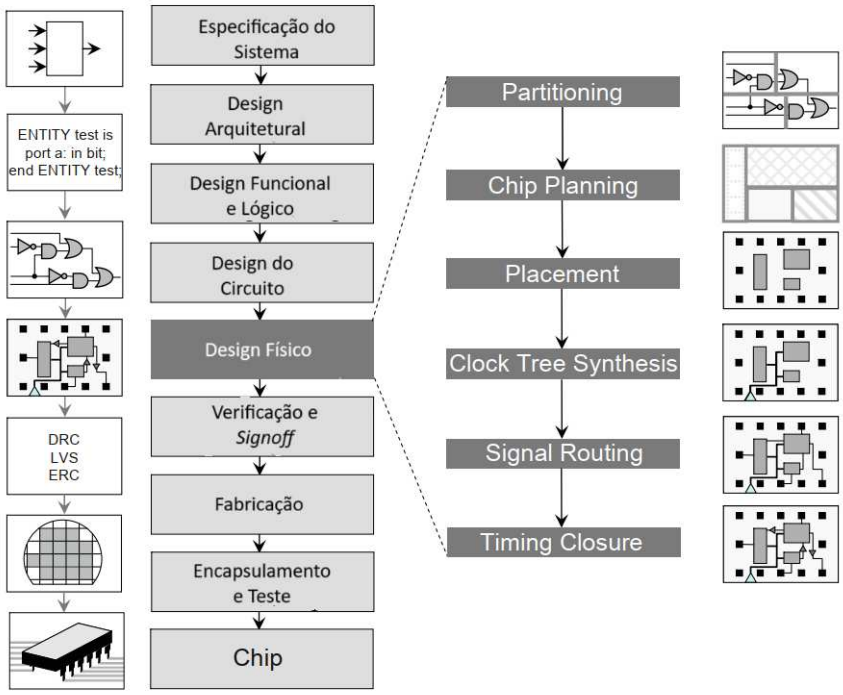
2.2 FLUXO DE *DESIGN* DIGITAL DE ASICS

O fluxo de *design* de ASICs é comumente dividido em duas etapas principais, são elas o *front-end*, que envolve os domínios comportamental e estrutural, e o *back-end*, que abrange parte do domínio estrutural e se desenvolve no domínio físico (WESTE; HARRIS, 2011). Esse processo, desde a especificação inicial do projeto até a obtenção do produto final, é ilustrado pelo fluxograma apresentado na Figura 2.

2.2.1 Especificações de *design*

A primeira etapa do fluxo consiste na definição das especificações do *design*. Dada a ideia do produto, um grupo geralmente composto por *designers* de chips, projetistas de circuitos, a equipe de marketing de produto, gerentes de operações e projetistas de layout colabora para definir coletivamente os objetivos gerais e requisitos de alto nível do sistema (KAHNG et al., 2011).

Figura 2 – Fluxo de *design* de ASIC



Fonte: (KAHNG et al., 2011)

2.2.2 Arquitetura do *design*

A etapa seguinte compreende o *design* arquitetural. Dadas as especificações do *design*, uma determinada arquitetura deve ser definida, envolvendo decisões como: tecnologia a ser utilizada, interfaces de entrada e saída, protocolos de comunicação, uso de blocos de IP (do inglês *Intelectual Property*), entre outras decisões dependendo dos requerimentos do *design*. Esta fase deve resultar em uma descrição funcional de alto nível, que permita estimar dados como desempenho, potência e tamanho do *die* (chip) a fim de verificar a viabilidade do produto (KAHNG et al., 2011), (SHERWANI, 2002).

2.2.3 *design* Funcional e Lógico (RTL)

Uma vez definida a arquitetura, inicia-se o *design* funcional, que serve como entrada para o *design* lógico. Também conhecido como *design* RTL (*Register-Transfer Level*), essa etapa envolve a descrição detalhada de todos os blocos funcionais do circuito utilizando linguagens de descrição de hardware (HDL), como VHDL ou Verilog. Além de capturar o comportamento lógico, esta fase permite especificar aspectos relacionados ao *timing*, como frequências de operação e margens de atraso. A etapa é concluída com a verificação funcional do *design* RTL, realizada para garantir que o circuito esteja em conformidade com os requisitos iniciais antes de avançar para a síntese lógica (BERTACCO, 2006).

2.2.4 Síntese Lógica

A síntese lógica compreende a geração do *netlist*, uma representação detalhada do circuito em nível estrutural que descreve os componentes lógicos, como, por exemplo, portas lógicas e registradores, e as conexões entre eles. O principal objetivo dessa etapa é gerar um modelo detalhado do circuito, otimizado com base nas restrições de projeto definidas previamente. Por exemplo, o *design* pode ser ajustado para minimizar o consumo de energia, reduzir a área do circuito integrado final ou facilitar a testabilidade do produto (BERTACCO, 2006). É nessa etapa onde geralmente se encerra o *front-end* e se inicia o *back-end*.

2.2.5 *design* Físico

No *back-end*, ocorre a síntese física, etapa que transforma a descrição lógica (*netlist*) em uma representação física (*layout*) pronta para a fabricação do chip. Para iniciar o *design* físico, são necessários dados como arquivos de tecnologia e bibliotecas, o *netlist* gerado a partir do *design* RTL, além de restrições de projeto e requisitos temporais previamente definidos (GOLSHAN, 2007). Essa fase engloba diversas etapas complexas e essenciais para o fluxo de *design*. De forma resumida, durante o *design* físico, são realizadas atividades como o dimensionamento do *die*, o planejamento da distribuição de alimentação elétrica, o posicionamento das células lógicas e o roteamento das interconexões entre os componentes do circuito (SHERWANI, 2002), resultando em um *layout* pronto para ser utilizado na etapa de fabricação do chip.

2.2.6 Fabricação

Após a finalização do *layout*, é realizada a verificação do *design* para que seja validado e enviado para fabricação em uma fundição de silício dedicada (*fab*). Esse envio, chamado de *tapeout*, recebe esse nome porque originalmente era realizado utilizando fita (do inglês, *tape*) magnética (SHERWANI, 2002; KAHNG et al., 2011).

2.3 ESTILOS DE *DESIGN*

A implementação eficiente de um ASIC requer uma abordagem de planejamento adequada, que otimize o tempo de ciclo do projeto e permita atingir metas como área e desempenho. Existem dois estilos principais para a implementação de ASICs: o *design* hierárquico e o *design flat*. Para projetos de pequeno a médio porte, como o abordado neste trabalho, o *design flat* é a escolha mais adequada. Nessa abordagem, o projeto é tratado como um único bloco, sem divisões em subprojetos, o que resulta em melhor utilização de área e simplifica a análise de *timing*, uma vez que o circuito é analisado de forma integral (GOLSHAN, 2007).

No entanto, essa metodologia demanda maior capacidade de memória e tempo de processamento, especialmente em projetos de grande escala, devido à necessidade de manipulação de um volume maior de dados de forma simultânea. A vantagem do *design flat* reside na eliminação da necessidade de reservar espaços extras para alimentação, aterramento e roteamento entre subprojetos, como ocorre no *design* hierárquico, tornando-o mais eficiente para projetos de complexidade moderada (GOLSHAN, 2007).

2.4 PERFORMANCE, POTÊNCIA E ÁREA

No *design* de ASICs, algumas figuras de mérito devem ser consideradas, sendo as principais: Performance, Potência e Área. O desempenho é geralmente avaliado pela frequência máxima de *clock* que o circuito pode alcançar, o que está diretamente relacionado ao tempo de propagação dos elementos internos. A potência abrange tanto a dissipação dinâmica, causada pela operação ativa do circuito, quanto a dissipação estática, que ocorre mesmo em estado de repouso, sendo essencial minimizar esses valores para melhorar a eficiência energética. Já a área refere-se ao espaço físico ocupado no die, influenciando diretamente os custos de fabricação, sendo necessário otimizar o *design* para reduzir a área utilizada sem comprometer as funcionalidades ou especificações desejadas (SAURABH, 2023).

2.4.1 *Constraints*

As *constraints* são restrições de projeto aplicadas em diferentes etapas do fluxo de *design*, desde a síntese lógica até o *layout* físico, para garantir que o circuito atenda aos requisitos de desempenho, área e consumo de energia. Os principais tipos de restrições são:

- **Restrições de *Timing***: Definem os requisitos temporais do circuito, como frequência de operação, tempos de *setup* e *hold*, e margens de atraso. O tempo de *setup* é o intervalo mínimo antes do *clock* em que o sinal de entrada deve estar estável para ser capturado corretamente. Já o tempo de *hold* é o intervalo mínimo após o *clock* em que o sinal de entrada deve permanecer estável para garantir que não haja interferência na captura (SHERWANI, 2002).
  Violações de *setup* ocorrem quando o sinal não chega a tempo antes do *clock*, enquanto violações de *hold* acontecem quando o sinal muda muito rápido após o *clock*, comprometendo a integridade dos dados. Essas restrições são críticas para assegurar o funcionamento correto do circuito na frequência desejada.
- **Restrições de Área**: Especificam o tamanho máximo que o circuito pode ocupar no chip, com base nos requerimentos iniciais de projeto e na aplicação. A otimização de área é essencial para reduzir custos de fabricação e melhorar a eficiência do *design*.

- **Restrições de Potência**: Limitam o consumo de energia do circuito, sendo especialmente relevantes em aplicações móveis e de baixo consumo. Técnicas como *clock gating* e *power gating* são frequentemente utilizadas para atender a essas restrições.

Além disso, as restrições de projeto devem ser balanceadas, pois a otimização de um parâmetro (por exemplo, *timing*) pode impactar negativamente outros (como área ou potência). Ferramentas de EDA modernas permitem a análise e ajuste iterativo dessas restrições, garantindo que o *design* final atenda a todos os requisitos.

2.5 FERRAMENTAS E RECURSOS NO PROCESSO DE *DESIGN*

2.5.1 Ferramentas de EDA

O processo de desenvolvimento de um ASIC, desde sua especificação até a fabricação, envolve uma elevada complexidade e demanda significativa de tempo. Antigamente, grande parte do *design* de circuitos VLSI era realizada manualmente, o que tornava o processo lento e ineficiente (SHERWANI, 2002). O avanço contínuo da tecnologia de semicondutores, conforme previsto pela Lei de Moore, permite hoje o desenvolvimento de circuitos integrados compostos por centenas de milhões de transistores, montados em pacotes com múltiplos chips e milhares de pinos, conectados a placas de circuito de alta densidade com diversas camadas de interconexão (KAHNG et al., 2011).

Neste sentido, surgiram as ferramentas de Automação de *design* Eletrônico (EDA) para viabilizar o processo de *design*. A introdução de *softwares* nesse domínio trouxe maior eficiência e precisão, permitindo que os projetistas analisem diversas opções e escolham a mais adequada para cada projeto. Apesar disso, algumas etapas do fluxo de *design* permanecem caras e difíceis de automatizar, tornando necessário o conhecimento de sistemas para auxiliar nesses casos (SHERWANI, 2002).

De acordo com EMA (2024), as principais empresas de EDA para *design* de ASIC em 2024 são Synopsys, Cadence e Siemens EDA (Mentor Graphics). Essas empresas oferecem ferramentas que integram processos de síntese lógica, *design* físico e verificação. Elas são projetadas para otimizar o fluxo de trabalho, garantindo eficiência e precisão na criação de *design*s complexos, essenciais para a fabricação de ASICs de alto desempenho.

2.5.2 Kit de *design* de Processo

Para que um chip possa ser fabricado, é necessário o uso de um conjunto de bibliotecas denominadas Kit de *design* de Processo (PDK). Essas bibliotecas consistem em uma coleção de arquivos que incluem *layout* físico, visões abstratas, modelos de temporização, modelos de simulação ou funcionais, e descrições de circuitos em nível de transistor (GOLSHAN, 2007). Além disso, o PDK inclui regras de *design* correspondentes

às especificidades do processo semicondutor a ser utilizado e, por isso, é fornecido pela *foundry*, a empresa responsável pela fabricação do chip.

Os principais arquivos do PDK utilizados para a síntese lógica são os arquivos ".lib"e ".lef". O arquivo LEF (*Library Exchange Format*) fornece informações sobre a posição dos pinos, camadas metálicas e limites de posicionamento de uma célula, com seções que definem tecnologia, site e macros, abordando camadas, regras de *design*, vias e capacitância. Já o arquivo LIB (*Liberty Timing File*) contém os modelos de temporização das células, incluindo atrasos, transições e requisitos de *setup* e *hold*. Cada arquivo LIB corresponde a um PVT (*Process, Voltage, Temperature*) *corner*, refletindo variações de temporização sob diferentes condições de operação. Além disso, define unidades de tempo, tensão, corrente, potência de fuga, carga capacitiva, taxa de subida e limiares de entrada e saída (SEMICONDUCTORS, 2025).

Para fins de pesquisa e desenvolvimento, existem alguns PDKs de código aberto (*open source*) que são disponibilizados para a comunidade acadêmica e de pesquisa sem custos, permitindo que projetistas acessem e modifiquem os arquivos e modelos necessários para o desenvolvimento de circuitos.

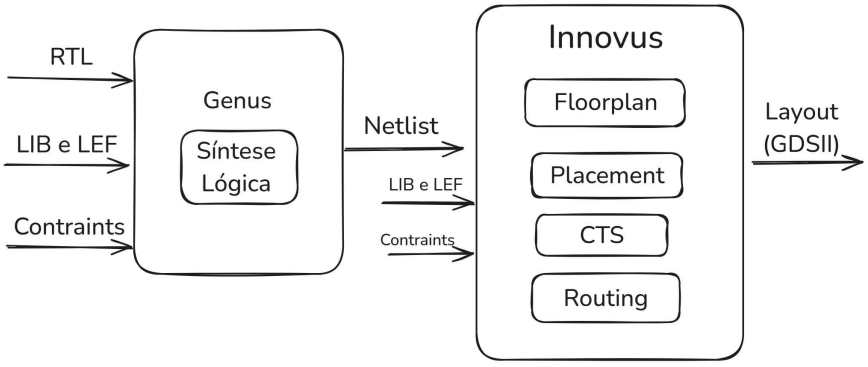
2.6 RISC-V

O RISC-V é uma arquitetura de conjunto de instruções (ISA) aberta e modular, desenvolvida na Universidade da Califórnia, Berkeley, a partir de 2010. Diferente de ISAs proprietárias, como ARM e x86, o RISC-V é de código aberto, o que permite sua livre implementação e adaptação para diversas aplicações, desde sistemas embarcados até supercomputadores. Sua modularidade permite que extensões sejam adicionadas conforme necessário, como suporte a operações de ponto flutuante (F), multiplicação e divisão (M), ou atomicidade (A), tornando-a altamente flexível. Essa característica, combinada com a simplicidade do conjunto base de instruções, torna o RISC-V uma escolha atraente para projetos acadêmicos, de pesquisa e comerciais, especialmente em cenários onde customização e redução de custos são prioritários (WATERMAN; ASANOVIć, 2019).

3 METODOLOGIA

Este capítulo descreve a abordagem adotada para a implementação do fluxo de síntese lógica e física do design digital. A Figura 3 mostra quais são os arquivos de entrada necessários para cada etapa e qual o resultado gerado; esse fluxo foi realizado três vezes, uma para cada PDK disponível. A seguir, são detalhados o ambiente de desenvolvimento, as ferramentas, bibliotecas e o IP utilizado para as sínteses lógica e a física.

Figura 3 – Arquivos de entrada e de saída no fluxo *design* digital para cada ferramenta



Fonte: Autoria Própria

3.1 FERRAMENTAS

Para a realização deste trabalho, foram utilizadas as ferramentas de síntese lógica e design físico da Cadence, Genus e Innovus, configuradas em um ambiente Linux. Essas ferramentas foram disponibilizadas pela UFERSA, que possibilitou o acesso remoto por meio da plataforma x2go. O x2go permitiu a conexão ao servidor remoto, oferecendo uma interface gráfica para o uso das ferramentas de design. Com isso, foi possível realizar o desenvolvimento do trabalho à distância, mantendo a sessão ativa e permitindo o retorno contínuo ao trabalho, mesmo após a desconexão.

3.2 PDKS

Foram utilizados três PDKs para o projeto, conforme descrito a seguir:

- O PDK IHP 130 nm é um kit de design de processo comercial, desenvolvido pelo Instituto IHP na Alemanha, para tecnologias BiCMOS de 130 nm, voltado para projetos de ASICs analógicos, digitais, mistos e RF. Os dados atuais do PDK estão disponíveis como uma versão de acesso antecipado, exclusivamente. Embora a tecnologia SG13G2 e o PDK comercial já tenham sido utilizados para o desenvolvimento de muitos designs, o PDK open-source ainda não

está programado para fins de produção no momento, estando em fase de desenvolvimento (MICROELECTRONICS, 2023).

- O PDK *Sky* 130 nm é uma colaboração entre o Google e a *SkyWater Technology Foundry* para fornecer um kit de design de processo totalmente open source e recursos relacionados, que podem ser usados para criar projetos fabricáveis nas instalações da *SkyWater*. Lançado em maio de 2020, o PDK foi empregado no desenvolvimento de muitos designs que foram fabricados com sucesso em grandes quantidades (GOOGLE, 2020). Sua versão open source é amplamente utilizada em pesquisas, testes iniciais de chips e verificação de design, sendo uma importante ferramenta para a comunidade acadêmica e de pesquisa.
- PDK Genérico da Cadence de 45 nm: O PDK genérico da Cadence para 45 nm é voltado para o design de chips utilizando tecnologia de 45 nm, uma das mais utilizadas para a fabricação de circuitos integrados de alta performance. Diferente dos PDKs open source mencionados anteriormente, este PDK não é de código aberto, sendo um kit comercial que requer a aquisição de uma licença das ferramentas da Cadence para seu uso.

A Tabela 1 mostra a quantidade de metais disponível para roteamento, os *layers*, para cada PDK.

	IHP 130	**Sky 130**	**GPDK045**
Top Layer	Metal 7	Metal 5	Metal 11

Tabela 1 – Quantidade de *layers* para cada PDK

3.3 DESIGN RTL: PICORV32

Por fim, foi escolhido o design RTL a ser sintetizado. O PicoRV32 é um processador RISC-V de 32 bits, altamente configurável e otimizado para aplicações com restrições de área e consumo de energia. Desenvolvido como um projeto open source e disponível no repositório oficial do YosysHQ (GitHub), ele é amplamente empregado em implementações de FPGA e ASIC devido à sua simplicidade e eficiência (YOSYSHQ, 2020).

O núcleo PicoRV32 foi selecionado devido à sua acessibilidade, código RTL bem estruturado e ampla compatibilidade com ferramentas de síntese lógica e física. Além disso, há muitos trabalhos já realizados com esse IP, o que reduz a necessidade de uma etapa de verificação funcional, que não faz parte do escopo deste trabalho.

3.4 SÍNTESE LÓGICA NO GENUS

Para a realização da síntese lógica no Genus, os arquivos de entrada necessários são os arquivos dos tipos .lef, .lib e o design RTL, no caso deste trabalho o arquivo em Verilog do PicoRV32. Além disso, é necessário um arquivo de *constraints*, que contenha

todas as informações de *clock* e restrições de tempo dependendo da aplicação. Na prática, esse arquivo é desenvolvido por quem faz o design RTL, guiado pela equipe responsável pela definição dos requerimentos de design. Para não fugir do objetivo deste trabalho, foi criado um arquivo simples de *constraints*, contendo apenas as informações mais básicas do *clock*, como nome, períodos e atrasos. Este mesmo arquivo foi utilizado para realizar a síntese para os três PDKs.

O processo de síntese lógica no Genus é facilitado, pois a própria ferramenta gera um script para ser preenchido com as especificações e com o caminho para cada arquivo de entrada. Nesse script, além dos arquivos já mencionados, a ferramenta sugere a inserção de um arquivo QRC ou CapTable, que são arquivos que contém dados relacionados à extração de parasitas (como resistência e capacitância). Esses arquivos foram utilizados em cada síntese conforme disponibilidade do PDK, como apresentado na Tabela 2

PDK	**.lib**	**.lef**	**QRC**	**CapTable**
IHP 130	sim	sim	sim	-
Sky 130	sim	sim	-	sim
GPDK045	sim	sim	-	-

Tabela 2 – Inputs disponíves para o Genus em cada PDK

3.5 DESIGN FÍSICO NO INNOVUS

Após a realização da síntese lógica no Genus, são gerados alguns arquivos que são utilizados com *inputs* para o *layout* no Innovus. Esses arquivos são o netlist em Verilog, o arquivo de *constraints* modificado pela síntese. Além disso, os arquivos do PDK também são utilizados como *inputs* no INNOVUS.

Diferente do fluxo no Genus, o fluxo no Innovus é bastante complexo, pois envolve todas as etapas de *layout* e demanda muito mais tempo de execução. O Innovus também permite a geração de scripts para facilitar a automação do *layout*, porém, o trabalho de configurar cada um desses scripts pode ser bastante extenso, uma vez que existem várias possibilidades de guiar um design. Diante disto e dada a simplicidade do projeto picoRV32, o fluxo de *Place & routing* foi realizado por meio da GUI (*Grafic User Interface*) em uma primeira execução, para que em seguida os comandos fossem registrados em um script em TCL e pudessem ser utilizados nas próximas execuções.

3.5.1 *floorplan*

O primeiro passo do *floorplan* consiste em definir a área do bloco (*die*). Com o fluxo alinhado ao Genus, ao carregar os inputs e configurações sugeridas do Genus no Innovus, a ferramenta gera um design inicial com uma estimativa de área, sendo necessário apenas adicionar o espaço desejado para o *core ring*, um anel de trilhas de VDD e VSS,

e a densidade desejada de ocupação de células. No *floorplan*, as únicas configurações diferentes para cada um dos testes são o nó tecnológico do PDK e a quantidade de *layers*, características que devem ser modificadas para cada execução.

3.5.2 Placement

Na etapa de *Placement*, as células lógicas geradas pela síntese são posicionadas no *die*. Existem comandos para direcionar a posição de determinados grupos de células, e isso é útil quando se tem uma definição de diagrama de bloco de um design pré-definida. No caso deste projeto, como não há especificações de posicionamento de células, o Innovus realiza o placement arbitrariamente.

Após o placement, é realizada uma otimização, para corrigir DRVs e preparar o design para a síntese da árvore de *clock* (CTS). Durante a otimização pré-CTS, células redundantes ou desnecessárias podem ser removidas, e o tamanho das células pode ser ajustado para reduzir o consumo de potência e a área ocupada.

3.5.3 CTS

Na etapa seguinte ocorre o processo de construção de uma rede de *clock* que distribui o sinal de *clock* a todos os elementos síncronos do circuito. Essa rede é chamada de árvore de *clock* porque, em muitos casos, assume uma estrutura hierárquica semelhante a uma árvore, com um ponto de origem (geralmente o gerador de *clock*) e ramificações que se estendem até os pontos de sincronização.

3.5.4 *Routing*

Na etapa do *routing*, todos os sinais restantes do circuito são roteados de forma detalhada, conectando os pinos das células lógicas de modo a gerar o *layout* físico do chip. A ferramenta realiza iterações para fazer o roteamento das trilhas da melhor maneira possível no espaço definido a fim de evitar congestionamento de rotas. Após essa etapa, foi possível realizar a comparação entre os resultados de cada teste.

4 RESULTADOS

4.1 SÍNTESE LÓGICA NO GENUS

A síntese lógica no GENUS ocorreu sem erros para os três testes. Após a realização da síntese, além do *netlist*, são gerados alguns relatórios para análise de qualidade dos resultados. Desses arquivos, foram selecionados alguns dados de área ocupada para serem utilizados na definição do tamanho do chip. Também foram coletados dados de potência para fins de comparação do resultado da utilização em cada PDK. A Tabela 3 mostra esses dados.

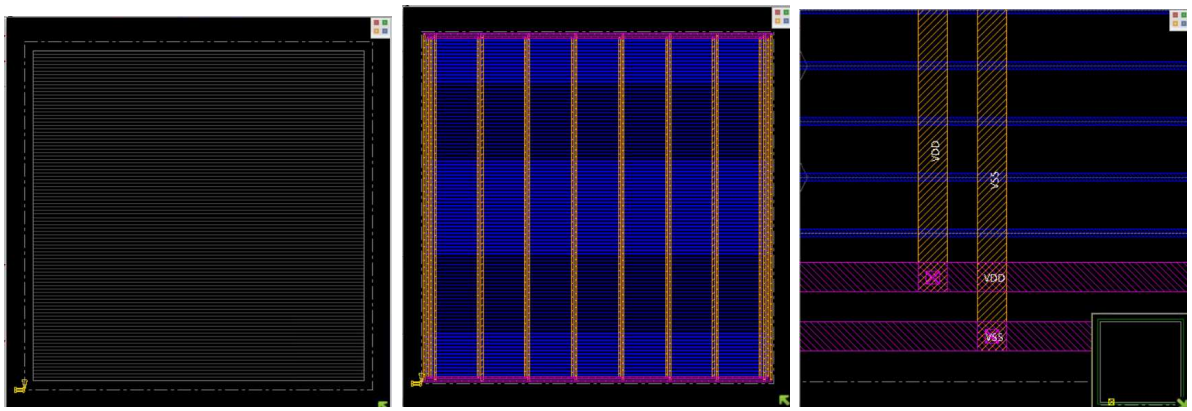
PDK	**Área de célula (μm^2)**	**Área da Net (μm^2)**	**Total (μm^2)**	**Leakage**
IHP 130	91279.824	42564.105	133843.929	2017.404
Sky 130	99059.270	124373.934	223433.205	13064.608
GPDK045	6333.920	6773.770	23107.690	372.990

Tabela 3 – Outputs do genus para cada PDK

4.2 DESIGN FÍSICO NO INNOVUS

4.2.1 *floorplan*

A Figura 4 mostra o resultado desta etapa, que visualmente, é o mesmo para cada uma das rodadas. A Figura 4a mostra o resultado do comando *Specify floorplan*, o qual define a área do core e das *rows*, que são as linhas nas quais as células serão inseridas. A Figura 4b mostra o resultado após a adição dos *core rings* e das demais trilhas do layer máximo (vertical) e do mínimo (horizontal).



(a) *floorplan* após definição da área

(b) *floorplan* após o planejamento de Power

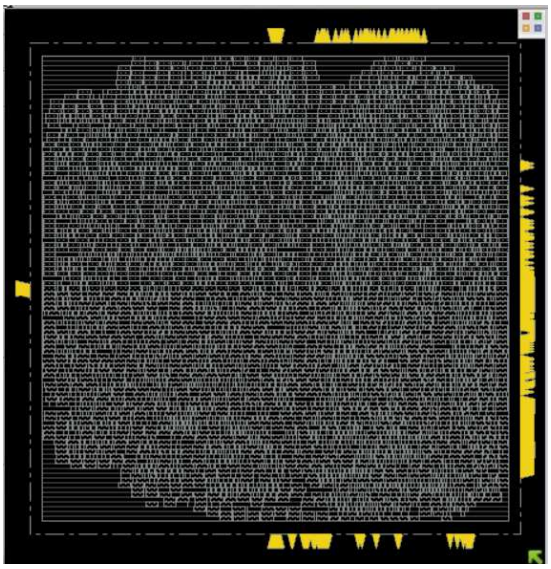
(c) Imagem aumentada do Planejamento de Power

Figura 4 – *floorplan*

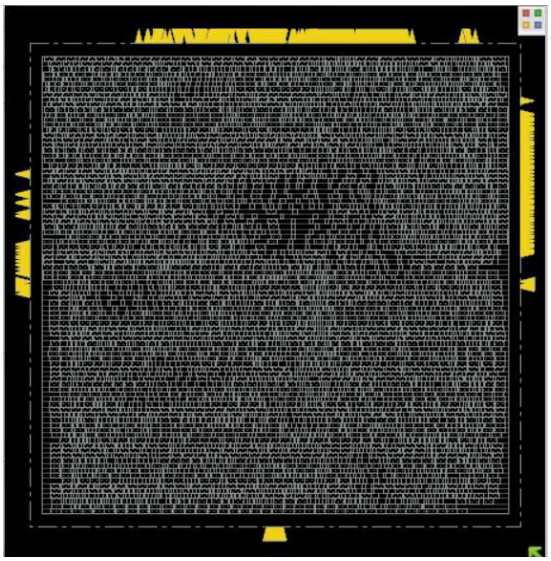
A Figura 4c mostra um trecho da imagem da Figura 4b aumentada, para facilitar a visualização das trilhas de metal utilizadas na grade de *power*.

4.2.2 Placement

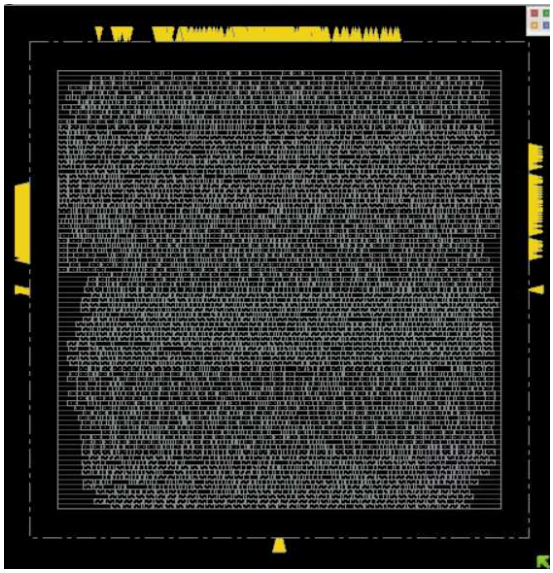
Os resultados pós-*placement* para cada design são mostrados na Figura 5. Se compararmos com a Figura 4a, nas imagens da Figura 5 agora contém uma nuvem de células lógicas, que foram adicionadas após o *placement*. Além disso, agora é possível visualizar os pinos, que são os objetos na cor amarela ao redor do bloco. É possível observar que tanto as células lógicas quanto a posição dos pinos diferem para cada design, devido à característica arbitrária da ferramenta.



(a) IHP130



(b) Sky130

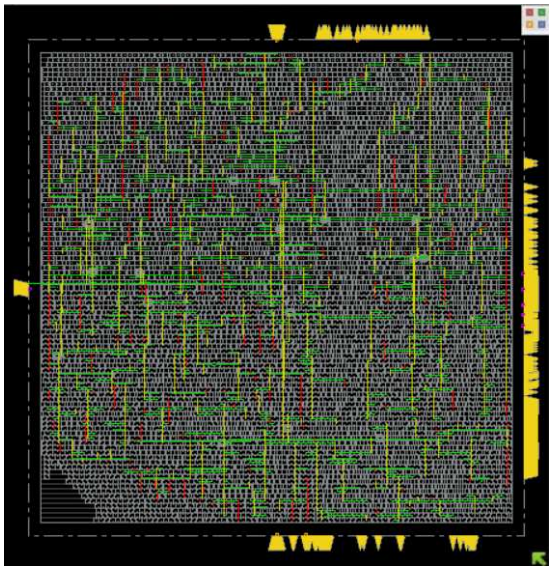


(c) GPDK45

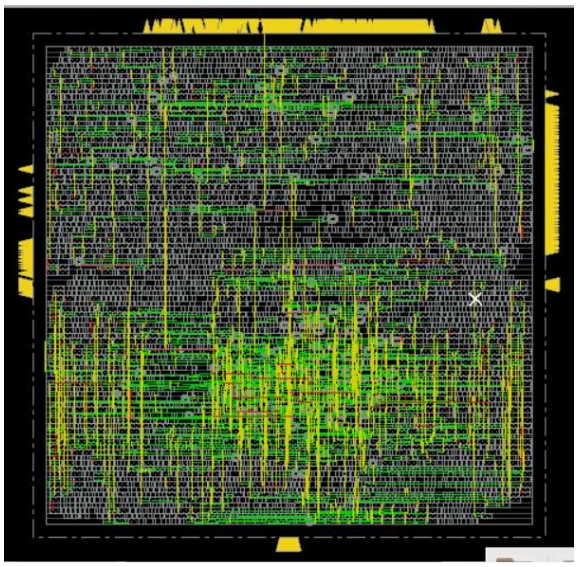
Figura 5 – Placement

4.2.3 CTS

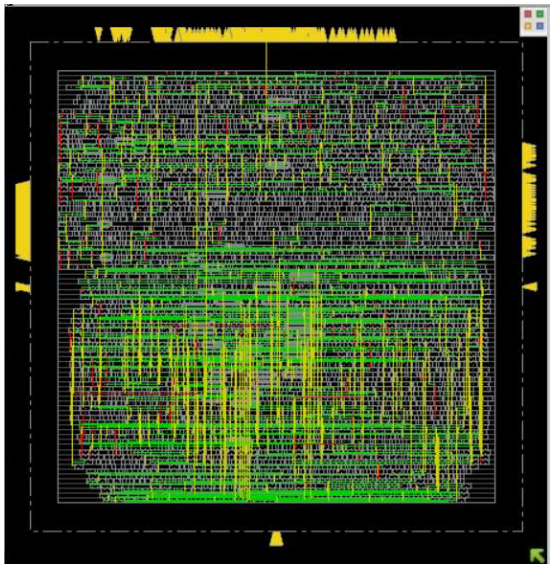
A Figura 6 mostra o resultado do CTS para cada design. Nessa fase, foram adicionadas as trilhas de metal da árvore de *clock*, que são representadas pelas linhas coloridas verticais e horizontais. É possível observar que a quantidade de *nets* se destaca para o GPDK45 e é menor no design com o IHP130.



(a) IHP130



(b) Sky130



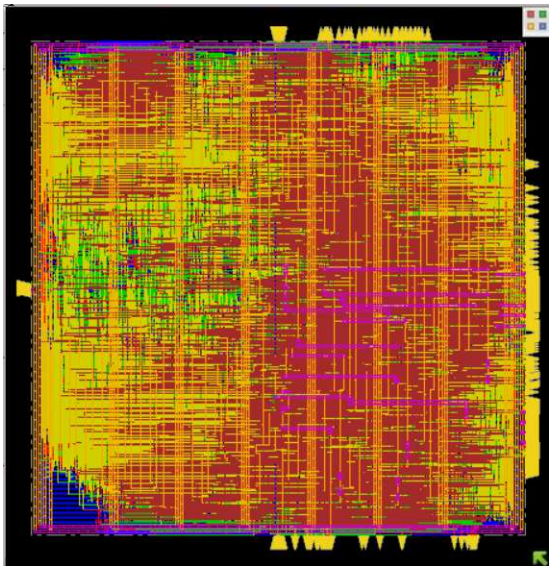
(c) GPDK45

Figura 6 – CTS

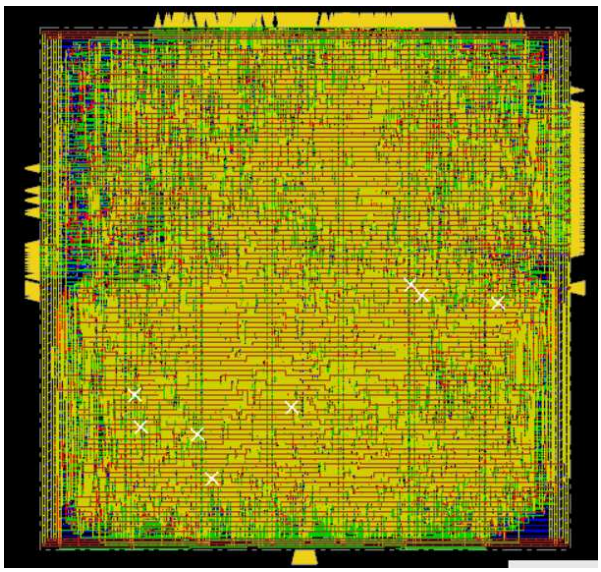
4.2.4 *Routing*

Na etapa do *Routing*, todo o roteamento restante foi realizado. A Figura 7 mostra o resultado desta etapa. Embora a alta densidade de informações dificulte a visualização dos detalhes, ao comparar as bordas de cada uma das figuras, é possível observar que o

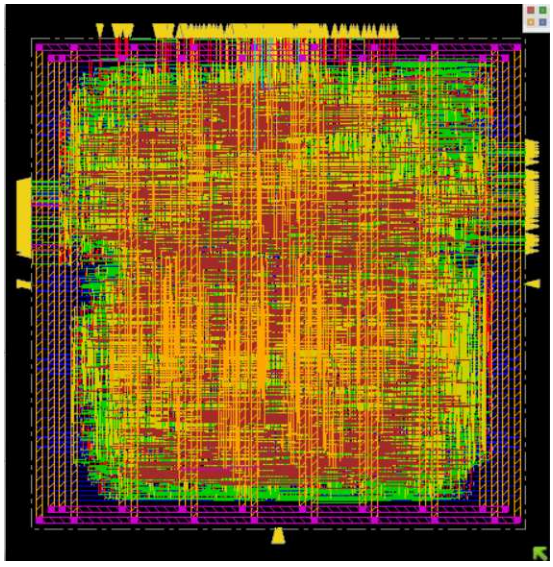
design utilizando o GPDK45 apresenta uma folga significativamente maior em relação ao roteamento quando comparado aos PDKs de 130nm. Entre estes, o design com o Sky130 demonstrou ser o mais congestionado, evidenciando áreas de alta densidade de interconexões e, inclusive, algumas violações de DRC (mostradas pelos símbolos em branco) que não foram observadas nos outros *layouts*.



(a) IHP130



(b) Sky130



(c) GPDK45

Figura 7 – *Layout* pós *routing*

Essa diferença ressalta o impacto das tecnologias e regras de design específicas de cada PDK no processo de roteamento.

4.3 MÉTRICAS

O Innovus, bem como outras ferramentas da Cadence, possui uma *feature* que permite comparar a qualidade dos resultados (QOR) de cada etapa do fluxo. Os resultados

do *Metrics* serão apresentados a seguir.

4.3.1 Resultados para Área

Foram extraídos dados das métricas e dos *logs* do Innovus para alimentar a Tabela 4, que apresenta informações relacionadas à área do chip, como densidade e indicadores de congestionamento. Na tabela, observa-se que o *layout* com o PDK IHP130 apresenta a maior densidade, enquanto os demais designs exibem valores inferiores e bastante próximos entre si.

Design	Densidade (%)	Routing Overflow H (%)	Routing Overflow V (%)
IHP 130	86,88	0,01	0,06
Sky 130	76,37	0,26	0,66
GPDK045	77,96	0,00	0,04

Tabela 4 – Estatísticas de área para cada design

O *routing overflow* é uma métrica que indica a quantidade de congestionamento no roteamento de um design. Ele ocorre quando o número de conexões que precisam passar por uma determinada área excede a capacidade de roteamento disponível naquela região. O congestionamento pode ser medido separadamente para as direções horizontal (H) e vertical (V). De acordo com os manuais de suporte da Cadence para o Innovus, esse valor deve ser nulo, ou muito próximo de zero. Na tabela, é visto que isso não ocorre para o design com o PDK Sky 130, o que é fácil de compreender, visto que esse PDK possui menor número de *layers* disponível, o que dificulta o roteamento para uma mesma área em relação aos outros.

4.3.2 Resultados para *Timing*

Para análise de *Timing* e Potência, foram utilizadas as tabelas do *Metrics* que mostram os dados para cada etapa do fluxo para cada *layout* (Figura 8). Observa-se que, em relação a *timing*, ocorreram apenas violações de *hold*, tais violações podem ser visualizadas nas células de cor vermelha. No design da Figura 8a, essas violações foram corrigidas no pós-CTS.

No segundo design (Figura 8b), as violações de *hold* também foram corrigidas nessa etapa, porém retornaram na fase de *routing*. Isso ocorre porque, nessa etapa, a ferramenta tem permissão para ajustar a posição das células de forma a otimizar o roteamento, o que pode reintroduzir violações de *hold*.

Já no design para o GPDK45 (Figura 8c, a otimização do pós-*routing* não conseguiu remover todas as violações de *hold*, mas isso pode ser corrigido com intervenção manual, trocando as células que estão causando esse atraso por outras de um atraso maior.

Snapshots	Setup (all)			Hold (all)			Power				
	WNS (ns)	TNS (ns)	FEPS	WNS (ns)	TNS (ns)	FEPS	Total (mW)	Leakage (mW)	Internal (mW)	Switching (mW)	Clock (mW)
place	13.303	0	0	-0.331	-490	1895	2.69	0.00	2.13	0.56	2.69
cts	13.243	0	0	-0.313	-451	1703	3.16	0.00	2.21	0.94	3.16
postcts_hold	12.572	0	0	0.000	0	2					
route	12.419	0	0	0.006	0	0	4.29	0.00	2.82	1.47	4.29
postroute	12.181	0	0	0.000	0	1					

(a) IHP130

Snapshots	Setup (all)			Hold (all)			Power				
	WNS (ns)	TNS (ns)	FEPS	WNS (ns)	TNS (ns)	FEPS	Total (mW)	Leakage (mW)	Internal (mW)	Switching (mW)	Clock (mW)
init	5.859	0	0				0.79	0.01	0.68	0.10	0.79
place	4.157	0	0	-0.023	-1	41	3.91	0.01	2.62	1.28	3.91
cts	3.892	0	0	-0.098	-0	1	4.89	0.01	2.73	2.14	4.89
postcts_hold	3.892	0	0	0.014	0	0					
route	3.919	0	0	-0.016	-0	1	5.01	0.01	2.72	2.28	5.01
postroute	3.755	0	0	0.036	0	0					

(b) Sky130

Snapshots	Setup (all)			Hold (all)			Power				
	WNS (ns)	TNS (ns)	FEPS	WNS (ns)	TNS (ns)	FEPS	Total (mW)	Leakage (mW)	Internal (mW)	Switching (mW)	Clock (mW)
init	11.230	0	0				0.05	0.00	0.04	0.00	0.05
place	10.361	0	0	-0.253	-33	416	0.18	0.00	0.12	0.06	0.18
cts	10.289	0	0	-0.286	-38	396	0.20	0.00	0.14	0.06	0.20
postcts_hold	9.772	0	0	0.000	0	0					
route	9.793	0	0	-0.005	-0	3	0.21	0.00	0.14	0.07	0.21
postroute	9.720	0	0	-0.012	-0	5					

(c) GPDK45

Figura 8 – QOR: Timing e Potência

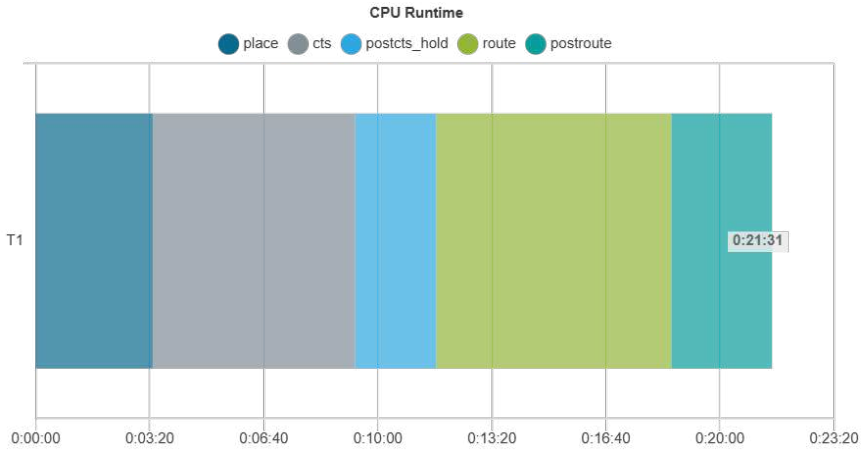
4.3.3 Análise de Potência

A análise de Potência requer a decisão prévia de requisitos de consumo. Como não está no objetivo deste trabalho realizar um projeto de aplicação específica, não há métrica inicial de potência. Entretanto, com os dados obtidos pelo *metrics*, pode ser realizada uma comparação entre os valores de potência para cada teste.

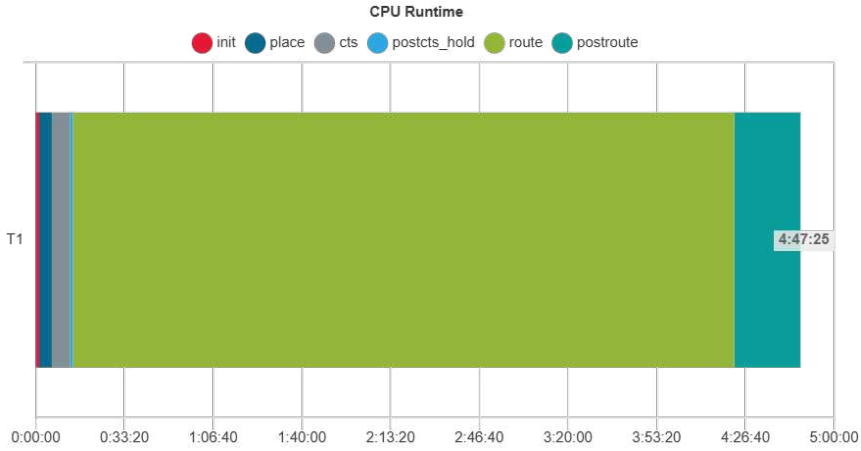
Ao analisar os valores para *power* apresentados na Figura 8, pode-se observar que a maior parte do consumo é devido ao *clock* para todos os casos. Além disso, os valores diferem consideravelmente para os projetos com 130 nm e o com 45 nm.

4.3.4 Tempo de Execução

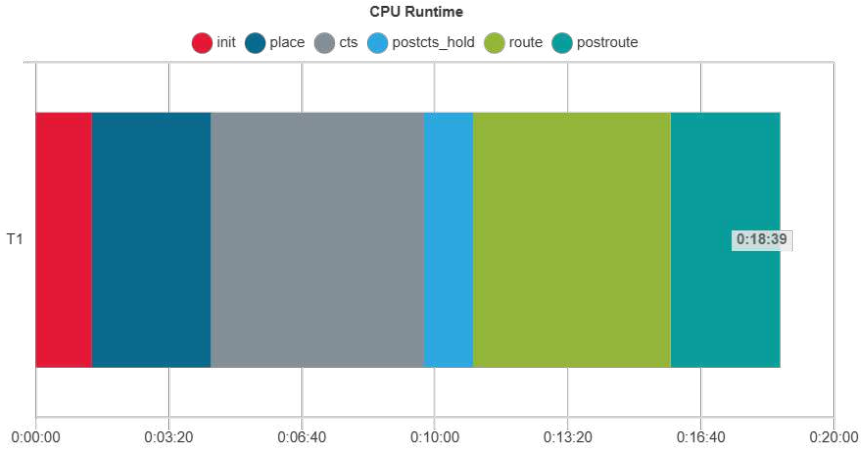
Um dado muito importante obtido a partir das métricas é o gráfico de *Runtime*, que indica o tempo de CPU necessário para a execução de cada etapa do fluxo. Na Figura 9, observa-se uma grande discrepância no tempo de execução do design com o PDK Sky130



(a) IHP130



(b) Sky130



(c) GPDK45

Figura 9 – Tempo de Execução

em relação aos demais. No gráfico (Figura 9b), nota-se que esse acréscimo é causado quase

exclusivamente pela etapa de roteamento, o que está de acordo com os dados apresentados nas seções anteriores.

Devido à quantidade limitada de *layers* e à grande área ocupada pelas células, a ferramenta enfrenta maiores dificuldades para realizar o roteamento desse design. Ao analisar o log do Innovus durante a execução desse design, observou-se que a ferramenta realizou várias iterações nesta etapa para alcançar um resultado aceitável, o que justifica o tempo elevado de processamento.

A Tabela 5 apresenta um resumo das principais métricas mencionadas anteriormente para os três designs distintos, comparando a área, a performance, representada pelo *Worst Negative Slack* (WNS), a potência e o tempo de execução (runtime).

Design	**Área (μm^2)**	***WNS* (ns)**	**Potência (mW)**	***Runtime (min)***
IHP 130	133843	12	4,29	0:21:31
Sky 130	223433	4	5,01	4:47:25
GPDK045	23107	10	0,21	0:18:39

Tabela 5 – Resumo das principais métricas para cada Design

Com base nas métricas apresentadas na Tabela 5, o design IHP 130 se destaca pelo maior WNS (12 ns), indicando melhor margem de temporização, além de um tempo de execução relativamente curto (21 min 31 s). No entanto, ele tem um consumo de potência moderado (4,29 mW) e uma área intermediária (133.843 µm²). O design Sky 130, apesar de possuir o maior consumo de potência (5,01 mW) e a maior área (223.433 µm²), apresenta um WNS de 4 ns, o que é pior que o IHP 130, e um tempo de execução elevado (4 h 47 min 25 s). Já o GPDK045 tem a menor área (23.107 µm²) e o menor consumo de potência (0,21 mW), além de um tempo de execução reduzido (18 min 39 s), mas possui um WNS inferior (10 ns) em relação ao IHP 130, indicando pior desempenho em temporização. Dessa forma, para um mesmo projeto, o IHP 130 pode ser considerado o melhor design para desempenho em performance e até mesmo em *runtime*, enquanto o GPDK045 é mais eficiente em termos de consumo de potência e compactação, e o Sky 130 apresenta o pior equilíbrio entre área, potência e tempo de execução em relação aos outros.

5 CONSIDERAÇÕES FINAIS

Este trabalho teve como objetivo explorar o fluxo de síntese lógica e design físico de um IP utilizando as ferramentas Genus e Innovus, empregando diferentes PDKs para análise comparativa. Durante a realização dos experimentos, foram observadas variações significativas nos resultados obtidos para cada tecnologia, tanto em termos de área quanto de desempenho do layout físico.

Na etapa de síntese lógica com o Genus, verificou-se que a ferramenta oferece um fluxo altamente automatizado, permitindo a inserção de arquivos de entrada e constraints de forma prática. Os resultados mostraram que cada PDK apresentou características próprias de ocupação de área e consumo de potência, sendo o GPDK045 aquele que demandou menor área total e apresentou o menor consumo estático (leakage power), enquanto o Sky130 exibiu a maior ocupação de área devido às diferenças em suas células padrão e regras de design.

Na etapa de design físico com o Innovus, constatou-se que o fluxo de Place & Route demanda um tempo considerável de execução e configurações específicas para cada tecnologia. O floorplan foi definido automaticamente com base nos inputs da síntese, sendo ajustado conforme as particularidades de cada PDK, como número de camadas metálicas disponíveis para roteamento. O placement apresentou diferenças visuais entre os designs, refletindo a disposição arbitrária das células pela ferramenta. Na síntese da árvore de clock (CTS), observou-se que o número de nets de clock variou entre os PDKs, sendo mais elevado para o GPDK045 e mais reduzido para o IHP130. Já na etapa de roteamento, o design com Sky130 demonstrou maior congestionamento e violações de regras de design (DRC), enquanto o GPDK045 apresentou folga significativa nos espaçamentos.

Os resultados quantitativos extraídos das ferramentas permitiram comparar a eficiência de cada PDK em relação às métricas de área, consumo de potência e qualidade do roteamento. Evidenciando que as características da tecnologia impactam diretamente a viabilidade e a qualidade final do layout.

Conclui-se que a escolha da tecnologia do PDK influencia fortemente o resultado final de um projeto de síntese lógica e design físico. Tecnologias mais avançadas, como o GPDK045, permitem um melhor aproveitamento de área e consumo de potência, enquanto tecnologias mais maduras, como o Sky130 e o IHP130, apresentam desafios adicionais relacionados ao congestionamento e otimização do roteamento.

Por fim, este trabalho forneceu uma visão prática e comparativa do fluxo de desenvolvimento de hardware digital, contribuindo para a compreensão das ferramentas de EDA e seus impactos na implementação de circuitos integrados.

REFERÊNCIAS

BERTACCO, V. **Design and Verification of Digital Systems**. 2006. 7-33 p.

DELGADO, R. **Microeletrônica: impulsionadora do desenvolvimento do Brasil**. 2023. Acessado em: 13 out. 2024. Disponível em: <https://revistati.com.br/inovacao/microeletronica-impulsionadora-do-desenvolvimento-do-brasil>.

EMA. **Top EDA Companies for 2024**. 2024. Accesso em: 20 jan. 2025. Disponível em: <https://www.ema-eda.com/ema-resources/blog/top-eda-companies-for-2024-emd>.

GOLSHAN, K. **PHYSICAL DESIGN ESSENTIALS: An ASIC Design Implementation Perspective**. New York: Springer, 2007.

GOOGLE. **SkyWater PDK**. 2020. <https://github.com/google/skywater-pdk>. Acessado em: 20 jan. 2025.

KAESLIN, H. **Digital Integrated Circuit Design: From VLSI Architectures to CMOS Fabrication**. New York: Cambridge University Press, 2008.

KAHNG, A. B. et al. **VLSI Physical Design: From Graph Partitioning to Timing Closure**. New York: Springer, 2011.

MICROELECTRONICS, I. **Open Source PDK**. 2023. Https://www.ihp-microelectronics.com/services/research-and-prototyping-service/fast-design-enablement/open-source-pdk. Acessado em: 20 jan. 2025.

SAURABH, S. **Introduction to VLSI Design Flow**. s.l.: Cambridge University Press, 2023.

SEMICONDUCTORS, S. **LEF, DEF, and LIB Files**. 2025. Acesso em: 20 jan. 2025. Disponível em: <https://signoffsemiconductors.com/lef-def-lib/>.

SHERWANI, N. **ALGORITHMS FOR VLSI PHYSICAL DESIGN AUTOMATION**. s.l.: KLUWER ACADEMIC PUBLISHERS, 2002.

WATERMAN, A.; ASANOVIć, K. **The RISC-V Instruction Set Manual, Volume I: User-Level ISA, Version 2.2**. 2019. Acesso em: 20 jan. 2025. Disponível em: <https://riscv.org/technical/specifications/>.

WESTE, N. H. E.; HARRIS, D. M. **CMOS VLSI Design: A Circuits and Systems Perspective**. s.l.: Pearson, 2011.

YOSYSHQ. **PicoRV32**. 2020. Acessado: 20 janeiro 2025. Disponível em: <https://github.com/YosysHQ/picorv32>.