Kaio Nikelisson de Lima Fernandes

# Relatório de Estágio:
# *NXP Semiconductors*

Campina Grande, PB

2019

Kaio Nikelisson de Lima Fernandes

# Relatório de Estágio:
# *NXP Semiconductors*

Relatório de estágio integrado submetido à Coordenação de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande, *Campus* Campina Grande, como parte dos requisitos necessários para obtenção do título de Graduado em Engenharia Elétrica.

Universidade Federal de Campina Grande – UFCG

Centro de Engenharia Elétrica e Informática

Departamento de Engenharia Elétrica

Orientador: Marcos Ricardo Alcântara Morais, D. Sc.

Campina Grande, PB

2019

Kaio Nikelisson de Lima Fernandes

# Relatório de Estágio: *NXP Semiconductors*

Relatório de estágio integrado submetido à Coordenação de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande, *Campus* Campina Grande, como parte dos requisitos necessários para obtenção do título de Graduado em Engenharia Elétrica.

Trabalho aprovado. Campina Grande, PB, 11 de dezembro de 2019:

---

**Marcos Ricardo Alcântara Morais, D. Sc.**
Orientador

---

**Gutemberg Gonçalves Júnior, D. Sc**
Convidado

Campina Grande, PB
2019

*Este trabalho é dedicado aos dispositivos eletrônicos,*
*que de alguma forma me fizeram perceber que*
*era isso que eu queria fazer.*

# Agradecimentos

Primeiramente, agradeço a minha mãe. Obrigado por tudo. Sou só o eco da orquestra que você toca. Você é minha vida e o meu caminho.

Agradeço também à minha família, á minha segunda mãe - que a vida resolveu presentear na forma de tia: sou eternamente grato, Beniza; ao meu irmão, Romeu; e ao meu padrasto, Lindomar.

Sou grato ao amor que tem nome de Mariana. Pela casa compartilhada por 5 anos, pelos aconchegos e pelas conversas. Pelo nosso futuro.

Aos Manolos (Felipe, Ítalo, João, Lucas, Luiz e Rômulo), muito obrigado pela amizade. De verdade. Em especial, sou grato à João, que embarcou no mesmo navio que eu e não me deixou lutar as batalhas do curso sozinho.

Ao Armário, que me acolheu e compartilhou comigo todos os momentos bons e ruins da engenharia. Não sei como o destino conseguiu reunir pessoas tão incríveis.

Por meio de Marcos Morais e Gutemberg Júnior, os mestres que me guiaram pelos caminhos da microeletrônica, agradeço à todos os professores que colaboraram para a minha formação.

Ao Laboratório X-MEN e todos que fizeram parte dessa história, muito obrigado. Essa organização é parte integral do meu sucesso e tudo o que eu tenho é carinho por tudo e todos.

À NXP Semicondutores, agradeço a incrível oportunidade de estágio. Obrigado pelos conhecimento compartilhado e pela chance de experimentar a sensação de desenvolver tecnologia. Gostaria de agradecer especialmente: à Luis P. Junqueira e à Juliê Marcolino, por terem me confiado à vaga; à Guilherme de Oliveira Coraucci, pelas orientações e ao tempo dedicado ao meu aprendizado e crescimento; à Arquilho Silva, pela paciência e contribuição para o meu desenvolvimento; à Leandro Campos e Marcelo Erigson, pela parceria e pelo esmero que tiveram para com minha pessoa.

Por fim, um homem deve reconhecer seus santuários e, por isso, finalizo com um ode ao café e aos jogos. Às séries e aos animes. Foram essas coisas simples que me fizeram suportar momentos difíceis e clarear a mente fadigada pelos estudos.

*"Still waters run deep."*
*(Provérbio do Latim)*

# Lista de ilustrações

# Lista de tabelas

# Lista de abreviaturas e siglas

| | |
|---|---|
| BSTC | Brazil Semiconductor Technology Center |
| IP | Intelectual Propertie |
| SoC | System on Chip |

# Sumário

# 1 Introdução

O presente relatório apresenta e discorre sobre as atividades desenvolvidas pelo aluno de Engenharia Elétrica da Universidade Federal de Campina Grande (UFCG), Kaio Nikelisson de Lima Fernandes, ao longo do seu estágio. As atividades foram desenvolvidas na sede brasileira da *NXP Semiconductors*, *Brazil Semiconductor Technology Center* (BSTC), com um ciclo completo de execução ocorrendo entre 14 de Janeiro à 30 de Novembro de 2019.

## 1.1 Empresa

A *NXP Semiconductors* é um companhia Holandesa com sede em Eindhoven dedicada ao desenvolvimento e manufatura de soluções em silício (circuitos integrados) para aplicações de microeletrônica embarcada. Atualmente, a empresa possui 66 anos de atuação no mercado de semicondutores - fundada em 1953 pela *Philips* - e conta com um quadro de aproximadamente 30.000 funcionários espalhados por mais de 30 países nos continentes da Ásia, Europa e América (Norte, Sul e Central).

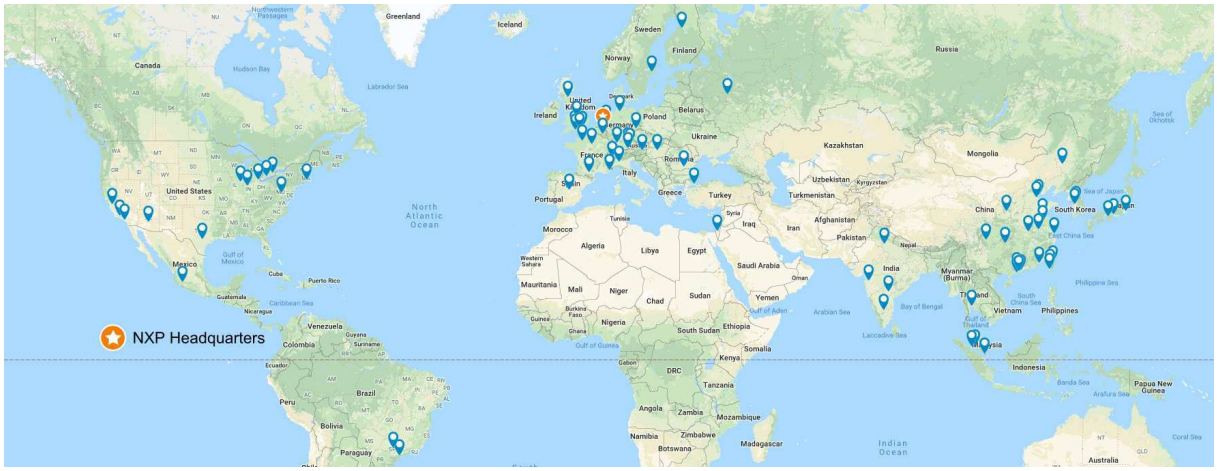


Figura 1 – Distribuição geográfica de filias e sedes da *NXP Semiconductors*.

Historicamente, a NXP surgiu primeiro na forma de divisão de semicondutores da Philips em 1953 (*Philips Semiconductos*). Apenas em 2006, foi concretizada como NXP quando a Philips anunciou o interesse em vender suas operações na área de semicondutores e foi comprada por um grupo de investidores que rebatizou a empresa. Hoje, a NXP trabalha no projeto e manufatura de sistemas digitais, analógicos e mistos destinados aos setores automobilísticos, de segurança, industrial e IoT (*Internet of Things*), comunicações móveis, gerenciamento de potência, entre outros.

Figura 2 – Sede da NXP em Eindhoven, Holanda.

Atualmente, a NXP atende à uma lista distinta de clientes, como: Amazon, Apple, Bosch, Continental, Ericsson, Gemalto, Giesecke and Devrient, Huawei, Hyundai, Kona, Nokia Networks, Panasonic, Samsung and ZTE e tem como missão possibilitar conexões seguras e uma infraestrutura para um mundo mais inteligente, propondo soluções que tornem a vida das pessoas melhores, mais fáceis e mais seguras.

## 1.2 Brazil Semiconductor Technology Center (BSTC)

As atividades que deram origem à NXP Brasil foram iniciadas pela Motorola Inc. em 1967 como uma forma de reconhecimento da qualidade dos profissionais brasileiros. Embora as atividades tenham sido iniciadas tão cedo, foi apenas em 1997 que a Motorola finalmente fundou o seu centro de operações nacional que perduraria e continuaria sendo a atual sede da NXP, o Brazil Semiconductor Technology Center, ou BSTC como é mais conhecido pela comunidade.

Com um crescimento satisfatório, o BSTC já contava com mais de 60 funcionários nos inícios dos anos 2000. Em 2004, a Motorola Inc. decidiu desvincular o seu setor de microeletrônica, tornando-o autônomo e dando origem à Freescale Semiconductors. Por fim, a NXP anunciou a aquisição da Freescale em 2015, fechando o acordo comercial no dia 7 de dezembro de 2015 - quando as duas empresas combinaram-se para formar a atual NXP Semiconductors.

Atualmente, a NXP conta com mais de 130 funcionários (95% engenheiros) contabi-

lizando mais de 100 projetos entregues, incluindo dezenas de microcontroladores e IP chave em áreas como gerenciamento de potência, redes automotivas, processamento digital de sinais e aceleradores criptográficos e de temporização. Além disso, continua sendo sediada no Tech Parque da cidade de Campinas, São Paulo, no Brazil Semiconductor Technology Center (BSTC).

# 2 Referencial Teórico

## 2.1 Microeletrônica

A microeletrônica é uma área da ciência e da engenharia que se ocupa do projeto e desenvolvimento de circuitos integrados baseados na tecnologia de semicondutores. Os semicondutores são dispositivos com características elétricas diferenciadas que permitiram o desenvolvimento do transistor - peça fundamental para o surgimento da eletrônica digital e a sua consequente evolução para a microeletrônica.

## 2.2 Fluxo da Microeletrônica

Atualmente, a produção de um circuito integrado de alta complexidade segue um rigoroso fluxo de projeto suficientemente robusto e refinado de forma que seja possível entregar, em pouco tempo, um sistema com centenas de milhões de transistores. Não obstante esse seja o padrão atual da indústria de microeletrônica, esse modelo de produção só começou a ser desenvolvido em meados da década de 80.

Diante da crescente diminuição do tamanho dos transistores e o consequente surgimento do termo VLSI (*Very Large-Scale Integration*) para descrever a magnitude e complexidade dos circuitos integrados que foram desenvolvidos a partir dos anos 80, a indústria percebeu a necessidade de organizar as etapas de projeto e produção e iniciou um processo de refinamento que culminou no surgimento de um fluxo sólido e genérico.

A Figura 3 apresenta um fluxograma genérico do atual modelo de desenvolvimento adotado tanto pela indústria como pelo meio acadêmico. Nas subseções a seguir, as grandes área de projeto da microeletrônica serão explanadas com mais detalhes.

### 2.2.1 Sistema

O primeiro grupo de profissionais que compõem a equipe de sistema são os engenheiros de marketing. Esses profissionais são responsáveis pela aquisição de projetos e por fechar acordos com as empresas que estão em busca de uma solução eletrônica em SoC. Tais profissionais negociam e entregam os requisitos do sistema para que eles sejam analisados e compilados por um segundo grupo: engenheiros de sistema e arquitetos.

Em geral, os arquitetos e engenheiros de sistema são profissionais experientes e capazes de visualizar o impacto de decisões sistêmicas no desempenho e estrutura final do SoC. Eles estudam os requisitos e desenvolvem um documento de especificação detalhado

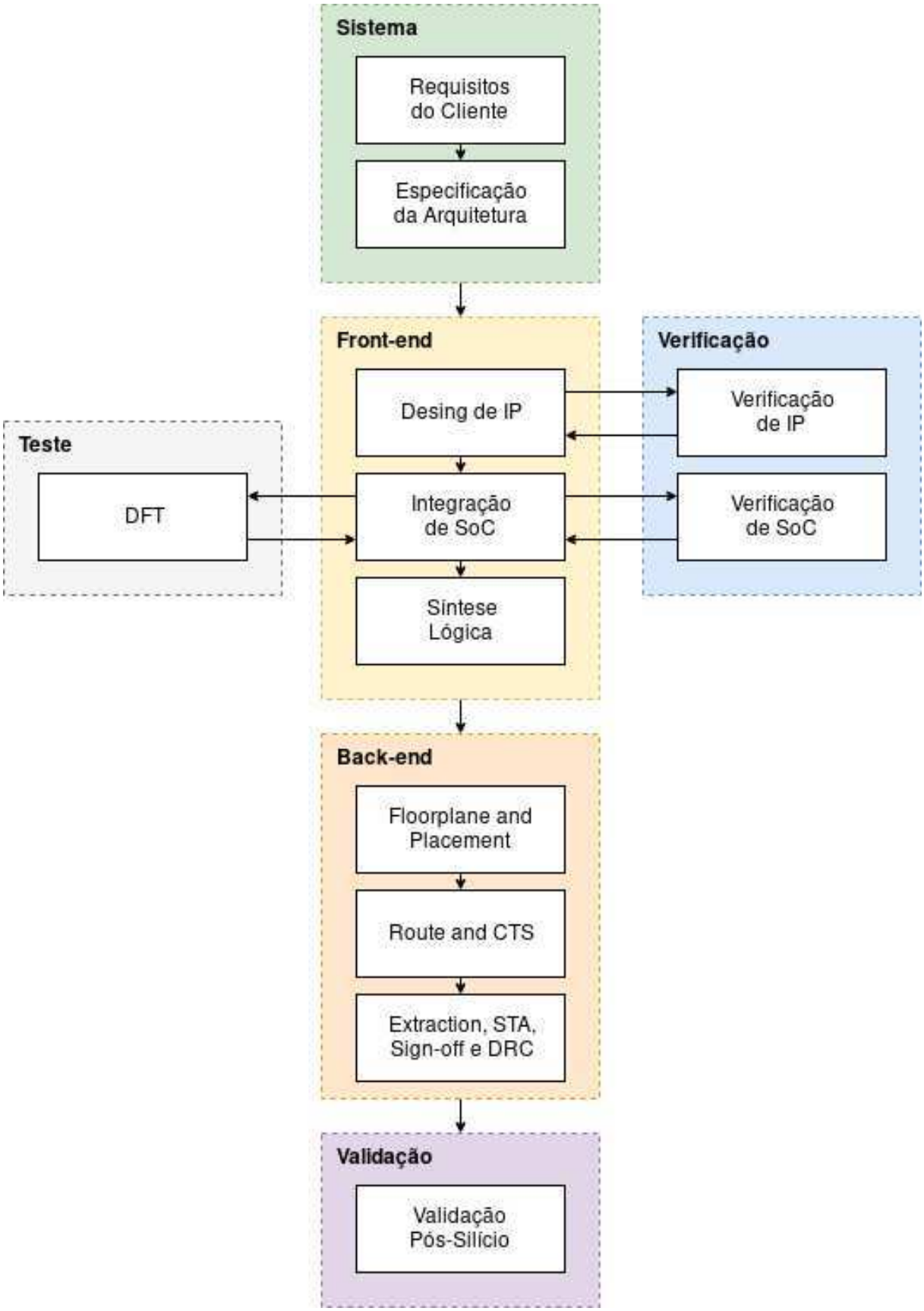


Figura 3 – Fluxo de projeto para a microeletrônica.

que será utilizado como base para o projeto dos IPs necessários, como também para a integração do sistema como um todo.

### 2.2.2 *Front-end*

O grupo de *front-end* trabalha como uma gama de responsabilidades, entre elas: entender especificações elétricas e algoritmos digitais; implementar soluções em RTL por meio de descrições em HDL; conhecer e integrar o SoC de modo a respeitar as especificações definidas pelo sistema e garantir o correto funcionamento de todos os blocos que o compõe; realizar a síntese lógica (tanto dos IPs como do SoC); e auxiliar em atividades posteriores que se relacionem com a síntese, como a extração e a STA (*Static Timing Analisys*).

O trabalho do engenheiro de *front-end* requer um vasto conhecimento de estruturas digitais, assim como um domínio de linguagens de descrição de hardware. No que tange as atividades de integração, espera-se que ele entenda o sistema, solucione problemas de conectividade, compreenda conceitos de *clock*, *reset* e estruturas básicas de um SoC e documento coerentemente as soluções implementadas. Além disso, essa atividade exige que o profissional tenha conhecimento do fluxo geral de uma síntese lógica e dos conceitos por trás de cada processo.

### 2.2.3 Teste

O grupo de teste é mais comumente denominado de DFT: *Design For Test* ou *Design For Testability*. Essa equipe opera em conjunto e paralelamente à praticamente todo o fluxo do projeto de um SoC - visto que sua função é desenvolver uma arquitetura coerente com a arquitetura proposta para ao SoC e que seja capaz de garantir o funcionamento das células digitais após a fabricação.

Para que isso seja possível, são inseridas lógicas de teste e cadeias de mux que permitem que o silício seja validado e um problema de fabricação não seja confundido com um erro de projeto.

### 2.2.4 *Back-end*

O grupo de *back-end* está diretamente associado com tarefas referentes ao projeto físico do SoC. Após a síntese, todas as funcionalidades implementadas em RTL foram sintetizadas e mapeadas para blocos lógicos básicos e podem ser posicionadas e interligadas em um silício real. Esse procedimento resume a etapa de *floorplan*, *placement* e *route*. Por se tratar de atividades com a lógicas física, é o grupo de *back-end* que projeta a árvore de relógio (ou *clock tree*) e realimenta as informações de atraso de roteamento, de carga e atributos da árvore de relógio para a equipe de *front-end* para que regras de tempo (*constraints*) possam ser verificadas e respeitadas.

Por tratar de todas as minúcias do projeto físico, o profissional de *back-end* deve conhecer muito bem o fluxo específico do seu trabalho e ter habilidades com muitas ferramentas de projeto. Além disso, deve entender conceitos elétricos fundamentais que regem a microeletrônica de forma que os resultados apresentados pelos relatórios das ferramentas sejam claramente compreendidos.

### 2.2.5 Verificação

O trabalho da verificação é realizar simulações e testes em nível de RTL ou *netlist* (RTL mapeado em blocos lógicos padrões de uma tecnologia) de forma que todas as funções propostas pelo bloco lógico (RTL ou SoC) estejam sendo executadas com perfeição.

Como o nível de complexidade entre um simples IP e uma coleção de IPs que formam um SoC é destoante, técnicas diferentes são aplicadas para cada caso de forma a atender melhor as necessidades de cada um. Portanto, é comum que existam dois grupos de verificação trabalhando simultaneamente: um dedicado à verificação funcional dos IPs e outro dedicado à verificação formal e geral do SoC.

### 2.2.6 Validação

O grupo de validação dedica-se à testar as funcionalidades do SoC de modo que atendam às especificações de projeto e, em última instância, os requisitos do cliente. Os engenheiros dessa área desenvolvem rotinas em linguagem de baixo nível que sejam capazes de escrever e ler nos registros de controle do SoC e estimulem as operações primárias do sistema embarcado.

## 2.3 *System on Chip* (SoC)

Um *System on Chip* pode ser entendido como uma combinação assertiva e refinada de *hardware* e *software* dedicada à atender uma demanda ou função específica. Por assim ser, os SoC são comumente mencionados na literatura como um sistema de soluções embarcadas e não podem ser confundidos com computadores genéricos na medida em que não possuem em seu projeto o intuito de produzir um sistema de cálculo generalista - como ilustra as diferenças a Tabela 1.

A arquitetura de um SoC pode variar bastante à depender das especificações para o qual ele está sendo desenvolvido. No entanto, é comum que seu projeto de hardware possua: um ou mais processadores; entradas e saídas (I/O) limitadas e específicas para sua função; periféricos dedicados; um sistema de gerenciamento em tempo real (*Real Time Operating System*, RTOS). Um exemplo de um sistema embarcado é o Emotion Engine utilizado no Playstation 2. Esse sistema possuía apenas duas funções: simulação comportamental e

| **Sistemas Embarcados** | **Computadores Generalistas** |
|---|---|
| Executam algumas aplicações já conhecidas durante o design. | Projetado para executar um conjunto geral de aplicações. |
| Não é programável pelo usuário final. | Programável pelo usuário final. |
| Possui limitações de performances valiosas para a aplicação. | Performance elevada. |

Tabela 1 – Diferenças relevantes entre sistemas embarcados e computadores de uso geral.

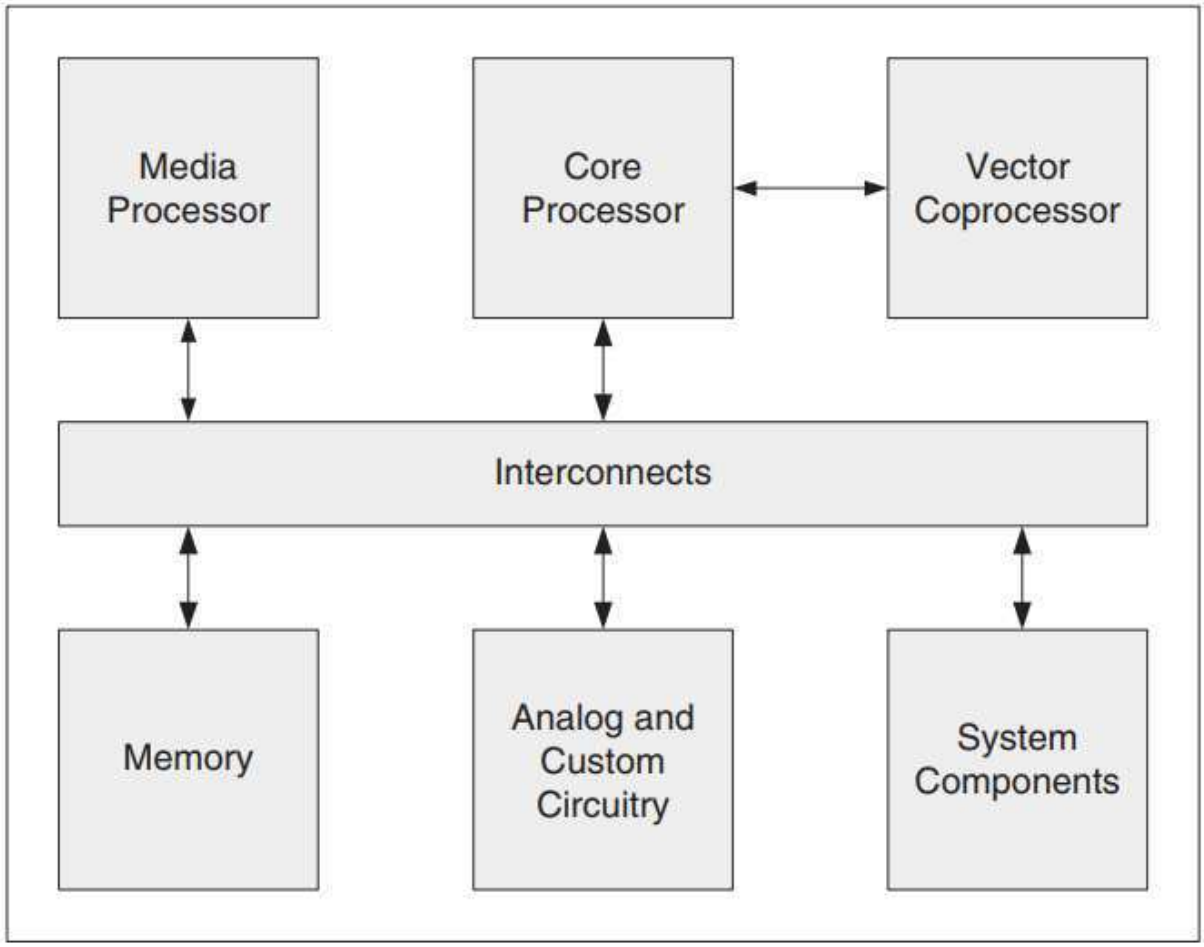


Figura 4 – Exemplo de SoC: arquitetura genérica do Emotion Engine.

interpretação geométrica, tornando-o uma solução embarcada extremamente especializada. A Figura 4 apresenta um diagrama lógico que sumariza genericamente a arquitetura do Emotion Engine.

Ademais, é ainda importante visualizar o projeto do SoC como um esforço conjunto da equipe de *hardware* e *software*, visto que todo o sistema de projeto em silício dedica-se diretamente ás aplicações que serão executadas em seu interior.

## 2.4 Linguagem de Descrição de Hardware: Verilog

O Verilog é uma Linguagem de Descrição de Hardware (Hardware Description Language, HDL) publicada pela Cadence em 1990 e adotada como padrão pelo IEEE em 1995. Como linguagem de descrição de hardware, o Verilog é capaz de descrever estruturalmente e comportamentalmente um circuito digital em nível de transferência de registro (*Register Transfer Level*, RTL).

O Verilog é inteiramente baseado em duas construções semânticas que se responsabilizam pela reprodução dos comportamentos combinacionais e sequenciais almejados em uma lógica digital. A primeira dessas construções trata-se da atribuição contínua, na qual fios são constantemente atualizados com valores de variáveis ou outros fios. Por outro lado, a atribuição procedural tem caráter mais comportamental na medida em que uma lista de sensibilidade é utilizada para indicar quando as atribuições lógicas internas à esse bloco devem ser realizadas. Em geral, as atribuições procedurais são utilizadas principalmente na descrição de lógicas sequenciais, podendo também ser utilizada para circuitos combinacionais.

Durante o design utilizando Verilog, as semânticas supracitadas devem ser encapsuladas em um módulo. O módulo pode ser entendido como uma caixa-preta que possui uma interface de entrada e saída e desempenha uma determinada função - a qual é descrita pelo seu conjunto de instruções lógicas internas. A Figura 5 ilustra os conceitos módulo e como eles são traduzidos para HDL.

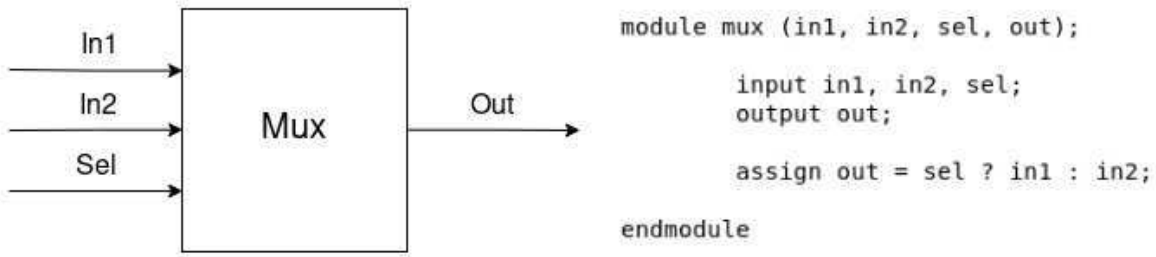


Figura 5 – Exemplo de design em Verilog.

Por fim, além de descrever com primazia circuitos eletrônicos, o Verilog apresenta suporte para todas as etapas do fluxo da microeletrônica, incluindo: sintaxes para ambientes de simulação e verificação, análise de *timing* e síntese.

# 3 Atividades Desenvolvidas

Essa seção tem como objetivo apresentar e detalhar todas as atividades desenvolvidas durante todos os meses de estágio na NXP Semiconductors. O planejamento geral do programa de estágio previa como ementa a realização das seguintes atividades no decorrer do ano completo:

1. Realizar atualização de sistemas de programas;
2. Auxiliar na elaboração de especificações sob supervisão do engenheiro responsável;
3. Auxiliar no suporte das ferramentas utilizadas;
4. Avaliar software e hardware já existentes.

Na primeira parte do estágio, foram realizados treinamentos e atividades na parte de integração de SoC; na segunda etapa, as atividades se concentraram no fluxo de síntese lógica (*constraints*) e suporte à verificação dos módulos integrados. As atividades serão apresentadas e pormenorizadas nos tópicos que completam esse capítulo.

## 3.1 Treinamentos

As primeiras atividades consistiram da realização de uma bateria de treinamentos diários por um período de aproximadamente 4 semanas. Os treinamentos foram divididos em 2 gêneros principais: treinamento em microeletrônica e em computação; e capacitação em ferramentas internas. Os tópicos a seguir detalham o escopo de cada um dos treinamentos fornecidos pela NXP.

### 3.1.1 Introdução ao fluxo da microeletrônica

Para contextualizar melhor os estagiários dentro do fluxo da microeletrônica, foi realizado um conjunto de treinamentos com o intuito de apresentar as atividades desenvolvidas por cada uma das equipes responsáveis pelo fluxo completo de um SoC.

a) **Design Flow:** teve duração de 2h e foi conduzido pelo chefe da divisão de SoC da empresa. Nesse treinamento, foram apresentados todos os passos necessários para o desenvolvimento de um SoC: iniciando-se nas especificações do cliente até à etapa de *tape-out* e validação. Cada etapa foi pincelada com comentários sobre estratégias adotadas pela empresa e detalhes conceituais importantes para a formação de um profissional da área.

b) **SoC Verification Overview:** treinamento de 2h liderado por um gerente da área de verificação. Nesse curso, obteve-se uma visão geral da importâncias da verificação no fluxo da microeletrônica e do processo de verificação de SoC adotado pela empresa.

c) **DFT:** teve duração de 2h e também foi conduzida por um gerente da área de DFT (*Design For Test/Testability*). Nele, foram apresentados os conceitos e a importância da DFT no desenvolvimento adequado de circuitos lógicos de alta complexidade (SoCs), além das técnicas mais comuns aplicadas na etapa de DFT.

As áreas de Analógico, Integração e Validação não tiveram treinamentos introdutórios dedicados à todos os estagiários. Parte deles disponibilizaram treinamentos exclusivos para estagiários membros da equipe em questão.

### 3.1.2 Verilog para simulação e síntese

O curso teve como objetivo apresentar o Verilog como uma ferramenta para a descrição de circuitos lógicos em nível de RTL (*Register Transfer Level*). Não obstante o curso tenha se focado nas estruturas sintetizáveis da linguagem, o instrutor fez questão de pontuar a diferenças e apresentar, na medida do possível, recursos da linguagem que seriam mais apropriados para um ambiente de simulação. Além disso, foram ensinadas técnicas de projeto relevantes para o desenvolvimento apropriado de um IP.

O treinamento em Verilog foi o curso com o maior número de horas e totalizou cerca de 20 horas de aulas teóricas e mais 5 á 10 horas de atividades práticas com o intuito de reforçar os conceitos e as técnicas apresentadas em sala de aula. A atividade final do treinamento consistiu na descrição de um receptor UART (UART-RX) em Verilog de forma que fosse possível sintetizá-lo. O projeto final permitiu que fossem exercitados conceitos de projeto e descrição com Verilog, assim como a elaboração de um ambiente básico de simulação.

### 3.1.3 Conceitos básicos para o projeto de SoCs

Nesses treinamentos, foram introduzidos alguns conceitos básicos do projeto de SoCs, sendo eles: (a) *clock* e *reset*; (b) protocolos de barramentos (AHB/AXI). O treinamento de *clock* e *reset* apresentou conceitos de circuitos síncronos e assíncronos, metaestabilidade, cruzamento de domínios de *clock* e sincronização - tudo sendo apresentado em 2 horas e meia de aula. O treinamento sobre barramentos introduziu a ideia de *handshake* e a estrutura básica de um protocolo de barramento, focando exaustivamente nos protocolos AHB e no AXI. Esse último treinamento teve duração de 3h.

### 3.1.4 Ferramentas Internas

Além dos treinamento mais focados em temas da microeletrônica, ocorreram uma série de cursos dedicados à apresentação das ferramentas internas da empresa que auxiliam no fluxo de projeto e são imprescindíveis para profissional da NXP. A seguir, serão listadas as ferramentas com uma breve explicação da sua função e um sumário da ementa do curso de cada uma delas.

a) **TR:** O TR trata-se de um configuração do sistema operacional do Linux que possibilita que ferramentas de microeletrônica possam ser utilizadas sem haja a necessidade que os arquivos estejam localmente instalados na máquina. Além disso, por meio do TR é possível configurar a versão de cada ferramenta que será utilizada em um determinado projeto, possibilitando sua padronização com o objetivo de evitar que ocorram incompatibilidade de versões entre os participantes durante o fluxo do projeto.

b) **Stingray:** treinamento de 2h liderado por um gerente da área de verificação. Nesse curso, focou-se na apresentação do *Stringray* que nada mais é do que uma ferramenta que encapsula *softwares* do fluxo da microeletrônica de forma a facilitar sua utilização.

c) **Controle de versão:** foram realizados um conjunto de treinamentos com o intuito de apresentar todas as ferramentas responsáveis pelo controle de versão interno na NXP. Esse treinamentos totalizaram 6h e foram divididas em 3 partes - ferramenta de banco de dados da empresa, instrumento de gerenciamento do banco de dados e ferramenta de controle de versão. Esses cursos foram essenciais para que fosse possível trabalhar no fluxo de produção da NXP.

d) **Shell script:** treinamento curto de apenas 2h com objetivo de apresentar os princípios de programação em shell e a importância dessa estrutura na automação e aceleração de procedimentos.

e) **VDI e processamento remoto:** apesar de o sistema operacional padrão das máquinas internas da NXP ser Windows, as ferramentas do fluxo da microeletrônica são direcionadas para o Linux. Assim sendo, a maior parte do trabalho é realizado utilizando uma máquina virtual que fica hospedada em um dos centros de maior porte da empresa. O repasse dessas informações e das instruções de uso da máquina virtual foi realizado em 2 horas. Ademais, foi realizado mais um treinamento de 2h para apresentação do processamento remoto que permite que processos mais demorados e que exijam mais processamento do que o disponível na sua máquina virtual sejam canalizados para um servidor de maior capacidade.

## 3.2 Integração de SoC

Antes que se iniciassem as atividades, o gerente da equipe de integração, Marcos Barros, indiciou um mentor para acompanhar o desenvolvimento do estagiário, repassando tarefas e estabelecendo prazos, assim como esclarecendo dúvidas e compartilhando a experiência e o conhecimento. Para esse estágio, Guilherme Coraucci foi encarregado de tutorar o aluno, Kaio Nikelisson de Lima Fernandes, durante a primeira etapa de treinamento.

Coraucci foi alocado como chefe da equipe de síntese da integração de SoC e, para facilitar a distribuição de atividades e o processo de treinamento, apresentou proposta de treinamento (e objetivos almejados) para o ano completo de estágio - que pode ser consultada na Tabela 2.

Embora a equipe no qual o aluno foi alocado tenha sido de síntese, grande parte das atividades inicias desempenhadas foram dedicadas à auxiliar no processo de integração a qual é caracterizada pela interligação dos blocos individuais (IPs) com os demais módulos e estruturas da arquitetura do SoC. Alguns periféricos que eram responsabilidade do mentor foram transferidos para estagiário na medida em que o próprio fosse capaz de desempenhar as tarefas sem atrasar o fluxo. Estão entre os IPs atribuídos ao estagiário: USB, FlexCAN, LPSPI (Low Power SPI), *semaphore*, entre outros.

A metodologia de integração baseia-se em: entender o macro-funcionamento do módulo do qual se estar integrando, ler as especificações e pretensões dos arquitetos no *Architecture Definition Document* (ADD) e, quando disponibilizado, ler o guia do bloco (*block guide*) e o guia de integração (*integration guide*) do módulo em questão; e realizar a instanciação, parametrização e interconexão das entradas do IP. As saídas, quando não realimentadas, são entradas de outros módulos e, portanto, responsabilidade de outros profissionais. Ainda assim, as saídas dos blocos de sua responsabilidade devem ser monitoradas e, quando necessário, conectadas.

A atividade de integração é auxiliada por meio de uma ferramenta interna que condensa as funcionalidades de diversos *softwares* de terceiros e semi-automatiza o processo - sendo necessário apenas o conhecimento das conexões do bloco e do uso da ferramenta de integração. Não obstante grande parte das conexões tenham um padrão pré-definido (barramentos, *resets* e *clocks*), algumas não se adéquam perfeitamente à arquitetura e requerem o projeto de lógicas intermediárias que são comumente reconhecidas como *glue logics* ou *integration logics*.

As *glue logics* são lógicas que intermedeiam as conexões entre IPs e possibilitam o reuso constante sem que haja a necessidade de pequenas modificações conforme a necessidade do projeto. Além disso, essas lógicas podem variar sua complexidade a depender da funcionalidade que se deseja obter e de como as interfaces dos módulos interconectados

| **Plano de Atividades** | |
|---|---|
| **Síntese** | |
| **Atividades** | - Auxiliar na geração de *constraints*;<br>- Executar a síntese de uma partição;<br>- Executar a síntese do topo;<br>- Explorar o arquivo *log*; |
| **Objetivos** | - Entender sobre o que se trata a síntese;<br>- Aprender e se familiarizar com o fluxo de síntese;<br>- Compreender o que são *constraints*;<br>- Assimilar conceitos técnicos: elétricos e *timing*;<br>- Conhecer as bibliotecas e como elas são utilizadas na síntese;<br>- Instruir-se sobre quais as entradas e saídas do trabalho;<br>- Desenvolver boas práticas de design. |
| **Expectativas** | - Ser capaz de identificar e consertar problemas durante a síntese;<br>- Ser capaz de auxiliar a equipe de síntese em qualquer critério;<br>- Obter conhecimentos robustos em *timing*;<br>- Obter conhecimentos robustos no fluxo de síntese. |
| **Integração** | |
| **Atividades** | - Auxiliar na integração do USB, FlexCAN, LPSPI e demais blocos;<br>- Utilizar ferramenta de integração;<br>- Escrever o Guia de SoC para os IPs integrados;<br>- Ler ADD e os Guias de Bloco e Integração de cada IP;<br>- Usar o sistema de gerenciamento de *tickets*. |
| **Objetivos** | - Desenvolver conhecimentos sobre sistemas e arquiteturas;<br>- Entender como especificações são traduzidas em *hardwares*;<br>- Cultivar espírito colaborativo ajudando e sendo ajudado;<br>- Desenvolver habilidades interpessoais (comunicação e apresentação);<br>- Instruir-se sobre quais as entradas e saídas do trabalho;<br>- Desenvolver boas práticas de design. |
| **Expectativas** | - Conhecimento considerável em arquitetura de microcontroladores;<br>- Capacidade de manipular a ferramenta de integração;<br>- Capacidade de gerenciar e atender *tickets*;<br>- Desenvolver comportamento proativo;<br>- Cultivar espírito de equipe;<br>- Criar e evoluir senso de gerenciamento de atividades; |

Tabela 2 – Planejamento inicial de atividades para o estagiário.

devem conversar entre si. As lógicas mais comuns e simples são inversores e sincronizadores. Contudo, é recorrente a necessidade de se implementar estruturas lógicas mais complexas. A Figura 6 ilustra o formato generalista de uma *integration logic* por meio de uma diagrama lógico.

No decorrer do estágio, verificou-se a necessidade da concepção e implementação de uma variedade de lógicas intermediárias de integração que atendessem as necessidades de interconexão dos IPs sobre responsabilidade do estagiário. A lista a seguir identifica e discorre sobre algumas dessas *glue logics*.

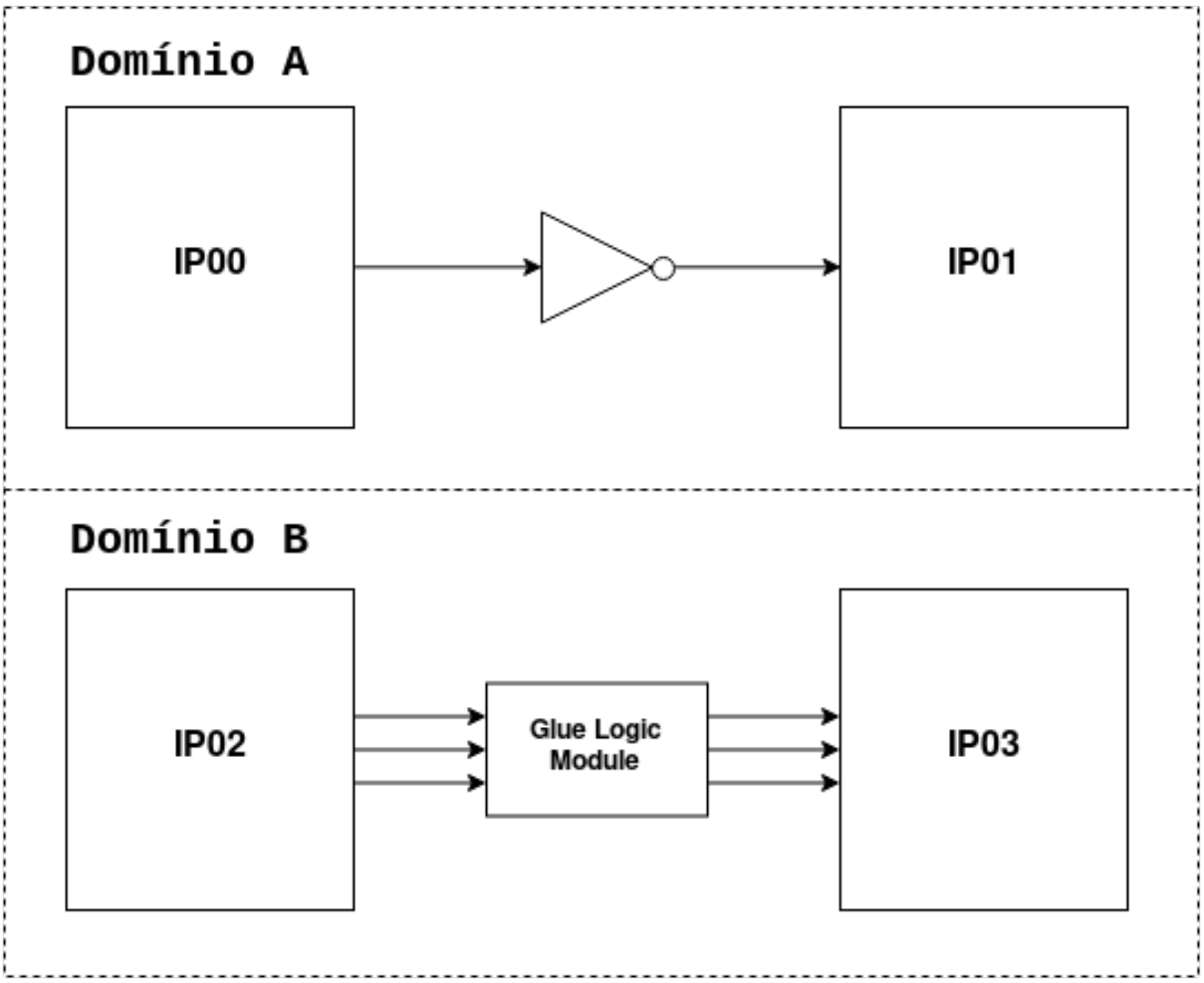


Figura 6 – Representação generalista das lógicas de integração (*glue logics*).

- Inversor: uma das lógicas mais recorrentes se tratava de uma simples inversão de sinal para a polaridade de um determinado sinal casasse com as especificações do IP no qual se desejava conectar.
- Lógicas primitivas: em algumas situações, um determinada informação estava contida em mais de um sinal e era resultado de uma operação lógica entre eles. Assim sendo, portas básicas como AND e OR eram constantemente reutilizadas para tornar coerente a comunicação entre os módulos.
- Sincronizador: em SoC com grande dimensões, é comum haver cruzamento de domínios de *clock* que precisam ser sincronizados. Além disso, os sincronizadores são módulos extremamente utilizados na sincronização de sinais de *reset* para impedir uma possível metaestabilidade.
- Arranjo/Decodificador de memórias: em determinadas situações faz-se necessário a implementação de lógicas de decodificação de memórias devido a estruturação dos blocos disponíveis para instanciação que agregam melhores resultados após a etapa

de *placement*. Em sistemas que utilizam compartilhamento de recursos, essas lógicas revelam-se demasiadamente complicadas e exigem um projeto cuidadoso da lógica que viabiliza o compartilhamento.

Além dos módulos listados acima, existia também a necessidade da criação de blocos lógicos com funcionalidades específicas, à exemplo de temporizadores que atendessem requisitos especiais e lógicas de controle dedicadas à administração de determinados aspectos operacionais do SoC.

Ao praticamente finalizar os módulos e as responsabilidades que lhes foram atribuídas inicialmente, o estudante foi realocado para ajudar na integração da plataforma - sistema de barramentos, controladores e processadores que compõem o SoC. Em tal função, trabalhou na integração de sinais específicos de barramento, assim como na produção de um sumário de conexões ainda não realizadas para que elas fossem devidamente atribuídas ou, quando fosse o caso, estudadas e prontamente conectadas.

Por fim, outra tarefa importante que foi realizada em paralelo ao projeto e a integração dos módulos mencionados diz respeito à documentação das atividades realizadas. Grande parte dessas informações deveriam ser registradas com detalhes em um documento interno intitulado *SoC Integration Guide*, ou Guia de Integração de SoC. Além de ser uma boa prática para consultas futuras em projetos diferentes, o arquivo servia de base para que outros engenheiros que também estivessem trabalhando na integração ou verificação pudessem obter informações relevantes de forma prática e direta.

## 3.3 Síntese

De início, o estagiário apenas participou das reuniões da equipe na fase de planejamento e acompanhou as discussões dos membros enquanto o ambiente estava sendo projetado e preparado para que as demais tarefas de configuração de *debbug* fossem realizadas.

Além disso, o tutor e líder da equipe de síntese, Coraucci, indicou livros e materiais de estudo para que o estagiário fosse se preparando na medida do possível para as atividades que seriam de sua responsabilidade assim que as prioridades de integração fossem finalizadas e o ambiente de síntese que estava sendo preparado se tornasse sólido o suficiente.

Utilizando os serviços disponibilizados pela NXP, o estagiário solicitou um curso de síntese da Cadence focado na utilização de sua ferramenta (Genus) para gerenciar e executar todo o fluxo da síntese. Para além dos comandos básicos e treinamentos utilizando o Genus, o curso também disponibilizava uma seção integral de síntese dedicada à sistemas de baixa consumo de potência (*low power*).

Assim que foram finalizadas as tarefas de integração e o ambiente de síntese estava rodando sem problemas visíveis, o estagiário foi alocado para o geração e verificação dos *clocks* de sistema. Aqui, declarou os relógios necessários e identificou problemas de arquitetura, reportando-os com propostas de solução.

Encerrada a primeira tarefa real de síntese, o estagiário foi alocado para desenvolvimento de *constraints* de IO em módulos de alto desempenho. Para tal, importou *constraints* de projetos passados e os adaptou para o projeto atual, removendo *clocks* e caminhos falsos desnecessários e incluindo e atualizando especificações de AC de *timing*. Essa segunda atividade foi essencial para que o pupilo adquirisse conhecimento da ferramenta e dos conceitos teóricos por trás dos arcos de temporização. Assim sendo, adquiriu experiência em técnicas para identificar e eliminar caminhos falsos e caminhos que precisariam de atenção especial no processo de síntese.

Adquirida a maturidade necessária, foi solicitado que o estagiário auxiliasse na declaração de *clocks* para mais módulos. O procedimento, apesar de semelhante, exigia reconhecimento e aprofundamento no que diz respeito aos caminhos de *clock* presentes em cada módulo individualmente. Tal atividade mostrou-se fundamental na formação inicial da árvore de relógio, além de permitir que os *clocks* com as frequências corretas alcançassem os registrados adequadamente - ajustando e limpando os arcos de temporização reais.

Por fim, o estagiário adquiriu experiência o suficiente para que fosse possível solicitar a sua ajuda na adaptação do fluxo de síntese quando o projeto migrava para uma nova *build* - marco importante de integração. Em paralelo com as atividades supracitadas, essa foi a última atividade desenvolvida pelo estagiário na NXP.

## 3.4 Reuniões

Como parte do time de integração e síntese, o estagiário participou de reuniões semanais que objetivavam o repasse do andamento do projeto e a discussão de contratempos e adversidades que pudessem vir a afetar o planejamento.

Mesmo que houvesse um pouco de interseção entre os componentes das equipes e as atividades de síntese e integração, as reunião dos times eram mantidas separadas para que os fóruns fossem dedicados aos seus devidos tópicos. A reunião de integração ocorria sempre às 9:30h das terças-feiras. Nela, discutiam-se o andamento da integração dos módulos e os principais problemas que deveriam ser ultrapassados ou que poderiam vir à atrapalhar o andamento do projeto. Nesses encontros, o mentor solicitou participação ativa do estudante quando a pauta da reunião tangenciasse módulos ou atividades de sua responsabilidade - providenciando um crescimento interpessoal considerável.

Em relação ao time de síntese, as reuniões eram semanais, duravam cerca de uma

hora, mas não possuíam horário fixo. Por ser uma atividade que dependia diretamente das definições e do andamento do processo de integração do SoC, as reuniões de síntese tiveram um início mais tardio. Nessas conferências, o estudante mais ouviu e se familiarizou com os problemas enfrentados no planejamento e no fluxo de síntese, assim como com os procedimentos adotados para a realização da atividade. Ao fim dos primeiros meses, quando o estagiário estava finalizando suas tarefas de integração, foi realocado para trabalhar diretamente com a síntese e pode participar mais ativamente das reuniões supracitadas.

# 4 Conclusão

Durante o programa de estágio, pode-se inferir que as atividades ocorreram como planejado e apresentaram resultados extremamente satisfatórios.

Dentro do ambiente de trabalho da NXP, o estudante teve a possibilidade de amadurecer os conhecimentos adquiridos durante a graduação, aplicando-os ativamente em um projeto de SoC de grande porte e complexidade. Além disso, a oportunidade de trabalhar diretamente com um projeto de real impacto industrial e social garantiu ao estagiário um poderoso arsenal de informações e detalhes da microeletrônica que não poderiam ter sido adquiridos somente com sua formação acadêmica.

Ademais, não foram desenvolvidas apenas competências técnicas. O convívio profissional diário e os desafios propostos permitiram o aprimoramento de atributos interpessoais, como: proatividade, trabalho em equipe, propósito, confiança, habilidades orais, gentileza e outras características sociais significativas para um bom convívio corporativo.

Não obstante o elevado grau de aprendizado obtido nos primeiros 6 meses do programa, esse tempo é insuficiente para a formação completa do profissional que trabalha com o desenvolvimento de tecnologia de ponta por meio da microeletrônica. Esse cenário torna-se ainda mais claro quando se percebe que as atividades planejadas não foram completamente finalizadas e que o encerramento do programa nos primeiros seis meses impossibilitaria o estudante a participar do fluxo completo de desenvolvimento de um SoC.

Assim, apesar do imprevisto e prejuízos, o estudante pode finalizar seu estágio de 1 ano completo atuando mais ativamente no fluxo de síntese e adquirindo habilidades essenciais para um bom profissional de *front-end*.

# Referências Bibliográficas

ARORA, M. **Embedded Sytem Design: Introduction to SoC System Architecture**. Austin: Learning Bytes Publishing, 2016.

FLYNN, M. J; LUK, W. **Computer system design: system-on-chip**. John Wiley & Sons, 2011.

INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS. **IEEE Standard for Verilog Hardware Description Language**. IEEE Computer Society, 2006.

KHAN, G. N. **Introduction to System-on-Chip**. Ryerson University.

NXP. **NXP no Brasil**. https://www.nxp.com/about/about-nxp/about-nxp/worldwide-locations/nxp-no-brasil:BRAZIL>. Acesso em: 24 de maio de 2019.

NXP. **NXP Corporate Overview**. Disponível em: <https://www.nxp.com/docs/en/supporting-information/NXP-CORPORATE-OVERVIEW.pdf>. Acesso em: 23 de maio de 2019.

WESTE, N. H. E.; HARRIS, D. M. **CMOS VLSI Design: A Circuits and Systems Perspective**. 4. ed. Addison Wesley, 2010.