Lucas Oliveira de Figueiredo

Relatório de Estágio:
NXP Semicondutores Brasil LTDA

Campina Grande, PB

2020

Lucas Oliveira de Figueiredo

Relatório de Estágio:
NXP Semicondutores Brasil LTDA

Relatório de estágio supervisionado 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: Rafael Bezerra Correia Lima

Campina Grande, PB

2020

Lucas Oliveira de Figueiredo

Relatório de Estágio:
NXP Semicondutores Brasil LTDA

Relatório de estágio supervisionado 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, _____/_____/_____.

Rafael Bezerra Correia Lima
Orientador

George Acioli Júnior
Convidado

Campina Grande, PB
2020

Este trabalho é dedicado a minha família e aos amigos ao longo do caminho que salvaram o curso.

Agradecimentos

Agradeço aos meus pais, Solange Oliveira e Pelágio Figueiredo, pela dedicação e sacrifícios para a minha formação. Por me instigarem a desenvolver ética e interesse no que faço na vida. Eu não poderia ter pedido por professores, amigos e parceiros no crime melhores que vocês dois.

Às minhas avós Salete e Margarida pelo imediato sorriso ao me ver e pelas histórias de outrora. Obrigado Edna pelo cuidado durante minha infância e pelos almoços às 10h da manhã. Agradeço aos meus tios e tias Etelânio, Katharine, Soleide, Sílvio, Soraya, Vieira, Fabiano e Nice pelo apoio sempre presente nessa nossa pequena família. À Cavalcante (Azi) e Demétrio por serem quase irmãos para mim, prontos para orientar e dar risadas, e que nossa amizade permeie por anos a vir. Obrigado meus primos e primas Ingrid, Luana, Leonardo e Leopoldo por enfrentarem os desafios de uma nova geração comigo.

Obrigado Adeilmo Júnior, que em sua carismática arrogância me fez muitos favores e melhorou minhas *playlists*. Sou grato aos meus amigos da Elétrica: Ravi, Porto, Danrley, Torres e Egydio, sem os quais a graduação teria sua dificuldade duplicada e minhas memórias seriam mais monótonas, porém talvez menos embaçadas. Aos amigos de Pau dos Ferros: Felipe, Rômulo e Nikelisson, por compartilharem interesses, vocações e um ombro amigo para o que a vida jogou na nossa frente. Obrigado Matheus Cavalcante, Brenda, Lucas Luna e William por dividirem cargas de trabalho e reflexões no curso. Também obrigado aos muitos outros colegas da Universidade da Criança, IFRN e UFCG que tiveram participação na minha história.

Ao LIEC, laboratório que me recebeu de portas abertas e me proporcionou uma segunda casa na universidade, como também experiências que agregaram a minha vida e ao meu currículo. Obrigado aos professores Rafael Bezerra, Péricles Barros e George Acioli em sua confiança e orientações. E aos colegas de laboratório Fabrícia, Marinho e Vinícius pelas boas memórias na sala 119.

À NXP Semicondutores, pela vaga de estágio e pela experiência profissional na engenharia. Obrigado a Clovis Lordello e Jacqueline Beray pela paciência e confiança, à Jeisson pelos conselhos, respostas, histórias e caronas ao longo deste ano. Obrigado Mario por ser o contato interno em verificação. E ao time de validação: Filipe, Felipe, Davi, Rocco, Hugo, Rodolfo, Claudionor e Tiago, pelos casos de teste resolvidos e risadas ao longo do caminho.

Por fim, agradeço ao professor Gutemberg, a Adail e Tchai da coordenação de engenharia elétrica, sem os quais este estágio não teria nem o plano aprovado.

"I wish it need not have happened in my time," said Frodo.
"So do I," said Gandalf, "and so do all who live to see such times. But that is not for them to decide. All we have to decide is what to do with the time that is given us."
J.R.R. Tolkien, The Fellowship of the Ring.

Lista de ilustrações

Lista de abreviaturas e siglas

BSTC	*Brazil Semiconductor Technology Center*
SoC	*System on Chip*
VLSI	*Very Large-Scale Integrated Circuits*
ASIC	*Application Specific Integrated Circuit*
IP	*Internal Peripheral*
RTL	*Register Transfer Level*
HDL	*Hardware Description Language*
FPGA	*Field Programmable Gate Array*
DFT	*Design for Test*
ROM	*Read-Only Memory*
RAM	*Random Access Memory*
DMA	*Direct Memory Access*
IDE	*Integrated Development Environment*
EDA	*Electronic Design Automation*
VDI	*Virtual Desktop Infraestructure*
VPN	*Virtual Private Network*
GIC	*Generic Interrupt Controller*
I²C	*Inter-Integrated Circuit*
I3C	*Improved Inter-Integrated Circuit*
EVB	*Evaluation Board*
SDHC	*Secure Digital Card Host Controller*

Sumário

1 Introdução

O presente relatório de estágio apresenta o desenvolvido pelo aluno de Engenharia Elétrica da Universidade Federal de Campina Grande (UFCG), Lucas Oliveira de Figueiredo. As atividades ocorreram na sede brasileira da empresa *NXP Semiconductors*, o *Brazil Semiconductor Technology Center* (BSTC), durante o período de tempo compreendido entre 06 de Janeiro de 2020 a 30 de Novembro de 2020.

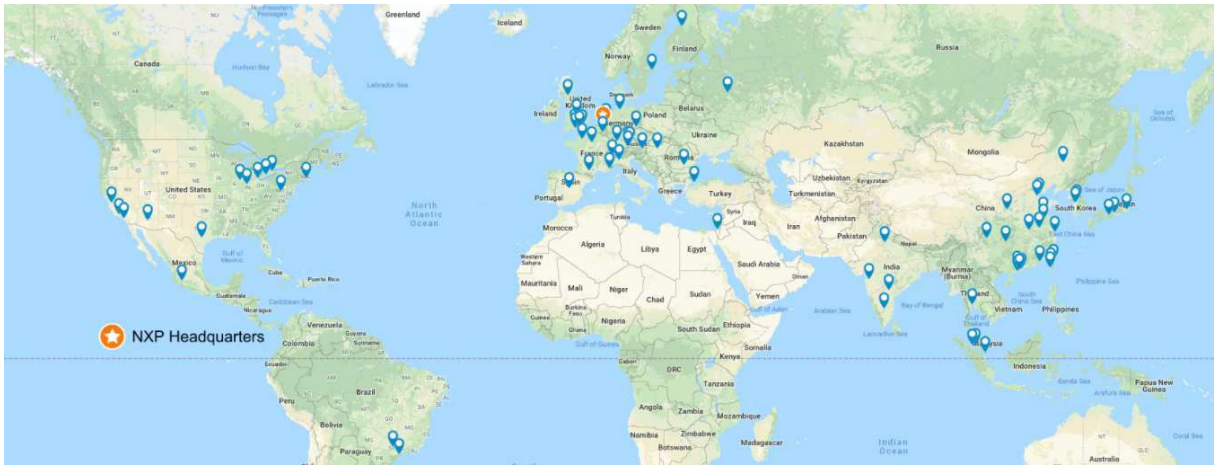

Figura 1 – Presença mundial da *NXP Semiconductors* [1].

A NXP Semiconductors, empresa com sede em Eindhoven nos Países Baixos, possui mais de 60 anos de experiência na criação de soluções que permitem conexões seguras e infraestrutura para um mundo mais inteligente. Estando presente em mais de 30 países e com aproximadamente 29.000 funcionários no mundo, sua sede no Brasil, o BSTC, está localizada no parque empresarial Techno Park em Campinas, São Paulo [2].

Em 1967 tiveram início atividades de desenvolvimento tecnológico da Motorola Inc. no Brasil, e em 2004 a Motorola Inc. tornou independente sua divisão de semicondutores, dando origem a empresa *Freescale Semiconductors*, já situada no BSTC. A NXP Semiconductors iniciou-se em 1953 como a divisão de semicondutores da Philips e foi formalizada em 2006 como uma empresa independente. A NXP realizou a aquisição comercial da Freescale no ano de 2015, culminando na *NXP Semiconductors* de hoje. A sede brasileira possui projetos entregues incluindo dezenas de microcontroladores, propriedades intelectuais chave em áreas como gerenciamento de potência, redes automotivas, processamento digital de sinais e aceleradores criptográficos e de temporização.

Figura 2 – Sede da NXP em Campinas, São Paulo (Fonte: o autor, 2020)

1.1 Plano de estágio

As atividades previstas para execução no plano de estágio e cumpridas ao longo dos meses na NXP foram:

- Participar do time de validação funcional, interagindo com times multifuncionais globais;
- Análise de requisitos e especificações técnicas para contribuir no planejamento da validação;
- Desenvolver o ambiente de validação, incluindo infraestrutura e software para estimular e checar as partes do circuito sendo avaliadas;
- Garantir que os protótipos funcionam de acordo com as especificações, rodando software em placas de desenvolvimento, e fazendo medições com equipamento de laboratório;
- Depurar falhas e reportar defeitos e os resultados da validação.

2 Referencial Teórico

Sistemas embarcados são uma combinação de hardware e software, embutidos (ou embarcados) em um sistema maior, com o propósito de interagir com o ambiente exterior [3]. Dispositivos computadores, como microcontroladores, circuitos para eletrônica de potência ou conversores analógico-digitais, por exemplo, quando integrados em um único hardware e programados com instruções de software definem, por fim, um sistema embarcado, que pode atuar em automóveis, eletrodomésticos ou aparelhos de celular, como ilustrado na Figura 3. Os microcontroladores presentes em tais sistemas são uma agregado de componentes menores, como processadores, memórias e componentes de entrada/saída, desenhados para aplicações já conhecidas e com limitações de performance em contraste com computadores pessoais, enquanto que SoCs (*System on Chip*), possuem definição semelhante aos microcontroladores, mas tendendo a integrar mais dispositivos de modo a ser uma solução dedicada.

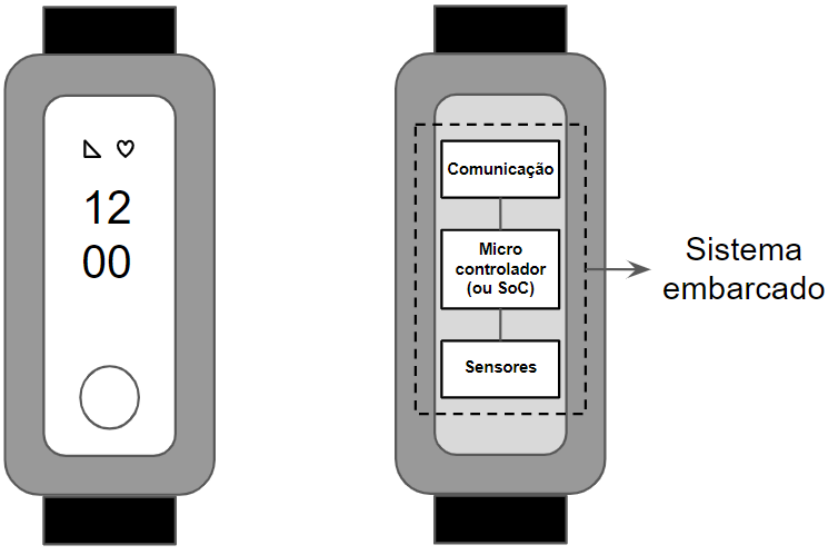

Figura 3 – Simples diagrama de blocos de um relógio inteligente (*smart watch*) (Fonte: o autor, 2020).

A microeletrônica é a área da engenharia que projeta e desenvolve, a partir de materiais semicondutores, circuitos integrados como os VLSI (*Very Large-Scale Integrated Circuits*), contendo milhões de transistores, e que tornaram viável a criação de produtos eletrônicos menores, seguros, mais rápidos e baratos do que o possível quando o número de transistores em um circuito era apenas na escala de dezenas de unidades [4]. O fluxo de projeto da microeletrônica descreve e regra desde a coleta de requisitos, o desenvolvimento de componentes até o controle de qualidade desses produtos de elevada complexidade e magnitude.

2.1 Fluxo do desenvolvimento de SoCs

O fluxo de projeto da microeletrônica, ou ASIC (*Application Specific Integrated Circuit*), utilizado para o desenvolvimento de SoCs, é um processo maduro para o design de produtos a serem impressos em silício [5]. Práticas testadas e aprovadas na indústria definem os métodos e etapas ilustrados de maneira simplificada na Figura 4.

Grupos de trabalhos distintos realizam as etapas do fluxo, que compreendem as atividades de:

Sistema: operações comerciais para a aquisição de projetos e contratos para o desenvolvimento de soluções por SoCs, com a listagem dos requisitos de modo que uma arquitetura do produto, com seus sistemas integrantes, possa ser cunhada e uma especificação gerada para nortear o desenvolvimento dos IPs (Internal Peripheral) necessários.

***Front-end*:** a implementação de soluções em RTL (*Register Transfer Level*) para os blocos componentes do SoC por meio de descrições em HDL (*Hardware Description Language*), como também a integração dos periféricos desenvolvidos, sua síntese lógica e a do sistema integrado. A emulação da solução em SoC é feita em FPGAs (*Field Programmable Gate Array*) de larga escala, permitindo que simulações em elevadas taxas de *clock* sejam feitas, porém com limitações na disponibilidade das funcionalidades do sistema, como ausência de componentes analógicos [6].

Teste: a inserção de lógica para testes com o propósito de identificar divergências da arquitetura original e a presença de defeitos de fabricação, compreende as atividades de DFT (*Design for Test*).

Back-end: o posicionamento e ligação dos blocos lógicos componentes do SoC em silício real. O projeto físico do SoC leva em conta a distribuição dos *clocks*, atrasos de roteamento e limitações de tempo.

Verificação: realização de simulações a nível de RTL ou *netlist* para a averiguar que as funções do bloco lógico viáveis de serem testadas nesta etapa, estão de fato presentes. A integração dos blocos lógicos componentes do SoC também é verificada, em virtude da funcionalidade conjunta de muitos deles.

A validação testa as funcionalidades do SoC a nível isolado de cada periférico e a integração dos blocos lógicos, com o propósito de verificar se as especificações do projeto e requisitos do cliente são atendidos. Com o artifício de dispositivos emuladores, a validação passou a ser também realizada em um estágio de pré-silício, paralela a verificação, porém com seus casos de teste compreendendo funções de mais alto nível quando comparados. Rotinas em linguagem de programação de baixo nível são feitas para interagir com os registradores do SoC e excitar desde operações simples de modificação de memória até

a execução de protocolos de comunicação. No estágio de pré-silício, uma suíte inicial de casos de teste é feita, já auxiliando na identificação de problemas no desenvolvimento, e reduzindo o tempo do estágio de pós-silício, onde uma placa de avaliação com o SoC é por fim testada em laboratório [7].

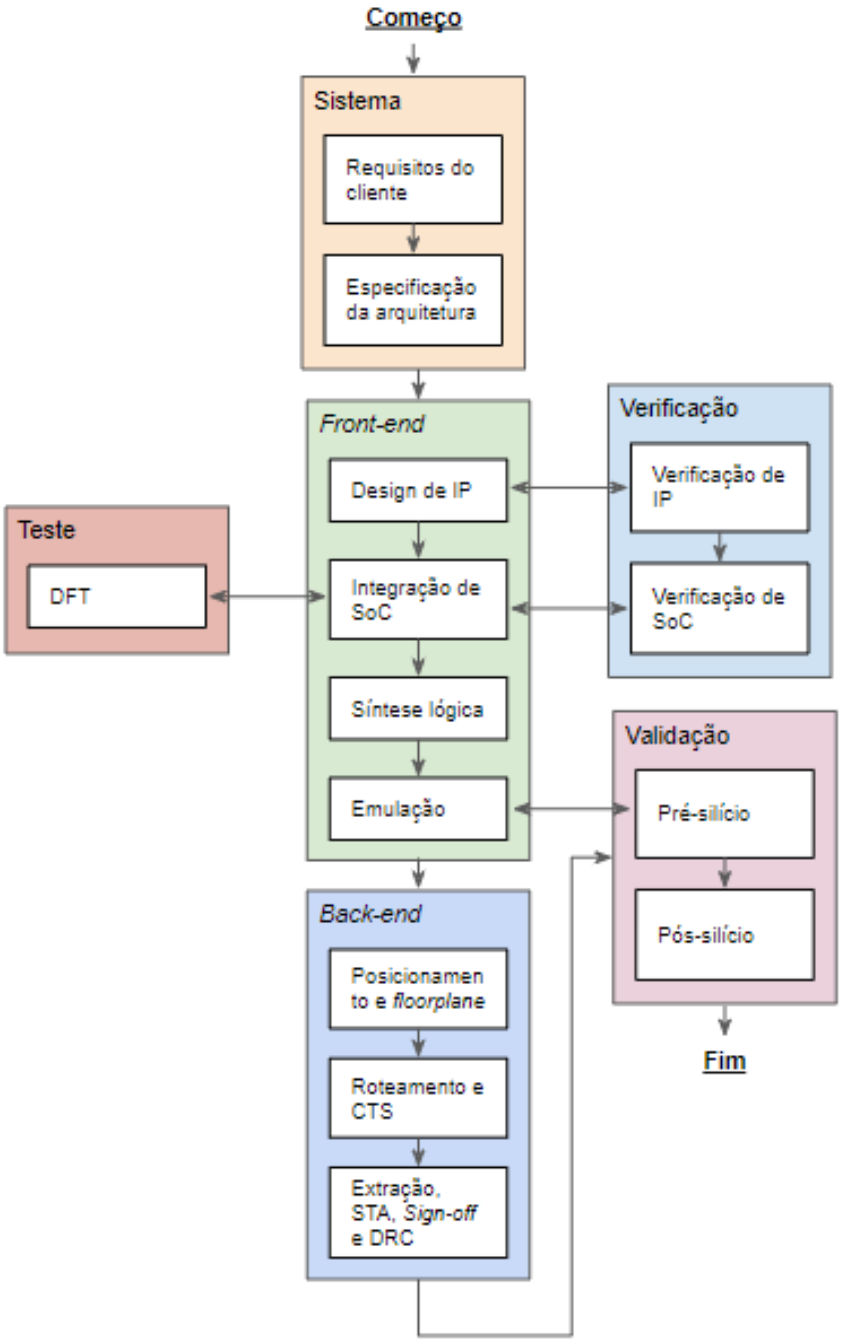

Figura 4 – Fluxo do desenvolvimento de SoCs (Fonte: o autor, com base no presente em [5]).

2.2 Programação de sistemas embarcados

Os processadores de uma solução em SoC executam um conjunto de instruções, denominado de *firmware*, armazenado em memória ROM (*Read-Only Memory*) lida quando o sistema é inicializado. A ativação ou não de partes do SoC é regida pelo conteúdo do *firmware*, com instruções capturadas da memória, decodificadas e executadas em sequência. As instruções suportadas são dependentes da arquitetura do processador, como por exemplo para um processador Arm Cortex-M33 e sua arquitetura Armv8-M.

A linguagem de programação C é uma linguagem de alto-nível no sentido de ser legível para o ser humano quando comparada com os zeros e uns da linguagem de máquina, porém a ausência de estruturas que permitem abstrações de tipos de dados, entre outras, tornam à de baixo-nível comparando-se com o C#, por exemplo. A tradução de um programa, ou código fonte, descrito em C para um formato interpretável pelo processador, é feita por várias ferramentas em um fluxo ilustrado na Figura 5 [3].

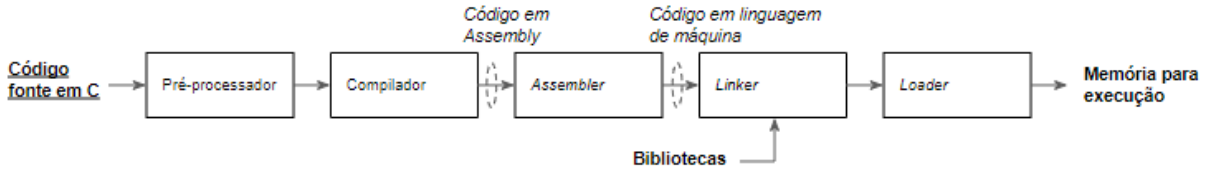

Figura 5 – Tradução de código fonte em C (Fonte: o autor, com base no presente em [3]).

O pré-processador prepara o código fonte ao ler diretivas de pré-processamento e gerar um arquivo final a ser compilado. O compilador traduz a linguagem C em linguagem *assembly*, a qual é dependente do processador alvo, em razão do conjunto de instruções suportado. O *assembler* converte o código *assembly* em linguagem de máquina com seus zeros e uns. O *linker* e o *loader* alocam as variáveis e dados atribuindo-as endereços, e neste estágio, o código resultante deverá ser alocado em uma memória ROM para que quando o sistema é iniciado, as instruções são movidas para uma RAM (*Random Access Memory*) e então os registradores do processador.

Em um nível mais alto de abstração para a programação de um SoC, sistemas operacionais podem ser utilizados em contraste com código *baremetal* por meio de *boot loaders*. Tais softwares realizam tarefas de, em primeira instância, configurar controladores de memórias e então carregar um sistema operacional a partir de múltiplas fontes, de modo que um usuário possa então controlar o processo de *boot* [8].

3 Atividades Desenvolvidas

Treinamentos em tarefas específicas da validação e em tópicos gerais do fluxo de desenvolvimento ocorreram na fase inicial do estágio. Nos meses subsequentes, a realização de atividades de pré-silício e pós-silício acompanhados de um supervisor compreenderam o previsto no plano de estágio.

O ambiente de trabalho incluiu uma baia com computador para funções de programação, laboratório com equipamentos para análise de circuitos elétricos e execução de testes em hardware, e sala de reuniões para encontros semanais de organização e atualização no projeto atual com o time. Nas segundas-feiras às 16:00h ocorriam as reuniões com o time por inteiro, embora a supervisão e interação com colegas foi presente ao longo de todo o período de estágio. A participação ativa do estudante nas reuniões foi incentivada pelo supervisor, reuniões estas que tratavam das etapas de pré-silício e pós-silício para os projetos atribuídos ao time brasileiro de validação. A determinação da propriedade (*ownership*) dos periféricos e funcionalidades a serem validadas eram distribuídas por conhecimento prévio e tempo disponível para realização, porém a orientação de se engajar com periféricos não trabalhados antes por um engenheiro era instigada. As prioridades definidas para as tarefas abertas e as escolhas de casos de teste com o propósito de abranger nas funcionalidades investigadas, permitiu uma familiarização rápida pelo estagiário com os obstáculos enfrentados na validação. Em seu ápice, o time continha dez integrantes, dentre eles um supervisor, um estagiário e oito engenheiros.

3.1 Treinamentos

Ao longo das primeiras quatro semanas ocorreram treinamentos com membros do time de validação e com engenheiros de outros times do fluxo de desenvolvimento. De caráter teórico com a motivação para as metodologias utilizadas, e prático com as ferramentas internas voltadas a validação, os treinamentos foram, respectivamente, ministrados a todos os estagiários no período, e por fim dedicado somente ao estudante, em virtude de ser o único nesta posição no time de validação.

3.1.1 Introdução ao fluxo da microeletrônica

A complexidade e escala de uma solução em SoC justifica a existência de grupos de trabalho específicos para cada etapa do fluxo de desenvolvimento. O conhecimento empírico por engenheiros de cada grupo é valioso para contextualização, e portanto, os treinamentos introdutórios tiveram um caráter generalista:

- **Fluxo de design:** ministrado pelo líder de verificação de SoC da sede, as etapas do desenvolvimento de SoC foram apresentadas com observações das metodologias e ferramentas em uso na empresa. O treinamento teve duração de 3h e apresentou conceitos importantes para que o dialogo entre os grupos de trabalho se tornasse mais uniforme.

- **Verificação de SoC:** ministrado por um gerente de verificação de SoC, o treinamento teve duração de 2h e discerniu sobre a importação da verificação de projetos de engenharia, em especial da microeletrônica, permitindo realizar ligações entre a motivação desta área e a de validação.

3.1.2 Fluxo de validação

O supervisor do time de validação, Clovis Lordello, indicou o mentor Jeisson Gutierrez para também auxiliar e acompanhar o desenvolvimento do estagiário. A apresentação e treinamento no fluxo de validação e ferramentas internas foram ministrados pelo mentor. Ao longo do primeiro mês de estágio, tarefas isoladas, compreendendo de horas até dias para conclusão, com o proposito de familiarizar o aluno foram atribuídas e as ferramentas correspondentes as tarefas foram apresentadas sob demanda. Periféricos e SoCs já validados foram utilizados para fins de ilustração e exercícios.

Com o propósito de averiguar as funcionalidades previstas para os periféricos de um SoC, a aquisição das especificações do projeto e dos periféricos componentes foi uma primeira instrução, com a apresentação do sistema de repositórios em nuvem dos documentos de projetos de SoC. O manual de referência é a coleção de tais informações, de modo que a identificação e seleção das funcionalidades de interesse para validação constitui uma etapa fundamental na construção do plano de validação por cada engenheiro.

O acesso as funcionalidades de um SoC a nível de validação, ou seja, por meio de *firmware* em linguagem C, se dá através de *drivers* para cada periférico e casos de teste para estímulos e visualização de resultados. A descrição dos registradores integrantes de um periférico e seus endereços em memória, presentes no manual de referência, são informações utilizadas para compor tal *driver*, de modo que bibliotecas de códigos fonte são construídas adicionando abstrações de nível mais alto para a geração de casos de teste. O periférico DMA (*Direct Memory Access*), o qual permite um acesso direto de escrita/leitura a memória sem a intervenção do processador, teve seu *driver* criado a partir do manual e funcionalidades estimuladas a partir de casos de teste codificados com o *driver* anteriormente definido.

O uso de uma IDE (*Integrated Development Environment*) para codificação, construção e depuração de rotinas em C para processadores alvo das fabricantes Arm e Cadence foi exercitado. A seleção da *toolchain* necessária e a conexão com o alvo, por meio de

debuggers virtuais e físicos, tornaram concreta a abordagem para carregamento de *firmware* e estímulos dos periféricos.

O sistemas de controle de versão e relato de defeitos, com seus detalhes para preservação da integridade dos códigos-fonte e bibliotecas criados até então, foram apresentados e colocados em uso com o driver e casos de teste previamente citados. O controle de versão permite uma manejo simples da progressão no desenvolvimento de programas tanto individualmente, quanto com um repositório compartilhado com vários desenvolvedores. Um sistema de relato de defeitos, ou *tickets*, torna visível problemas já conhecidos do projeto em questão, e permite atrair a atenção necessária para inconsistências encontradas por casos de teste.

3.1.2.1 Ferramentas internas

O fluxo de validação é dependente dos periféricos componentes do SoC e dos processadores alvo, acessados por ferramentas proprietárias dos fabricantes e internas da própria NXP, em virtude da experiência agregada ao longo de várias validações. As ferramentas utilizadas para o fluxo de validação durante o estágio foram:

- **IDEs Arm DS-5 e Arm Development Studio:** ambientes de desenvolvimento de software em C/C++ para sistemas embarcados. Conta com o Arm *Debugger* para depuração em processados Arm.
- ***Toolchains* Arm e Cadence:** conjunto de compiladores, bibliotecas e outras ferramentas para a tradução de códigos-fonte C em linguagem de máquina com alvos em processadores Arm e Cadence.
- **Repositório Git Bitbucket:** repositório Git em núvem com funcionalidades para CI/CD (*Continuous Integration*/*Continuous Delivery*).
- **Sistema para desenvolvimento ágil Jira:** ferramenta de desenvolvimento ágil de software, com acompanhamento de itens e projetos.
- ***Debugger* DSTREAM:** ferramenta Arm para *debug* e *trace* de software em processadores da fabricante, com USB e Ethernet para conexões diretas e remotas de um usuário com o alvo.
- ***Debugger* TRACE32:** ferramenta da Lauterbach com suporte a emulador e capacidades de depuração com alvos Arm e Cadence.
- **VDI e processamento remoto:** muito embora disponibilizadas máquinas com o sistema operacional Windows, as ferramentas de EDA (*Electronic Design Automation*) disponibilizadas em Linux implicaram na necessidade do acesso a distribuições

deste outro sistema. VDIs (*Virtual Desktop Infraestructure*) tornam viável o processamento remoto, com a disponibilidade de máquinas virtuais com distribuições Linux hospedadas em servidores da empresa, e com conexão segura por VPN (*Virtual Private Network*). A configuração TR do Linux permitia a seleção das versões das ferramentas de desenvolvimento necessárias.

- **Microsoft Office:** para fins de elaboração de relatórios, listagem de casos de teste e controle do estado da validação, as ferramentas Word, Excel, entre outras, do Microsoft Office foram utilizadas.

3.2 Validação pré-silício

Em sua natureza, de estimular funcionalidades dos periféricos de um SoC e comparar os resultados de um caso de teste, as etapas de pré-silício e pós-silício diferem apenas no alvo dos testes, onde um sistema de emulação é o da primeira, enquanto que o silício real é o da segunda. Há um propósito duplo no pré-silício, de abrir uma oportunidade para identificação de erros mais cedo no desenvolvimento do projeto, porém também de acelerar o estágio de pós-silício, aproveitando-se dos casos de teste já escritos no primeiro estágio.

A emulação do SoC em desenvolvimento permitia o acesso de depuração por meio do *debugger* TRACE32 da Lauterbach, seguindo rotinas de conexão no Linux. Ferramentas para a coleta de formas de onda (*waveforms*) estavam disponíveis para auxiliar na depuração em um nível mais baixo, e a descrição de *drivers* e casos de teste foi feita por meio do Arm DS-5 disponibilizado na VDI utilizada.

Após o treinamento introdutório ao desenvolvimento de microeletrônica na NXP e nos detalhes do grupo de validação, o mentor do estagiário repassou tarefas e estabeleceu prazos para o estagiário realizar a validação em pré-silício de periféricos do projeto atual, uma vez que a maturidade para o trabalho com as ferramentas do grupo foi atingida. Sob o acompanhamento e auxílio de Jeisson, a infraestrutura para o manejo de interrupções com o GIC-500 (*Generic Interrupt Controller*) da ARM foi codificada, e os IPs I²C (*Inter-Integrated Circuit*) e I3C (*Improved Inter-Integrated Circuit*) de comunicação foram validados em um estágio de pré-silício, compreendendo desde o estudo dos protocolos de comunicação, leitura da descrição dos periféricos no manual de referência, listagem de casos de teste, descrição de *drivers* e codificação de casos.

3.3 Validação pós-silício

Fazendo-se uso de *drivers* e casos de teste já validados para a etapa de pré-silício, a adaptação destes estímulos aos detalhes da placa de desenvovimento em que o SoC se encontra são as tarefas da etapa de pós-silício. Porém, caso o periférico em questão

não possuía funcionalidades suportadas por emulação, o desenvolvimento de código ainda ocorre nessa etapa.

Por padrão, o SoC compreende uma solução em único chip, de modo que para facilitar sua análise e teste de funcionalidades, uma placa de desenvolvimento (EVB - *Evaluation Board*) é projetada. O posicionamento de placas secundárias e roteamento de circuitos é feita pelo setor de produção sob a demanda e instrução do grupo de validação, com reuniões para o refinamento da solução final de EVB, lista de população de resistores e conectores personalizados para os casos de teste descritos.

Em etapa de pós-silício para um projeto distinto ao validado na etapa anterior, o estagiário realizou a validação dos periféricos SDHC (*Secure Digital Card Host Controller*) e I3C, averiguando os resultados da regressão dos casos de teste já cunhados, porém no silício real e em um ambiente de laboratório. Esta foi a última atividade realizada pelo estagiário na NXP.

A validação pós-silício pode apontar falhas de projeto no SoC, implicando no redesenho de partes da solução, seja na integração ou na natureza dos próprios periféricos, de modo que novas versões do SoC são testadas e os problemas reportados são por fim validados.

4 Conclusão

As atividades definidas no plano de estágio foram cumpridas satisfatoriamente pelo estagiário, implicando no auxílio ao grupo de validação em suas tarefas e na valiosa aquisição de experiência em um fluxo de trabalho profissional na microeletrônica. A pandemia do COVID-19 não chegou a afetar o desempenho das atividades, uma vez que medidas de distanciamento social e o uso de EPIs (*Equipamento de Segurança Individual*) permitiram a continuação das tarefas previstas.

Os conhecimentos adquiridos durante o curso de graduação em Engenharia Elétrica foram postos em uso no decorrer da participação em projetos no estágio, amadurecendo o aprendido em sala de aula e provando sua relevância. As disciplinas de *Introdução a Programação*, *Técnicas de Programação* e *Cálculo Numérico*, em suas ementas para o ensino de linguagens de programação C, C++ e MATLAB, agregaram propriedade para a descrição de rotinas em software. Para tarefas em laboratório, por sua vez, as disciplinas e seus co-requisitos de *Eletrônica*, *Arquitetura de Sistemas Digitais* e *Circuitos Elétricos*, forneceram práticas para análise e desenvolvimento de hardware. Por fim, as cadeiras de ênfase em Controle e Automação: *Informática Industrial* e *Gerência, Planejamento e Controle da Produção*, com suas propostas de controle de qualidade e desenvolvimento ágil, tiveram seus conhecimentos postos em prática.

Ademais, a oportunidade de desenvolver habilidades sociais e de trabalho em equipe com profissionais de áreas e culturas distintas foi de grande valia. O treinamento e atuação na NXP pelo estudante foi um experiência por demais gratificante neste ambiente de estado da arte na indústria da microeletrônica.

Referências Bibliográficas

[1] NXP. **NXP Corporate Overview**. Disponível em: <https://www.nxp.com/docs/en/supporting-information/NXP-CORPORATE-OVERVIEW.pdf>. Acesso em: 01 de Junho de 2020.

[2] NXP. **NXP no Brasil**. Disponível em: <https://www.nxp.com/company/our-company/worldwide-locations/nxp-no-brasil:BRAZIL>. Acesso em: 01 de Junho de 2020.

[3] PECKOL, J. K. **Embedded Systems: A Contemporary Design Tool**, 2nd Edition. Wiley, 2019.

[4] MANO, M.; KIME, C.; MARTIN, T. **Logic Computer Design Fundamentals**, 5th Ediiton. Prentice Hall, 2015.

[5] TARAATE, V. **Digital Logic Design Using Verilog: Coding and RTL Synthesis**. Springer, 2016.

[6] RAJEWSKI, J. **Learning FPGAs: Digital Design for Beginners with Mojo and Lucid HDL**. O'Reilly, 2017.

[7] AHMET, B. **Electronics for Embedded Systems**. Springer, 2017.

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