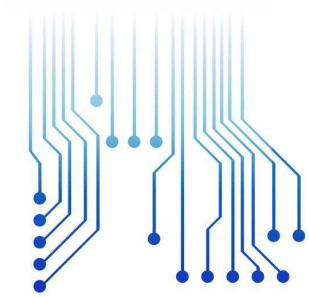


CURSO DE ENGENHARIA ELÉTRICA

MATHEUS ANDRADE DE ALMEIDA

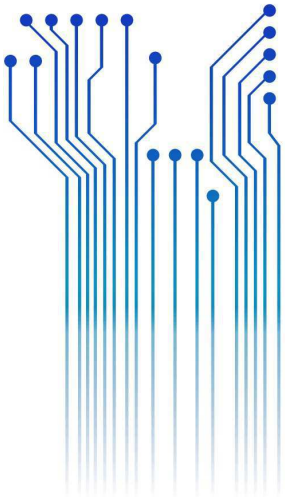
Universidade Federal
de Campina Grande

Centro de Engenharia
Elétrica e Informática

RELATÓRIO DE ESTÁGIO INTEGRADO

NXP SEMICONDUCTORS

Departamento de
Engenharia Elétrica



CAMPINA GRANDE
2018

MATHEUS ANDRADE DE ALMEIDA

NXP SEMICONDUCTORS

Relatório de Estágio Integrado submetido à Unidade Acadêmica de Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciências no Domínio da Engenharia Elétrica.

Área de concentração: Microeletrônica

Orientador:
Professor Gutemberg Gonçalves dos Santos Júnior, D.Sc.

CAMPINA GRANDE
2018

MATHEUS ANDRADE DE ALMEIDA

NXP SEMICONDUCTORS

Relatório de Estágio Integrado submetido à Unidade Acadêmica de Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciências no Domínio da Engenharia Elétrica.

Área de concentração: Microeletrônica

Aprovado em: 21/12/2018

Professor Marcos Ricardo Alcântara Morais, D.Sc.
Universidade Federal de Campina Grande
Avaliador

Professor Gutemberg Gonçalves dos Santos Júnior, D.Sc.
Universidade Federal de Campina Grande
Orientador

CAMPINA GRANDE
2018

Dedico este trabalho à Deus por toda a força e orientação que me concedeu ao longo desses anos.

AGRADECIMENTOS

Agradeço a Deus, em primeiro lugar, pela minha vida e pelo dom da perseverança, que me permitiu concluir este trabalho.

Agradeço também aos meus pais, Antonio e Fatima, por terem se esforçado tanto para me proporcionar uma boa educação, por terem me alimentado com saúde, força e coragem, as quais que foram essenciais para superação de todas as adversidades ao longo desta caminhada.

Agradeço também a minha noiva, Natalya, por ser a pessoa maravilhosa que ela é na minha vida, por todo apoio e força que ela me deu em todos os momentos, fáceis e difíceis nesses anos.

Agradeço também a toda minha família, pelo auxilio ao longo dos meus estudos sem contar todo carinho e apoio, não medidindo esforços para eu chegar a esta etapa da minha vida.

Aos meus queridos amigos do grupo dos "Normais" que sempre estiveram presentes em todos os momentos bons e ruins, desde antes como depois da graduação.

Agradeço também aos meus amigos advindos do meio universitário, em particular, Jesney, Thiago e Yago pois sem eles não teria chegado até aqui.

A todos os membros do projeto de Excelência em Microeletrônica do Nordeste, por todos os momentos que passamos durante o período no laboratório tanto os de trabalho como os de vadiagem.

Aos professores Marcos Morais e Gutemberg Júnior, por todos os ensinamentos, orientação e oportunidades concedidas, permitindo que eu descobrisse áreas de interesse da minha formação e por acreditarem na microeletrônica no Brasil.

Enfim, agradeço a todos que de alguma forma, passaram pela minha vida e contribuíram para a construção de quem sou hoje.

"Tudo o que fizeres, façam de todo o coração, como para o Senhor, e não para os homens, sabendo que receberão do Senhor a recompensa da herança. É a Cristo, o Senhor, que estão servindo."

Colossenses 3:23-24

RESUMO

Neste trabalho estão descritas as atividades desenvolvidas durante o período de estágio do aluno Matheus Andrade de Almeida do curso de engenharia elétrica da UFCG na empresa NXP Semiconductors – BSTC (Brazil Semiconductor Technology Center), dentro da equipe de verificação de SoC. Ao longo do ano de 2018, foram realizados trabalhos na área de microeletrônica direcionados para a verificação de SoC. Durante o estágio foi possível realizar tarefas dentro dos projetos vigentes nesse ano, contribuindo ativamente com o time nas realizações desses trabalhos. Além das atividades práticas, foram realizados treinamentos sobre assuntos relacionados a empresa, bem como, as metodologias utilizadas para realizar a verificação em SoC.
Palavras-chave: NXP Semiconductors, Verificação de SoC.

ABSTRACT

This work describes the activities developed during the traineeship of Matheus Andrade de Almeida from the UFCG electrical engineering course at the company NXP Semiconductors - BSTC (Brazil Semiconductor Technology Center), within the SoC verification team. During the year 2018, work was done in the area of microelectronics aimed at the verification of SoC. During the internship it was possible to carry out tasks within the projects in force that year, contributing actively to the team in the accomplishments of these works. In addition to the practical activities, training was conducted on subjects related to the company, as well as the methodologies used to carry out the verification in SoC.

Keywords: NXP Semiconductors, SoC Verification.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

AHB	*advanced High-performance Bus*;
AMBA	*Advanced Microcontroller Bus Architecture*;
AXI	*Advanced eXtensible Interface*;
BSTC	*Brazilian Semiconductor Technology Center*;
CIEE	Centro de Integração Empresa-Escola;
DFT	*Design for Testability*;
FlexIO	*Flexible input and output peripheral*
IP	*intellectual property*;
JTAG	*Joint Test Action Group*;
NIC	*Network Interconnect CoreLink*;
RTL	*Register transfer level*;
SoC	*System On Chip*;
SWD	*Serial Wire Debug*;
TAP	*Test Acess Port*;

SUMÁRIO

1 Introdução

Este relatório tem como objetivo descrever a experiência de estágio integrado do estudante Matheus Andrade de Almeida, do curso de Engenharia Elétrica da Universidade Federal de Campina Grande, na empresa NXP Semiconductors, sob a tutoria do Engenheiro Jonathan Janampa e supervisão do Engenheiro Jorge Sabino. O referido estágio teve início no dia 09/01/2018 a 17/12/2018 cumprindo com a carga horaria miníma de 660 horas da disciplina de estágio integrado, disciplina a qual é componente curricular obrigatório e o cumprimento de sua carga horária é requisito para aprovação e obtenção de diploma de bacharel em engenharia elétrica.

O contrato firmado entre a empresa e a universidade foi feito tendo o CIEE, Centro de Integração Empresa-Escola, como intermediário. Segundo LUPI (2008) esse órgão, é uma associação civil de direito privado, sem fins lucrativos e de fins não econômicos, reconhecida como um agente de integração no mercado de trabalho para adolescentes e jovens, e tem por responsabilidade fazer o acompanhamento administrativo das oportunidades de estágio, tendo sedes em vários estados ao longo do país.

Figura 1 – BSTC, Base de Operações Brasileira da NXP Semiconductors

Fonte: Autoria Propriá.

A NXP opera no Brasil junto ao BSTC, ou *Brazilian Semiconductor Technology, Center*, dentro do complexo empresarial do Technopark em Campinas, São Paulo. Suas instalações permitem o desenvolvimento de notáveis produtos na área de tecnologia, especificamente de microprocessadores e microcontroladores que são, então, produzidos e distribuídos ao redor do mundo.

Durante o período do estágio, diversas atividades foram desenvolvidas dentro da empresa, auxiliando em projetos e permitindo crescimento pessoal e profissional. A NXP trabalha no setor de microeletrônica fabricando produtos na linha de chips semicondutores para utilização de terceiros. Assim sendo, as atividades realizadas auxiliaram na criação desse tipo de produto, sendo efetivas como modo de ensino e tendo resultados práticos, observáveis no fluxo de trabalho da empresa.

1.1 Objetivos

Dessa maneira, o estágio integrado atingiu os objetivos firmados até presente momento. Tais objetivos serão discutidos a seguir.

1.1.1 Gerais

Este relatório tem como objetivo descrever as atividades realizadas durante o estágio na *NXP Semiconductors*, relacionando as experiências vividas no mercado de trabalho com os ensinamentos obtidos na Universidade, principalmente relacionado ao conteúdo apresentado nas disciplinas de Circuitos Lógicos, Arquitetura de Sistemas Digitais e Arquiteturas Avançadas para Computadores.

1.1.2 Específicos

- Abordar uma breve descrição sobre a história da empresa e sua atuação no mercado;
- Apresentar o fluxo de elaboração do um projeto de um chip;
- Descrever as atividades realizadas durante o estágio que se concentraram na área de Verificação de SoC (*System On Chip*).

2 Revisão Bibliográfica

Neste capítulo será apresentado a história da empresa bem como sua atuação no mercado mundial, além disso será apresentado a fundamentação teórica necessária para a compreensão do fluxo de atividades realizadas durante um projeto de um chip.

2.1 História

A Freescale, que tinha sua matriz na cidade Austin no Texas, teve suas operações iniciadas no Brasil em 1967, lançando as bases para o que, em 1997, viria a ser o Brazil Semiconductor Technology Center (BSTC). O BSTC começou suas operações com 8 experientes engenheiros e atualmente conta com mais de 130 funcionários (95% engenheiros) contabilizando vários projetos entregues, nas áreas de gerenciamento de potência, redes automotivas, processamento digital de sinais e aceleradores criptográficos e de temporização About NXP|Brasil (2016).

A Freescale e NXP no ano de 2015 anunciaram o acordo de fusão sendo a NXP a compradora. A partir desse momento, todos os recursos pertencentes à Freescale se tornaram da NXP, empresa esta que atua no mercado de microcontroladores. No entanto, cada uma delas teve uma trajetória distinta até se tornarem o que são atualmente.

2.1.1 Freescale Semiconductor Employees

A Freescale foi fundada no ano de 1948, a partir da separação do setor de semicondutores da Motorola, se tornando uma das primeiras empresas dessa setor. Ela foi de extrema importância para o programa espacial dos Estados Unidos durante a corrida espacial e os Programas Apollo, fornecendo diversos materiais e equipamentos para as missões. Inicialmente voltada para Radiofrequência e Comunicações, a divisão eventualmente se voltou para o desenvolvimento comercial de produtos para o mercado automotivo e de semicondutores.

Figura 2 – Antiga Base de Operações da Freescale no Oak Hill em Austin, Texas

Fonte: (FREESCALE, 2005).

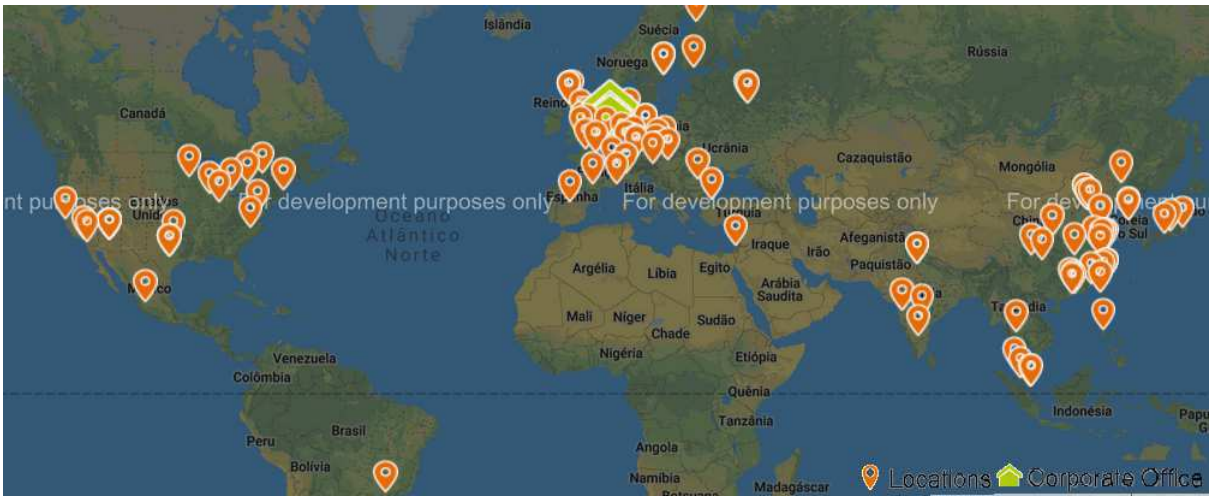
Graças ao grande desenvolvimento e pelo aumento crescente dos números de lançamentos no mercado, a Motorola anunciou, no ano de 2003, que a sua divisão de semicondutores seria oficialmente separada dando origem a *Freescale Semiconductor Employees*. Já como empresa independente, novas linhas de produtos e melhorias foram apresentadas, entrando em setores como geração de energia solar e embarcados na medicina. Também foram criados os produtos da linha Kinetis, considerados à época os menores chips ARM do mundo, rivalizando os chips oferecidos pela líder no setor, Atmel.

2.1.2 NXP Semiconductors

A historia da NXP começa no ano de 1953 a partir da criação da Phillips Semiconductors, um setor de pesquisa da empresa Phillips. Esse setor ganhou grande relevância dentro da empresa graças a criação de tecnologias inovadoras e ímpares no mercado, como o NFC, *Near Field Communication*, e chips de identificação utilizados em passaportes, sem contar as importantes aquisições realizadas, como a Signetics, responsável pela criação do temporizador 555, e a VLSI Technology.

Em 2006, a Phillips anunciou a separação do seu setor de semicondutores, tornando-se independente da sua matriz a empresa passou a se chamar de NXP Semiconductors. A transação decorreu por meio de um consórcio de investidores que realizaram a compra de 80,1% do setor, fazendo parte deste grupos como Kohlberg Kravis Roberts(KKR), Bain Capital, Silver Lake Partners, Apax Partners e AlpInvest Partners.

Figura 3 – Bases de Operações da NXP no mundo.



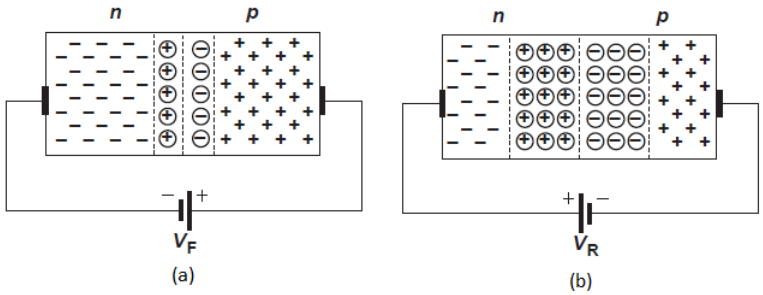
Fonte: (ABOUT NXP|WORLDWIDE LOCATIONS, 2016).

Já então como NXP, a empresa adquiriu diversas companhias ao longo de sua trajetória auxiliando na expansão de novas áreas de mercado. Dessa maneira, suas aquisições começaram já no início de sua separação, em 2007, com a AeroFONE, e se seguiram nos anos seguintes, variando desde *joint ventures* até processos de restruturações e compra de empresas estratégicas para sua própria expansão. Culminando em dezembro de 2015, na compra da multinacional de semicondutores americana Freescale Semiconductors. Atualmente, a empresa está presente 33 países, possui 31.000 funcionários e cerca de 130 filiais pelo mundo e registrou uma renda de $9,5 Bilhões de Dólares em 2016. About NXP|Worldwide Locations (2016).

2.2 Semicondutores

Semicondutores são materiais que possuem como característica mais importante a capacidade de alterar sua condição de isolantes para condutores com certa facilidade. Isso ocorre graças à criação de uma "banda proibida" na sua região intermediária a partir do momento que é submetido a uma diferença de potencial elétrico. As figuras 4 (a) e (b) representam um semicondutor quando é submetido a uma polarização direta, presente na Figura 4(a), e a uma polarização reversa, apresentada na Figura 4(b). Dessa forma pode-se perceber os momentos em que o semicondutor se encontra em condição de condução e isolação respectivamente.

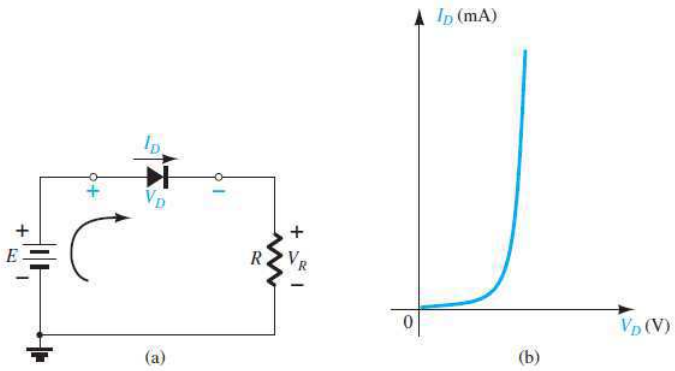
Figura 4 – Semicondutor sobre polarização: (a) direta; (b) reversa.



Fonte: (RAZAVI, 2014).

Dentre os semicondutores mais relevantes podemos dar destaque a dois, os diodos e os transistores. Segundo Razavi (2014) o diodo é um dispositivo composto de dois terminais, catodo e anodo, sendo o anodo representado por um triângulo o qual indica a direção permitida para o fluxo de corrente e o catodo representado por uma barra vertical que significa o bloqueio do fluxo de corrente na direção oposta.

Figura 5 – Configuração do diodo serial: (a) circuito; (b) curva característica.

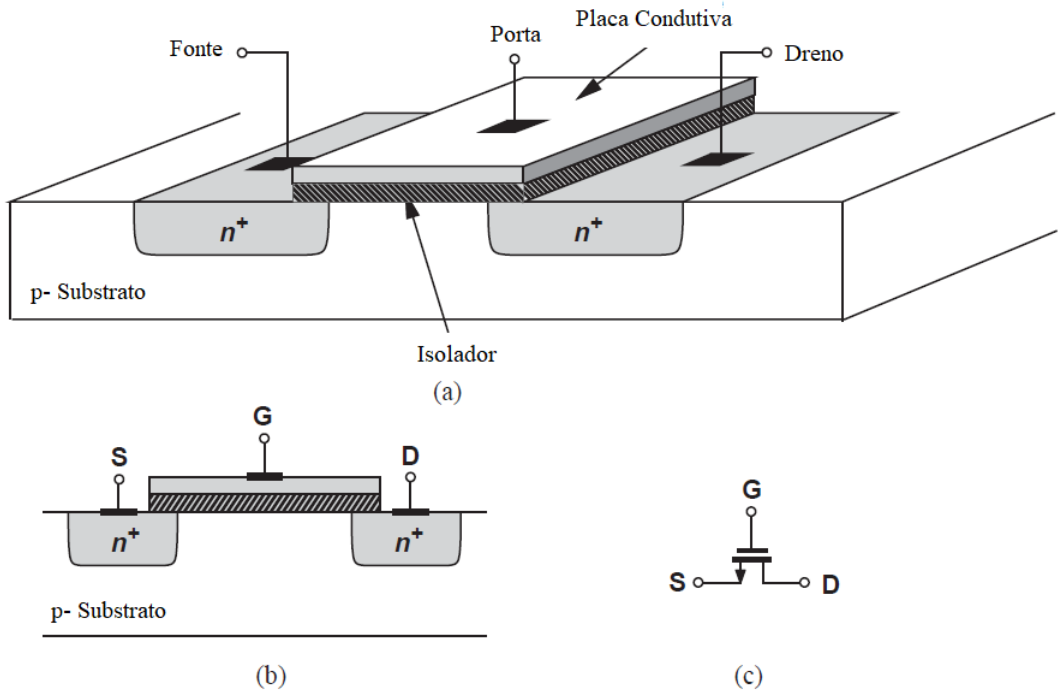


Fonte: (BOYLESTAD, 2013).

Transistores por sua vez são estruturas mais complexas que permitem o desenvolvimento de sistemas muito mais efetivos e controlados que aqueles possibilitados por diodos. Segundo Amos (2000), o transistor diferindo do diodo, funciona com três terminais. O primeiro deles representa a entrada de corrente, chamado de Fonte ou *Source*, o segundo a saída, denominado Dreno ou *Drain*, e por fim o terceiro, sendo o maior diferencial, é um terminal de controle chamado de Porta ou *Gate* funcionando como uma chave que determina se a corrente será

ou não transmitida pelo transistor. Caso seja alimentado, o Gate permite a passagem, não a permitindo no caso oposto. A maneira com que esses terminais estão dispostos no transistor pode ser observada na Figura 6.

Figura 6 – (a) Estrutura de um transistor MOSFET, (b) Visão lateral, (c) Simbolo.



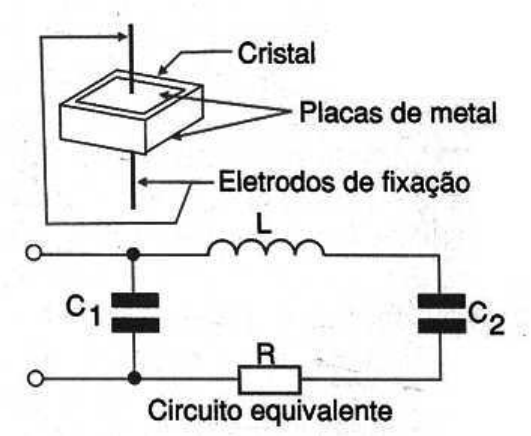
Fonte: (RAZAVI, 2014).

Graças a possibilidade de se realizar esse nível de controle, eventualmente foi descoberto que, utilizando um conjunto de transistores, era possível desenvolver uma linguagem computacional baseado no seu estado elétrico. A codificação dessa linguagem foi regida por diretrizes que determinavam que o sinal pode ser representado de maneira binária, com os valores 0 ou 1, valores esses determinados pela intensidade da corrente que atravessa o fio num dado momento. Definida essa representação foi possível aplicar estruturas de tratamento e transmiti-los de maneira controlada através de um circuito. A partir disso, foi possível gerar dados num componente e transmiti-los de maneira compreensível para outros, que os utilizariam nas suas próprias funções, aumentando, assim, a gama de aplicações eletrônicas que poderiam ser desenvolvidas para o mercado.

2.3 Clock

Em um SoC, um dos componentes mais importantes para o seu funcionamento é o *clock*, ou relógio, esse componente é responsável por realizar a sincronização das atividades do chip. A medição do *clock* é feita em hertz (Hz), a unidade padrão de medidas de frequência, indica o numero de oscilações ou ciclos que ocorrem dentro de um segundo.

Figura 7 – Circuito equivalente de um cristal.



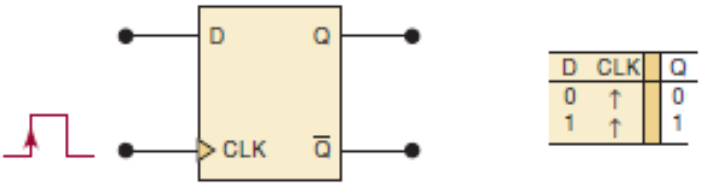
Fonte: (PINTO, 2000).

O *clock* consiste num contador que varia de acordo com a frequência de um cristal oscilatório, demonstrado na Figura 7, inserido dentro do SoC. Um sistema permite a utilização de vários *clocks*, porém é necessário cuidado, uma vez que o *clock* é um sinal elétrico o mesmo pode sofrer com perda de intensidade devido as resistências dos componentes e das trilhas, podendo gerar um problema de dessincronização entre os diferentes *clocks*, causando atrasos e perdas de informações nas operações do chip.

2.4 Registradores

Os registradores são pequenas partições de memória localizadas dentro de cada IP, *intellectual property*, permitindo acessos e armazenamento rápidos de dados dentro do IP, aumentando dessa forma a velocidade de execução da lógica interna. Os registradores, em sua maioria, são compostos basicamente por flip-flops, estrutura básica de armazenamento de dados num chip. Seu funcionamento, segundo Mano (2015), é dado através de portas dispostas de maneira a permitir a passagem de dados de maneira intercalada. Dessa maneira, é possível alimentar o componente com um dado e, após um ciclo de clock, ele será propagado no circuito. Essa estrutura é uma das mais importantes na construção de componentes eletrônicos, sendo vital em qualquer aplicação que requeira controle sob os dados sendo transmitidos em seus terminais.

Figura 8 – Estrutura de um Flip-Flop Tipo D.



Fonte: (TOCCI, 2007).

A estrutura interna do Flip-flop mais utilizado, o tipo D representado na Figura 8, tem seu funcionamento descrito, segundo Tocci (2007), da seguinte forma. O sinal de entrada D será transmitido para saída Q normalmente e inversamente para a saída $\overline{Q}$. Em outras palavras, o nível presente em D será armazenado no flip-flop no instante em que o *clock* oscilar.

2.5 SoC - *System on Chip*

A evolução tecnológica, proporcionada pelo desenvolvimento do mercado de semicondutores, possibilitou para o mundo a criação de diversos dispositivos capazes de realizar operações ainda mais complexas e que ocupassem espaços bem menores. A integração desses dispositivos, tanto analógicos quanto digitais, em um único componente passou a ser chamada de chip. Esses produtos, em geral, comportam de milhares a milhões de transistores dentro de seus circuitos.

Por sua vez os produtos chamados SoC, ou *System on Chip*, são componentes eletrônicos complexos que, segundo Flynn (2011), envolvem processadores, memórias e um sistema de entrada e saída criadas para um certo campo de aplicação. A gama de aplicações que eles podem realizar é virtualmente infinita, dependendo apenas do seu processo de fabricação, sendo dessa forma utilizados desde circuitos puramente digitais ou analógicas a circuitos mistos, analógico-digitais, e circuitos de rádio frequência. Eles também são vastamente utilizados no mercado de embarcados, sendo, usualmente, o componente principal da aplicação, agindo como controlador ou o processador de outros componentes.

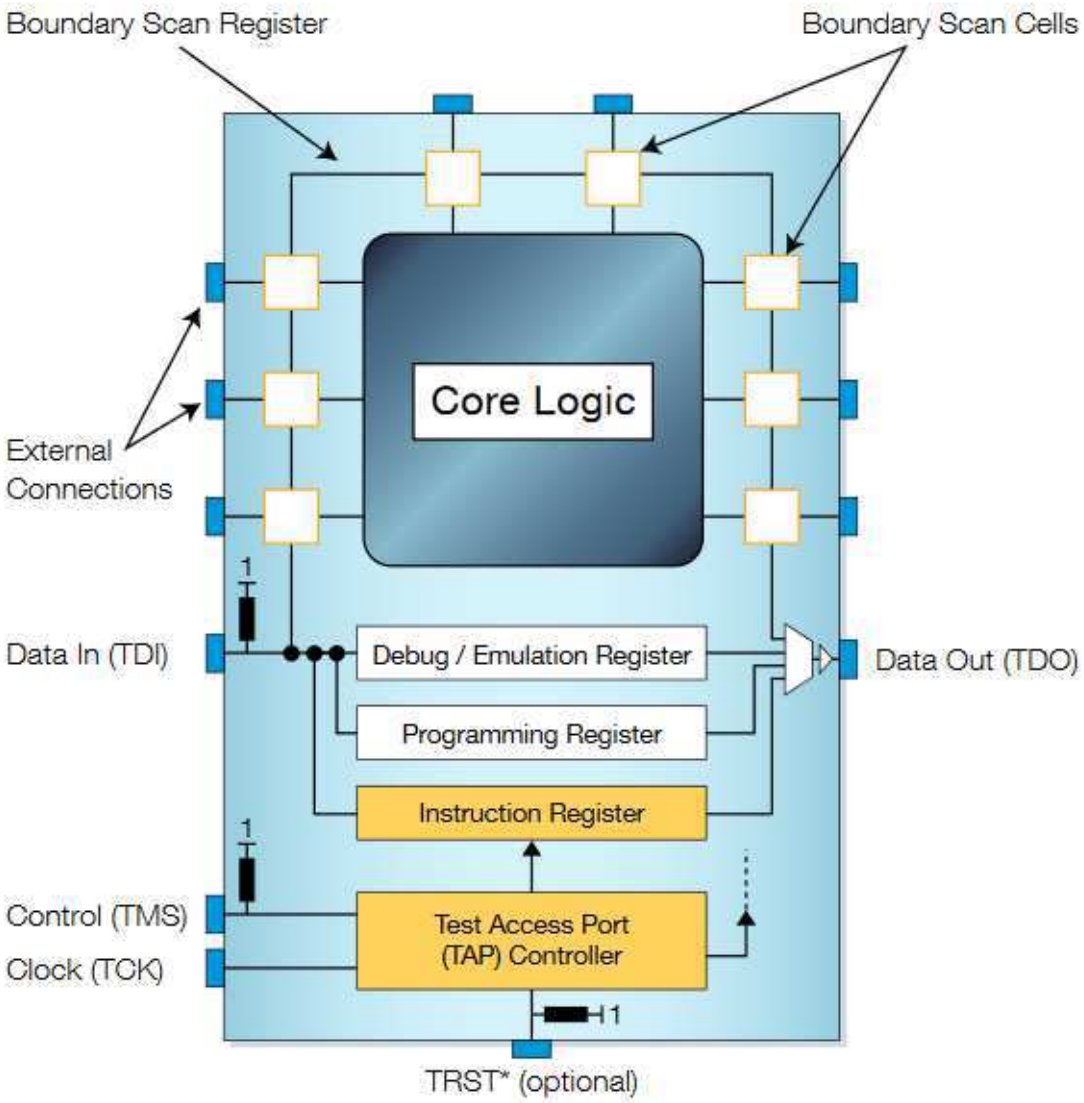
O desenvolvimento de um SoC passa por várias etapas antes de chegar até o consumidor final, desde o momento em que foi fechado o contrato com o cliente, passando pela análise financeira do projeto, pelo planejamento e definições estruturais, até chegar nas etapas de criação do chip as quais são finalizados com análises de qualidade, para assim poder ser entregue ao cliente.

2.6 JTAG

O JTAG é uma interface de programação e teste de circuitos digitais, padronizada como IEEE 1.149,1. Originalmente desenvolvido para a programadores lógicos. O mesmo recebeu esse nome em homenagem ao grupo ao qual o codificou o *Joint Test Action Group*, o JTAG também é frequentemente utilizado para microcontroladores. Conectando-se a uma porta de acesso de teste no chip (*Test Acess Port* - TAP) a qual foi projetada para interagir através de quatro linhas seriais, Controle (TMS), *Clock* (TCK), Dados de entrada (TDI) e Dados de saída (TDO), podendo ou não ser utilizada uma quinta porta para o *reset* (TRST). Dessa maneira as portas de memória, bits de segurança, registros e etc, podem ser lidos e escritos em alta velocidade, um exemplo de uma arquitetura com JTAG é apresentado na figura 9.

Inicialmente as primeiras aplicações com JTAG tinham como alvo testes de nível de placa, atualmente o padrão é projetado para auxiliar no teste de dispositivos, placas, sistemas, diagnóstico e isolamento de falhas. Hoje o JTAG é usado como o principal meio de acessar sub-blocos de circuitos integrados, tornando-se um mecanismo essencial para *debugar* sistemas embarcados que podem não ter nenhum outro canal de comunicação capaz de depuração.

Figura 9 – Diagrama de blocos utilizando JTAG.



Fonte: (XJTAG, 2015).

Os *hosts* JTAG se comunica com os TAPs manipulando o caminho de controle e de dados de entrada em conjunto com o *clock* obtendo os resultados por meio dos dados de saída, o qual é a única entrada padrão do lado do *hosts*. As transições do *hosts* usam o caminho mais curto entre dois estados, algumas camadas construídas sobre o JTAG monitoram as transições de estado e usam caminhos incomuns para acionar operações de nível superior. Alguns núcleos ARM usam essas sequências para entrar e sair de um modo SWD de dois fios.

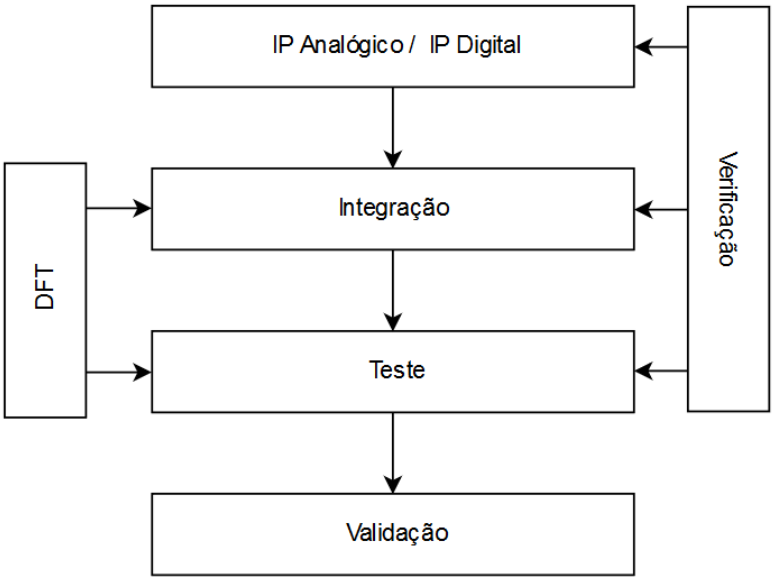
O SWD, Serial Wire Debug, nada mais é que uma interface elétrica alternativa de 2 pinos que usa o mesmo protocolo. Ele usa a conexão com a referência existente. O SWD usa um protocolo de fio bidirecional padrão da CPU ARM, definido na interface de *debug* ARM v5 ARM (2017). Isso permite que o depurador se torne outro mestre do barramento AMBA para acesso à memória do sistema e registros de *debug* ou periféricos. Em dispositivos JTAG com capacidade SWD, o controlador e o *clock* são usados como sinais SWDIO e SWCLK, proporcionando uma programação em paralelo.

2.7 Fluxo do Projeto

O processo de criação de um chip é complexo e necessita de uma equipe qualificada, a qual deve possuir membros com habilidades específicas para cada uma das diferentes etapas do projeto. Por isso, existem diferentes grupos dentro de uma empresa de semicondutores, visando auxiliar a gerência e a liderança sobre o estado atual do projeto facilitando as tomadas de ações necessárias para otimizá-lo.

Analisando o escopo da produção e assumindo o maior nível de abstração possível, é viável traçar um diagrama do fluxo do projeto, representado na Figura 10.

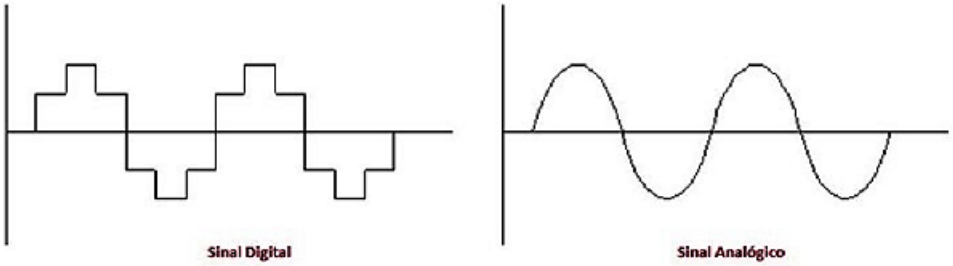
Figura 10 – Fluxo do Projeto de criação de um Chip.



Fonte: Próprio Autor.

Como todo bom projeto, deve-se começar do mais baixo nível para que a evolução seja feita gradualmente, em um processo de criação de um chip isso não é diferente. A primeira etapa no fluxo envolve a criação dos IPs ou *Intellectual Properties*, estes são componentes básicos do chip apresentando funcionalidades específicas. Essa etapa envolve a criação tanto de IPs digitais quanto analógicos, podendo existir IPs que possuem componentes digitais e analógicos funcionando em sincronia.

Figura 11 – Representação de um sinal: (a) digital; (b) analógico.



Fonte: Próprio Autor.

A principal diferença entre IPs digitais e analógicos é a maneira com que seus sinais são tratados, podendo perceber a diferença entre esses tipos de sinais na Figura 11. Segundo Schaum (1995), se um sinal de tempo contínuo poder assumir qualquer valor no intervalo contínuo esse sinal é chamado de analógico, enquanto que se um sinal de tempo discreto pode assumir apenas um número finito de valores distintos, então chamamos este sinal de sinal digital.

2.7.1 IP Analógico

IPs analógicos são estruturas as quais precisam que seu desenvolvimento seja feito a nível de transistores, uma vez que, é necessário a realização de estudos dos vários limites eletrônicos para o componente, afim de que atendam aos requisitos necessário para a aplicação à qual estão sendo desenvolvidos. Estes limites são geralmente a tensão, corrente, capacitância e impedância.

2.7.2 IP Digital

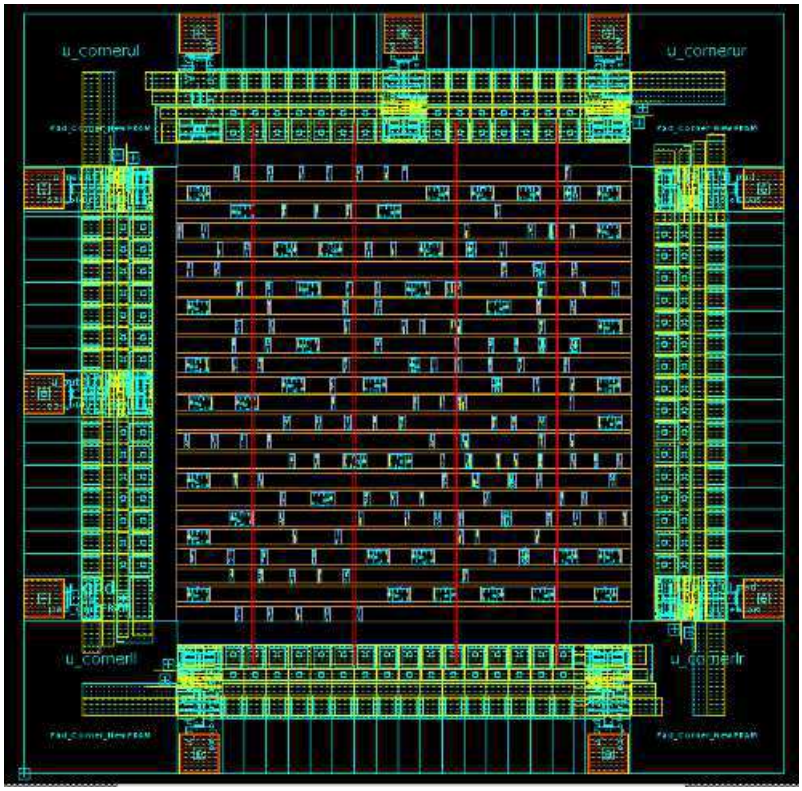
IPs digitais são sintetizados por ferramentas capazes de interpretar as linguagens descritivas em portas logicas, como AND, OR, XOR, NAND e NOT. Dessa forma, linguagens como Verilog, SystemVerilog e VHDL são normalmente utilizadas no processo de criação desses componentes.

Em casos de IPs analógicos mais complexos, é geralmente necessária a criação de uma interface digital para que possa existir uma comunicação mais eficiente entre o IP e o resto do chip. Para tal, os times responsáveis comunicam entre si para compartilhar as especificações desejadas e trabalhar na criação do componente.

2.7.3 Integração

Uma vez criados os componentes é feita uma lista destes para o chip, e os mesmos são adicionados dentro de um ambiente de controle de versão para que se possa prosseguir com para a etapa de integração. Nesta fase, IPs são conectados entre si e sua funcionalidade é aplicada de maneira a funcionar de modo harmonioso, possibilitando a geração de lógicas mais complexas e com características específicas, como maior potência, menor consumo. Junto a essas conexões, é necessário distribuir os componentes no espaço físico destinado ao chip, tomando cuidado com as posições de cada IP para que as conexões não excedam distâncias muito grandes, isso é feito através de ferramentas de posicionamento como a representada na Figura 12, o que geraria problemas de sincronismo no sistema.

Figura 12 – Representação da Colocação de Componentes no Espaço Físico do Chip.



Fonte: (SYNOPSYS, 2015).

Ao fim dessa etapa ocorre o *Tape Out*, isto é, o projeto sai do âmbito virtual e passa para o real, com a produção do silício contendo o chip. A partir desse ponto, é então possível verificar o controle de qualidade ao promover testes e análises que vão observar o comportamento do produto num ambiente real, e não simulado.

2.7.4 Teste

Com o chip fisicamente disponível, a etapa de teste consiste em verificar que padrões e rotinas, geradas na etapa de integração, retornem resultados positivos quando exercitadas. Segundo JAS (2002) a maneira com que esses testes acontecem é através de máquinas especialmente desenvolvidas, chamados testadores, para exercitar os pinos do chip. Elas possuem agulhas ou *sockets*, produzidos especificamente para cada modelo de chip, e são, então, programadas para gerar valores iguais aos dos padrões, testando a funcionalidade da parte lógica e verificando se existe algum tipo de erro gerado pela má produção do chip.

2.7.5 Validação

A etapa final do processo é a validação, responsável por realizar análises no chip que confirmem suas características determinadas ao longo do projeto. Antes da chegada do chip, a equipe realiza estudos seguindo as especificações do projeto e determina quais testes são pertinentes para serem feitos, simulando os mesmos em FPGA e em softwares. Já com o chip, o time confirma via testes de bancada, que são estímulos e medições com aparelhos de laboratório em placas de teste que contém o chip, que ele funciona, é interessante também exercitar o chip com valores máximos de forma a verificar seus limites de atuação.

2.7.6 Verificação

O time de verificação atua de forma paralela ao fluxo linear do projeto, uma vez que se tem a necessidade de garantir que não haja erros ao longo do projeto. Um erro que se propague sem controle traria prejuízos não mensuráveis ao produto final e à empresa. Dessa maneira, existem divisões responsáveis por verificar as atividades dos outros times, através das diferentes etapas do projeto.

No caso do grupo de verificação o mesmo é dividido em IP e SoC, para a equipe de IP cada modificação feita em IPs individuais requerem a ação de um verificador para garantir que o componente mantém a funcionalidade descrita em sua especificação. De maneira análoga, o time de SoC verifica que as modificações feitas ao código do chip bem como as ligações dos componentes, que ocorre na etapa de integração, continuam funcionais e que é possível transmitirem informações apropriadamente através dos componentes. Dessa maneira, o grupo de verificação começa seu trabalho já no início da vida de um projeto, e continua ativamente até o momento em que não houver mais nenhuma alteração física ou lógica no chip.

2.7.7 DFT

A etapa de DFT, sigla para *Design for Testability*, é definida segundo Wang (2006), como responsável por adicionar, principalmente durante a integração, componentes e lógica no chip que permitam que o time de testes efetue, nas etapas futuras, testes que proporcionem ao projeto a maior taxa de cobertura possível. Ao testar o chip, é possível detectar discrepâncias entre a lógica e o físico. assim promovendo maior segurança e qualidade para o projeto.

Esse grupo é considerado parte do grupo de Integração, devido às atividades desenvolvidas e ao período de atuação da equipe. Sua representação única no fluxo se faz necessária, entretanto, por sua atuação persistir após o *Tape Out*, continuando durante o período de testes. Nessa nova fase, geralmente existe a geração de novos padrões para melhor cobertura ou eventual análise do RTL, *Register transfer level*, para correções. Uma testabilidade eficiente garante que problemas de fabricação sejam detectados e que a peça defeituosa seja descartada antes de chegar ao cliente.

3 Atividades Realizadas

Durante o período de estágio que decorreu ao longo do ano de 2018 na NXP, foram realizadas diversas atividades na areá de verificação de SoC, que é responsável pelo processo de verificação da conexão entre os diferentes IPs utilizados nos SoCs dentro dos projetos. Todas as atividades contaram com o acompanhamento de ao menos um tutor, o qual prestou suporte e auxílio na solução de eventuais dúvidas.

Todas as atividades foram realizadas dentro da equipe de verificação de SoC, a qual apresenta como objetivos garantir a integridade das conexões feitas entre os componentes utilizados, mais precisamente garantir que a experiência do usuário final estará de acordo com as especificações do projeto. Isso inclui então, verificar o funcionamento dos processadores e aceleradores, escritas e leituras na memória, condições de consumo, e principalmente se as interligações de cada IP estão sendo realizadas. Portanto, é de responsabilidade da equipe elaborar um plano de verificação e garantir que todas as possíveis falhas de integração possam ser corrigidas através desse plano.

A verificação era feita por meio de testes em C/C++ e em alguns casos específicos feitos em conjunto com SystemVerilog, os quais visavam estimular todas as conexões dos IPs a serem analisados, simulando dessa forma o funcionamento do chip em uso normal por parte do usuário final. Essa metodologia de verificação diverge da utilizada pelo time de verificação de IP, o qual foca na garantia de que um IP específico está cumprindo com suas especificações.

Bem como em todo início de trabalho foi necessário um período para se familiarizar com o ambiente, dessa forma a empresa ofereceu alguns treinamentos voltados principalmente para explicar as ferramentas da próprias da empresa, embora tenham sido ofertados também cursos para maior familiarização com as linguagens básicas para a realização das atividades práticas.

3.1 Treinamento

O processo de treinamento foi realizado com o auxilio de funcionários mais experientes os quais se dispuseram a realizar apresentações, a fim de que todos os estagiários ficassem a par de diferentes ferramentas, sistemas e protocolos utilizados pela empresa, assim como para que conhecessem as diferentes áreas e as etapas de desenvolvimento dos produtos produzidos pela empresa.

Inicialmente os treinamentos foram focados em apresentar o fluxo de projeto, bem como o uso das ferramentas internas da empresa, apresentando a metodologia e o conceito de qualidade aplicados na empresa.

Ao término dessa etapa, os estagiários começaram a desenvolver atividades referentes aos seus times. Dessa maneira foi iniciado o trabalho em conjunto com a equipe de verificação de SoC, que decorreu até o término do estágio.

3.2 Contextualização das Atividades Praticas

Durante o período de estágio, foi possível a participação em diversas frentes de conhecimento, sendo as mais importantes:

- Desenvolvimentos de software para verificação funcional e formal;
- Verificação do ambiente de teste realizado pela equipe de DFT;
- Estudo de ferramentas para detecção de falhas na verificação;

Estas atividades serão explanadas de forma mais detalhada nos tópicos à seguir.

3.2.1 Verificação Funcional

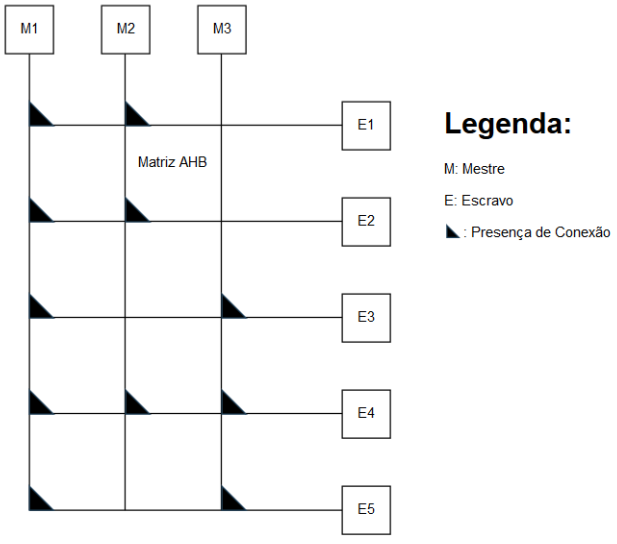
A tarefa principal da equipe de verificação de SoC gira em torno de garantir que os IPs integrados pelo time de integração, estão conectados e funcionando de acordo com a documentação. Dessa maneira, são desenvolvidos softwares que serão executados pelos processadores, ou aceleradores disponíveis no chip, afim de estimular cada IP configurando todas as opções de *clocks*, *resets* e *triggers* externos e demais configurações do IP de forma a garantir que todos os sinais que entram e saem dele, estão corretamente conectados.

Antes do inicio de um projeto, é passado para os gerentes responsáveis por cada equipe as especificações que serão utilizadas no chip bem como os prazos de entrega. Partindo dessas informações cada gerente escolhe líderes dentro do seus times para serem responsáveis pelo projeto. Após a definição dos líderes os gerentes de cada time juntamente com os líderes de projeto escolhidos, traçam planos de como serão executadas as atividades da equipe alocando cada um dos demais membros em atividades que lhe sejam mais adequadas, partindo do pressuposto de conhecimentos já adquiridos em trabalhos anteriores.

Conforme Bergeron (2003), a verificação funcional é um processo usado para demonstrar que o objetivo do projeto é preservado em sua implementação. Tendo essas definições em mente, o verificador de SoC inicia suas atividades realizando estudos do funcionamento do IP ao qual foi alocado, por meio da documentação disponibilizada, bem como de demais fontes que possam servir de auxílio ao seu trabalho. Traçando dessa forma o plano de verificação mais adequado para as especificações do projeto em questão, de maneira que venha a estimular todos os sinais do bloco ao qual foi designado.

O estagiário foi responsável pela verificação da matriz AHB, *Advanced High-performance Bus*, iniciando as atividades com a criação de uma planilha contentado todos os mestres bem como todos os escravos, com o objetivo de preencher as mesmas com pelo menos um teste ao qual estimulasse o caminho de um determinado mestre ate um determinado escravo, tomando como base um diagrama semelhante ao representado na Figura 13. Garantindo dessa forma que toda a matriz estava conectada e que os sinais de transmissão de dados e enderenço estavam sendo estimulados.

Figura 13 – Representação demostrativa da matriz AHB.



Fonte: Autoria Própria.

Como uma das formas de otimizar o projeto as equipes de verificação e integração são adaptas da filosofia da reusabilidade do trabalho, uma boa parte das atividades nesta parte consistiu apenas em adaptar os softwares existentes dos IPs para o SoC em questão. No entanto, em alguns casos existe uma nova funcionalidade no projeto em que é necessário o desenvolvimento de um novo estímulo que a teste.

Devido a algumas especificações do projeto foi necessário a realização de adaptações nos testes da matriz, mais especificamente em testes que verificavam as conexões com as memória. Graças a isso foi possível a identificação de algumas incongruências entre a documentação e o design, as quais foram reportadas para o time de integração realizar as devidas correções.

3.2.2 Verificação Formal

Devido ao fato de alguns sinais apresentarem características específicas que não conseguem ser estimuladas funcionalmente devido a limitação da ferramenta, vê-se a necessidade de estimulá-los formalmente por meio da adição, em um códigos *SystemVerilog*, de monitores e *assertions* que tem como objetivo estimular esses sinais que não são verificados funcionalmente.

Alguns desses casos são comumente encontrados na verificação de protocolos de comunicação, onde as partes determinam uma sequência de sinais que devem ser obrigatoriamente obedecidas, com padrões definidos para casos de falha e sucesso da transação. Outro caso recorrente ocorre nas configurações dos pinos externos que aumentam ou diminuem a corrente nos transistores, realizando assim um controle na velocidade com o qual o sinal se propaga para dentro ou para fora do SoC. Apesar de ser uma funcionalidade dos pinos, não pode ser emulada pela ferramenta de verificação, resultando em um sinal exatamente igual do ponto de vista do simulador, e que portanto, não pode ser verificada de formas tradicionais. Para estes casos, e outros, é utilizada a verificação formal.

Foi utilizada para verificação da propagação de sinais elétricos que não podiam ser diferenciados funcionalmente, como sinais que eram estimulados na matriz por meio de dois mestres simultaneamente diferindo a região de conexão por uma lógica interna do próprio IP, e também para a verificação de alguns pinos que precisavam ser estimulados, onde o estímulo destes por meio da verificação funcional é altamente custoso, podendo ser facilmente modelado na verificação formal, além de trazer uma confiança a mais no âmbito geral da verificação, aumentando a confiabilidade do RTL simulado.

3.2.3 Verificação do Ambiente de Teste

Dentro de um SoC uma fração significativa do espaço ocupado pelo mesmo não é utilizado durante o seu uso pelo cliente final, esse espaço é destinado exclusivamente aos blocos de testes que são inseridos com a intensão de realizar testes funcionais no chip para melhor garantia de que seu funcionamento está de acordo com o esperado.

A inclusão desses blocos, durante a fase de integração, é realizado pela equipe de DFT. Sendo possível chegar a conclusão da importância de realizar a verificação dessas conexões, bem como de qualquer outro IP, atividade ao qual também é de responsabilidade do time de verificação de SoC.

Durante essa etapa do projeto o estagiário ficou responsável pela criação de padrões que configurassem os blocos de testes, de forma que a partir do momento que o chip era habilitado para entrar no modo de teste, os IPs de testes deveriam começar a realizar as suas funcionalidades, sendo assim possível verificar de maneira tanto funcional como formal, por meio tanto do ambiente de verificação padrão como por meio do ambiente do JTAG com

capacidade SWD, realizar os estímulos de todos os sinais, bem como, os resultados dos testes verificando se os mesmos estavam funcionais e de acordo com a documentação.

3.2.4 Falhas na Verificação

Afim de validar a real eficiência de uma nova ferramenta, que estava em estado de estudo de viabilidade para projeto, todos os estagiários de verificação de SoC foram alocados para trabalhar com ela, cada um com um bloco especifico. A ferramenta tem como objetivo a detecção de falhas, mas diferente das normalmente utilizadas, essa ferramenta detecta falhas na verificação de SoC e não falhas de RTL.

O seu funcionamento se da por meio dá análise dos padrões criados pelos verificadores, bem como do relatório de cobertura gerado no final de um projeto. A ferramenta não tem o intuito de ser utilizada durante o andamento do projeto, mas sim após o seu término, ou seja, após todo o relatório de cobertura ter sido gerado, e todos os verificadores estejam com suas atividades completas. Nesse momento que a ferramenta será utilizada para que possa garantir que realmente todos os testes criados estão verificando corretamente o SoC.

A ferramenta tem como ideia garantir que todos os sinais do SoC de fato estejam sendo testados. Para isso ela se divide em três etapas de funcionamento, sendo elas a fase que gera um modelo do RTL para ser utilizados nos testes, seguido de uma fase que detecta quais sinais estão sendo ativados por cada teste, e por fim temos uma etapa de validação dos testes na qual a ferramenta insere falhas no RTL e espera que os padrões que estimularam o respectivo sinal, reportem o erro.

O funcionamento de todo esse processo da ferramenta demora um tempo considerável, já que para uma regressão de um único bloco pequeno que demorava algo entre 30 minutos e 2 horas para apresentar os relatórios de cobertura a ferramente chegou ao ponto de que com o mesmo bloco demorar de 6 à 12 horas, onde o caso de um bloco maior chegou a demorar 35 horas. Todavia independente de tudo esse tempo foi discutido por toda a equipe e a conclusão realmente foi a de que a ferramenta deveria ser utilizada no fluxo de projeto da verificação de SoC.

3.3 Detalhamento das Atividades Práticas

No decorrer do estágio atividades praticas especificas foram alocadas para o estagiário realizar a verificação, sendo as mesmas:

- Verificação dos barramento AMBA: AHB e AXI;
- Verificação da interface de teste do MIPI/DSI;
- Estudo de ferramentas para detecção de falhas na verificação, analisando o IP FlexIO;

Estas atividades serão explanadas de forma mais detalhada nos tópicos à seguir.

3.3.1 Barramentos AMBA: AHB e AXI

A Arquitetura de Barramento Avançado de Microcontrolador (AMBA, *Advanced Microcontroller Bus Architecture*) foi introduzida em 1996 e é largamente utilizada como um padrão de comunicação on-chip. Sua função é fazer com que os blocos interajam uns com os outros em um SoC, proporcionando dessa forma a comunicação entre os módulos. O protocolo AMBA possui alguns modelos de barramentos de sistema, dentre eles analisaremos apenas os que foram relevantes durante o trabalho do estagiário, sendo estes AMBA AHB e AXI. O AHB atua como o sistema de alto desempenho, suportando conexões eficiente de processadores, memórias on-chip e memória externa sem chip interfaces com funções macrocélulas periféricas

de baixa potência. Enquanto que protocolo AXI é utilizado para altas performances, designs de sistemas de alta frequência e uma série de características de interconexão de alta velocidade.

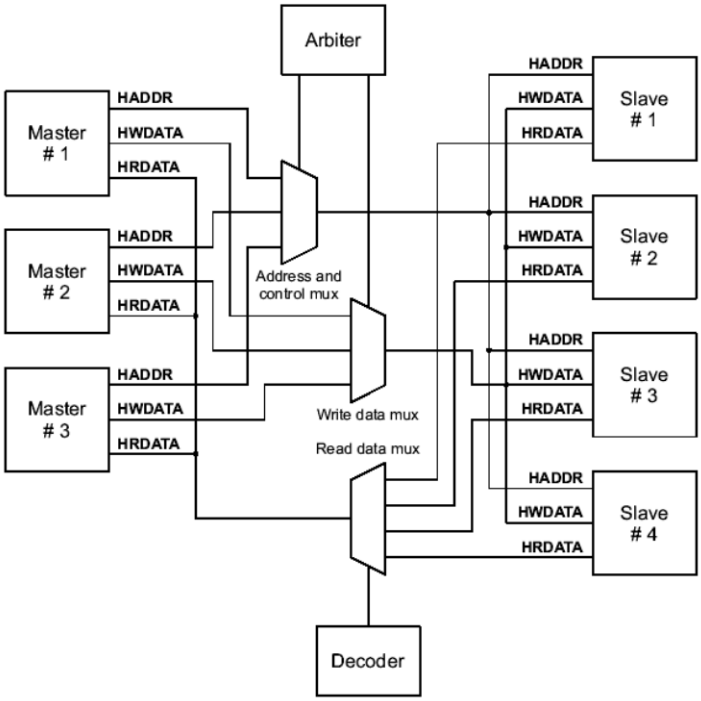
Partindo desse dados o estagiário realizou a verificação dos barramentos AMBA: AHB e AXI, dentro do projeto fazendo uso tanto de verificação formal quanto funcional, como foi citado anteriormente.

3.3.1.1 AHB

Atividade começou com um estudo a respeito da arquitetura dos barramentos, priorizando inicialmente a verificação do AHB o qual era o barramento principal do sistema, foi utilizado como referência a documentação disponibiliza pela ARM ARM (1999), a qual explica detalhadamente o funcionamento do barramento analisado.

O desenvolvimento dos testes foram realizados tomando como base o diagrama do projeto, estrutura essa semelhante ao apresentado na Figura 14, dessa maneira os testes foram criados visando estimular todos os sinais internos do AHB. Sendo assim foi iniciado com a verificação da prioridade de acesso dos mestres perante os escravos, bem como se estava respeitando a hierarquia de mestres do projeto, para com o barramento. O teste visava estimular os sinais específicos do AHB responsáveis por realizar o controle de acesso a matriz, de forma que fosse observado a variação dos sinais no momento em que um acesso de baixa prioridade era substituído por um acesso de alta prioridade, dessa forma foram inseridas *assertions* em conjunto com uma lógica funcional em C para realizar esse monitoramento.

Figura 14 – Ilustração da estrutura necessária para implementar um projeto AMBA AHB.



Fonte: (ARM, 1999).

Outros testes relevantes de se comentar foram testes que tinham como objetivo monitorar as regiões de acesso reservado da memória. Dessa forma foram criados padrões que realizavam tentativas de escrita e leitura em regiões da memória as quais não era permitido acesso por meio da AHB, estimulando assim os sinais de resposta da matriz. Por meio desses

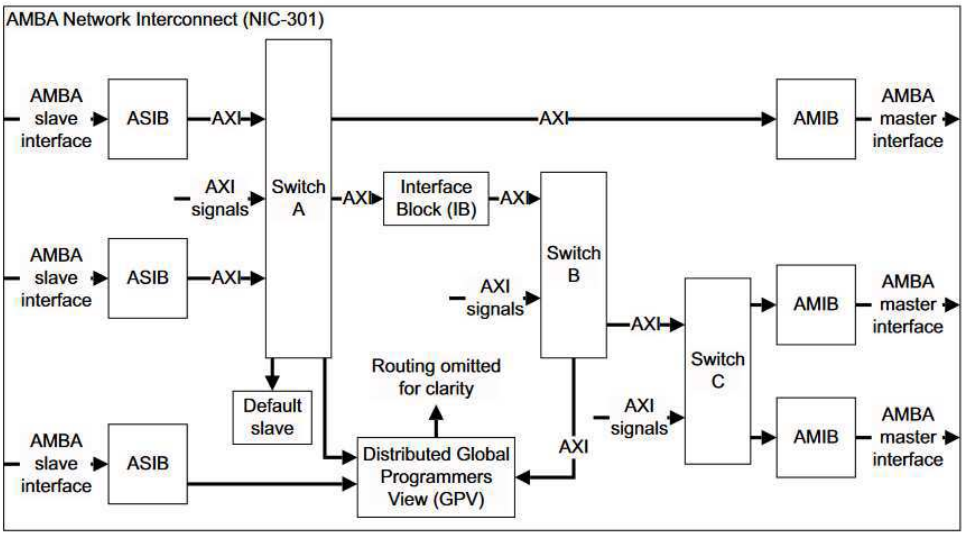
testes foi possível a detecção de inconsistências no *design*, uma vez que certas regiões de memoria que deveriam ser reservadas, não estavam retornando aviso de erro quando era solicitado acesso. Analisando a fundo foi possível detectar que algumas dessas regiões ainda estavam alocadas para blocos que não eram aplicáveis para o projeto em andamento, o que poderia acarretar em futuros problemas no chip. Esses erros foram reportados ao time de integração o qual realizou os devidos ajustes no RTL.

Por fim, a parte mais importante da verificação do barramento AHB era referente ao preenchimento da planilha, citada anteriormente, cujo objetivo era garantir que toda a matriz estava devidamente conectado no chip. Portanto, foram desenvolvidos diversos testes, cada teste visa garante um caminho especifico da matriz, sendo esses similares ao apresentado na Figura 13. Graças a esses padrões foi possível garantir a total cobertura dos sinais da matriz AHB, bem como verificar as conexões do barramento dentro do chip.

3.3.1.2 AXI

Para o barramento AXI o tratamento foi um pouco diferente, uma vez que o foco foi dado ao NIC301, ou AMBA *Network Interconnect CoreLink*, o qual é um componente altamente configurável permitindo criar uma infraestrutura de rede completamente compatível com AMBA e de alto desempenho. As configurações para o NIC301 foram usadas visando-o como um componente de ponte, ou seja, uma ponte de protocolo AHB para AXI, exemplificada na Figura 15 a qual apresenta o digrama de conexão do bloco.

Figura 15 – Um diagrama de blocos da interconexão do AMBA NIC301.



Fonte: (ARM, 2010).

A relação mestre escravo do NIC301 pode ficar um pouco confusa se for analisado do ponto de vista do AXI, uma vez que, para o barramento os seus mestres são conectados a interface de escravo enquanto que a matriz AHB, que seria o escravo para o AXI, é conectada na interface de mestre. Sabendo disso foram desenvolvidos testes de conexão para o AXI similares aos realizados no AHB, com o acréscimo de alguns testes exclusivos da interconexão de rede. Esses teste foram necessários devido a necessidade de se garantir que todos os sinais da lógica interna do NIC301 estavam funcionando de acordo com as especificações do projeto.

Com relação aos padrões do NIC301, foi verificado inicialmente a partir da análise do seu banco de registradores, uma modalidade de teste básica e padrão da verificação de SoC,

esse teste visava garantir que os registros internos do bloco estavam funcionando e endereçados corretamente de acordo com a documentação. Esse tipo de verificação é feita a partir de estímulos realizados nos registros por meio de escritas e leituras diversas, que variam de acordo com a característica do registro, que podem ser "*Read Only*", "*Write Only*", "*Read/Write*" e reservado.

Outro teste básico realizado foi a verificação do multiplexador responsável pela seleção dos *clocks* de entrada do bloco, ou seja, garantir que o bloco funciona corretamente independente da fonte de *clock* selecionada no momento. Nesse padrão é feita um operação simples qualquer muitas vezes, sendo que cada vez com uma frequência de funcionamento diferente, tambem é feito no teste o "check" do *reset* que é analisado da seguinte forma, após forçar o chip durante uma transação a entrar em *reset* é verificado quais são os valores esperados nos registros e o que acontece com o resultado da transação.

3.3.2 Interface de teste do MIPI/DSI

Segundo MipiAlliance (2015), a interface do processador da indústria móvel (MIPI - *The Mobile Industry Processor Interface*) trabalha em conjunto com uma interface de *display* serial de alta velocidade (DSI - *Display Serial Interface*) entre um processador, *host*, e um módulo de exibição. A interface permite que os fabricantes integrem os monitores para obter alto desempenho, baixa potência e baixa interferência eletromagnética, reduzindo a quantidade de pinos e mantendo a compatibilidade entre diferentes fornecedores.

O foco da verificação feita pelo estagiário não foi no funcionamento do MIPI/DSI, mas especificamente no modo de *debug* que funciona com os módulos de teste usados pela equipe de DFT. Essa etapa do estágio foi a mais complexa e a mais trabalhosa em comparação com as outras atividades, devido a complexidade do trabalho bem como a necessidade de se entender um pouco do funcionamento dos blocos de DFT.

Os testes foram inicialmente desenvolvidos no ambiente padrão de verificação usado pelo time de SoC. Esses testes visavam a verificação das conexões dos blocos de DFT dentro do RTL, sendo feitos de maneira funcional e formal, começando com o estimulo dos pinos externos de teste, simulando por software os sinais que seriam inseridos pelo testador. Após isso foram feitas as configurações dos blocos de DFT forçando alguns valores de sinais que não poderiam ser configurados no ambiente padrão, e ativando as configurações de teste segundo a documentação do MIPI/DSI. Por fim nesses testes foram comparados os resultados obtidos com o que era esperado de acordo com a especificação.

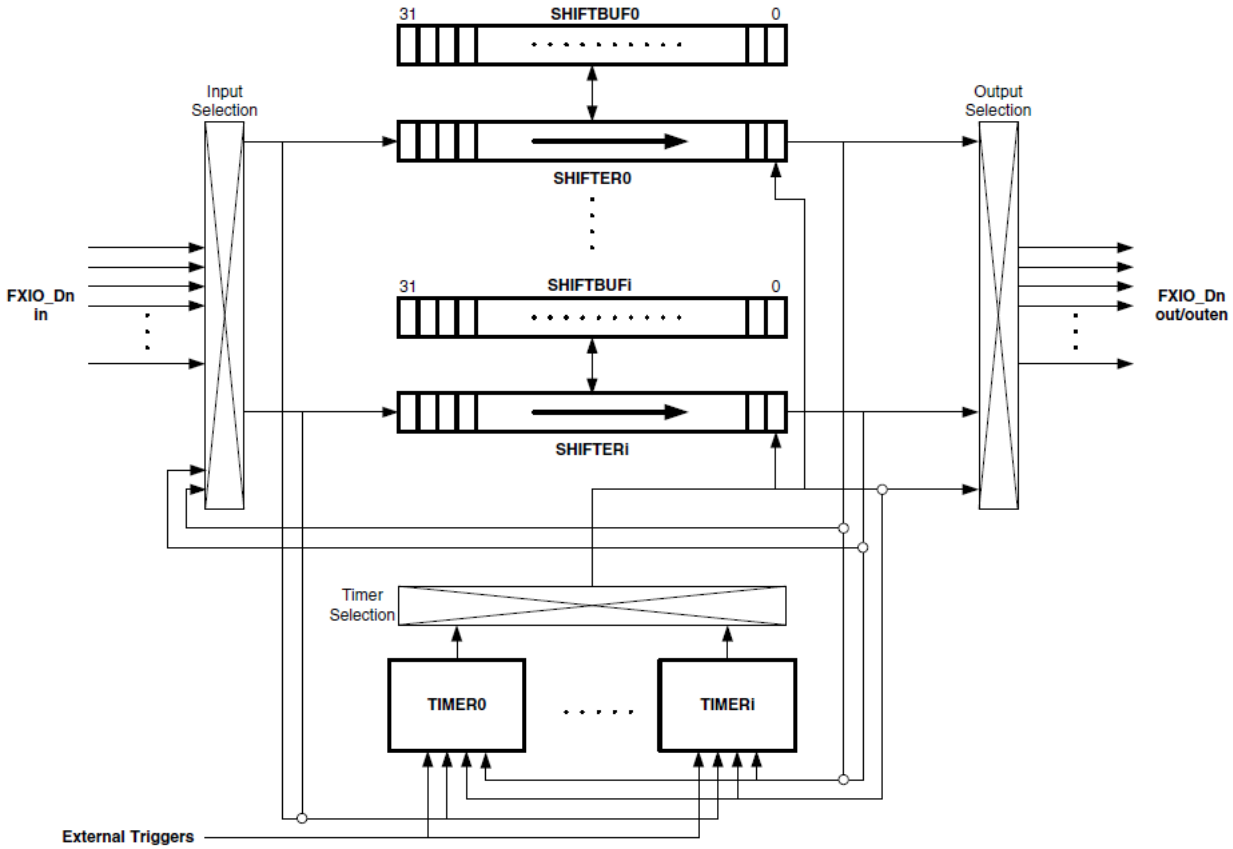
Como o ambiente normal de verificação não consegue realizar a configuração dos sinais do JTAG, foi necessário realizar testes utilizando o ambiente próprio do JTAG com capacidade SWD. Dessa foi necessário realizar um estudo prévio do funcionamento do JTAG com SWD, para poder iniciar a codificação dos testes. Estes são escrito em nível lógico baixo, ou seja, foram codificados testes em nível de escrita de registros, onde cada linha de código nada mais é do que o endereço seguido com o dado a ser escrito ou lido pelo mesmo. Após realizar todas as configurações necessárias para iniciar o chip, foi realizada as configurações de *debug* do MIPI/DSI e comparado os resultados com o esperado, garantindo assim que a interface do testados estava funcionando corretamente para o bloco.

3.3.3 Flex I/O

O FlexIO, ou *Flexible input and output peripheral*, é um modulo muito flexível de entrada e saída capaz de emular vários protocolos de comunicação serial, incluindo: UART, SPI e I2C, sendo configurável de acordo com suas necessidades de comunicação. Os principais

componentes do FlexIO são os *shifters*, *timers* e pinos. Os dados são carregados num *shifter* juntamente com um *timer* logo após são atribuído para gerar o *clock* do *shifter* e usar um pino para produzir os dados do *shifter*, na Figura 16 é apresentado o diagrama de bloco do IP.

Figura 16 – Diagrama de blocos do FlexIO.



Fonte: (NXP, 2015).

O trabalho com o FlexIO diferenciou dos demais, visto que não teve como enfoque principal a verificação do design. Isso ocorreu, pelo fato da verificação já está "madura" o suficiente para retornar 100% de cobertura. Partindo desse estado o trabalho teve enfoque em analisar quão eficientes eram os testes de verificação de SoC, esse processo foi realizado com o uso de uma ferramenta específica, descrita anteriormente na Seção 3.2.4. Por meio dos resultados dessa ferramenta foi possível verificar critérios importantes que não tinham sido percebidos pelo verificador inicialmente, o que acabou gerando alterações nos padrões do FlexIO melhorando dessa forma a cobertura do seus sinais, uma vez que os mesmos agora estavam garantindo as condições de erro trabalhadas pela ferramenta.

As primeiras condições de erros analisadas foram a de *Stuck-at-zero*, que consistem em "cravar" um determinado sinal em zero e *Stuck-at-one*, que consiste em "cravar" um determinado sinal em um, logo após a inserção do erro a ferramenta copila todos os testes aos quais estimularam o sinal durante a fase de ativação, e verifica se esses padrões são capazes de detectar o erro, reportando uma falha durante a sua copilação. Caso nenhum padrão detecte o erro durante a fase de detecção da ferramenta, no relatório final é reportado que o sinal não foi detectado na condição *Stuck-at-zero* ou em *Stuck-at-one* a depender de qual o erro não foi devotável. Por fim são apresentados quais testes foram capazes de estimular o sinal durante a copilação e as formas de ondas dos padrões com o erro estimulado, facilitando o trabalho do verificador no que diz respeito em como e onde alterar os testes para que os mesmos se tornem

capazes de detectar esse erro. Não necessariamente todos os testes precisam detectar o erro, apenas um já é suficiente para que o erro se torne detectável do ponto de vista da ferramenta.

Nessas condições de erros os sinais os quais apresentaram problemas em sua maioria estavam relacionados aos modos de *power* ou *trigger*. Isso ocorreu devido a forma que os testes foram estruturados, porque para os sinais analisados os mesmos não interferiam nos resultados esperados pelo verificador naquelas condições a qual o padrão estava proposto. Dessa maneira foram realizadas alterações nos testes de modo que os mesmos além de analisarem se o chip estava funcionando nesse casos, também fossem capazes de detectar alterações nos sinais de controle de *power modes* reportando erros caso os sinais não estivam de acordo com a sua especificação de funcionamento.

Para a outra modalidade de erro estimulado que foi *Port Negated*, a qual consiste em colocar um inversor no sinal, ou seja, o sinal é invertido em todos os momentos de seu funcionamento, a ferramenta avaliou esse erro da mesma maneira que os anteriores. Nessa análise os sinais que apresentaram maiores números de falhas na detecção eram sinais de *clock*. Isso ocorria muito devido a esse tipo de erro apenas gerar um atraso de um ciclo no sinal o que não gerava problema do ponto de vista do padrão. Para detecção desse tipo de erro foi necessário realizar testes formais, uma vez que a detecção do erro funcionalmente ou era muito complexa ou era inviável. Portanto foram inseridas *assertions* dentro dos padrões para que os mesmos pudessem por fim detectar esses erros, limpando o relatório de falhas da ferramenta e assim possibilitando uma cobertura completa dos sinais tanto do ponto de vista normal do *Toggle* como do ponto de vista dos erros de *Stuck-at-zero*, *Stuck-at-one* e *Port Negated*.

4 Conclusão

O período decorrido ao longo do estágio foi consideravelmente proveitoso para o crescimento pessoal e profissional do estagiário, uma vez que, por muitas vezes, foi necessária a comunicação com colegas de diferentes setores para melhor entendimento do problema a ser resolvido. A responsabilidade tomada, por mais que limitada devido à posição, ainda sim foi grande a ponto de que as atividades tivessem impacto direto no desenvolvimento de projetos. Dessa forma, a imersão no ambiente empresarial se deu de maneira completa, com prazos a serem cumpridos, consequências de decisões equivocadas, o que acabou gerando um maior desempenho por parte do aluno. Com isso, cumpriu-se o objetivo acadêmico e pessoal, que era o de experienciar o funcionamento dentro de uma empresa, aprender e se adequar a ele.

Quanto as atividades efetuadas, tinham como foco prover suporte para os demais integrantes do time de verificação de SoC, assim como realizar o desenvolvimento de testes dos blocos aos quais fosse alocado no projeto vigente. Dessa maneira, o processo de estágio tomou caráter prático e teve impacto no fluxo de trabalho da empresa, já que, as atividades desenvolvidas tinham impacto direto no funcionamento da empresa e agregassem valor aos produtos desenvolvidos.

Por fim, durante toda a extensão dessas atividades, foi necessário o emprego de várias habilidades comportamentais, denominadas *soft skills*. Essas habilidades envolvem capacidades interpessoais, e seu emprego é altamente desejado no ambiente de trabalho. Comportamentos de interesse geralmente envolvem proatividade, criatividade, gentileza, interesse e empenho ao se realizar atividades. Dessa maneira, todas as atividades empregadas durante o período de estágio tiveram, como embasamento, o emprego destes valores para que o ambiente de trabalho se mantivesse sempre saudável. A partir do retorno de colegas de trabalho, foi possível então aprender como se portar de maneira mais agradável e produtiva no ambiente empresarial, assim como descobrir pontos a serem trabalhados e aperfeiçoados.

Referências

ABOUT NXP|BRASIL. **Sobre a NXP Semiconductors no Brasil**. [S.l.], 2016. Disponível em: <https://www.nxp.com/about/about-nxp/about-nxp/worldwide-locations/nxp-no-brasil:BRAZIL>. Acesso em: 23 de Outubro de 2018. Citado na página 3.

ABOUT NXP|WORLDWIDE LOCATIONS. **Sobre a NXP Semiconductors no mundo**. 2016. Disponível em: <https://www.nxp.com/about/about-nxp/about-nxp/worldwide-locations:GLOBAL_SITES>. Acesso em: 23 de Outubro de 2018. Citado na página 4.

AMOS, M. J. S. **Principles of Transistor Circuits**. 6. ed. United Kingdom: Newnes, 2000. Citado na página 5.

ARM. **AMBA Specification (Rev. 02)**. 1999. Disponível em: <http://soc.eecs.yuntech.edu.tw/Course/SoC/doc/amba.pdf>. Acesso em: 11 de Novembro de 2018. Citado na página 18.

ARM. **AMBA Network Interconnect (NIC-301)**. 2010. Disponível em: <https://static.docs.arm.com/ddi0397/g/DDI0397G_amba_network_interconnect_nic301_r2p1_trm.pdf>. Acesso em: 11 de Novembro de 2018. Citado na página 19.

ARM. **Arm® Debug Interface Architecture Specification ADIv5.0 to ADIv5.2**. 2017. Disponível em: <https://static.docs.arm.com/ihi0031/d/debug_interface_v5_2_architecture_specification_IHI0031D.pdf>. Acesso em: 11 de Novembro de 2018. Citado na página 9.

BERGERON, J. **Writing Testbenches: Functional Verification of HDL Models**. 2. ed. Norwell,: Kluwer Academic Publishers, 2003. Citado na página 15.

BOYLESTAD, L. N. R. L. **Electronic devices and circuit theory**. Boston: Vernon R. Anthony, 2013. Citado na página 5.

FLYNN, W. L. M. J. **Computer System Design: System-on-Chip**. 1. ed. London: Wiley, 2011. Citado na página 8.

FREESCALE. **Sobre a Freescale no mundo**. [S.l.], 2005. Disponível em: <http://phx.corporate-ir.net/phoenix.zhtml?c=196520&p=irol-newsarticle_print&id=832346>. Acesso em: 23 de Outubro de 2018. Citado na página 3.

JAS, A. Deterministic test vector compression/decompression for systems-on-a-chip using an embedded processor. **Journal of Electronic Testing**, 2002. Citado na página 12.

LUPI, C. **Nova cartilha esclarecedora sobre a lei do estágio**: Ministério do trabalho e emprego. Brasília, 2008. Citado na página 1.

MANO, M. M. **Logic and Computer Design Fundamentals**. 5. ed. Los Angeles: Julie Bai, 2015. Citado na página 7.

MIPIALLIANCE. **MIPI Display Serial Interface (MIPI DSI)**. 2015. Disponível em: <https://mipi.org/specifications/dsi>. Acesso em: 11 de Novembro de 2018. Citado na página 20.

NXP. **Understanding FlexIO**. 2015. Disponível em: <https://community.nxp.com/docs/DOC-105640>. Acesso em: 11 de Novembro de 2018. Citado na página 21.

PINTO, L. A. V. **Osciladores**. 2000. Disponível em: <http://www.vargasp.com/download/livros/Oscilers.pdf>. Acesso em: 29 de Outubro de 2018. Citado na página 7.

RAZAVI, B. **Fundamentals of Microelectronics**. 2. ed. Los Angeles: Don Fowley, 2014. Citado 2 vezes nas páginas 5 e 6.

SCHAUM, H. P. H. **Theory and Problems of Signals and Systems**. 1. ed. New Youk: The McGraw-Hill Companies, 1995. Citado na página 11.

SYNOPSYS. **Synopsys ASIC Tutorial**. 2015. Disponível em: <http://www.ece.utep.edu/courses/web5375/Labs_files/Synopsys_tutorial_v11.pdf>. Acesso em: 03 de Novembro de 2018. Citado na página 12.

TOCCI, R. J. **Digital Systems Principles and Applications**. 10. ed. New Jersey: Vern Anthony, 2007. Citado na página 7.

WANG, L.-T. **VLSI Test Principles and Architectures**. 1. ed. Boston: Morgan Kaufmann, 2006. Citado na página 13.

XJTAG. **What is JTAG and how can I make use of it?** 2015. Disponível em: <https://www.xjtag.com/wp-content/uploads/xjtag-ebook_what-is-jtag-en.pdf>. Acesso em: 11 de Novembro de 2018. Citado na página 9.