



Universidade Federal
de Campina Grande

Centro de Engenharia Elétrica e Informática
Departamento de Engenharia Elétrica

PEDRO HENRIQUE SILVA CAVALCANTE

RELATÓRIO DE ESTÁGIO SUPERVISIONADO
LABORATÓRIO DE EXCELÊNCIA EM MICROELETRÔNICA DO NORDESTE - XMEM

Campina Grande
2020

PEDRO HENRIQUE SILVA CAVALCANTE

RELATÓRIO DE ESTÁGIO SUPERVISIONADO
LABORATÓRIO DE EXCELÊNCIA EM MICROELETRÔNICA DO NORDESTE - XMEM

Relatório de Estágio Supervisionado submetido à Coordenação do Curso de Graduação em 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.

Orientador:

D. Sc. Marcos Ricardo de Alcântara Morais

Campina Grande

2020

PEDRO HENRIQUE SILVA CAVALCANTE

RELATÓRIO DE ESTÁGIO SUPERVISIONADO
LABORATÓRIO DE EXCELÊNCIA EM MICROELETRÔNICA DO NORDESTE - XMEM

Relatório de Estágio Supervisionado submetido à Coordenação do Curso de Graduação em 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.

Aprovado em / /

D. Sc. Marcos Ricardo de Alcântara Morais
UFCG

D. Sc. Antonio Marcus Nogueira Lima
UFCG

Campina Grande
2020

AGRADECIMENTOS

Agradeço primeiramente à meus pais, Nenem e Boião, por terem feito de tudo para que eu pudesse hoje estar onde estou.

Agradeço também a toda minha família por terem me dado muito amor e apoio durante toda a jornada e por sempre acreditarem em mim.

A minha companheira Fernanda que esteve junto durante todos os momentos, sempre me lembrando da minha capacidade e me dando forças para vencer todos os desafios.

A todos que fazem parte do laboratório XMEN em especial a Klynger Dantas, José Iuri e Matheus Andrade, responsáveis pelo incentivo para o desenvolvimento deste trabalho e indispensáveis para o seu término dentro do prazo.

RESUMO

Neste relatório, descrevem-se as atividades que foram realizadas durante o período de estágio supervisionado na área de Microeletrônica no Laboratório de Excelência em Microeletrônica do Nordeste. Este trabalho descreve o processo de verificação funcional de blocos digitais que foram realizadas durante o estágio. Para que isso fosse possível, um estudo do fluxo de desenvolvimento utilizado pelos parceiros do laboratório foi realizado. A verificação desses blocos necessitou de pesquisa na área de comunicações digitais para que a funcionalidade dos mesmos fosse compreendida.

Palavras-chave: Microeletrônica, Verificação, DSP, Relatório de Estágio.

ABSTRACT

This report describes the activities developed during the period of supervised training in the area of Microelectronics at the Laboratory of Excellence in Microelectronics in the Northeast. This work describes the functional verification process of digital blocks that were performed during the internship. For this, a study of the development flow used by the partner laboratories was carried out. The verification of these blocks required research in the area of digital communications to understand their functionality.

Keywords: Microelectronics, Verification, DSP, Report.

LISTA DE ILUSTRAÇÕES

Figura 1 – Equipe de verificadores do Laboratório XMEN	10
Figura 2 – Fluxo do Projeto de criação de um Chip.	11
Figura 3 – Porcentagem do tempo total utilizado na verificação de um projeto de CI/ASIC.	12
Figura 4 – Histórico de utilização de metodologias de verificação em pesquisa es- pontânea com desenvolvedores.	13
Figura 5 – Exemplo de Arquitetura de <i>testbench</i> UVM.	15
Figura 6 – Modelo conceitual da PMD.	17
Figura 7 – Blocos de informação com a presença de símbolos piloto	18
Figura 8 – <i>Testbench</i> com a presença de VIPs	20
Figura 9 – Fluxo de análise de cobertura.	21
Figura 10 – Exemplo de relatório de cobertura.	22

LISTA DE ABREVIATURAS E SIGLAS

UFCG	Universidade Federal de Campina Grande
DEE	Departamento de Engenharia Elétrica
DSP	Processador Digital de Sinais
ASIC	Circuitos Integrados de Aplicação Específica
VIP	Verification Intellectual Property
IP	Intellectual Property
RTL	Register Transfer Level

SUMÁRIO

	Lista de ilustrações	6
1	INTRODUÇÃO	9
1.1	Laboratório de Excelência em Microeletrônica do Nordeste - XMEN	9
2	FUNDAMENTAÇÃO TEÓRICA	11
2.1	Fluxo do Projeto	11
2.2	Verificação Funcional	12
2.3	<i>Universal Verification Methodology - UVM</i>	13
3	ATIVIDADES DESENVOLVIDAS	16
3.1	Verificação de blocos	16
3.1.1	Bloco de recuperação de polarização	16
3.1.2	Bloco de detecção de Piloto	18
3.1.3	Bloco estimador de tempo	19
3.2	Utilização e análise de VIP	19
3.2.1	O VIP	19
3.2.2	Análise da utilização do VIP	19
3.3	Análise de Cobertura	20
4	CONCLUSÕES	23
	REFERÊNCIAS	24

1 INTRODUÇÃO

Este relatório tem como objetivo descrever as atividades desenvolvidas durante o período de Estágio Supervisionado realizado no Laboratório de Excelência em Microeletrônica do Nordeste - XMEN tendo o estágio ocorrido durante o período do dia 28 de setembro de 2020 a 9 de dezembro de 2020 totalizando uma carga horária de 188 horas.

O estágio em questão foi realizado em Microeletrônica, especificamente com verificação funcional, na qual foram atribuídas ao estagiário atividades as seguintes atividades:

- Compreender o funcionamento dos principais blocos que compõem os modelos de Processadores de Sinais Digitais (DSP – Digital Signal Processors) de um transceptor coerente para comunicação em alta velocidade;
- Estudo e familiarização com o fluxo de referência da Idea para projeto lógico dos circuitos digitais e estratégias de verificação;
- Verificação dos circuitos digitais seguindo a linguagem de descrição de hardware de referência usado na empresa (SystemVerilog) em conjunção com a metodologia de verificação UVM;
- Análise de cobertura;
- Análise de uso de um VIP dentro da estrutura convencional de testbench.

1.1 LABORATÓRIO DE EXCELÊNCIA EM MICROELETRÔNICA DO NORDESTE - XMEN

A iniciativa foi criada em 2016 por pesquisadores da UFCG, visando o estabelecimento de um modelo de formação de recursos humanos, pesquisa, desenvolvimento e inovação na área de microeletrônica anteriormente chamado de PEM - Programa de Excelência em Microeletrônica.

Atualmente no laboratório conta com a participação de 19 alunos de graduação e 9 engenheiros, os desenvolvedores que compõem o laboratório são divididos nas três equipes de desenvolvimento de um *chip*, a saber *frontend*, responsável pela construção lógica e arquitetural do circuito a ser projetado, a verificação, responsável por garantir que a correta funcionalidade do circuito projetado e, por fim, a equipe de *backend* responsável por realizar a construção física do circuito projetado.

Além da divisão entre as equipes de desenvolvimento existem também divisão em frentes de trabalho, em que há desenvolvedores trabalhando em projetos de P&D, projetos internos do laboratório e projetos junto a parceiros.



Figura 1 – Equipe de verificadores do Laboratório XMEN

Fonte – Própria

2 FUNDAMENTAÇÃO TEÓRICA

Nessa sessão será apresentado uma contextualização sobre o trabalho da verificação na área da microeletrônica, para evidenciar os termos utilizados ao longo do texto, serão explicados conceitos fundamentais do setor.

2.1 FLUXO DO PROJETO

O processo de criação de um chip requer uma equipe capacitada, contendo integrantes com habilidades específicas em cada etapa do projeto. Portanto, existem diferentes grupos dentro de uma empresa de semicondutores que visam ajudar a gerenciar e conduzir o estado atual do projeto, promovendo as ações necessárias para otimizar o projeto.

No caso de analisar o escopo de produção e assumir o maior nível de abstração, pode-se traçar um fluxograma do projeto, conforme mostrado na Figura 2.

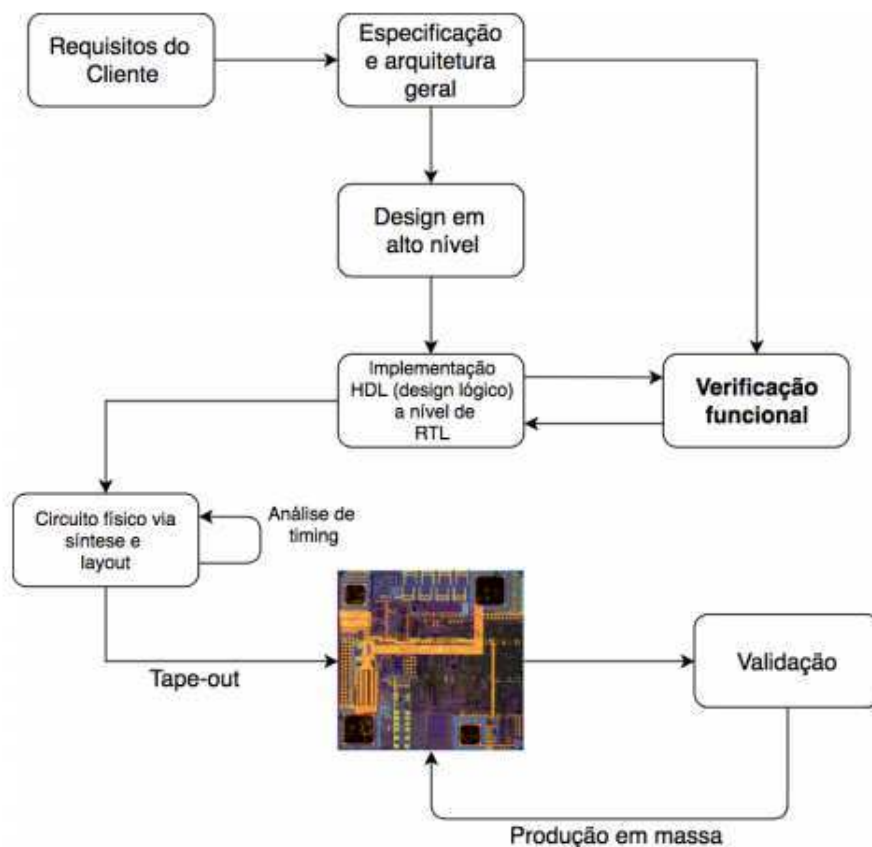


Figura 2 – Fluxo do Projeto de criação de um Chip.

Fonte – (CARVALHO, 2018)

A primeira etapa é o requerimento do cliente com a ideia do projeto, após isso, é

gerada uma especificação do mesmo e proposto uma arquitetura a ser implementada, com essas etapas concluídas o trabalho do projetista e do verificador são iniciados. Finalmente obtemos um RTL verificado, no qual pode ser sintetizado gerando assim uma *netlist* que será utilizada nos processos subsequentes.

2.2 VERIFICAÇÃO FUNCIONAL

Verificação funcional busca uma equivalência entre o que foi especificado e o RTL implementado. Essa é uma tarefa que geralmente toma uma parcela significativa do tempo e esforço na maioria dos projetos de design de sistemas eletrônicos.

É apresentado na Figura 3 a porcentagem do tempo total utilizado na verificação de um projeto de CI/ASIC. Pode-se ver dois extremos neste gráfico. Em geral, projetos que demandam parcela menor de tempo na verificação normalmente estão trabalhando em projetos com uma boa quantidade de IP de design pré-verificado, que é integrado para criar um novo produto. No outro extremo, projetos que gastam uma quantidade significativa de tempo na verificação, muitas vezes têm uma alta porcentagem de IP de design recém-desenvolvido que deve ser verificado (FOSTER, 2020).

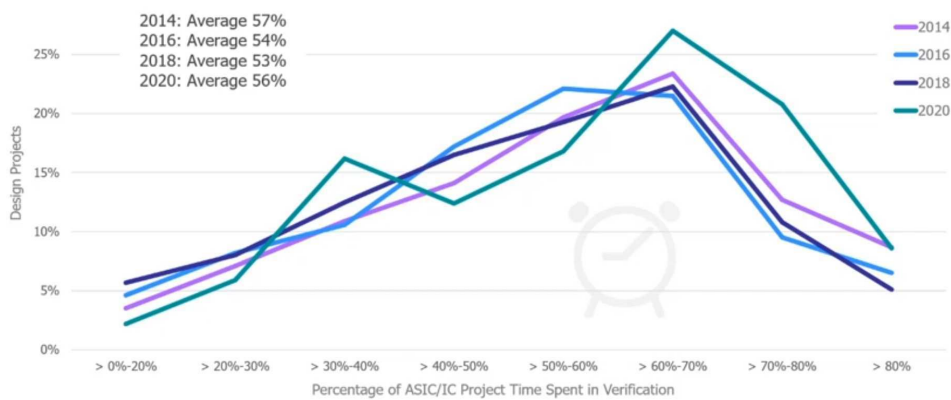


Figura 3 – Porcentagem do tempo total utilizado na verificação de um projeto de CI/ASIC.

Fonte – (FOSTER, 2020)

O design de um chip pode consistir de centenas de milhares de linhas de código, tornando o processo de depurar manualmente algo impraticável, cabe então ao engenheiro de verificação o trabalho de buscar problemas na implementação do hardware, falhas em sinais ou *bugs*, quando o design não funciona conforme a especificação. O engenheiro de verificação identifica e expõe essas falhas executando simulações exaustivas no design.

O processo de verificação funcional pode ser trabalhoso devido as diversas possibilidades de casos de teste que podem existir até em um simples design. Entretanto, a

verificação pode ser abordada por vários métodos que em conjunto, podem ser de bastante auxílio para um completo trabalho de verificação.

2.3 UNIVERSAL VERIFICATION METHODOLOGY - UVM

A Metodologia Universal de Verificação (UVM) é um método padronizado para verificação de circuitos digitais. Os principais objetivos do UVM são a reutilização e modularização dos seu componentes a fim de determinar um padrão na engenharia da indústria e de verificação.

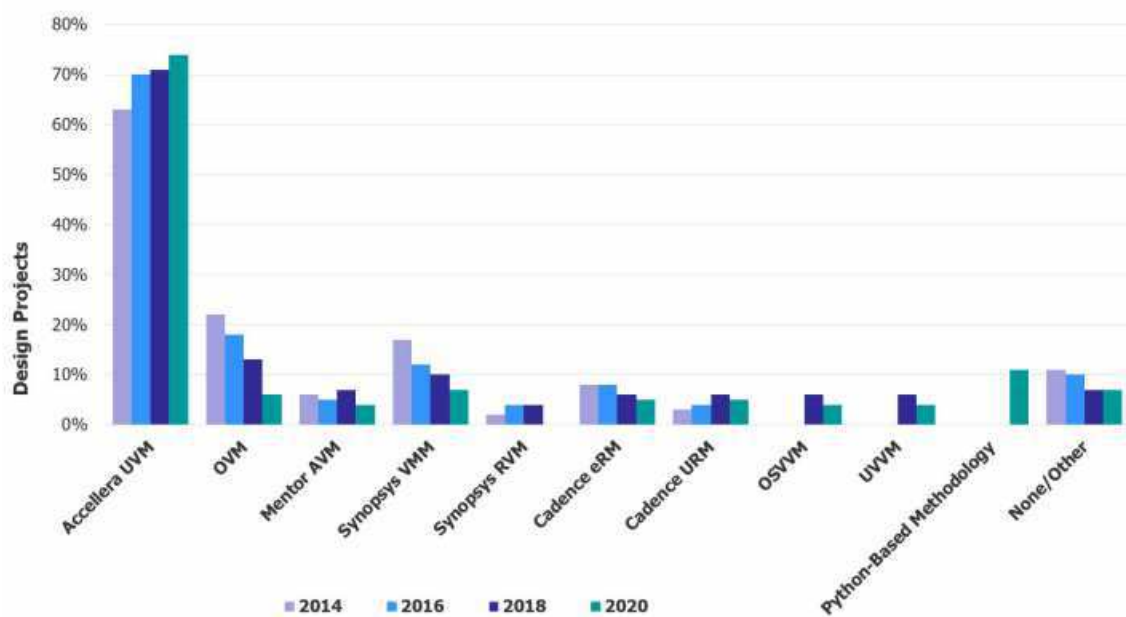


Figura 4 – Histórico de utilização de metodologias de verificação em pesquisa espontânea com desenvolvedores.

Fonte – (FOSTER, 2020)

Dentro de um fluxo de verificação utilizando a biblioteca e a metodologia do UVM é necessário criar os UVCs (Universal Verification Component). Cada um deles desempenha um papel dentro da topologia do *testbench*. Esses componentes herdam características das classes padrão da biblioteca e expandem as funções e tarefas de acordo com as necessidades do verificador. A utilização de orientação a objetos é uma das principais razões que justificam a adoção da metodologia e sua larga aceitação na indústria, como pode ser visto na Figura 4.

Os ambientes de verificação utilizando UVM utilizam a metodologia de modelagem TLM (*Transaction Level Modeling*) por isso é necessário que um dos componentes criados pelo verificador é a transação, nela os sinais do DUT são abstraídos em termos de com-

plexidade e passam a compor uma classe de alto nível em que as características temporais são ignoradas e as características funcionais dos sinais são levadas em consideração.

O próximo componente do ambiente de verificação é a sequência. Nela são definidas a forma como as transações serão geradas para serem convertidas futuramente em estímulos. Cada uma das sequências identificam, em via de regra, um caso de uso do bloco ou agrupam um conjunto de estímulos relativos a uma única interface independente no DUT.

Outro componente típico dos ambientes escritos utilizando a metodologia UVM é o *driver* que funciona como um fonte de estímulos. Esse componente é responsável por receber as transações geradas pelas sequências e converter o nível de abstração de TLM, ou seja, de transação, para o nível dos sinais. Esse componente acessa diretamente a interface de sinais dos blocos e realiza toda a lógica de controle temporal e *handshake*.

O *monitor* é o componente responsável por fazer a tarefa oposta ao *driver*, converter os sinais em transação. Esse componente monitora constantemente as trocas de informação entre o *driver* e o DUT de modo que após uma transação ser completamente tratada pelo *driver* o *monitor* possa reconstruir ela de modo exatamente igual ao que foi gerado na sequência. Essa independência é essencial para garantir que os estímulos foram corretamente gerados de acordo com as transações que chegaram no *driver*. Nesse mesmo componente também é encapsulado as declarações de cobertura funcional, os *covergroups* e *coverpoints*. Dentro de um fluxo de verificação a cobertura é a parte responsável por checar o nível de exercício que foi testado no DUT. A principal diretiva de métricas que indicam o fim da verificação funcional é o nível de cobertura atingido.

O modelo de referência é a parte do *testbench* responsável por reproduzir o comportamento ideal do bloco, para que as saídas geradas por esse componente sejam consideradas a ideal para referências em casos que esses valores divergem dos observados no DUT. A implementação desse modelo é feita normalmente em linguagens de alto nível, a exemplo de Python e C++, por isso o nível de confiabilidade é bastante elevado.

Por fim, outro componente que desempenha papel fundamental nos ambientes de verificação funcional é o comparador. Esse UVC é o responsável por comparar as saídas do model de referência e o DUT de modo que, dada a politica de tolerância, as saídas não diverjam.

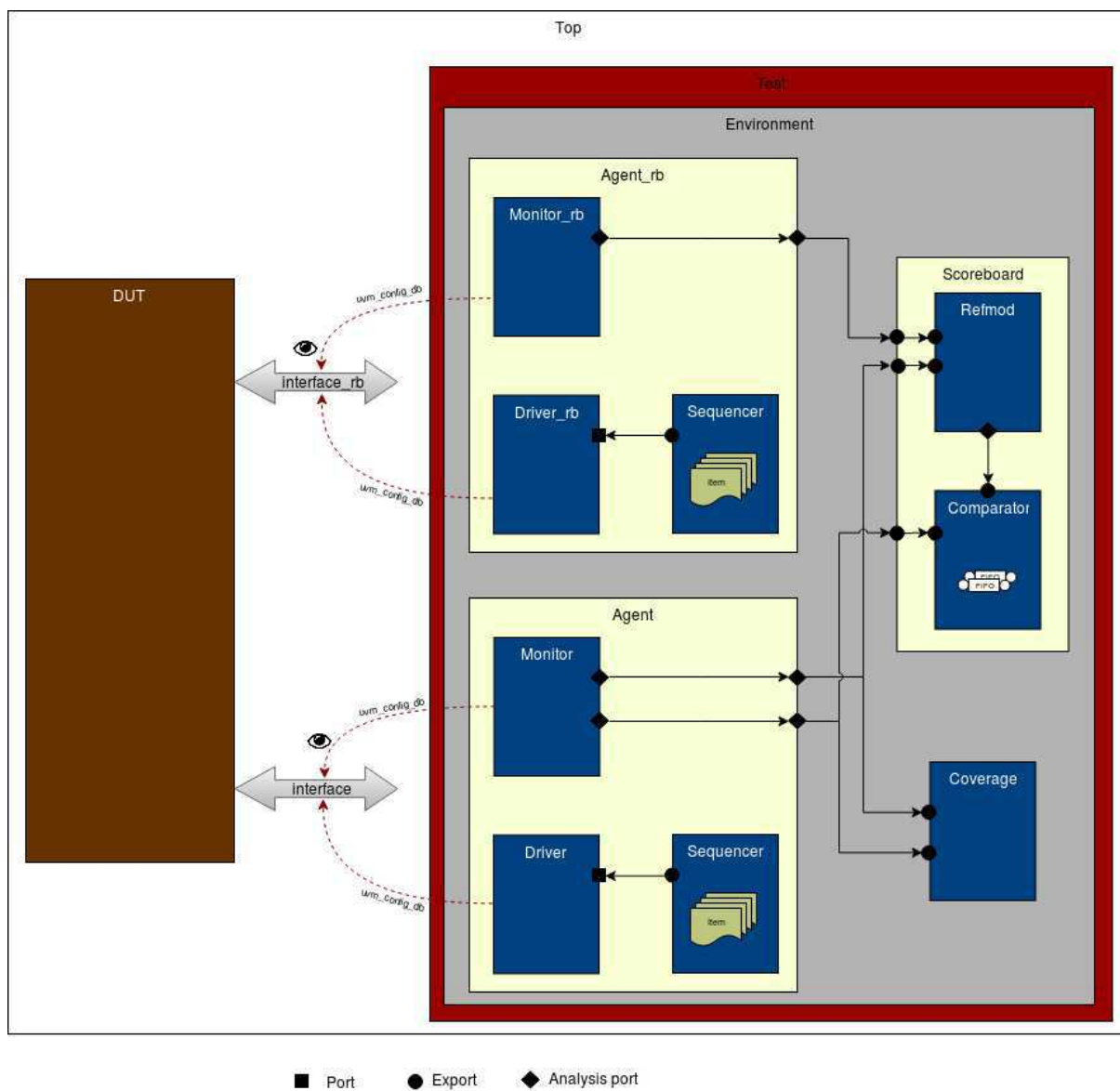


Figura 5 – Exemplo de Arquitetura de *testbench* UVM.

Fonte – Própria

3 ATIVIDADES DESENVOLVIDAS

Nessa seção é descrito as atividades que foram realizadas pelo estagiário durante a vigência do período correspondente no Laboratório de Excelência em Microeletrônica do Nordeste - XMEN.

3.1 VERIFICAÇÃO DE BLOCOS

Para as atividades de verificação de bloco duas ferramentas foram utilizadas para que se seguisse o fluxo da empresa, sendo elas um gerador de *testbench*, esse entregava ao estagiário uma estrutura inicial padronizada para o *testbench*, a outra ferramenta foi um gerador de dados, esse era responsável por gerar os dados para todos os blocos do DSP, sendo completamente configurável, era possível adicionar efeitos como *jitter*, dispersão cromática, erro de fase entre outros efeitos encontrados em canais de fibra óptica.

3.1.1 BLOCO DE RECUPERAÇÃO DE POLARIZAÇÃO

A dispersão dos modos de polarização, PMD, é uma propriedade fundamental da fibra óptica monomodo e componentes, no qual a energia do sinal em um dado comprimento de onda é decomposta em dois modos de polarização ortogonais com velocidades de propagação levemente diferentes (JUNIOR, 2003).

Essa pequena diferença entre os modos de polarização é chamada de atraso diferencial de grupo, esse atraso causa sérias degradações na capacidade de transmissão.

Os efeitos da PMD se parecem com os da dispersão cromática, sendo o último caracterizado por uma diferença entre as velocidades de comprimentos de onda distintos, porém existe uma importante diferença. A dispersão cromática é um fenômeno relativamente estável e pode ser calculada a partir da soma das suas partes. O local e valor dos compensadores de dispersão cromática podem ser planejados antecipadamente.

A PMD em fibras ópticas monomodo, em qualquer comprimento de onda do sinal não é estável, forçando os projetistas de sistemas a fazer previsões estatísticas dos efeitos da PMD e fazendo a compensação passiva impossível.

Uma fibra longa é representada na Figura 6, como uma série de comprimentos aleatórios, segmentos birrefringentes de rotações aleatórias. Cada segmento exibe modos de polarização lenta e rápida, geralmente com algum sinal de luz propagando em cada um. Este processo é repetido diversas vezes ao longo do enlace de fibras ou enlaces de comunicação, e é este o mecanismo que causa o atraso diferencial de grupo na maioria

dos sistemas de fibras já instaladas mostrando uma dependência do comprimento de onda das condições ambientais (JUNIOR, 2003).

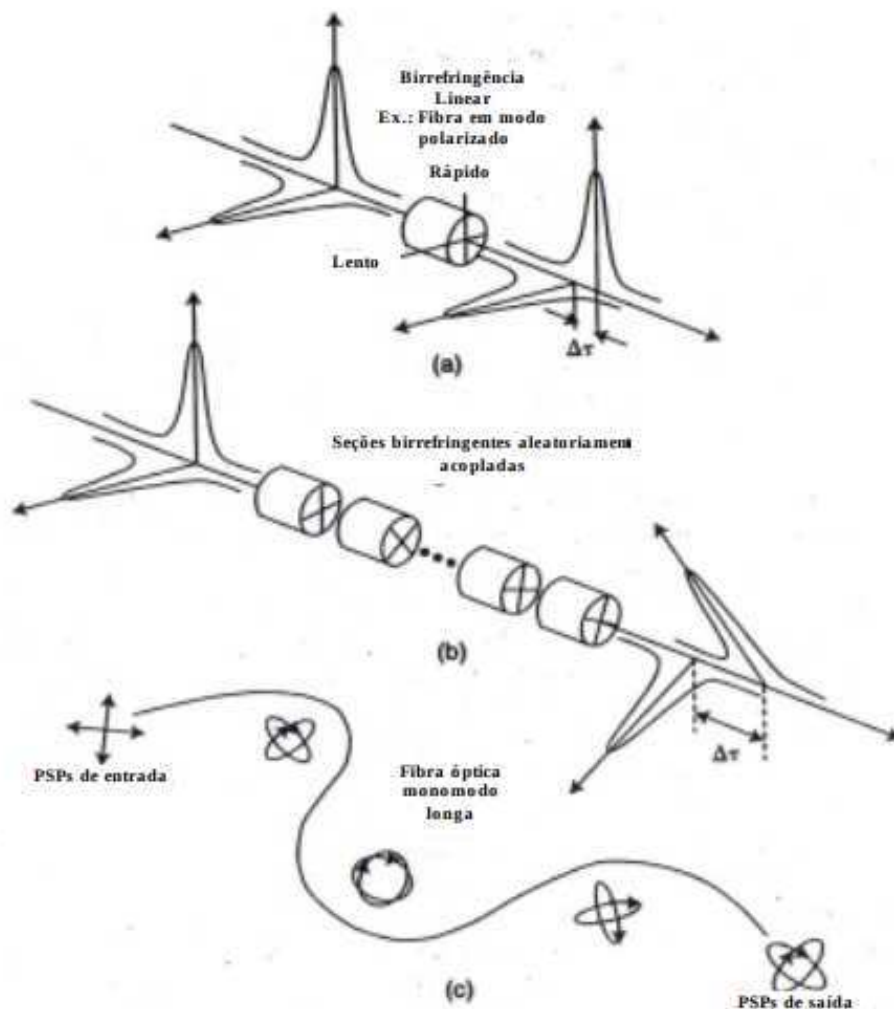


Figura 6 – Modelo conceitual da PMD.

Fonte – (JUNIOR, 2003)

Portanto, dado o exposto, faz-se necessário a adição de um bloco que seja capaz de corrigir esse efeito. Para isso foi desenvolvido esse módulo que junto com um filtro adaptativo recupera o sinal eliminando os efeitos relativos a polarização. Ao estagiário coube realizar as atividades relativas a verificação funcional desse bloco.

Para a verificação funcional desse bloco, foi utilizado o ICM para gerar o *testbench*. No decorrer do processo a correspondência entre o modelo funcional e a implementação foi alcançada. Entretanto o RTL não atingiu as especificações desejadas em termos de área e potência, portanto foi solicitado ao projetista que refizesse o bloco assim coube ao estagiário aguardar a conclusão do mesmo com uma nova arquitetura para reiniciar o fluxo de verificação e isso não ocorreu durante a vigência do estágio.

3.1.2 BLOCO DE DETECÇÃO DE PILOTO

No contexto de comunicações digitais a estimação dos efeitos que acometem o sinal recebido é uma tarefa extremamente custosa, tanto em termos matemáticos como em termos temporais. Devido a isso uma das técnicas largamente utilizadas é a inserção de símbolos com informação já conhecida pelo receptor, a estes denominamos símbolos pilotos.

Na situação de um DSP-RX, uma das primeiras atividades realizadas pelo bloco é a identificação dos símbolos pilotos que foram transmitidos. Após essa identificação os demais blocos conseguem operar de maneira adequada e corrigir os efeitos corretamente. Portanto foi desenvolvido um bloco responsável pela identificação da posição do piloto e coube ao estagiário as atividades de verificar esse bloco.

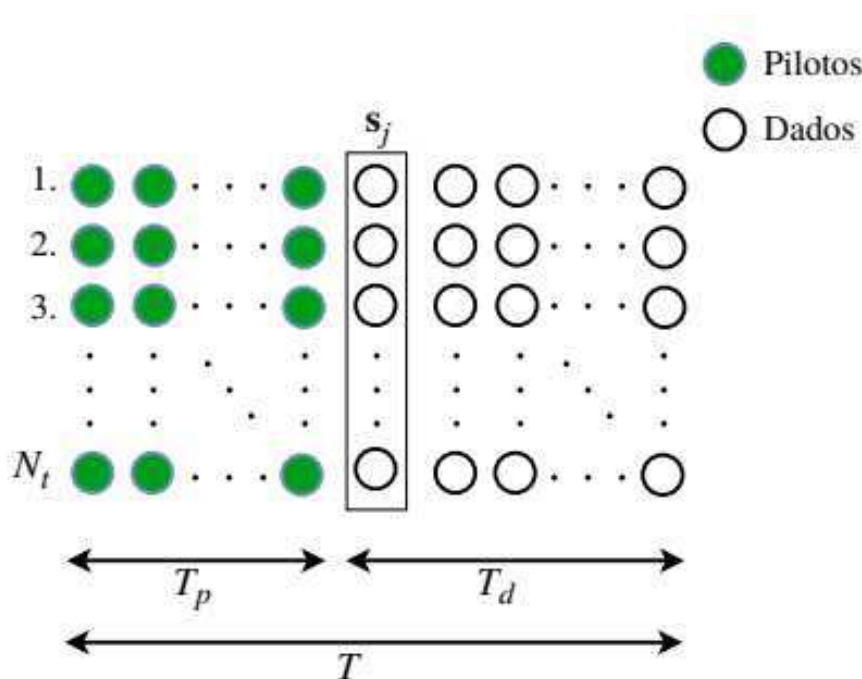


Figura 7 – Blocos de informação com a presença de símbolos piloto

Fonte – (SANTOS, 2016)

Assim como os demais blocos foi utilizado o ICM e o gerador de dados. Nesse aspecto ocorreram problemas com relação a convergência do bloco, ou seja, a posição do piloto não permanecia fixa, causando diversas problemáticas para o sistema como um todo. Foi identificado a fonte do problema, se tratava de um erro nos dados gerados e não no bloco. Devido a isso, o gerador de dados precisou ser corrigido impactando a verificação de todos os outros blocos do receptor. Na vigência do estágio, após identificação de vários *bugs* e solução por parte do projetista, conseguiu-se atingir o nível de cobertura funcional necessário e não houve divergências entre o modelo de referência e o RTL.

3.1.3 BLOCO ESTIMADOR DE TEMPO

A sincronização de símbolos ou recuperação de tempo é uma das funções de receptor mais críticas em sistemas de comunicação síncrona. O relógio do receptor deve ser continuamente ajustado em sua frequência e fase para otimizar os instantes de amostragem do sinal de dados recebido e para compensar os desvios de frequência entre os osciladores usados nos circuitos de relógio do transmissor e receptor.

Esse bloco, diferentemente dos demais, está localizado no lado *host* do DSP. Essa parte do DSP realiza a comunicação por meios elétricos e não ópticos, portanto a codificação utilizada é a codificação de linha ou PAM. Devido a isso foi utilizada outra configuração do gerador de dados para gerar estímulos com a modulação correta. Vale ressaltar, também, que a codificação nesse caso é serial por isso a recuperação de tempo se torna uma atividade crítica para o funcionamento do sistema.

As atividades desenvolvidas pelo estagiário foi também a verificação funcional desse bloco. Diferente das demais, para esse ambiente foi desenvolvido um *monitor* funcional para checar se a estimação de amostras na saída do bloco correspondia a que foi inserida efetivamente, assim adicionando mais uma camada de checagem da correção funcional. Durante a vigência desse estágio o bloco foi completamente verificado.

3.2 UTILIZAÇÃO E ANÁLISE DE VIP

Em paralelo à verificação dos blocos, o estagiário ficou responsável por testar um VIP recém desenvolvido a fim de reportar possíveis falhas e avaliar a viabilidade do uso.

3.2.1 O VIP

Chamado de *vip_data_gen* este tinha o papel de facilitar e acelerar o processo de integração do gerador de dados, já mencionando anteriormente, ao *testbench* UVM. Em resumo o VIP entregava além do gerador um *driver*, *monitor* e cobertura para todos os dados gerados fazendo com que o verificador passe a concentrar seus esforços (estimular, monitorar e cobrir) apenas aos sinais de controle.

3.2.2 ANÁLISE DA UTILIZAÇÃO DO VIP

Apesar de ser a primeira experiência com a utilização de um VIP, o processo de conectar e configura-lo em algo com simples e intuitivo, o uso se mostrou bem eficaz ao ponto de poder reduzir o tempo de desenvolvimento do *testbench* a poucas horas e o processo da verificação ficar mais dedicado a busca de *bugs* no bloco. Após o uso do VIP foram dadas algumas sugestões como por exemplo:

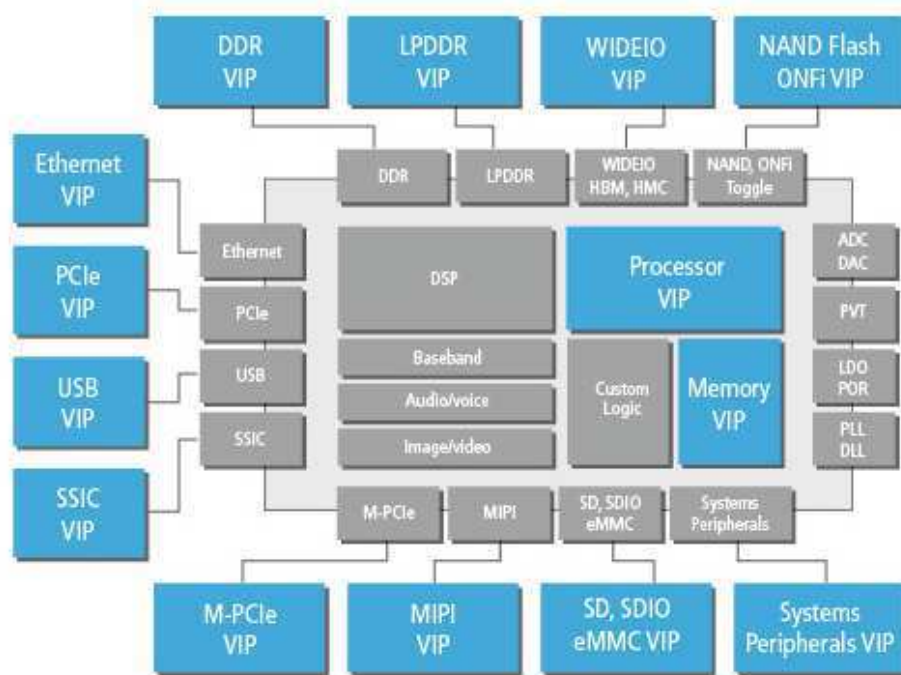


Figura 8 – *Testbench* com a presença de VIPs

Fonte – (YUN et al., 2011)

- Melhorias na cobertura implementada pelo VIP;
- Ajustes em valores padrão de configuração;
- Implementação de função para atingir a cobertura de forma mais rápida.

3.3 ANÁLISE DE COBERTURA

Para todos os blocos que foram verificados foi realizada uma análise de cobertura mais rigorosa. Para essa análise foi considerada diferentes métricas de cobertura, sendo elas: funcional, alternância, expressões e blocos. A utilização dessas métricas garantem que o código foi devidamente exercitado e nenhum buraco de cobertura funcional escapou do processo de verificação.

A primeira métrica utilizada foi cobertura funcional. Esta é a mais comum em atividades de verificação e envolve a utilização de recursos próprios da linguagem System-Verilog, como por exemplo, *covergroups*, *coverpoints*, *bins*, *cross*, etc. A análise é feita em uma ferramenta própria de cobertura que gerencia todas as execuções e coleta os dados respectivos.

Cobertura de alternância refere-se a checagem de integridade e exercício dos sinais. A métrica para isso é a alternância de cada um dos bits de todas as variáveis utilizadas. Com isso é possível identificar potenciais erros com relação as operações lógicas e arit-

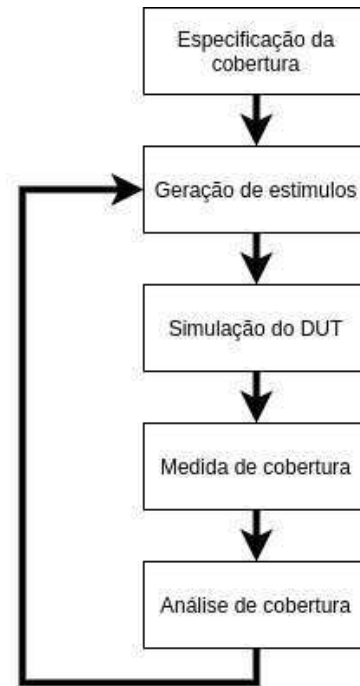


Figura 9 – Fluxo de análise de cobertura.

Fonte – Autoria própria

métricas que podem afetar os bits. Além disso com esse identificador é capaz de indicar possíveis variáveis não utilizadas.

A cobertura de expressões é utilizada para checar se as expressões lógicas foram devidamente exercitadas, ou seja, se as partes constituintes da sentença lógica em algum momento dos testes tiveram valor verdadeiro e falso. Com isso, torna-se possível identificar trechos de código que nunca foram exercitados pois a condição para sua execução nunca foi alcançada.

A cobertura de bloco visa checar se houve algum bloco de código que não foi executado, portanto para completar esse indicador é necessário que todas as linhas do código testado seja executada ao menos uma vez. É comum que falhas nessa métrica estejam associadas a falhas também na cobertura de expressões.

The screenshot shows a software coverage report interface with a tree view on the left and a data table on the right. The tree view includes categories like Overall, Code, Block, Statement, Expression, Toggle, FSM, Functional, Assertion, CoverGroup, and FaultNode. The data table provides metrics for each category, including Overall Average Grade and Overall Covered percentage.

Ex	UNR	Name	Overall Average Grade	Overall Covered
		Overall	99.99%	168810 / 168947 (99.92%)
		Code	99.78%	168324 / 168461 (99.92%)
		Block	100%	16816 / 16816 (100%)
		Statement	100%	15638 / 15638 (100%)
		Expression	98.69%	11784 / 11921 (98.85%)
		Toggle	100%	139724 / 139724 (100%)
		FSM	100%	11 / 11 (100%)
		Functional	100%	475 / 475 (100%)
		Assertion	100%	15 / 15 (100%)
		CoverGroup	100%	460 / 460 (100%)
		FaultNode	n/a	0 / 0 (n/a)

Figura 10 – Exemplo de relatório de cobertura.

Fonte – Autoria própria

4 CONCLUSÕES

No relatório foram apresentadas as atividades desenvolvidas pelo estagiário no Laboratório de Excelência em Microeletrônica do Nordeste, proporcionando um desenvolvimento na área de microeletrônica, processamento de sinais e verificação de IPs.

Durante a vigência do estágio, foram realizadas atividades relacionadas a verificação funcional a nível de blocos digitais. Essas atividades incluíram a elaboração do plano de verificação, estruturação do *testbench*, execução dos testes e análise de cobertura. Além disso, o estagiário trabalhou em conjunto com uma equipe de engenheiros, participando de reuniões, escrevendo relatórios e reportando *bugs*.

Dentro do ambiente de trabalho ficou evidente a importância de assuntos apresentados nas disciplinas Técnicas de Programação, Circuitos Lógicos e Arquitetura de Sistemas Digitais relacionados a área de microeletrônica e as disciplinas Princípios de Comunicações, Comunicações Digitais e Transmissão Digital da Informação relacionados a área de comunicações. Entretanto para a execução das atividades propostas se fez necessário, ao estagiário, conhecimentos em verificação digital e tal conhecimento não foi conseguido através de disciplinas do curso de engenharia elétrica, esse deficit foi sanado com a participação no projeto de Pesquisa e Desenvolvimento dentro do laboratório durante quatro anos que proporcionou ao estagiário um conhecimento prévio bem próximo ao de situações comumente encontradas na indústria da microeletrônica.

REFERÊNCIAS

- CARVALHO, H. de L. *Ambiente de verificação funcional de um IP-core em UVM*. [S.l.], 2018. 11
- FOSTER, H. *2020 Wilson Research Group functional verification study*. [S.l.], 2020. 12, 13
- JUNIOR, J. A. B. *Especificação estatística da dispersão dos modos de polarização em enlaces ópticos*. [S.l.], 2003. 16, 17
- SANTOS, D. M. S. d. Seleção de símbolos piloto em sistemas de comunicação sem fio. 2016. 18
- YUN, Y.-N. et al. Beyond uvm for practical soc verification. In: IEEE. *2011 International SoC Design Conference*. [S.l.], 2011. p. 158–162. 20