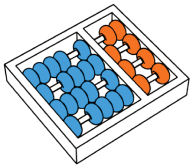
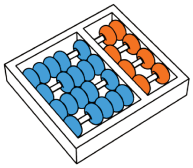


Marcelo Fontes Santana

"Autenticação de Circuitos Integrados Usando Physical Unclonable Functions"

CAMPINAS
2012



Universidade Estadual de Campinas
Instituto de Computação

Marcelo Fontes Santana

"Autenticação de Circuitos Integrados Usando Physical Unclonable Functions"

Orientador(a): **Guido Costa Souza de Araújo**

Co-Orientador(a): **Mario Lúcio Côrtes**

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Computação da Universidade Estadual de Campinas para obtenção do título de Mestre em Ciência da Computação.

Este exemplar corresponde à versão final da Dissertação defendida por Marcelo Fontes Santana, sob orientação de Guido Costa Souza de Araújo.

Assinatura do Orientador(a)

CAMPINAS
2012

FICHA CATALOGRÁFICA ELABORADA POR
MARIA FABIANA BEZERRA MULLER - CRB8/6162
BIBLIOTECA DO INSTITUTO DE MATEMÁTICA, ESTATÍSTICA E
COMPUTAÇÃO CIENTÍFICA - UNICAMP

Santana, Marcelo Fontes, 1983-
Sa59a Autenticação de circuitos integrados usando physical unclonable functions / Marcelo Fontes Santana. – Campinas, SP : [s.n.], 2012.

Orientador: Guido Costa Souza de Araújo.
Coorientador: Mario Lúcio Côrtes.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Função física inclonável. 2. Autenticação (Circuitos integrados). 3. Variação do processo de fabricação. 4. Hardware (Medidas de segurança). 5. Pares de desafio-resposta. I. Araújo, Guido Costa Souza de,1962-. II. Côrtes, Mario Lúcio,1950-. III. Universidade Estadual de Campinas. Instituto de Computação. IV. Título.

Informações para Biblioteca Digital

Título em inglês: Authentication of integrated circuits using physical unclonable functions
Palavras-chave em inglês:
Physical unclonable function
Authentication (Integrated circuits)
Manufacturing process variations
Hardware (Security measures)
Challenge-response pairs
Área de concentração: Ciência da Computação
Titulação: Mestre em Ciência da Computação
Banca examinadora:
Guido Costa Souza de Araújo [Orientador]
Wang Jiang Chau
Paulo César Centoducatte
Data de defesa: 13-11-2012
Programa de Pós-Graduação: Ciência da Computação

TERMO DE APROVAÇÃO

Dissertação Defendida e Aprovada em 13 de Novembro de 2012, pela Banca examinadora composta pelos Professores Doutores:

Prof. Dr. Wang Jiang Chau
POLI / USP

Prof. Dr. Paulo Cesar Centoducatte
IC / UNICAMP

Prof. Dr. Guido Costa Souza de Araújo
IC / UNICAMP

Instituto de Computação
Universidade Estadual de Campinas

Autenticação de Circuitos Integrados Usando Physical Unclonable Functions

Marcelo Fontes Santana[1]

Novembro de 2012

Banca Examinadora:

- Guido Costa Souza de Araújo (Orientador)
- Wang Jiang Chau
  Departamento de Engenharia de Sistema Eletrônicos/USP
- Paulo Cesar Centoducatte
  Institudo de Computação/UNICAMP
- Sandro Rigo (Suplente)
  Institudo de Computação/UNICAMP
- Alexandro José Baldassin (Suplente)
  Departamento de Estatística, Matemática Aplicada e Computação/UNESP

[1] FAPESP - Fundação de Amparo à Pesquisa do Estado de São Paulo (processo 2010/14492-4) 2011–2012.
CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (processo 01-P-0/965/2010) 2010–2011.

Resumo

Ao longo dos anos os equipamentos eletrônicos, especialmente os móveis, vêm se tornando o principal meio de comunicação. Muitas empresas e organizações disponibilizam serviços para que seus clientes acessem remotamente. Entretanto, pessoas não autorizadas podem também acessar indevidamente os serviços de outras através da autenticação com dados do usuário. Para coibir o acesso a serviços por pessoas não autorizadas, surgiu a autenticação do dispositivo eletrônico. Ela funciona como o reconhecimento de uma identidade digital do circuito integrado.

PUF - Physical Unclonable Function - é a técnica de projeto recente usada para identificar circuitos integrados. Usando um PUF é possível identificar qualquer dispositivo eletrônico. Ele permite extrair informações do chip, funcionando como uma impressão digital não clonável.

Um PUF se baseia no fato de que dois dispositivos eletrônicos podem possuir características elétricas distintas. Isso ocorre devido à variação do processo de fabricação dos semicondutores fazendo com que cada dispositivo físico seja diferente. Através desta estrutura é possível extrair informações que podem ser comparadas ao DNA do chip.

Dentro deste contexto, esta dissertação traz um novo modelo de PUF eletrônico digital baseado em atrasos, o CRPUF - Cylindrical Reconvergence PUF. Duas comparações são feitas entre o CRPUF e o APUF - Arbiter PUF - que é um modelo de PUF baseado em atrasos consagrado no mercado. As comparações identificam vantagens do modelo CRPUF em comparação ao APUF.

O CRPUF possui uma maior capacidade de expor as características físicas distintas dos dispositivos físicos usando uma mesma quantidade de hardware. Da mesma forma, é demonstrado que o APUF necessita de um hardware maior para obter a mesma cobertura do CRPUF. Isso torna atraente o uso do CRPUF em arquiteturas de sistemas de identificação que possuam restrição em relação à quantidade de hardware, como, por exemplo, RFID. Resultados dos experimentos demonstram que o CRPUF possui desempenho três vezes melhor na cobertura que o APUF, com a mesma quantidade de hardware.

Abstract

Over the years electronic devices, especially mobile, are becoming the mainly mean of communication. Many companies and organizations provide services for its clients to access remotely. However, unauthorized people may also access their services unduly through the authentication of user data. To restrain the access to unauthorized people, came authentication of electronic device. It serves as the recognition of a digital identity of the integrated circuit.

PUF - Physical Unclonable Function - is a recent proposal to identify integrated circuit. With a PUF it is possible to identify any electronic device. It allows to extract information from the chip to act as a unclonable fingerprint.

A PUF is based on the fact that electronic devices may have diferent electrical characteristics. This occurs due to the variation of the semiconductor manufacturing process making each physical device different. Through this structure is possible to extract information that can be compared to DNA of the chip.

Within this context, this Dissertation presents a new model of delay-based digital electronic PUF, the CRPUF - Cylindrical Reconvergence PUF. Two comparisons are made between the CRPUF and APUF - Arbiter PUF - wich is a delay-based model of PUF well established in the market. The comparisons identify the advantages of the CRPUF model in comparison to the APUF.

The CRPUF has a greater ability to expose the different physical characteristics of the physical devices using the same amount of hardware. Likewise, it is shown that the APUF requires more hardware to obtain the same coverage of the CRPUF. This makes attractive the use of CRPUF on identification systems architectures that have restriction on the amount of hardware, such as RFID. Results of the experiments demonstrate that CRPUF has three times better performance on the coverage than the APUF, using the same amount of hardware.

Agradecimentos

Eu agradeço primeiramente a Deus por sempre estar ao meu lado. Somente ele teve condições de me entender, de me dar forças para que superasse as grandes dificuldades e sempre acreditasse em mim.

Agradeço também à minha mãe, Iracy Fontes Santana (in memorian), que durante a infância insistentemente me obrigava ir para escola. Me incentivava ainda que eu estivesse trabalhando como ajudante de obras ou como vendedor na feira livre durante o ensino médio, ainda assim, sempre acreditou em mim. Dedico este trabalho à Senhora. Sempre será a Pessoa mais importante na minha vida, Amor inigualável. Minha eterna gratidão!

À Família, em especial ao meu pai Raildo Silva Santana e às minhas três irmãs Rosângela Fontes Santana Moreira, Ione Fontes Santana Sena e Tatiane Fontes Santana Cunha por sempre estarmos juntos nos momentos de alegrias e dificuldades de cada um. Saibam que são a minha maior fortaleza! Não posso esquecer dos meus sobrinhos Thais, Amanda, Valentin e Alice!

Agradeço ao meu orientador, Professor Guido Costa Souza de Araújo, e ao meu coorientador, Mario Lúcio Côrtes, que me orientaram com muita responsabilidade no desenvolvimento deste trabalho. Um enorme aprendizado com pessoas tão experientes, excelentes professores na sala de aula e apaixonadas pela pesquisa.

Além disso, também agradeço ao Professor Wagner Luiz Alves de Oliveira que me incentivou durante o período da graduação a seguir este caminho. Sua ajuda também foi fundamental para a início deste trabalho.

Aos amigos e colegas, pessoas que tiveram participação direta e ainda indireta no desenvolvimento deste trabalho. Não posso listar o nome dessas pessoas, porque seria injusto ao esquecer o nome de alguém. Meu muito obrigado a vocês!

E por fim, agradeço à FAPESP e à CAPES, pelo apoio financeiro e institucional.

Sumário

Resumo **ix**

Abstract **xi**

Agradecimentos **xiii**

1 Introdução **1**

2 Conceitos e Trabalhos Relacionados **5**

2.1 Conceitos Básicos 5

2.1.1 Noções Conceituais sobre PUF 5

2.1.2 Desafios e Respostas 7

2.1.3 Propriedades de um PUF 8

2.1.4 Parâmetros de um PUF 10

2.1.5 Classificação dos PUFs 11

2.2 Trabalhos Relacionados 12

2.2.1 PUFs Não-Elétricos 13

2.2.2 PUF Analógicos 15

2.2.3 PUFs Baseados em Atraso 17

2.2.3.1 PUF usando Árbitro - APUF 17

2.2.3.2 PUF usando Tri-state 21

2.2.3.3 PUF Usando Anéis Oscilatórios - ROPUF 22

2.2.3.4 Glitch PUF 24

2.2.4 PUF Baseados em Memória 24

2.2.4.1 SRAM PUF 24

2.2.4.2 Butterfly PUF 25

2.2.4.3 *Latch* PUF 26

2.2.4.4 MECCA PUF 26

2.2.5 Técnicas Utilizadas com PUFs 26

2.2.5.1 PUF Controlado 27

2.2.6 Comparação entre PUFs 28
2.2.7 Considerações Finais 29

3 CRPUF - Cylindrical Reconvergence PUF **31**
3.1 Concepção da Arquitetura 31
3.2 Arquitetura do CRPUF 33
3.2.1 Organização do CRPUF 35
3.2.2 Modelo Geral do CRPUF 37
3.3 Exemplo: CRPUF(3x3) 38
3.4 Desafio e Resposta 39
3.5 Captura da Saída do CRPUF 41
3.5.1 $CRPUF_1$ 42
3.5.2 $CRPUF_2$ 43
3.6 Considerações Finais 44

4 Resultados Experimentais **45**
4.1 Plataforma Experimental 45
4.1.1 Geração dos Atrasos 46
4.1.2 Geração dos Desafios 49
4.1.3 Cálculo da Cobertura 49
4.1.4 Ferramentas 50
4.1.5 Infraestrutura Utilizada 51
4.1.6 Descrição dos Arquivos Utilizados 51
4.2 Experimentos 51
4.2.1 Descrição 52
4.2.2 Caracterização do APUF 54
4.2.3 Caracterização do CRPUF 57
4.2.4 Comparações APUF x CRPUF 63
4.2.4.1 Experimento com APUF como Referencial 63
4.2.4.1.1 Medida C**x**L para **Hardware Igual** 63
4.2.4.1.2 Medida C**x**L para **Cobertura Igual** 65
4.2.4.2 Experimento Com CRPUF Como Referencial 68
4.2.4.2.1 Medida C**x**L para **Hardware Igual** 68
4.2.4.2.2 Medida C**x**L para **Cobertura Igual** 69
4.3 Análise Quantitativa Teórica 73
4.3.1 Cálculo da Quantidade de Hardware em Transistores 73
4.3.1.1 CRPUF 74
4.3.1.2 APUF 76
4.3.2 Análise de Limite Superior de Respostas 77

4.3.2.1 CRPUF 78
4.3.2.2 APUF 79
4.3.2.3 CRPUF x APUF 80
4.4 Considerações Finais 81

5 Conclusões **83**
5.1 Contribuições da Dissertação 84
5.2 Propostas de Trabalhos Futuros 85

Bibliografia **86**

A Avaliação do APUF **91**

B Avaliação do CRPUF **99**

Lista de Tabelas

2.1 Diferentes Tipos de PUFs [27] 12
2.2 Imprevisibilidade X Tamanho da Implementação em Relação ao Tamanho do Desafio. 28

3.1 Desafios e Saídas do CRPUF Para a Forma de Onda da Figura 3.11. 41

4.1 Número de Desafios para o APUF Alcançar a Cobertura de 95%. 57
4.2 Número de Desafios para o CRPUF Alcançar a Cobertura de 95%. 62
4.3 Dimensões do CRPUF para Hardware do APUF(64x1). 63
4.4 Conjunto de APUFs(MxN) com Hardware Total Igual a 1418 Transistores. 65
4.5 Conjunto de APUFs(MxN) com Hardware Total Próximo a 362 Transistores. 71
4.6 Quantidade de Transistores CMOS por Bloco/Porta Lógica. 73
4.7 Dimensão e Número de Transistores do $CRPUF$. 76
4.8 Dimensão e Quantidade de Transistores do APUF. 77
4.9 Número Total de Comportamentos Possíveis para o APUF(2x1). 80
4.10 Resumo da Limite Superior de Respostas do $CRPUF$ e do $APUF$. 80
4.11 Imprevisibilidade X Tamanho da Implementação em Relação ao Tamanho do Desafio com CRPUF. 82

Lista de Figuras

2.1 Modelo Geral do Funcionamento de um PUF. 6
2.2 Identificação de um *chip*. 6
2.3 Funcionamento do PUF Óptico [26]. 14
2.4 Formação da Camada e Funcionamento do Coating PUF. 16
2.5 Estrutura dos MUX utilizado no PUF. 18
2.6 PUF usando Árbitro [33]. 19
2.7 Organização do APUF MxN. 20
2.8 Modelo do *Buffer Tristate*. 21
2.9 Modelo do Unidade de Atraso Usando *Buffer Tristate*. 21
2.10 PUF usando tristate [28]. 22
2.11 Anel oscilador utilizado pelo ROPUF. 22
2.12 PUF usando anéis osciladores [33]. 23
2.13 Funcionamento do Glitch PUF. 24
2.14 Célula SRAM [adaptada de [38]] 25
2.15 *Butterfly PUF*. 25
2.16 Circuito Lógico do *Latch* PUF. 26
2.17 Comparação entre PUF Não Controlado e o PUF controlado. 27
2.18 PUF Controlado Proposto por [6]. 27

3.1 Dependência da Saída. 32
3.2 Malha Reticulada de XORs 33
3.3 Malha Interconectada em Forma de Cilindro. 34
3.4 Malha de Organização e Propagação dos Sinais no CRPUF. 34
3.5 Estágio do CRPUF. 36
3.6 Organização dos Estágios do CRPUF. 36
3.7 CRPUF(MxN). 37
3.8 $CRPUF_1 3x3$ 38
3.9 Entradas e Saídas do CRPUF. 39
3.10 Entradas, Saídas, Sistema de Captura e Respostas do CRPUF. 40
3.11 Representação do Funcionamento do CRPUF. 40

3.12 $CRPUF_1(3x3)$. 43
3.13 $CRPUF_2(3x3)$. 44

4.1 Distribuição Gaussiana dos Atrasos. 46
4.2 Destaque do Atraso dos Trechos Menores de um Estágio do APUF. 47
4.3 Destaque do Atraso dos Trechos Maiores de um Estágio do APUF. 48
4.4 Destaque do Atraso do CRPUF 48
4.5 Como Determinar a Cobertura para 6 PUFs com 3 bits de resposta. 50
4.6 Organização do APUF(MxN). 55
4.7 Cobertura para o APUF(Mx1). 56
4.8 Cobertura para o APUF(Mx8). 56
4.9 Cobertura para o $CRPUF_1$ com 3 linhas. 58
4.10 Cobertura para o $CRPUF_2$ com 3 linhas. 59
4.11 Cobertura para o $CRPUF_1$ com 5 linhas. 59
4.12 Cobertura para o $CRPUF_2$ com 5 linhas. 60
4.13 Cobertura Para o $CRPUF_1$ com 2 Estágios. 61
4.14 Cobertura Para o $CRPUF_2$ com 2 Estágios. 61
4.15 APUF(64x1) x $CRPUF_1$ com Hardware Semelhante. 64
4.16 APUF(64x1) x $CRPUF_2$ com Hardware Semelhante. 64
4.17 APUFs(Mxn) x com Hardware Total Próximo a 1418 Transistores. 66
4.18 APUF(MxN) x $CRPUF_1(MxN)$ com mesmo Hardware do APUF(64x1). . 67
4.19 APUF(MxN) x $CRPUF_2(MxN)$ com mesmo Hardware do APUF(64x1). . 67
4.20 Comparação APUF(16x1) e $CRPUF(5x3)$. 69
4.21 Cobertura do APUF para Vários Tamanhos. 70
4.22 APUFs(MxN) para mesmo Hardware do $CRPUF_1(5x3)$. 71
4.23 APUFs(MxN) para mesmo Hardware do $CRPUF_2(5x3)$. 72
4.24 APUF 16 bits x $CRPUFs(5x3)$. 73

A.1 Cobertura para o APUF(12xN). 92
A.2 Cobertura para o APUF(16xN). 92
A.3 Cobertura para o APUF(32xN). 93
A.4 Cobertura para o APUF(64xN). 93
A.5 Cobertura para o APUF(Mx1). 94
A.6 Cobertura para o APUF(Mx2). 95
A.7 Cobertura para o APUF(Mx3). 95
A.8 Cobertura para o APUF(Mx4). 96
A.9 Cobertura para o APUF(Mx6). 96
A.10 Cobertura para o APUF(Mx8). 97

B.1 Cobertura para o $CRPUF_1$ com 3 linhas. 99
B.2 Cobertura para o $CRPUF_2$ com 3 linhas. 100
B.3 Cobertura para o $CRPUF_1$ com 5 linhas. 100
B.4 Cobertura para o $CRPUF_2$ com 5 linhas. 101
B.5 Cobertura para o $CRPUF_1$ com 7 linhas. 102
B.6 Cobertura para o $CRPUF_2$ com 7 linhas. 102
B.7 Cobertura para o $CRPUF_1$ com 9 linhas. 103
B.8 Cobertura para o $CRPUF_2$ com 9 linhas. 103
B.9 Cobertura para o $CRPUF_1$ com 2 Estágios. 104
B.10 Cobertura para o $CRPUF_2$ com 2 Estágios. 104
B.11 Cobertura para o $CRPUF_1$ com 3 Estágios. 105
B.12 Cobertura para o $CRPUF_2$ com 3 Estágios. 105
B.13 Cobertura para o $CRPUF_1$ com 4 Estágios. 106
B.14 Cobertura para o $CRPUF_2$ com 4 Estágios. 106
B.15 Cobertura para o $CRPUF_1$ com 5 Estágios. 107
B.16 Cobertura para o $CRPUF_2$ com 5 Estágios. 107
B.17 Cobertura para o $CRPUF_1$ com 6 Estágios. 108
B.18 Cobertura para o $CRPUF_2$ com 6 Estágios. 108
B.19 Cobertura para o $CRPUF_1$ com 7 Estágios. 109
B.20 Cobertura para o $CRPUF_2$ com 7 Estágios. 109
B.21 Cobertura para o $CRPUF_1$ com 8 Estágios. 110
B.22 Cobertura para o $CRPUF_2$ com 8 Estágios. 110
B.23 Cobertura para o $CRPUF_1$ com 9 Estágios. 111
B.24 Cobertura para o $CRPUF_2$ com 9 Estágios. 111

Capítulo 1

Introdução

Dispositivos eletrônicos estão cada vez mais onipresentes e interconectados. O funcionamento seguro dos dispositivos eletrônicos é uma característica muito exigida em diversas áreas, pois os dados que manipulam precisam ser executados e transmitidos de forma privada. Equipamentos eletrônicos fazem parte do cotidiano das pessoas nas mais diversas atividades que vão desde atividades simples às que exigem um alto grau de sigilo, como por exemplo, atividades de segurança nacional.

Atividades, tais como, acesso ao e-mail, operações bancárias com equipamentos móveis e transmissão de dados para outras bases, precisam muitas vezes que, além do usuário, o dispositivo físico também seja reconhecido. Desta forma, a identificação do dispositivo eletrônico é utilizada para evitar que outras pessoas se comuniquem ou transmitam dados no lugar da pessoa autorizada, que é a dona do equipamento. Como exemplo, alguns bancos e provedores de e-mail tem utilizado a liberação do dispositivo para que seus usuários acessem seus serviços. Essa alternativa vem sendo utilizada porque a segurança da autenticação através da identificação do usuário e a sua senha correspondente não se mostram tão seguras. Assim, mesmo que outra pessoa não autorizada possua o par usuário e senha, ela não seria capaz de acessar os serviços, pois o dispositivo eletrônico dessa pessoa não estaria habilitado.

Durante anos a indústria incluiu a execução de algoritmos criptográficos dentro dos circuitos integrados e a utilização de chaves pré-definidas já durante o projeto do circuito integrado. Recentemente, este modelo esbarrou em limitações físicas que dificultam a sua continuidade para dispositivos que exigem baixo consumo de energia e restrições em relação à área utilizada. O aumento da complexidade dos algoritmos criptográficos e do tamanho das chaves para estes algoritmos necessárias à uma melhor segurança, fez com que a academia e indústria buscassem novas alternativas para o projeto de sistemas criptográfico que atendessem estas restrições. Por outro lado, a inclusão de identificadores pré-definidos pode ser facilmente realizada e em seguida emulada por outros dispositivos.

Além disso, a identificação de circuitos integrados é uma necessidade que vêm sendo requisitada por muitas aplicações de segurança. A garantia que determinada informação seja recebida por apenas um dispositivo estabelecido e que outros não tenham acesso, é fundamental para o funcionamento privado de muitas aplicações.

Uma das soluções encontradas que atende os requisitos de baixo consumo de energia e restrições de área consiste em obter a identificação através de variações que ocorrem durante o processo de fabricação de cada circuito integrado. Esta variação faz com que a disposição do silício e dos materiais depositados nele seja arranjada de forma única em cada circuito integrado produzido. Tais características podem ser comparadas ao DNA de cada circuito integrado.

Assim, várias técnicas que exploram estas características da disposição do silício foram propostas para identificar circuitos integrados. Entre estes modelos estão os PUFs (*Physycal Unclonable Functions*) que é uma função pela qual se envia uma determinada pergunta (desafio) e se obtém uma resposta para aquela pergunta. Desta forma, por meio das respostas apresentadas por cada PUF é possível identificar qual a identidade do circuito integrado.

PUF é um sistema físico complexo que tem por objetivo definir uma função de sentido único, ou seja, dado um resultado, não é possível descobrir qual o desafio aplicado. Também não é possível prever qual a resposta para um determinado desafio. A replicação do PUF também é muito difícil (praticamente impossível) mesmo para quem produziu.

Desta forma, os PUFs têm o propósito de capturar e expor as variáveis incontroláveis e aleatórias resultantes do processo de fabricação, tais como a densidade do metal e a profundidade da difusão, e representá-las em resultados mensuráveis. Os PUFs são funções embutidas no hardware que ocupam pouca área e normalmente consomem pouca energia. Entre os seus princípios estão o de que os resultados devem ser reproduzidos pelo mesmo dispositivo.

Sendo assim, considera-se um PUF como um recurso constituído com características aleatórias para geração de dados confiáveis visando a identificação do dispositivo físico e geração de chaves para sistemas criptográficos.

Ao longo dos trabalhos relacionados são apresentados vários modelos de PUFs que utilizam diferentes características de aleatoriedade ocorridas durante o processo de fabricação. Muitos desses PUFs já estão consagrados no mercado, como por exemplo o *Arbiter PUF* - APUF - que é baseado em atrasos.

Diante desse contexto, essa Dissertação possui o objetivo de propor um novo modelo de PUF baseado em atraso que tem uma estrutura visando melhorar o desempenho custo/benefício com relação ao APUF.

Os resultados esperados para o CRPUF (Cylindrical Reconvergence PUF) são:

1. demonstrar que o novo modelo de PUF apresenta melhor desempenho que o APUF

quando ambos possuem a mesma quantidade de hardware;

2. identificar que o novo modelo de PUF possui, com relação ao aumento do seu tamanho, aumento exponencial da imprevisibilidade das respostas com aumento linear de hardware.

Os dois modelos de PUFs (CRPUF e APUF) são colocados em condições iguais de hardware e a partir daí é identificado o desempenho de cada um deles utilizando a simulação de Monte Carlo. Além disso, a partir do desempenho de um deles, também é encontrado o hardware do outro para que este tenha o mesmo desempenho.

Também é apresentada uma análise do CRPUF que mostra que o tamanho do hardware é determinado por uma equação linear e que a partir do seu aumento é possível melhorar exponencialmente a imprevisibilidade das respostas.

Os objetivos, conceitos e os resultados deste trabalho são apresentados a partir da seguinte organização: no Capítulo 2 são apresentados os conceitos básicos e os principais trabalhos relacionados de vários modelos de PUFs, dando destaque especial aos modelos que possuem maior semelhança com o PUF baseado em atraso desenvolvido neste trabalho; no Capítulo 3 são apresentadas as motivações e a arquitetura do novo modelo CRPUF (*Cylindrical Reconvergence PUF*) proposto neste trabalho, além de descritas as suas características fundamentais quando comparadas com outro modelo de PUF baseado em atraso; no Capítulo 4 são apresentados os resultados experimentais obtidos na comparação entre o CRPUF e o APUF (*Arbiter PUF*) e apresentada a metodologia experimental; no Capítulo 5 são apresentadas as conclusões;

Capítulo 2

Conceitos e Trabalhos Relacionados

Este Capítulo apresenta os conceitos e as características gerais que são úteis para mensurar qualitativamente os PUFs (seção 2.1) e apresenta também detalhes do funcionamento de vários modelos de PUFs, agrupados de acordo com a classificação que é apresentada (seção 2.2).

2.1 Conceitos Básicos

Esta seção apresenta noções conceituais sobre os PUFs. Além disso, apresenta também as características principais de vários modelos de PUFs.

2.1.1 Noções Conceituais sobre PUF

Conforme apresentado em [6], *Physical Unclonable Functions* - PUF - é uma função que mapeia desafios em respostas explorando características aleatórias adquiridas durante o processo de fabricação [6, 14, 33].

Como apresentado na Figura 2.1, o PUF recebe um estímulo (desafio) e a partir gera uma resposta que é uma assinatura individual das características físicas aleatórias do material constituinte. Trata-se genericamente de uma função que ao ser estimulada por desafios, gera resultados distintos para cada PUF por causa das propriedades aleatórias que identificam cada um deles. Assim, um PUF é identificado pelo conjunto de pares de desafios e respostas.

Figura 2.1: Modelo Geral do Funcionamento de um PUF.

Um PUF de um dispositivo físico qualquer, como apresentado na Figura 2.1, pode incorporar diversos tipos de aleatoriedade durante o processo de fabricação. Essa aleatoriedade, a depender da forma como for medida, pode ser caracterizada através do resultado de várias propriedades, tais como a reflexão da luz, a capacitância, a potência dissipada, a diferença dos atrasos e etc. As características físicas particulares de cada dispositivo diferencia cada indivíduo como se fosse o DNA do dispositivo físico, como apresentado na Figura 2.2.

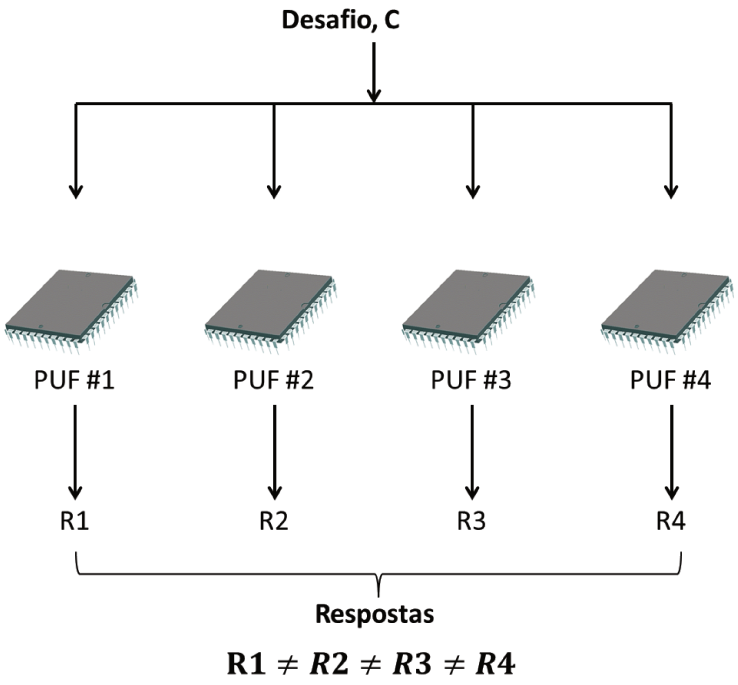


Figura 2.2: Identificação de um *chip*.

De acordo com a representação da Figura 2.2, cada um dos 4 *chips* possui um PUF (PUF #1, PUF #2, PUF #3 e PUF #4) que recebem o mesmo desafio C e cada um deles retorna uma resposta diferente, $R1$, $R2$, $R3$ e $R4$ respectivamente. Assim é possível identificar os 4 dispositivos físicos, pois cada PUF possui características distintas.

A função física inclonável é incorporada ao dispositivo físico e possui resultados diferentes devido à variação durante o processo de fabricação.

Além da característica de inclonável, de acordo com [6, 33], os PUFs seguem as seguintes propriedades:

Fácil de Avaliar: O dispositivo físico que incorpora o PUF é capaz de avaliá-lo em pouco tempo.

Difícil de Caracterizar: a partir de um número limitado de medidas de pares de desafio-resposta (CRP - *Challenge-Response Pairs*) escolhido, um atacante que não tem mais o dispositivo só pode extrair uma quantidade insignificante de informações sobre o desafio e a resposta.

Estas duas propriedades dos PUFs, fácil de avaliar e difícil de caracterizar, são utilizadas pelos principais trabalhos de pesquisas relacionados.

O PUF deve ser facilmente avaliado (estimulado) pelo dispositivo físico que o incorpora. A aplicação do desafio e recebimento da resposta deve ser um processo simples de comunicação.

Difícil de caracterizar representa que ao ser obtido alguns pares de desafios e respostas do PUF, não será possível prever quais são os outros pares de desafios e respostas. Assim, o PUF deve ser matematicamente inclonável, pois o seu funcionamento não pode ser caracterizado por funções matemáticas, ou seja, não é possível prever o resultado do PUF. Além disso, dizer que o PUF é físicamente inclonável também significa que ele não pode ser reproduzido fisicamente nem mesmo pelo pelo próprio fabricante porque a sua individualização ocorre devido à variações aleatórias durante o processo de fabricação.

Na seção 2.1.3 é apresentado um conjunto de propriedades resultantes de um trabalho de pesquisa que visa uma definição formal e conceitual sobre as propriedades gerais dos PUFs. Assim, aqui nesta visão inicial dos PUFs são dadas apenas algumas propriedades mais conceituais sobre os PUFs. Na subseção 2.1.2 seguinte são dados maiores detalhes sobre a importância e os conceitos relacionados ao desafio e à resposta.

2.1.2 Desafios e Respostas

Em [26], o PUF - *Physical Unclonable Function* - é representado por uma função $\Pi(x) = y$, onde o domínio, valor x, representa o desafio utilizado para estimular o PUF. A imagem da função, valor y, é o resultado (resposta) obtido ao aplicar o desafio x à função $\Pi(x) = y$.

O desafio e a resposta dessa operação são conhecidos como par desafio-resposta (*Challenge-Response Pair - CRP*). Os pares desafio-resposta compõem um conjunto que caracteriza o comportamento individual de cada PUF de forma que se possa identificá-lo.

Apesar de representar uma função, o PUF não deve ser considerado genericamente como uma função no conceito matemático em si, pois, em alguns casos, para determinado desafio a que o PUF for submetido, é possível que seja obtida mais que uma saída possível [26]. Estes casos podem ser indesejáveis para PUFs que são utilizados para identificação ou autenticação do dispositivo. No entanto, eles podem ser utilizados para a geração de números aleatórios usando PUFs.

Resumindo, o PUF é uma função física inclonável que quando submetido a um desafio obtêm-se uma resposta, sendo assim cada indivíduo caracterizado pelo conjunto de pares desafio-resposta [4, 6, 9, 14, 28, 33]. Esta função é incorporada ao dispositivo físico e, de acordo com [6, 33], o conjunto dos pares desafio-resposta representa uma identidade do dispositivo que contém o PUF. Sendo assim, neste trabalho, os indivíduos representam dispositivos físicos que podem ser caracterizados através das suas propriedades físicas definidas durante o processo de fabricação.

2.1.3 Propriedades de um PUF

As propriedades representam um conjunto de características que são observadas nos PUFs. Elas são qualidades e atributos dos PUFs. Entretanto, nem todos os modelos de PUFs possuem todas as propriedades.

Em [26] são elencadas sete propriedades dos PUFs visando uma sistematização e consolidação das definições. Esse foi um dos primeiros trabalhos a elencar as principais propriedades.

Antes de apresentar as propriedades, em [26] também considera-se a notação $\Pi : \chi \rightarrow \Upsilon : \Pi(x) = y$ como representação da funcionalidade do desafio-resposta de um determinado PUF Π, onde x representa o desafio e y a resposta. As propriedades são definidas por [26] da seguinte forma:

1. **Avaliável:** dados Π e x, é fácil calcular $y = \Pi(x)$. De um ponto de vista teórico, fácil pode significar tempo ou recurso utilizado, enquanto do ponto de vista prático, pode significar baixo *overhead* de tempo de resposta.

2. **Único:** $\Pi(x)$ contém algumas informações sobre a identidade da entidade física embutida Π. Isso significa que o conjunto de CRPs de um determinado PUF é suficiente para identificá-lo unicamente numa determinada população de PUFs.

3. **Reprodutível:** dado $y = \Pi(x)$, as respostas para várias aplicações de um mesmo desafio x devem ser iguais, sendo aceita também respostas próximas até um "pequeno erro"[1]. A reprodutibilidade é a propriedade que distingue os PUFs dos geradores de

[1] O termo informal "pequeno erro"utilizado em algumas propriedades é considerado como uma de-

números aleatórios verdadeiros (TRNGs), pois estes últimos apresentam respostas distintas quando aplicado um mesmo desafio várias vezes.

4. **Inclonável:** dado o PUF Π, é difícil construir outro PUF $\Gamma \neq \Pi$ de tal modo que $\forall x \in \chi : \Gamma(x) \approx \Pi(x)$ com até um "pequeno erro". Isso significa que a partir de todo o conjunto de CRPs de um PUF, é difícil construir outro PUF do mesmo modelo que tenha CRPs iguais ao primeiro PUF.

5. **Imprevisível:** dada uma parte do conjunto de desafios e respostas $Q = (x_i, y_i = \Pi(x_i))$, com $i = 1...q$, é difícil de prever a resposta $y_c = \Pi(x_c)$ do PUF ao desafio aleatório x_c, sendo que $x_c \notin Q$ e (x_c, y_c) pertence ao conjunto de CRPs do PUF.

6. **Sentido Único:** dado apenas y e Π, é difícil encontrar $x \in \chi$ de tal modo $\Pi(x) = y$.

7. **Inviolável:** alterando a entidade física embutida Π ocorre a transformação $\Pi \rightarrow \Pi'$ de modo que com alta probabilidade $\exists x \in \chi : \Pi(x) \neq \Pi'(x)$, nem mesmo com um pequeno erro. Isto significa que um ataque invasivo no PUF deixa traços permanentes no comportamento dos CRPs.

Assim, [26] classifica inclonabilidade em duas formas: física e matemática. A inclonabilidade física está relacionada com a incapacidade de construir (clonar) fisicamente outro PUF com comportamento igual. Já a inclonabilidade matemática está relacionada com a incapacidade de construir matematicamente uma função que apresente comportamento igual ao PUF. Uma construção pode ser fácil de clonar fisicamente, mas não matematicamente, ou vice-versa. A fim de ser verdadeiramente inclonável, o PUF Π precisa ser fisicamente e matematicamente inclonável. A inclonabilidade física faz com que a dificuldade de produzir um clone físico também seja a mesma para o fabricante do PUF original Π, e é por esta razão que ela também é chamada de resistência (dificuldade) de replicação. Assim, a inclonabilidade física implica em unicidade. Para matematicamente inclonável, dado Π, é difícil construir um procedimento matemático (abstrato) f_r de tal modo que $\forall x \in \chi : f_{r(x)} \approx \Pi_x$ tenha um pequeno erro.

Apresentadas as principais propriedades dos PUFs, a seção seguinte apresenta os parâmetros que são calculados e utilizados para verificar as propriedades dos PUFs através de testes experimentais.

---

finição abstrata, ficando assim, como relata [26], como um trabalho futuro que ainda deverá definir uma representação exata de qual a precisão do erro a ser considerado pequeno. Portanto, não existe ainda nos trabalhos relacionados um valor associado que seja considerado como "pequeno erro"

2.1.4 Parâmetros de um PUF

Em [27], Maiti et al. avaliam a definição de parâmetros obtidos por pesquisadores em suas implementações em vários trabalhos. Os parâmetros de um PUF são medidos para avaliar a qualidade dos modelos de PUFs e desta forma definir algumas de suas propriedades. Os trabalhos que definem as propriedades dos PUFs e os que definem os parâmetros não estão correlacionados, por isso, cada parâmetro determinado por [27] não está diretamente relacionado com uma propriedade do PUF. Assim, esta Dissertação apresenta o conjunto de parâmetros definidos em [27]. Os parâmetros apresentados são:

Uniformidade: Estima o quão uniforme é a proporção de '0's e '1's num bit das respostas de um PUF. A proporção ideal é 50% para '0' e para '1'.

Confiabilidade: Identifica se o PUF reproduz sempre os mesmos bits de resposta quando estimulado por um desafio várias vezes em condições diversas. Este parâmetro é correlacionado com a propriedade *Reprodutível*.

Estabilidade Identifica se há tendência de um bit da resposta para um mesmo valor, ou seja, se um determinado bit da resposta sempre apresenta um mesmo valor. É uma estimativa da variação *intra-chip* em termos das respostas do PUF.

Unicidade Representa a capacidade de um PUF de um chip particular ser identificado univocamente entre um grupo de chips do mesmo tipo. É uma estimativa da variação *inter-chip* em termos das respostas dos PUFs.

Difusabilidade Representa o quão diferentes são as respostas para desafios próximos aplicados ao mesmo PUF. Este parâmetro é correlacionado com a propriedade *Imprevisível*.

Distorção de bit (*bit-aliasing*) Se *bit-aliasing* acontece, diferentes chips podem produzir respostas de PUFs parecidas ou próximas, o que é um efeito indesejado. Este parâmetro é correlacionado com a propriedade *Único*.

Probabilidade de falso-positivo Estima a probabilidade de um chip ser identificado falsamente como outro devido aos bits da resposta. Este parâmetro é correlacionado com a propriedade *Único*.

Estas qualidades dos PUFs são utilizadas para avaliar a implementação e o funcionamento real dos PUFs de acordo com parâmetros mensuráveis. A seção 2.1.5 apresenta a classificação geral dos PUFs.

2.1.5 Classificação dos PUFs

Os diversos modelos de PUFs foram classificados por [26] de acordo com a natureza dos componentes que contribuem para a estrutura aleatória que faz o PUF único. Os PUFs são classificados em:

- **PUFs Não-Elétricos: não possuem** características elétricas no fator que determina a sua aleatoriedade;
- **PUFs Analógicos:** são caracterizados por medições analógicas no fator que determina a sua aleatoriedade;
- **PUFs Baseados em Atrasos Intrínsecos:** caracterizados por medições digitais baseados nos atrasos no fator que determina a sua aleatoriedade;
- **PUFs Baseados em Memória Intrínsecas:** caracterizados por medições digitais baseados na concorrência entre sinais para determinar o valor inicial no *startup* de uma célula de memória no fator que determina a sua aleatoriedade.

Embora os PUF Não-Elétricos em alguns modelos possuam componentes elétricos para armazenar ou até mesmo transmitir as respostas, a sua aleatoriedade não é determinada por estes componentes.

Assim, definidos os parâmetros, as classificações e as propriedades dos PUFs, na seção seguinte 2.2 são apresentados os principais trabalhos relacionados que trazem modelos de PUFs e suas classificações.

2.2 Trabalhos Relacionados

Nesta seção é apresentada uma breve descrição dos modelos de PUF desenvolvidos em vários trabalhos de pesquisas nos últimos anos. Pretende-se aqui evidenciar os vários trabalhos de pesquisas realacionadas a PUFs, comparando-os e analisando suas principais caractérísticas.

Embora existam diferentes modelos de PUFs, nesta Dissertação é detalhado o funcionamento dos PUFs Baseados em Atrasos, pois eles estão diretamente relacionados com o modelo desenvolvido neste. Os demais são apresentados de forma resumida, com indicação de referenciais aos principais. Pretende-se, desta forma, oferecer uma visão geral dos modelos de PUFs, descrevendo principalmente os PUFs que utilizam a variação dos atrasos como elemento de aleatoriedade.

Em [27], Maiti et al. apresentam cronologicamente vários trabalhos de pesquisas que têm desenvolvido novos modelos de PUFs, evidenciando assim a criação de um novo modelo de PUF, quase a cada ano. Conforme informam os autores, estes modelos de PUFs apresentados na Tabela 2.1 não representam uma lista exaustiva dos modelos de PUFs existentes, já que nos últimos anos novos modelos de PUFs vêm sendo propostos. Uma discussão mais aprofundada sobre vários modelos de PUFs pode ser obtida em [26]. A Tabela 2.1 relaciona o surgimento de alguns modelos de PUFs ao longo dos últimos anos. Nessa Dissertação são apresentados também, além destes PUFs da Tabela 2.1, outros modelos de PUFs.

Tabela 2.1: Diferentes Tipos de PUFs [27]

Ano	***Título do trabalho sobre o PUF***
2000	IC Identification Using Device Mismatch [24]
2001	Physical One-Way Function [29, 30]
2002	Physical Random Function [10]
2004	Arbiter PUF [21]
2006	Coating PUF [36]
2007	Ring Oscillator PUF [33], SRAM PUF [17]
2008	Butterfly PUF [20]
2009	PUF Using Power Distribution System of an IC [16]
2010	Glitch PUF [34]
2011	Mecca PUF [19]

Como apresenta a Tabela 2.1, novos modelos de PUFs têm sido propostos a cada ano. Existem ainda trabalhos que dão outras contribuições aos diferentes modelos de PUFs, como por exemplo, a utilização de técnicas mais aperfeiçoadas para melhorar o desempenho do PUF.

2.2.1 PUFs Não-Elétricos

Nesta seção são apresentados os principais modelos de PUFs Não-Elétricos. Estes modelos não possuem características elétricas no parâmetro que determina a sua aleatoriedade. Embora alguns deles possuam elementos elétrico-eletrônicos para processar ou armazenar as respostas aos desafios, sua aleatoriedade advém apenas de elementos não elétrico-eletrônicos. Os modelos de PUFs Não-Elétricos aprensentados neste trabalho são:

1. **PUF Óptico**
2. **Paper PUFs**
3. **CD PUFs**
4. **RF-DNA**
5. **PUF Magnético**
6. **PUF Acústico**

O funcionamento de cada um desses modelos de PUFs Não-Elétricos são descritos a seguir.

1. O **PUF Óptico** foi o modelo de PUF pioneiro que desencadeou os mais diversos modelos de PUFs utilizando outras caracterísicas. De acordo com [26], [35] foi o primeiro trabalho a dar direção ao desenvolvimento dos PUFs. Muitos autores caracterizam tal feito ao trabalho desenvolvido por [29,30]. O trabalho [35] foi desenvolvido ainda no início dos anos 90 e apresenta historicamente uma primeira ideia de PUF. Quando o PUF em [35] foi proposto ainda não existia a definição conceitual de PUF. Este conceito começou a ser definido a partir dos trabalhos [6–11]. O PUF em [35] é baseado no padrão de reflexão óptica aleatória, que foi chamado de Tag de Partícula Reflexiva (*Reflective Particle Tagging - RPT*). O resultado deste trabalho foi inicialmente utilizado para identificação de armas estratégicas em tratados de controle de armas [26].

Um modelo de PUF baseado em meio transparente foi proposto em [29,30]. Chamado de *Physical One-Way Functions* (POWF), este trabalho despertou o interesse dos pesquisadores no desenvolvimento de PUFs utilizando outros recursos. Como apresentado na Figura 2.3, ele utiliza uma microestrutura ótica transparente em epóxi construída

pela mistura com esferas de vidro refrativas. Esta estrutura é atravessada por um laser de hélio-neônio. A forma de onda que atravessa o vidro se torna irregular devido aos múltiplos espalhamentos refrativos do feixe. O padrão da luz que chega é capturado por uma câmera CCD para o processamento digital. Posteriormente, é aplicado ao padrão de luz capturado o filtro Gabor para obtenção de um *hash*. Este filtro é utilizado na computação forense para autenticação de imagens e utilizado como um procedimento de extração de características de imagens [37].

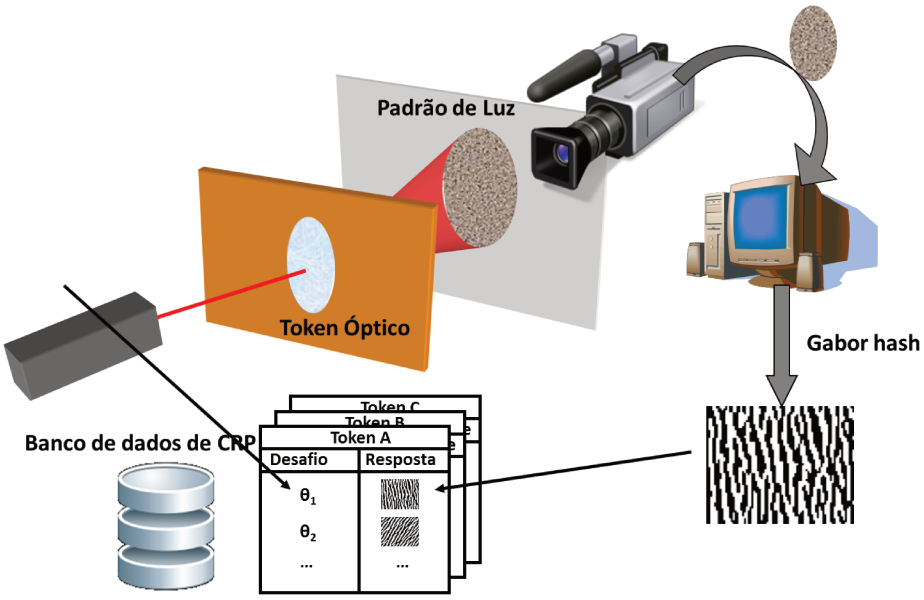


Figura 2.3: Funcionamento do PUF Óptico [26].

O resultado é uma string de bits representando um *hash* para o padrão da luz capturada. Cada estrutura construída é chamada de *token* óptico e considerado um PUF, como apresenta a Figura 2.3. Os desafios estão relacionados com a forma de incidência e orientação do laser [26,28].

2. Assim como no Optical PUF, o **Paper PUF** também foi proposto antes da definição do conceito de PUF. Ele é um modelo proposto na literatura que consiste em *scanear* a estrutura única da fibra aleatória de um papel normal ou modificado. Alguns projetos utilizam as estruturas fixas do papel, e outros inserem explicitamente fibras ultravioletas no papel durante o processo de fabricação. Ambos podem ser medidos por um *scanner* convencional. Alguns deles também introduzem um método para correlacionar fortemente o dado no documento com o papel, usando uma assinatura digital de dados e impressão digital do papel que ao final é impressa no documento [1–3, 26].

3. O **CD PUF** é um modelo de PUF que observa a medida do tamanho de *lands* e *pits* sobre um CD (*Compact Disk*). *Lands* e *pits* são estrutura físicas microscópicas que

representam a informação binária do dado armazenado no disco. *Lands* refletem o laser em um sensor (valor 1), já *pits* não refletem o laser (valor 0). Assim, observou-se que o CD possui desvios nos tamanhos das *lands* e *pits* desejados devido a uma variação aleatória durante o processo de fabricação. Este desvio é grande o suficiente para que seja observado monitorando o sinal elétrico do foto sensor de um tocado de CD regular [26]. Este método foi testado para um grande número de CDs e locais em cada CD. Os resultados deste trabalho são apresentados em [15].

4. O **RF-DNA**, ou simplesmente PUF baseado em radio-frequência, foi proposto em [5]. Este modelo de PUF foi construído de forma semelhante ao PUF Óptico, no entanto foi utilizado um *token* menor e mais barato quando comparado com o usado no PUF Óptico. Fios finos de cobre são colocados de forma aleatória em uma estrutura selante de silício. Ao invés de observar o desvio da luz como no PUF Óptico, é observada a dispersão do campo de ondas eletromagnéticas pelos fios de cobre em outros comprimentos de ondas, entre 5 a 6 Ghz.

5. O **PUF magnético** utiliza a unicidade inerente do padrão de particulas em meio magnético, por exemplo em cartões magnéticos [18, 26].

6. Os **PUFs Acústicos** são construídos observando o espectro de frequência característico de uma linha de atraso acústica. Linhas de atrasos acústicas são componentes utilizados para atrasar sinais elétricos. Eles convertem sinal elétrico alternado em vibração mecânica. Uma string de bits é obtida através da análise da componente principal e é estimado que ao menos 160 bits de entropia podem ser extraídos [26].

2.2.2 PUF Analógicos

Nesta seção são discutidos os principais modelos de PUF que possuem como elemento de aleatoriedade a medição analógica de uma característica elétrica ou eletrônica. Assim, podem ser considerados nestes modelos de PUFs: tensão, corrente, capacitância, potência etc. Os PUF elétricos com características analógicas apresentados neste trabalho são:

1. V_T **PUFs**
2. ***Coating PUF***
3. ***Power Distribution PUFs***

No decorrer da seção é descrito o funcionamento de cada um desses modelos de PUFs Analógicos.

1. O $V_T PUF$ utiliza a variação da tensão de *treshold* (V_{TH}) para identificar cada circuito [24]. No V_T **PUF** um conjunto de transistores igualmente projetados são dispostos em um array endereçável. O *driver* do transistor aciona uma carga resistiva e por causa do

efeito da variação do processo de fabricação nas tensões de limiar (V_T) desses transistores, a corrente durante esta carga será aleatória. A tensão sobre a carga é medida e convertida para uma sequência de bits usando comparador [22, 23, 26].

2. O ***Power Distribution PUF***, proposto em [16], aproveita as variações de resistência inerentes nas camadas de metal que definem uma rede de distribuição de potência disposta no *chip*. As quedas de tensão e a resistência equivalente no sistema de distribuição de potência são medidas através de instrumentos externos e assim é observado que estes parâmetros elétricos são afetados pela variação do processo de fabricação.

3. O ***Coating PUF***, ou simplesmente PUF em camada, utiliza uma camada adicional de metal sobre o circuito em formato de pente. Esta camada de metal é utilizada para que sejam calculadas capacitâncias distribuídas pelo circuito através de sensores [26]. Ele foi proposto em [36] e tem aleatoriedade decorrente não apenas da variação no processo de fabricação que é comum em diversos PUFs. Ele também insere explicitamente elementos aleatórios por meio de uma camada dielétrica distribuída diretamente sobre os sensores [26]. Considerando que a camada física inserida é opaca e inerte quimicamente, ele também oferece uma forte proteção contra ataques físicos ao PUF [26, 36].

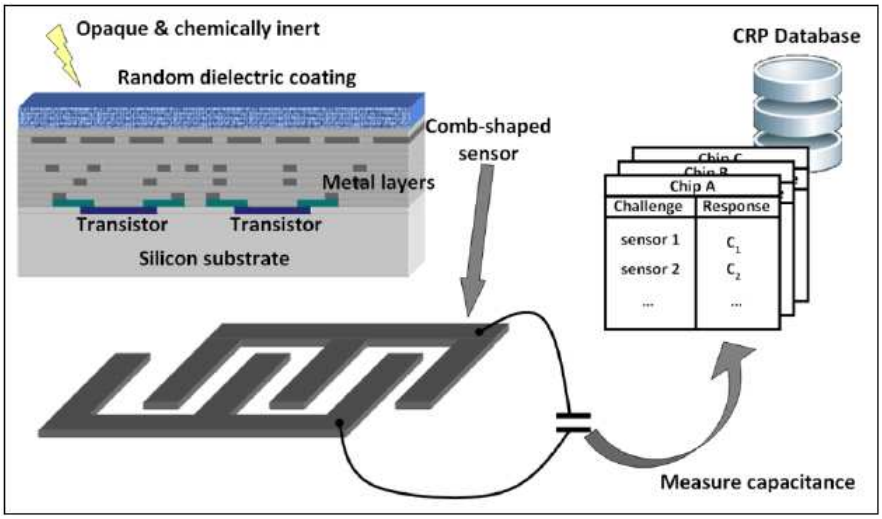


Figura 2.4: Formação da Camada e Funcionamento do Coating PUF.

Como se pode observar na Figura 2.4, a camada de metal inserida no circuito em formato de pente possui sensores que auxiliam na medição da capacitância em locais específicos do circuito. No lado direito da Figura 2.4 é possível identificar que a partir de um conjunto de desafios é possível capturar diversas respostas para cada capacitor utilizado no circuito. Maiores detalhes sobre este modelo de PUF podem ser encontrados em [36].

Na subseção seguinte são descritos os modelos de PUFs baseados em atrasos.

2.2.3 PUFs Baseados em Atraso

Nesta seção são discutidos os principais modelos de PUF que possuem como elemento de aleatoriedade a medição dos atrasos em um determinado caminho do circuito. Genericamente, trata-se de circuitos que possuem corridas de sinais em caminhos de comprimento nominal idêntico, onde ao final é avaliado o resultado da corrida. Os PUFs elétricos baseados em atrasos discutidos neste trabalho são:

- **APUF** - PUF usando multiplexadores que utiliza um árbitro (*flipflop*) nas duas saídas para determinar qual o caminho pelo qual um mesmo sinal disparado chegou mais rápido;
- **PUF Usando *TriState*** - Funcionamento semelhante ao PUF usando árbitro, só que usa *tristate* para determinar o caminho;
- **RO PUF** - PUF Usando Oscilador em Anel que, dentre um conjunto de osciladores em anel, seleciona um par de osciladores em anel e compara qual o mais rápido;
- **Glitch PUF** - Este PUF recebe uma forma de onda composta de *glitches* e converte em um conjunto de bits de resposta.

Os PUFs baseados em atrasos que utilizam corridas entre sinais (como o APUF, PUF usando tristate e RO PUF) necessitam que os caminhos possuam tamanhos idênticos. A diferença de atrasos ocorre apenas devido à variação no processo de fabricação.

Nas subseções seguintes são descritos e discutidos detalhadamente cada um destes modelos de PUFs.

2.2.3.1 PUF usando Árbitro - APUF

O APUF é um PUF baseado em atrasos. Desenvolvido ao longo dos trabalhos [6,10,13,33], o APUF utiliza multiplexadores para selecionar os caminhos percorridos por dois sinais disparados simultaneamente após um desafio. O APUF recebe um desafio X de N bits de entrada e retorna *1* bit de resposta de saída *Resposta* (Figura 2.6). Cada bit de desafio é responsável por selecionar o caminho para um estágio, funcionando como um *crossbar*. Cada MUX recebe dois sinais de entrada e através do bit de desafio (*select*) permite a passagem de apenas um para a saída, como apresentado na Figura 2.5.

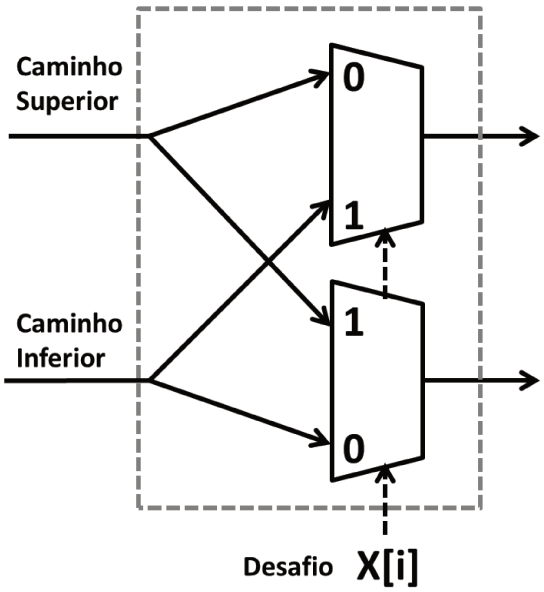


Figura 2.5: Estrutura dos MUX utilizado no PUF.

Cada estrutura da Figura 2.5 representa um estágio do APUF de N estágios (desafios) [6,10,13,33]. Conforme mostrado na Figura 2.5, se o bit de controle do MUX, *X[i]*, (o bit do desafio respectivo relativo àquele estágio) é igual a:

- *0* - as entradas dos caminhos superior e inferior são transferidas para as suas respectivas saídas de cada MUX (superior e inferior);
- *1* - as entradas dos dois caminhos são trocadas (cruzadas), ou seja, a entrada do caminho superior é direcionada para a saída do MUX inferior e a entrada do caminho inferior é transferida para a saída do MUX superior.

Como na organização do circuito da Figura 2.6, os MUXes funcionam como um *crossbar*. A saída de cada MUX é conectada à entrada dos MUXes consecutivos. No final, as saídas dos últimos MUXes são conectadas a um *flipflop*, chamado de árbitro.

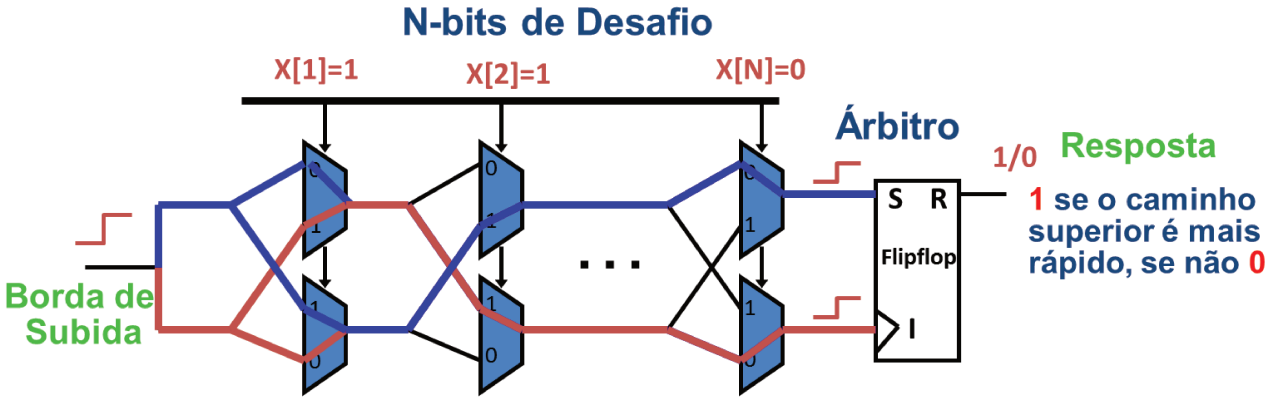


Figura 2.6: PUF usando Árbitro [33].

O circuito cria um par de caminhos de atraso para cada desafio *X[i]*. Para obter uma resposta na saída após um determinado desafio, um pulso (borda de subida) é dado a ambos os caminhos ao mesmo tempo. A partir daí, começa uma corrida deste sinal através de dois caminhos de atraso. Ao final o árbitro (*flipflop*) identifica e resolve qual o sinal foi mais rápido. Como mostrado na Figura 2.6, a saída no *flipflop* tem como resultado 1, se o sinal da entrada de dados (D) for mais rápido, e 0 caso contrário [6, 10, 13, 33].

A utilização de *flipflops* como árbitro deve ser feita com alguns cuidados. A chegada próxima dos sinais no árbitro após a corrida pelo circuito do PUF ocasionam valores imprevisíveis na saída quando os tempos de *setup* e *hold* do *flipflop* não são respeitados [33]. Este fenômeno ocorre devido à metaestabilidade. Para diminuir estatisticamente a ocorrência da metaestabilidade, [33] utiliza APUF com tamanho mínimo de 64 bits de desafio. Este tamanho mínimo possui 2^{64} pares de desafio resposta, o que também dificulta o mapeamento total de um PUF em bancada e também dificulta a criação de um modelo matemático ou computacional do funcionamento do APUF através de alguns desafios e respostas. Nos experimentos de [33], após estimular o PUF em bancada, foi selecionado um conjunto de desafio no qual os resultados não apresentam características de metaestabilidade.

No processo de identificação o mesmo APUF pode ser utilizado várias vezes com desafios diferentes para a obtenção de um conjunto de bits de resposta. Além disso, o APUF pode ser também replicado [33] para obtenção de um conjunto de bits de respostas com um desafio. Neste caso, são N cópias do mesmo projeto do APUF de M bits com uma a organização apresentada na Figura 2.7, sendo que cada APUF é responsável por um bit da resposta.

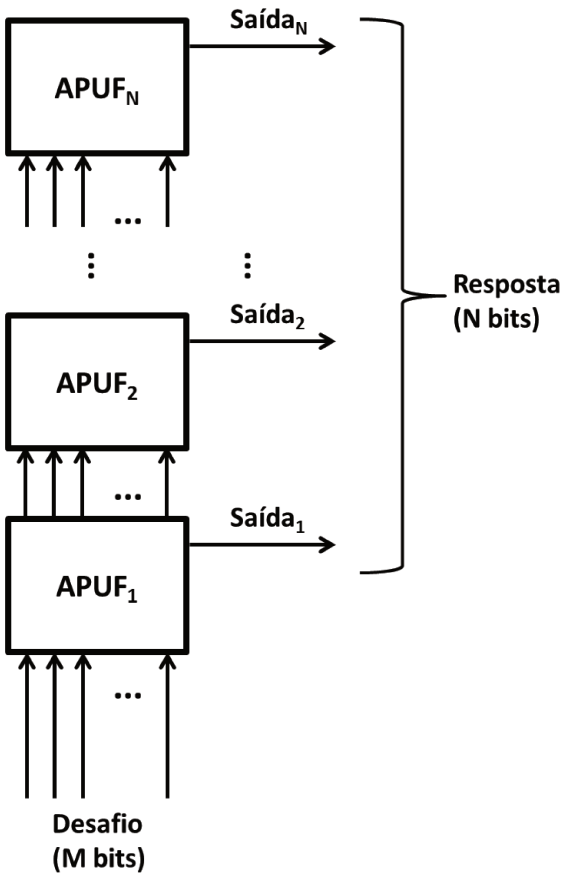


Figura 2.7: Organização do APUF MxN.

Como se observa na organização dos APUFs da Figura 2.7, N APUFs são dispostos em paralelo e submetidos simultaneamente ao mesmo desafio de M bits. Os bits das respostas de cada APUF (1 até N) são concatenados e assim é formada uma única resposta de N bits.

Dado que o APUF é um circuito simples, alguns ataques podem construir um modelo linear temporal e entender os parâmetros de alguns pares de desafio e resposta [33]. Para prevenir estes ataques, a saída do circuito do PUF pode ser ofuscada pela utilização de *XOR* em múltiplas saídas dos APUFs ou ainda uma saída do PUF pode ser usada como um dos sinais de controle de um outro MUX [6, 10, 13, 33]. No primeiro, a resposta é um XOR dos bits de cada APUF, no segundo caso, o desafio aplicado depende de um resultado anterior do APUF. Em [31] são apresentados resultados de ataque ao APUF através de modelagem matemática do seu funcionamento. Os ataques são baseados em técnicas de aprendizagem de máquinas, tais como Regressão Logística (RL) e Estratégias Evolutivas (EE). Os resultados apresentados em [31] demonstram que o APUF com 128 estágios pode

ter predição de 99,9% em 2,1s de treinamento, após a coleta de $39,2 \cdot 10^3$ CPRs. Estes experimentos foram realizados num *cluster* com 30 nós com *AMD Opteron Quadcores*. Entretanto, técnicas utilizadas para ofuscar o funcionamento do circuito interno do APUF, tal como no *Light APUF*, aumentam esse tempo para 267 dias de treinamento, sendo necessário $1 \cdot 10^6$ CPRs. Estas técnicas utilizadas para melhorar o APUF também podem ser utilizadas em outros modelos de PUFs para dificultar a predição dos CRPs.

2.2.3.2 PUF usando Tri-state

O PUF usando *Tristate* é semelhante ao APUF que utiliza multiplexadores (MUXes), ou seja, utiliza caminhos a serem percorridos pelos sinais. No entanto, o PUF baseado em *tristate*, proposto em [28], ao invés de ter dois caminhos de atraso que se interconectam, ele possui um circuito com dois caminhos exclusivos para cada sinal superior e inferior (Figura 2.10). Além disso, o circuito do PUF foi construído utilizando as variações de atraso dos *buffers tristate*. A célula do *buffer tristate* é mostrado na Figura 2.8.

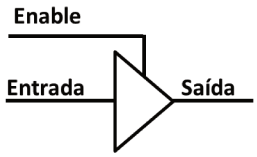


Figura 2.8: Modelo do *Buffer Tristate*.

O PUF usando *tristate* utiliza a estrutura apresentada na Figura 2.8 para fazer o modelo da unidade de atraso, como apresentado na Figura 2.9.

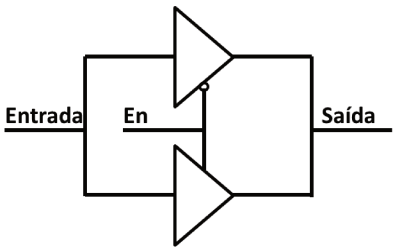


Figura 2.9: Modelo do Unidade de Atraso Usando *Buffer Tristate*.

As entradas e as saídas de dois *buffers tristate* são conectadas formando a estrutura da Figura 2.9, sendo que o *buffer tristate* superior mostrado possui um inversor na entrada. Esta estrutura permite a propagação do sinal por apenas um dos caminhos de acordo com o valor do sinal *En*, pois ele habilita apenas um dos dois *buffers*. Assim é determinado por qual *buffer tristate* passa o sinal iniciado pelo pulso. Isso altera a contribuição de cada unidade de atraso a partir do bit de desafio (C_N). A integração de unidades de

atraso feita com *buffer tristate* forma uma estrutura do PUF semelhante à apresentada na Figura 2.10.

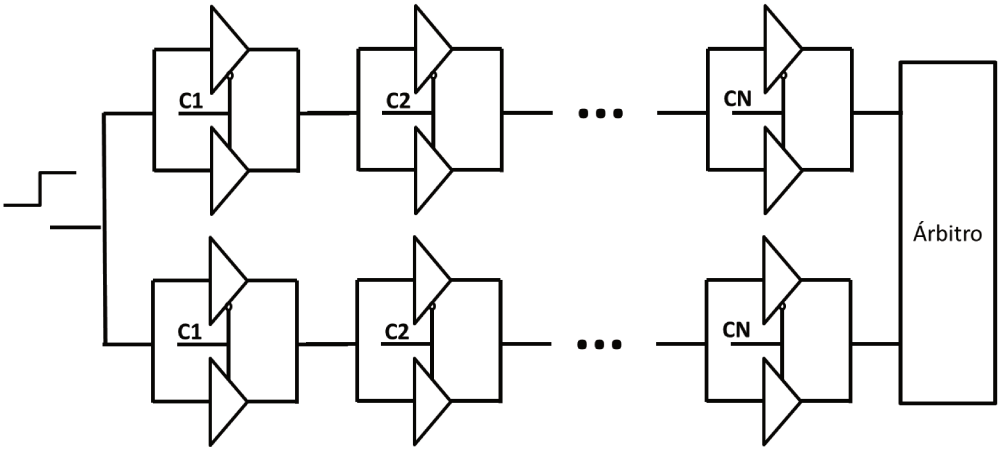


Figura 2.10: PUF usando tristate [28].

De acordo com o funcionamento do PUF usando *buffer tristate* o pulso de entrada da Figura 2.10 se propaga pelos caminhos superior e inferior. Em cada caminho, o sinal se propaga por N *buffers tristate* superior, ou *buffers tristate* inferior, de acordo com o desafio C. Por fim, o resultado da corrida é capturado pelo árbitro que, semelhante ao APUF, é um *flipflop*.

Este tipo de implementação de PUF melhorou a área e a consumo de energia em relação ao PUF usando multiplexadores (APUF) apresentado na seção 2.2.3.1 [28].

2.2.3.3 PUF Usando Anéis Oscilatórios - ROPUF

O PUF usando anéis oscilatórios (**ROPUF**) foi proposto em [33]. Este tipo de PUF utiliza um contador ao final da seleção de cada par de osciladores. Na Figura 2.11 é apresentado um oscilador em anel.

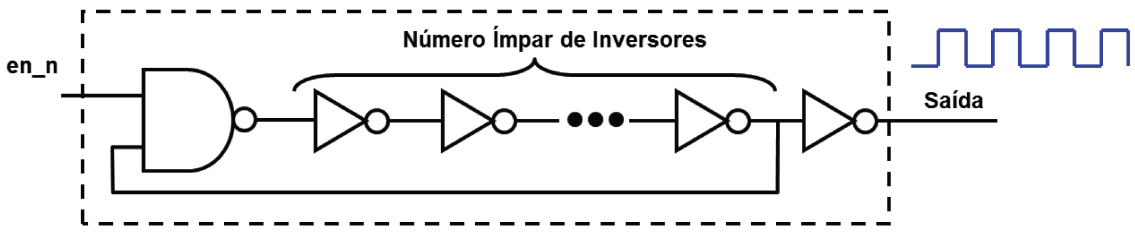


Figura 2.11: Anel oscilador utilizado pelo ROPUF.

Devido ao processo de fabricação, cada oscilador pode apresentar uma frequência de oscilação diferente. Assim, com a combinação de vários osciladores apresentada na Figura

2.11 foi construído o ROPUF (Ring Oscilator PUF) apresentado na Figura 2.12. Nele pode ser observado que há N osciladores e cada um deles está conectado a um par de MUXes. O desafio é aplicado nos chaveadores dos MUXes através da entrada *desafio*.

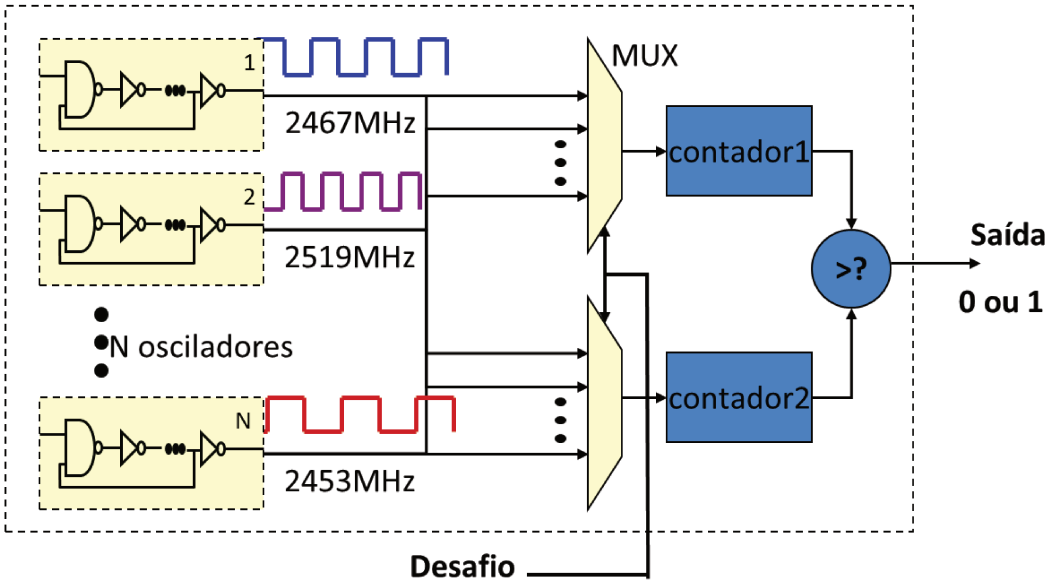


Figura 2.12: PUF usando anéis osciladores [33].

Como mostrado na Figura 2.12, com a seleção de dois osciladores os contadores iniciam a contagem dos pulsos. Ao final, a soma das corridas (número de oscilações) é contabilizada e a partir daí os valores dos contadores são comparados. O resultado desta comparação determina um bit 0 ou 1 indicando se o oscilador selecionado do multiplexador superior foi mais rápido ou mais lento que o oscilador do multiplexador inferior. Vale observar que os anéis oscilatórios são idênticos, dependendo, assim, o resultado apenas da variação no processo de fabricação.

No entanto, este PUF apresenta variações em relação a transitividade, pois quando um anel oscilador A possui maior frequência que um oscilador B e o oscilador B possui maior frequência que o oscilador C, fica claro que o oscilador A também possui uma maior frequência que o oscilador C. Tal fenômeno diminui bastante o número de pares de osciladores que geram diferenças de contagem substanciais entre si [33].

Além disso, o PUF usando anéis osciladores é suscetível à interferências externas. Osciladores com frequências próximas podem apresentar desvios quando submetidos a variações de temperatura ou quando expostos à ondas eletromagnéticas. Isto pode ocasionar num funcionamento incorreto do PUF [33].

2.2.3.4 Glitch PUF

O ***Glitch PUF*** utiliza uma fonte de *glitches* aleatórios e transforma esta forma de onda em um conjunto de bits de resposta, como apresenta a Figura 2.13.

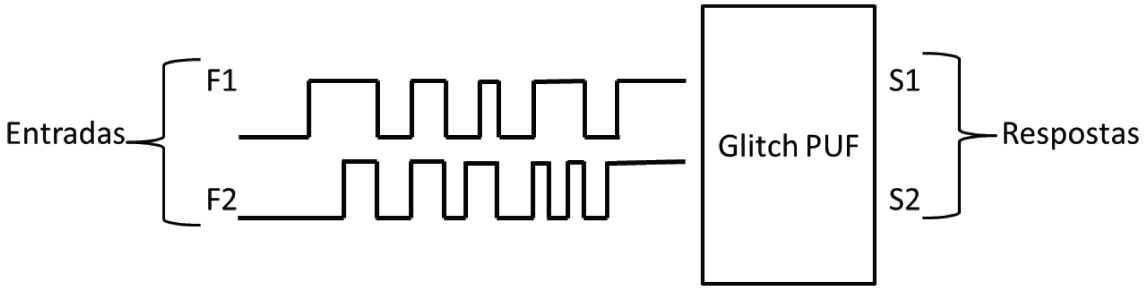


Figura 2.13: Funcionamento do Glitch PUF.

Glitch PUF é um modelo de PUF que não é caracterizado por corridas entre sinais. Ele utiliza as variações das formas de ondas geradas por circuitos combinacionais como fonte de *glitches* para gerar respostas distintas para cada forma de onda. Como apresenta a Figura 2.13, quando aplicadas as formas de ondas $F1$ e $F2$ são obtidas as respostas distintas $S1$ e $S2$.

O *Glitch PUF* consiste internamente por registradores de controle, registradores de dados e circuito de amostragem [34].

2.2.4 PUF Baseados em Memória

Nesta subseção são apresentados os principais modelos de PUFs que utilizam células de memória específica para que o seu funcionamento seja determinado pela aleatoriedade da variação do processo de fabricação.

2.2.4.1 SRAM PUF

Este PUF introduzido em [17] utiliza células de mémoria SRAM (*Static Random-Access Memory*) na sua construção. Uma célula de memória SRAM é composta por dois inversores em anti-paralelo conforme apresentado na Figura 2.14.

Como os valores lógicos dos bits, *bit* e *bit_b*, são complementares, a definição dos valores destes bits na inicialização é determinada pela corrida entre esses dois inversores. Assim, o que determina o valor lógico inicial de determinado bit é a diferença causada pela variação ocorrida no processo de fabricação [17, 26].

Destaca-se que em células SRAM tradicionais o tamanho dos transistores é dimensionado adequadamente de forma que essse efeito não ocorra para que a célula armazene o bit corretamente.

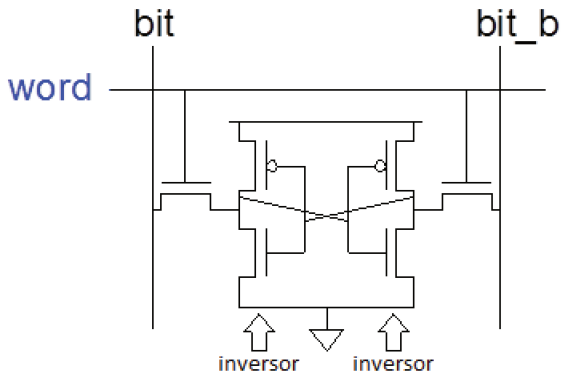


Figura 2.14: Célula SRAM [adaptada de [38]]

2.2.4.2 Butterfly PUF

O *Butterfly* PUF foi introduzido por [20]. A sua estrutura básica utiliza dois *latches* interconectados num acoplamento cruzado como é apresentado pela Figura 2.15.

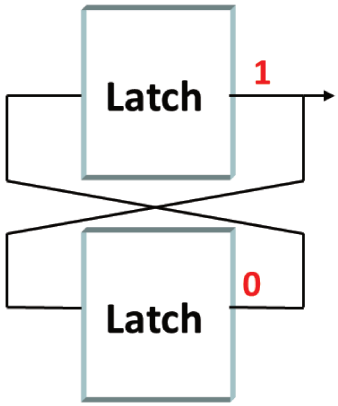


Figura 2.15: *Butterfly PUF*.

Como o *latch* é transparente e possui dois estados estáveis, é possível utilizá-lo como PUF através da funcionalidade de *clear/preset*. Desta forma um estado instável é introduzido após o qual o circuito converge para um dos dois estados estáveis. O estado para o qual cada célula do *Butterfly PUF* se estabiliza após a introdução do estado instável depende da variação do processo de fabricação das células de memória usadas na confecção dos *latches* [26].

2.2.4.3 *Latch* PUF

O *Latch* PUF proposto por [32] tem funcionamento parecido com o SRAM PUF e o Butterfly PUF. Em vez de ter dois inversores ou dois *latches*, o *Latch* PUF utiliza um *gated latch* SR, como apresentado na Figura 2.16 [26].

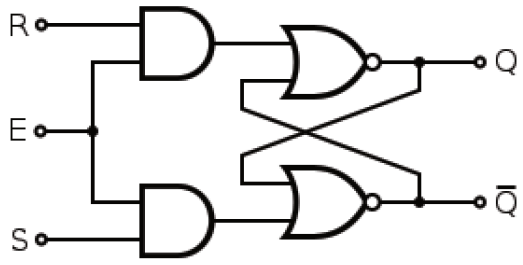


Figura 2.16: Circuito Lógico do *Latch* PUF.

Pelo funcionamento normal do *latch*, ao colocar o sinal E em '0' e os sinais R e S em '1', e em seguida alterar o sinal E para '1' (habilitar o *latch*), o *latch* se torna instável e novamente converge para um estado estável dependendo da diferença de atraso entre seus componentes resultante da variação no processo de fabricação. Assim, o *Latch PUF* possui o mesmo comportamento que o *Butterfly PUF* e o *SRAM PUF* [26].

2.2.4.4 MECCA PUF

O *(ME)mory (C)ell-based (C)hip (A)uthentication* PUF - MECCA PUF - é utilizado para autenticação e geração de chaves. Ele é baseado no mecanismo de falhas num *array* de memória. Proposto em [19], o MECCA PUF realiza a autenticação explorando variações intrínsecas do processo de leitura/escrita das células das memórias estáticas.

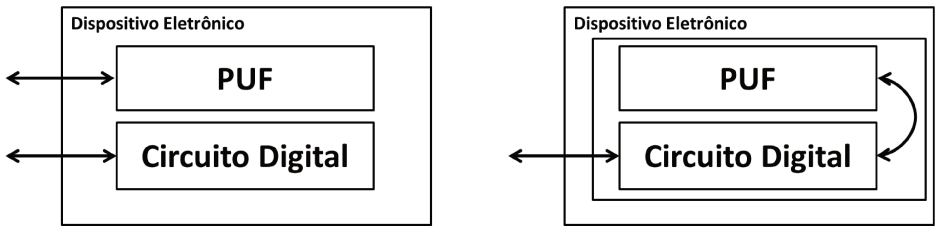
A confiabilidade das células é caracterizada após a fabricação e testes através da indução de falhas temporais, tais como falhas de gravação e de acesso, utilizando para isso um controlador programável do ciclo de funcionamento da *word line* que é inserido no circuito [19].

2.2.5 Técnicas Utilizadas com PUFs

Nesta seção são descritas algumas técnicas relacionadas com PUFs. Elas são utilizadas para encapsular ou até mesmo ofuscar o funcionamento do PUF dificultando assim a sua modelagem.

2.2.5.1 PUF Controlado

Segundo a definição em [6], um PUF é controlado se o desafio e a resposta não podem ser acessados diretamente. O PUF fica isolado fisicamente do processo de comunicação. Isto faz com que um atacante não tenha acesso direto aos valores gerados pelo PUF. A Figura 2.17 apresenta a diferença entre um processo tradicional de identificação de um PUF e um outro processo que tem uma organização que utiliza o PUF controlado.



(a) PUF Não Controlado. (b) PUF Controlado.

Figura 2.17: Comparação entre PUF Não Controlado e o PUF controlado.

A Figura 2.17(a) apresenta uma estrutura na qual o PUF é acessado diretamente. Já na Figura 2.17(b) o PUF não é acessado diretamente, pois somente o circuito digital sabe qual o desafio aplicado e a resposta obtida do PUF.

Assim, no PUF controlado é aplicado um pseudo-desafio. A partir deste pseudo-desafio, normalmente calcula-se o *hash*. Este *hash*, então é submetido ao PUF. Posteriormente, calcula-se o resultado da resposta quando aplicado o desafio *hash* ao PUF. A Figura 2.18 apresenta um projeto de PUF Controlado proposto em [6].

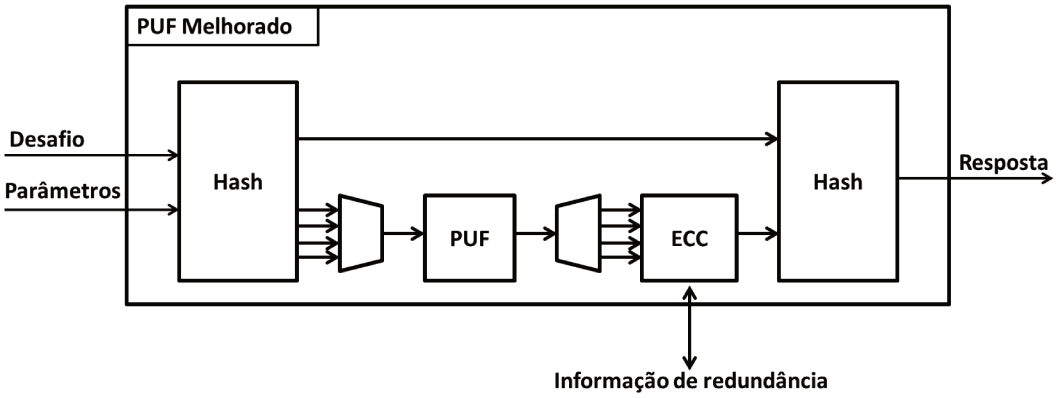


Figura 2.18: PUF Controlado Proposto por [6].

O processo de identificação apresentado na Figura 2.18 recebe como entradas um desafio (pseudo-desafio) e parâmetros utilizados pela função *hash*. O PUF recebe como

desafio o resultado do *hash*. A resposta do PUF é processada por outros sistemas, ECC e outra função *hash*. Ao final é obtida uma resposta (pseudo-resposta). Nesse procedimento, não é possível descobrir qual o desafio aplicado ao PUF e nem a resposta dele.

Vale ressaltar que o controle realizado pelo algoritmo pode restringir os desafios que são submetidos ao PUF, limitando também o conjunto de respostas. O controle é a ideia fundamental que permite aos PUFs fazerem uma identificação de uma aplicação [12].

2.2.6 Comparação entre PUFs

Nesta seção são apresentadas comparações de alguns modelos de PUF apresentada em [25], onde PUFs são comparados de acordo com o crescimento do número de bits do desafio do PUF. Os modelos que não foram listados possuem semelhanças ou são aperfeiçoamentos dos modelos de PUFs discutidos neste trabalho. Os dados apresentados na Tabela 2.2 apresentam como alguns modelos de PUFs escalam com o aumento do número de bits do desafio afetando assim o aumento da imprevisibilidade das respostas e do tamanho da implementação dos PUFs.

Tabela 2.2: Imprevisibilidade X Tamanho da Implementação em Relação ao Tamanho do Desafio.

Modelo	**Imprevisibilidade das Respostas**	**Tamanho da Implementação**
Arbiter PUF	Linear	Linear
Basic RO PUF	Linear	Linear
Optical PUF	Superlinear	Linear
Coating PUF	Exponencial	Exponencial
SRAM/FF PUF	Exponencial	Exponencial
Latch/Butterfly PUF	Exponencial	Exponencial
Paired RO PUF	Exponencial	Exponencial

Na comparação realizada em [25], foram analisadas e estudadas diversas características destes PUFs. Questionam se seria possível desenvolver um modelo de PUF em que o aumento no tamanho do desafio implicasse em um crescimento exponencial da imprevisibilidade das respostas mas também em um aumento linear do tamanho da implementação.

A imprevisibilidade das respostas está na complexidade de modelar o funcionamento do PUF. Como se observa na Tabela 2.2, ainda não havia sido desenvolvido algum modelo de PUF com estas características.

No Capítulo 3 é apresentado um novo modelo de PUF que responde à esta questão.

2.2.7 Considerações Finais

Esse Capítulo 2 apresentou os principais conceitos relacionados aos PUFs e apresenta uma descrição de vários modelos de PUFs encontrados na literatura.

Como existem vários modelos de PUFs e ainda uma grande variação de muitos deles, focou-se este trabalho apenas nos modelos de PUFs baseados em atrasos porque o novo modelo de PUF aqui proposto também é baseado em atrasos e possui características em comum aos primeiros.

Capítulo 3

CRPUF - Cylindrical Reconvergence PUF

Dada uma visão geral dos principais modelos de PUFs já desenvolvidos e destacadas as suas principais características, este Capítulo 3 apresenta as motivações e as características de um novo modelo de PUF proposto com o intuito de introduzir melhorias com relação a alguns modelos de PUFs baseados em atrasos. A seção 3.1 apresenta as principais ideias que foram utilizadas para desenvolver o CRPUF. A seção 3.2 descreve a arquitetura e a nomenclatura definida para o CRPUF. A seção 3.3 apresenta um exemplo do CRPUF para descrever a sua estrutura. A seção 3.4 especifica os procedimentos para aplicar o desafio e pra obter a resposta no CRPUF. A seção 3.5 apresenta dois tipos do CRPUF.

3.1 Concepção da Arquitetura

Após a caracterização e a comparação dos outros modelos de PUFs apresentados no Capítulo 2, destaca-se que uma das motivações iniciais deste projeto foi a de explorar outras possibilidades de modelos de PUFs baseados em atrasos. Uma destas possibilidades foi a de expandir o modelo APUF. Nesse modelo que foi apresentado na subseção 2.2.3.1 há apenas a corrida simultânea em dois caminhos definidos por um desafio. A expansão da quantidade de caminhos no CRPUF traz maiores possibilidades de resultados, pois há aumento de trajetos para as corridas simultâneas.

É de grande importância que um PUF apresente muitos resultados diferentes na saída, pois esta característica, tende a reduzir a probabilidade de *aliasing*, ou seja, a probabilidade que dois indivíduos produzam as mesmas respostas quando submetidos ao mesmo conjunto de desafios. Indivíduos são PUFs físicos do mesmo modelo a serem identificados através da aplicação do desafio e a obtenção da reposta. Por exemplo, no modelo APUF, independente do tamanho do desafio, o resultado dos indivíduos sempre terá duas possibi-

lidades na resposta, '0' ou '1'. Observa-se que esta expansão pretendida pelo novo modelo CRPUF tem o objetivo de dificultar uma possível clonagem do funcionamento do PUF, pois aumenta consideravelmente o custo computacional para a modelagem das respostas, pois esta apresentaria mais combinações de resultados.

A estrutura do CRPUF é apresentada em seguida descrevendo sua construção de forma incremental.

Inicialmente, considere uma organização semelhante à estrutura apresentada na Figura 3.1. Assim, a disposição espacial em formato de cone, como apresenta a Figura 3.1, foi utilizada porque há inúmeros caminhos das entradas para a saída.

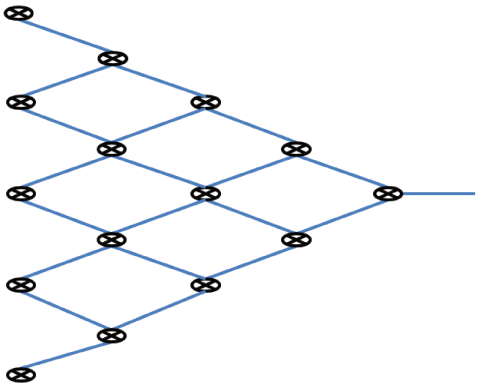


Figura 3.1: Dependência da Saída.

Com esta configuração da Figura 3.1, a saída do PUF se torna dependente da alteração de qualquer entrada. E assim, para que as mudanças não fossem mascaradas pela mudança de um bit, são utilizadas portas lógicas XOR, pois a alteração de um bit faz com que necessariamente a saída seja modificada. Desta forma, uma cadeia de portas lógicas XOR na estrutura da Figura 3.1 é capaz de modificar a saída ocorrendo a alteração de qualquer bit da entrada.

Em um segundo passo, a estrutura da Figura 3.1 foi expandida para que o PUF também tenha uma maior quantidade de saídas. Para isso foi utilizada uma expansão que resultou numa perspectiva semelhante à malha apresentada na Figura 3.2.

Assim, com o aumento da quantidade de saídas representada pela evolução da representação da Figura 3.1 para a Figura 3.2, o número de saídas e caminhos foram ampliados.

Entretanto, uma única mudança na entrada da estrutura da Figura 3.2 ainda pode não afetar todas as saídas e a quantidade de caminhos não é a mesma das entradas para as saídas, conforme ilustrado na Figura 3.2 através da quantidade de caminhos possíveis das entradas até as saídas $S1$, $S2$, $S3$, $S4$ e $S5$.

Em um terceiro passo, a malha foi fechada na forma de um cilindro, como apresentada na Figura 3.3. Agora qualquer saída tem o mesmo número de caminhos a partir

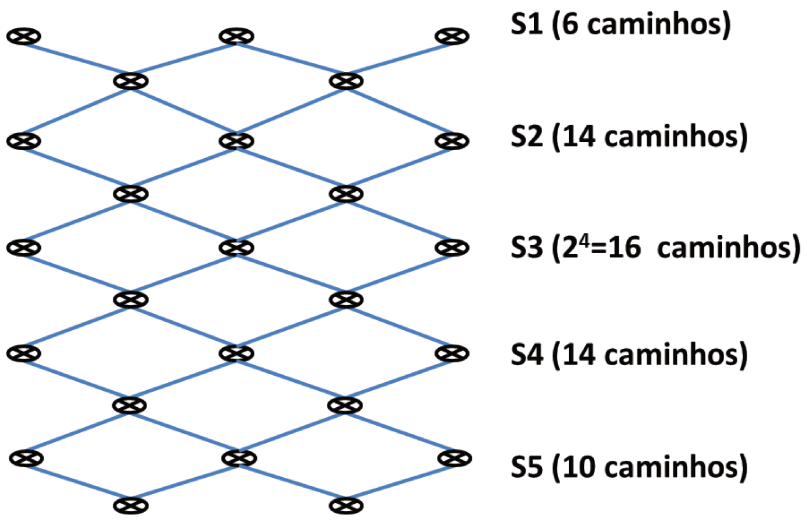


Figura 3.2: Malha Reticulada de XORs

das entradas. A modificação aplicada na Figura 3.2 resultam em 2^3 caminhos (o máximo) para qualquer entrada.

Na estrutura da Figura 3.3 qualquer mudança na entrada se propaga para todas as saídas. Assim, o cilindro se apresenta como uma malha, onde cada estágio interfere diretamente em duas células do estágio seguinte. O objetivo desta estrutura é cascatear estas modificações de forma que sejam criadas uma grande quantidade de caminhos de propagação com atrasos equilibrados.

Finalmente, parte da estrutura (3 estágios) da Figura 3.3 foi rearranjada no plano como mostrada na Figura 3.4. Este rearranjo da disposição topológica das portas busca uniformizar os comprimentos das interligações em todos os caminhos. Isto facilita o projeto físico do PUF. Essa estrutura lógica de propagação dos sinais é apresentada na Figura 3.4.

Na Figura 3.4, destaca-se que a estrutura tem o projeto totalmente escalável, pois apresenta uma organização regular e pode ser apenas replicável. A composição e a organização da Figura 3.4 são descritas na seção 3.2.

3.2 Arquitetura do CRPUF

Nesta seção são descritas todas as estruturas que compõem o $CRPUF$, quais suas funcionalidades, organização e contribuições para a arquitetura.

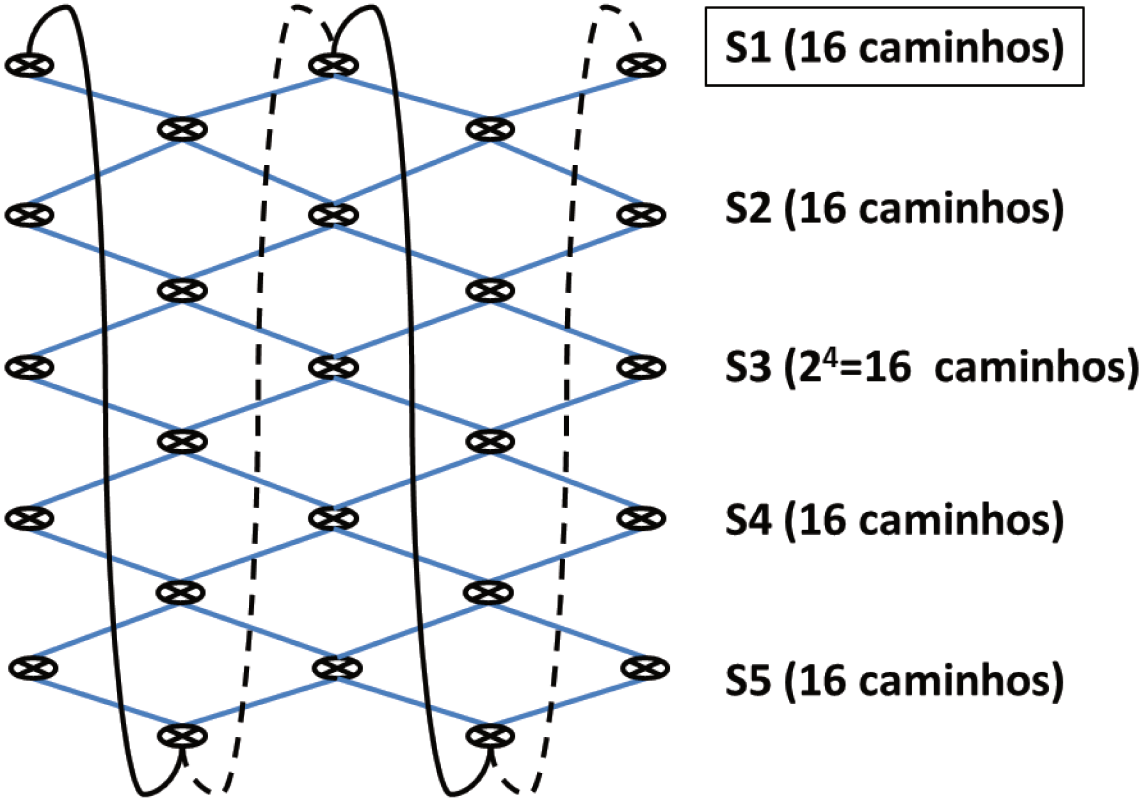


Figura 3.3: Malha Interconectada em Forma de Cilindro.

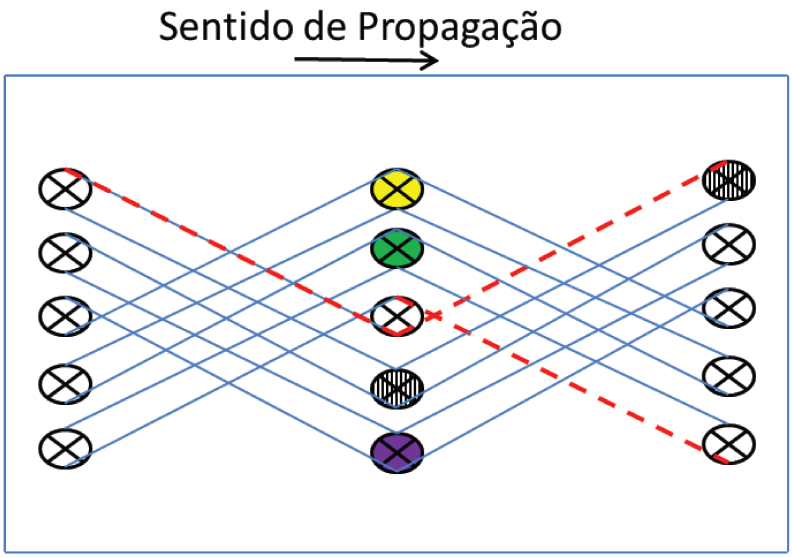


Figura 3.4: Malha de Organização e Propagação dos Sinais no CRPUF.

3.2.1 Organização do CRPUF

A organização lógica do CRPUF é composta por estágios. Cada estágio do PUF possui uma mesma quantidade ímpar de portas lógicas XORs. Detalhes sobre esta restrição da quantidade de células são apresentados na seção 3.3. Assim, por enquanto, a composição de cada estágio é apresentada na Figura 3.5.

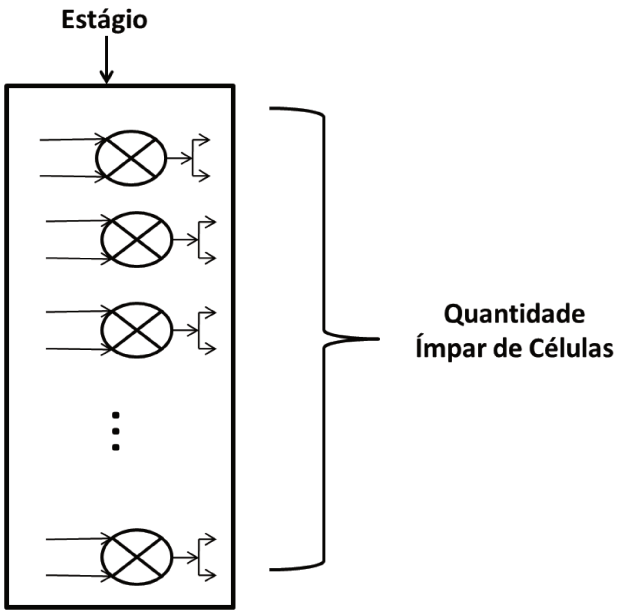


Figura 3.5: Estágio do CRPUF.

No projeto lógico como apresentado na Figura 3.5, as células dentro do estágio são dispostas na vertical. Pra esta organização não há interconexão entre elas dentro do mesmo estágio. As conexões entre as células são realizadas entre células de estágio diferentes como apresentado na 3.6.

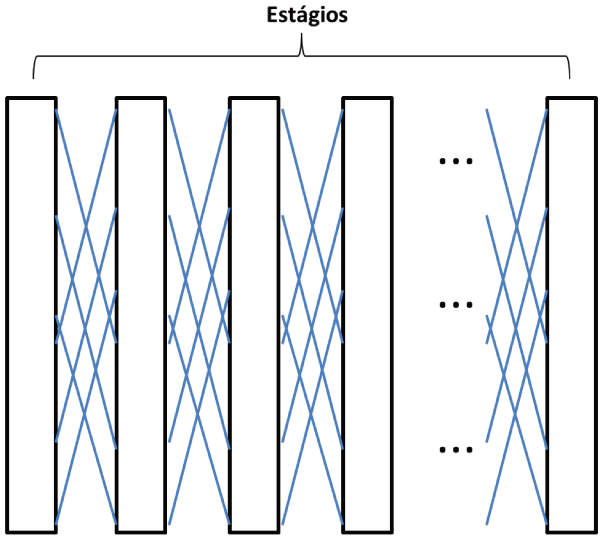


Figura 3.6: Organização dos Estágios do CRPUF.

Assim, o CRPUF é composto por vários estágios. Observa-se, entretanto, que um

CRPUF com apenas dois estágios apresenta uma variação mínima no resultado da diferença dos atrasos decorrentes da corrida entre os sinais. No outro sentido, muitos estágios aumentam o percurso dos caminhos que serão estimulados e os atrasos acumulados, colaborando estatisticamente para maiores diferenças e contribuições para captura final do resultado, que é apresentada na subseção 3.5. A quantidade de estágios do CRPUF afeta a quantidade de hardware do dispositivo, devendo para isso ser considerado o custo e o benefício.

3.2.2 Modelo Geral do CRPUF

Esta subseção descreve uma abordagem genérica para a configuração do tamanho do CRPUF. Assim as dimensões do CRPUF são determinadas pela quantidade de células XORs por estágio ou quantidade de linhas, e pela quantidade de estágios que compõe o PUF ou quantidade de colunas, como apresentado na figura 3.7.

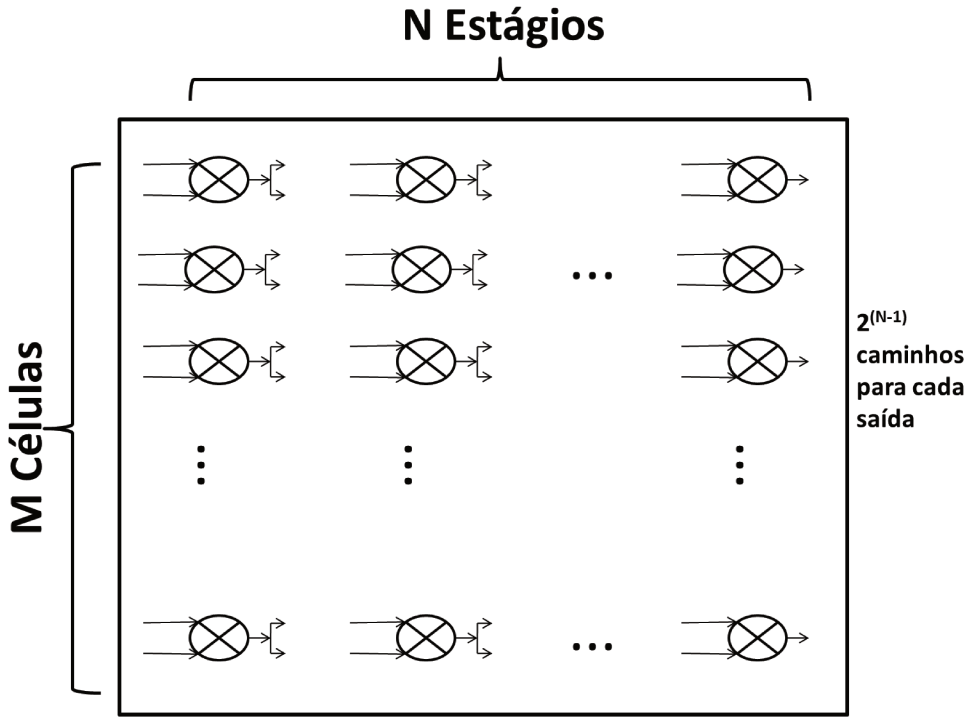


Figura 3.7: CRPUF(MxN).

O $CRPUF_{MxN}$, como apresentado na Figura 3.7, representa um CRPUF que tem M células XOR por estágio ou número de linhas e ainda N estágios ou número de colunas. Por exemplo, um $CRPUF(3x5)$, possui três células XORs por estágio (linhas) e é composto por 5 estágios (colunas).

3.3 Exemplo: CRPUF(3x3)

Na organização em formato de cilindro, em encontro de duas linhas no estágio existe uma porta XOR cuja saída está ligada em duas outras células do estágio (coluna) seguinte. Isso faz com que, dependendo do comprimento (número de estágios) do cilindro, a alteração em apenas um bit na entrada se propague de alguma forma como estímulo para todas as saídas do cilindro. Para que haja uma propagação em todas as saídas do $CRPUF$, é necessário que a quantidade de estágios (colunas) seja maior que a quantidade células por estágio (linhas).

O projeto lógico de um $CRPUF(3x3)$ com estrutura semelhante à Figura 3.4, é apresentado pela Figura 3.8.

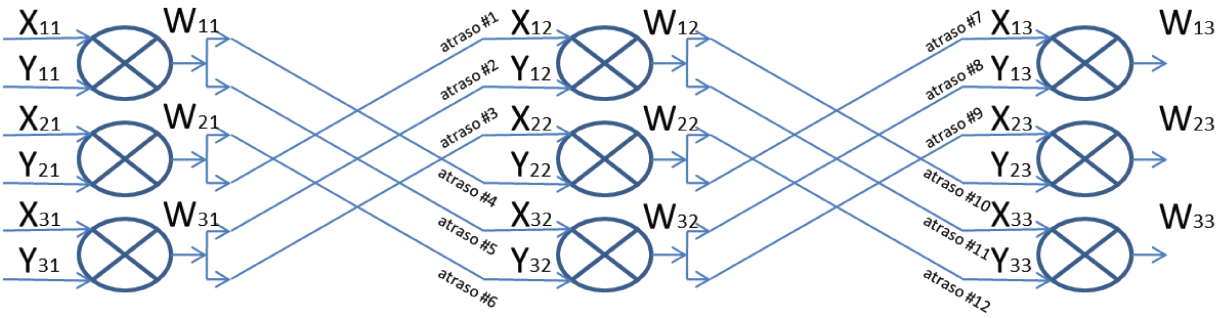


Figura 3.8: $CRPUF_1 3x3$

De acordo com a nomenclatura utilizada, o PUF da Figura 3.8 é um $CRPUF(3x3)$. Ele possui 3 linhas (portas XOR por estágio) por 3 colunas (número de estágios). Assim, no total são 9 células XOR no $CRPUF(3x3)$.

Como pode ser observado na Figura 3.8, por projeto todos os caminhos geometricamente já possuem os mesmos tamanhos do ponto de vista lógico. Isso facilita o projeto físico (*place* e *route*) em qualquer tecnologia que se pretenda implementar o PUF (FPGA, *Standard Cell*, *Gate Array* ou *Full Custom*).

Na Figura 3.8 é possível observar que cada célula de um estágio se conecta com outras duas células consecutivas do estágio seguinte. A concepção do roteamento, possibilitando caminhos igualmente distribuidos, resulta em uma célula central na coluna com conexões com as células extremas da coluna (estágio) seguinte, sendo assim diferente das demais, por exemplo, W_{21}, W_{22} ou W_{23}. Por este motivo, o número de XORs de uma coluna deve ser ímpar. Esta restrição garante que já no projeto lógico é possível obter a igualdade entre os tamanhos dos caminhos. Apesar de existirem duas células nas extremidades dos estágios, estas células são continuas e interconectadas como uma malha do ponto de vista lógico.

A variação do processo de fabricação faz com que os atrasos dos caminhos da Figura 3.8 (atraso #1 ao atraso #12) sejam diferentes. As diferenças entre estes atrasos fazem com que as saídas possuam comportamentos dinâmicos distintos. A diferença se dá de acordo com a chegada do primeiro sinal até que todos se estabilizem. Assim, ao aplicar um estímulo ao circuito, a saída final será diferente para os dispositivos físicos porque as transições na saída serão diferentes. Desta forma, a ordem, o momento e a quantidade das mudanças na saída serão determinados pelo espalhamento entre os atrasos que serão diferentes devido à variação durante o processo de fabricação do semicondutor.

3.4 Desafio e Resposta

Esta seção define o conceito de desafio e de sua respectiva resposta para o CRPUF. Para descrever o mecanismo utilizado, supõe-se no exemplo a seguir a existência de dois indivíduos IND_1 e $IND2$ do modelo $CRPUF$ a serem identificados, como apresentado na Figura 3.9.

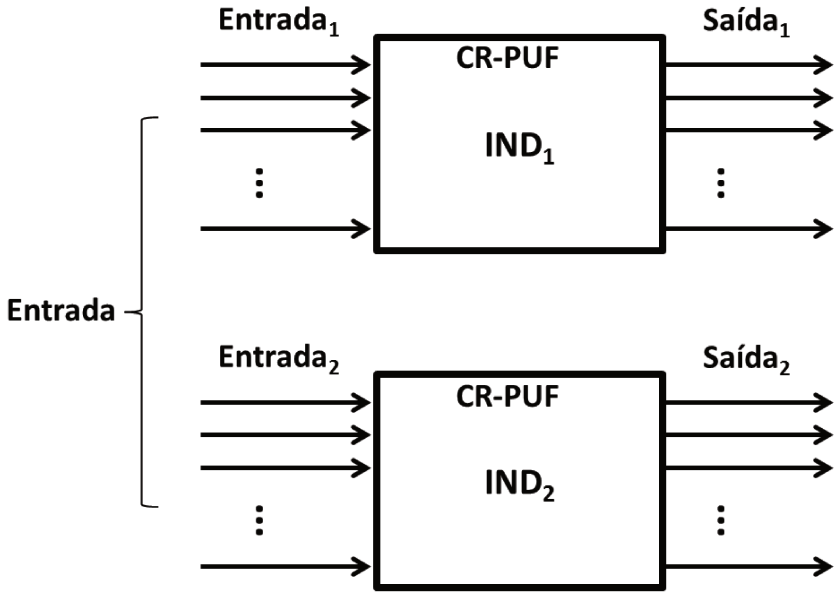


Figura 3.9: Entradas e Saídas do CRPUF.

Para identificar cada indivíduo da Figura 3.9 é necessária a aplicação de desafios para que seja obtido o perfil das saídas através do mecanismo de captura apresentado na Figura 3.10.

O mecanismo de captura para a obtenção da resposta no CRPUF, como no caso do APUF, também utiliza *flipflops* para extrair as informações que são utilizadas para identificar de forma única cada indivíduo quando ele for estimulado por um desafio. Assim

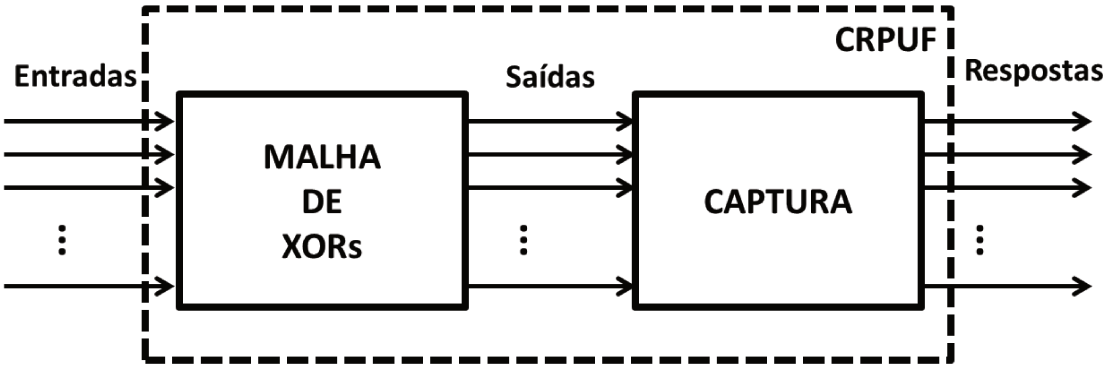


Figura 3.10: Entradas, Saídas, Sistema de Captura e Respostas do CRPUF.

como o APUF, o CRPUF também é suscetível à metaestabilidade de acordo com a diferença dos atrasos dos sinais das entradas dos *flipflops*. Isso significa que as mesmas técnicas utilizadas no APUF podem ser utilizadas no CRPUF para reduzir o efeito da metaestabilidade na identificação do circuito integrado.

Para entender a diferença entre entrada, desafio, saída e respostas do CRPUF, são apresentados a seguir na Figura 3.11 um diagrama de uma forma de onda hipotética dos dois indivíduos apresentados na Figura 3.9. Para manter a clareza da explicação e sem prejuízo de generalização, a saída dos indivíduos da Figura 3.9 possuem apenas dois sinais na saída.

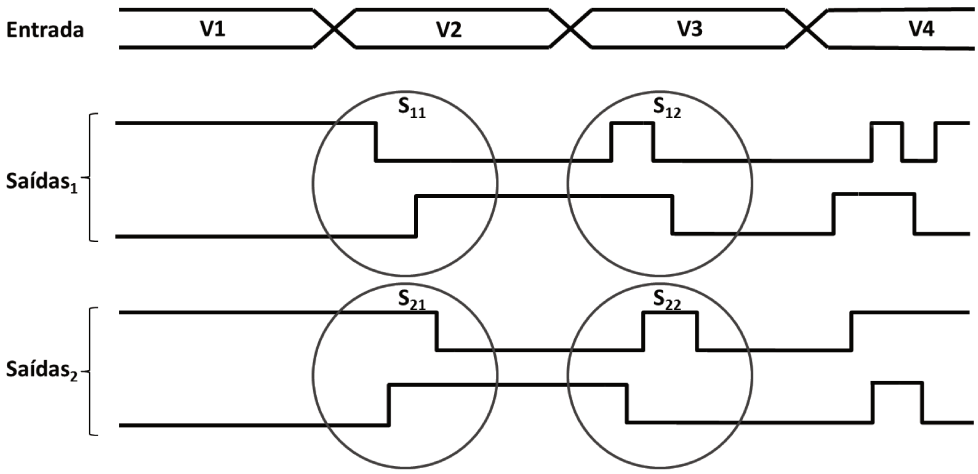


Figura 3.11: Representação do Funcionamento do CRPUF.

Por definição deste trabalho, o conjunto de bits aplicados nas entradas dos CRPUF são chamados de vetores. Assim, Os vetores $V1$,$V2$,$V3$ e $V4$ são aplicados nas entradas dos indivíduos IND_1 e IND_2. Desta forma, estas são as entradas para a forma de onda recebidas pelos dois indivíduos do modelo CRPUF na Figura 3.11.

Um desafio é a aplicação de dois vetores consecutivos. O primeiro vetor traz o PUF para um estado conhecido, em seguida o segundo vetor aciona a corrida no CRPUF para que a sequência das transições sejam capturadas pelos *flipflops*. O segundo vetor do desafio anterior pode ser aproveitado como primeiro vetor para a aplicação de um novo desafio. Desta forma, por exemplo, 4 vetores podem ser utilizados para a aplicação de 3 desafios.

Observa-se que de acordo com a forma de onda apresentada na Figura 3.11 os dados que indicam a diferença são apresentados na Tabela 3.1.

Tabela 3.1: Desafios e Saídas do CRPUF Para a Forma de Onda da Figura 3.11.

Desafio	**Saída** IND_1	**Saída** IND_1
($V1$,$V2$)	S_{11}	S_{12}
($V2$,$V3$)	S_{21}	S_{22}

A resposta do PUF para cada desafio da forma de onda hipotética da Figura 3.11 e destacada na Tabela 3.1 depende da captura pelo *flipflop*.

A sequência de duas entradas diferentes (um par de entradas) determina um desafio aplicado ao CRPUF. Como apresenta a Tabela 3.1, a entrada $V2$ aplicada ao CRPUF que havia recebido previamente a entrada $V1$ (desafio $(V1, V2)$) pode ter uma resposta diferente caso a entrada $V2$ fosse aplicada após o CRPUF ter recebido a entrada $V3$ (desafio $(V3, V2)$).

Por se tratar de um circuito combinacional e que ainda pode ser observado na Figura 3.11, a saída estável do CRPUF após transientes é sempre a mesma quando aplicada uma determinada entrada, no entanto, as transições dos sinais são potencialmente diferentes e, por isso, são utilizadas para identificar o PUF.

Uma métrica importante para um PUF é o NDP (**Número de Desafios Possíveis**). No $CRPUF(MxN)$ o NDP é determinado pela combinação entre os pares de entradas possíveis menos o número em que eles são iguais, ou seja, menos os casos em que $V1 = V2$.

$$NDP = 2^M \cdot 2^M - 2^M \tag{3.1}$$

Os pares de entradas iguais são descartados pois eles não modificam os resultados dos sinais no CRPUF.

3.5 Captura da Saída do CRPUF

A captura da saída como apresentada na Figura 3.10 é feita por uma estrutura que é acionada pela chegada dos sinais do $CRPUF$ na saída. Igualmente ao $APUF$, o $CRPUF$

utiliza *flipflops* para capturar a diferença do atraso na corrida entre dois sinais. Esta diferença, portanto, também deve ser o suficiente para não violar o tempo de *setup* e *hold* de um *flipflop* que é utilizado para capturar a resposta no CRPUF.

Neste trabalho foram abordados apenas duas formas de implementação para identificar a riqueza de transição dos sinais na saída do $CRPUF$. No entanto, outras combinações podem ser utilizadas para que seja aproveitado o máximo da diversificação sem aumentar muito a quantidade de hardware causada pela inserção de *flipflops*. Desta forma, é possível aumentar a capacidade de identificação e distinção dos dispositivos utilizando várias combinações das saídas.

Para o CRPUF desenvolvido, são apresentados dois tipos de CRPUF que utilizam agrupamento dos sinais na entrada dos *flipflops* de formas distintas. O $CRPUF_1$ utiliza uma combinação de M saídas de 2 em 2 entre os sinais da saída do $CRPUF$, ou seja, C_2^M. Já o outro tipo, o $CRPUF_2$, utiliza um arranjo M saídas de 2 em 2, ou seja, (A_2^M). Desta forma, é possível caracterizar substancialmente a riqueza inserida no circuito pela variação do processo de fabricação. O ideal é utilizar todas as combinações possíveis para que seja possível capturar as corridas com maior quantidade de detalhes. No entanto, em alguns casos que tenham grande quantidade de células XOR por estágio isso pode se tornar inviável pela grande quantidade de hardware que deve ser inserida no circuito com a adição dos *flipflops*. Sendo assim, nas duas subseções seguintes são apresentados cada um destes tipos de CRPUF.

3.5.1 $CRPUF_1$

O $CRPUF_1$ é um tipo de $CRPUF$ que utiliza uma combinação C_2^M das saídas para os *flipflops* para capturar o resultado da corrida entre os sinais do CRPUF. Para um CRPUF de M saídas, utilizando a fórmula de combinação entre as saídas de 2 em 2, temos uma quantidade de combinações possíveis que produzem N_R resultados (igual ao número de *flipflops*):

$$N_{R1} = \frac{M \cdot (M-1)}{2} \tag{3.2}$$

Portanto, existem $\frac{M \cdot (M-1)}{2}$ possibilidades de combinações de sinais que é igual ao número de *flipflops* (número de resultados N_R).

A Figura 3.12 apresenta a estrutura de um $CRPUF_1(3x3)$.

Assim, podem ser feitas várias combinações entre os sinais de entrada de cada *flipflop*. No exemplo da Figura 3.12 são sinais de entrada e *clock* dos três *flipflops* respectivamente os pares (W_{13}, W_{23}), (W_{23}, W_{33}) e (W_{33}, W_{13}). Observa-se que a quantidade de *flipflops* depende apenas da quantidade de linhas ou número de células XOR por estágio do

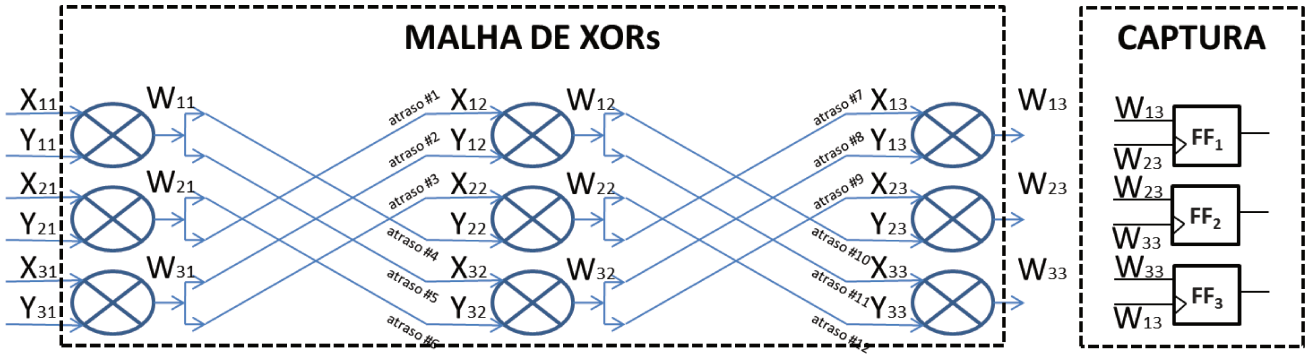


Figura 3.12: $CRPUF_1(3x3)$.

$CRPUF_1$. Desta forma, a quantidade de colunas ou estágios não interfere na quantidade dos *flipflops*.

3.5.2 $CRPUF_2$

O $CRPUF_2$ é um tipo de $CRPUF$ que utiliza um arranjo A_2^M dos sinais das saídas para os *flipflops* para que ele capture o resultado da corrida entre os sinais do CRPUF. O funcionamento do $CRPUF_2$ é idêntico ao $CRPUF_1$, no entanto, a quantidade de bits na saída é diferente. No $CRPUF_2$ é possível capturar uma maior diversidade de corridas entre os sinais que o $CRPUF_1$. Ele tem uma capacidade maior de capturar a riqueza de comportamentos manisfestados por *glitches* e *hazard* dinâmicos, que são tanto mais frequentes quanto for maior o número de estágios (N) do CRPUF. Pode ser que a quantidade de hardware não seja tão significativa em relação ao hardware total. E sendo assim, ele apresenta o resultado das corridas com maior fidelidade. Desta forma, este modelo utiliza *flipflops* em todas as combinações de saídas. Assim, para um $CRPUF_2$ de M saídas, utilizando a fórmula de arranjo (A_2^M) entre as saídas de 2 em 2, temos uma quantidade de combinações possíveis entre os sinais com *flipflops* iguais a:

$$N_{R2} = M \cdot (M-1) \tag{3.3}$$

Desta forma teremos $M \cdot (M-1)$ bits de saída para o $CRPUF_2$. Combinando as saídas do $CRPUF_2(3x3)$ (W_{13},W_{23} e W_{33}) da Figura 3.8 como entradas e *clocks* de *flipflops*, como apresentado na Figura 3.13, obtêm-se um $CRPUF_2(3x3)$.

No exemplo da Figura 3.13 são sinais de entrada e *clock* dos seis *flipflops* respectivamente os pares (W_{13}, W_{23}), (W_{23}, W_{13}), (W_{13}, W_{33}), (W_{33}, W_{13}), (W_{23}, W_{33}) e (W_{33}, W_{23}). Assim, o número de *flipflops* do $CRPUF_1$ é igual ao dobro do número de *flipflops* do $CRPUF_1$.

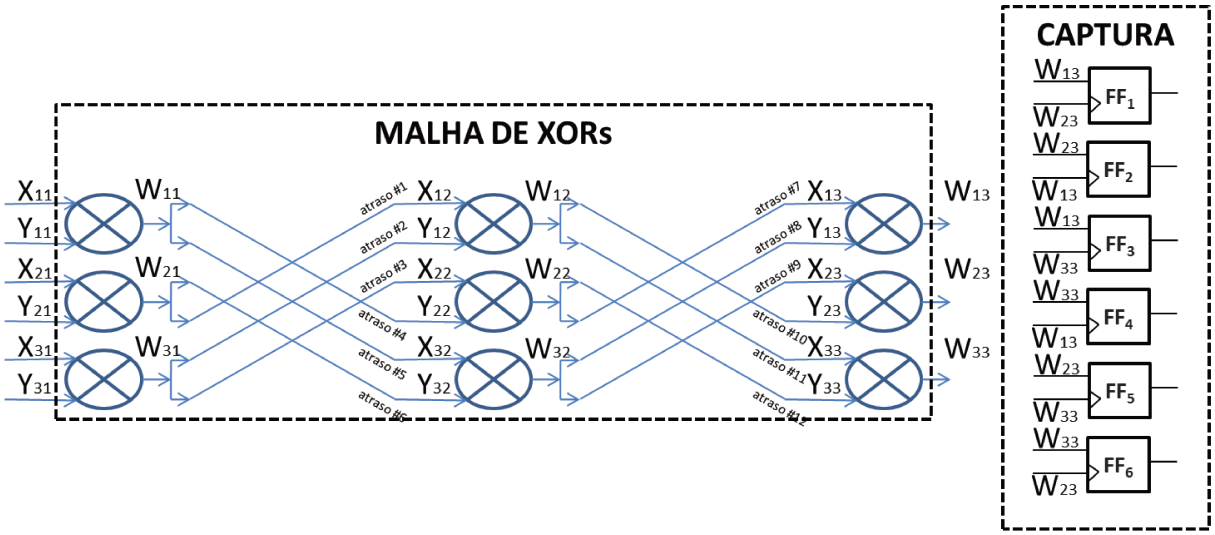


Figura 3.13: $CRPUF_2(3x3)$.

3.6 Considerações Finais

Este Capítulo 3 descreveu o novo modelo de PUF, o *Cylindrical Reconvergence PUF* (CRPUF).

Entre as principais características do CRPUF podemos citar que ele apresenta:

- estrutura regular com várias saídas;
- identificação do indivíduo via corrida de propagação durante a aplicação de 2 vetores de estímulo;
- sistema de captura utilizando *flipflops*.

Foram apresentados dois tipos de CRPUF: $CRPUF_1$ e $CRPUF_2$. O primeiro utiliza uma combinação das saídas aos pares, já o segundo, utiliza um arranjo das saídas aos pares.

O próximo Capítulo 4 apresenta os resultados experimentais através do planejamento (seção 4.2.1), execução (seção 4.1) e análise (seção 4.3) para verificação das propriedades e desempenho do CRPUF quando comparado ao APUF.

Capítulo 4

Resultados Experimentais

Feita uma visão geral dos principais modelos de PUFs já desenvolvidos (Capítulo 2) e destacadas as principais características do CRPUF desenvolvido neste trabalho (Capítulo 3), este Capítulo 4 apresenta os experimentos (seção 4.2) que comparam o APUF (*Arbiter PUF*) com o CRPUF (*Cylindrical Reconvergence PUF*).

Nos experimentos é utilizado o APUF porque ele apresenta características semelhantes às do CRPUF, tais como ser baseado em atraso e utilizar *flipflop* como árbitro. Além disso, o APUF também é um modelo de fácil implementação e consagrado na literatura e no mercado.

Os dois modelos de PUFs (APUF e CRPUF) são comparados com relação à cobertura. Foi utilizado o método de Monte Carlo para a geração de cada PUF utilizado nas simulações. Este Capítulo apresenta inicialmente a plataforma e os recursos utilizados para a execução dos experimentos (seção 4.1). Em seguida, são apresentados os experimentos realizados e os seus resultados (seção 4.2). Por fim, são apresentadas uma avaliação teórica e analítica do CRPUF e do APUF para determinação do limite superior máximo de respostas e ainda a metodologia utilizada para calcular o número de transistores (hardware total) dos dois modelos (seção 4.3)

4.1 Plataforma Experimental

Nesta seção é descrita a plataforma para a simulação dos PUFs. São apresentados os detalhes sobre os recursos necessários para o funcionamento da plataforma e também os recursos utilizados para o desenvolvimento e simulação dos PUFs.

Foi utilizado a simulação de Monte Carlo para cada tamanho dos PUFs utilizado nas simulações. São gerados os atrasos para a constituição de cada PUF e os desafios a serem aplicados aos PUFs.

Nesta seção são apresentados os recursos necessários para a realização das simulações com os APUF e o CRPUF.

4.1.1 Geração dos Atrasos

Por se tratar da simulação de PUFs baseados em atrasos foi necessária a geração dos atrasos para simular o comportamento e funcionamento das portas lógicas e dos fios do circuito integrado. Os atrasos são gerados de forma que sejam semelhantes ao que ocorre com a variação durante o processo de fabricação do circuito integrado. Para isso, utilizou-se uma função que apresenta uma aproximação de uma gaussiana normalizada para determinação do conjunto de atrasos de cada indivíduo específico do modelo de PUF. Essa função foi utilizada porque ela possui um valor mediano como referência e uma variação. Dessa forma, o valor mediano funciona como o atraso determinado para a porta lógica e a variação é decorrente do processo de fabricação.

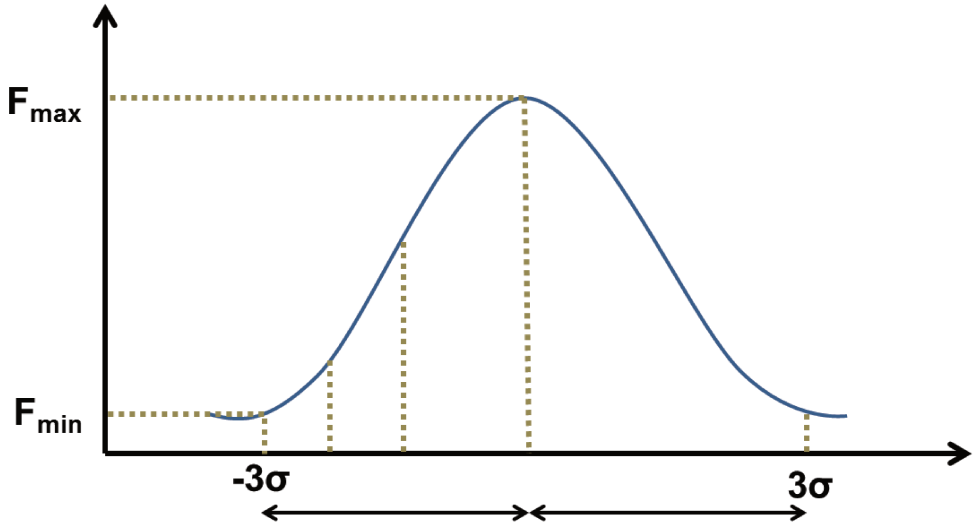


Figura 4.1: Distribuição Gaussiana dos Atrasos.

Como apresentado na Figura 4.1, cada atraso pertence ao intervalo $[-3\sigma, 3\sigma]$, sendo que a frequência destes valores extremos são iguais a F_{min}. Valores intermediário são gerados com maiores frequências chegando ao máximo em F_{max}. Para isso, foi utilizada uma função em JAVA (*RandomGaussian*) que garante essa distribuição.

A variação mínima de um atraso para outro na simulação é na escala de um femto-segundo, ou seja, $1 \cdot 10^{-15}$ segundo. Essa variação é a mínima que pode ser utilizada no VHDL do simulador GHDL.

Para o CRPUF a distribuição utilizada para cada célula foi de $1,5ns$ com variação até

$0,5ns$. Para o APUF a distribuição utiliza $1,5ns$ e $2ns$ com variação também de $0,5ns$ para os caminhos menores e maiores respectivamente.

Como o objetivo do trabalho é o de comparar o CRPUF proposto nesse trabalho com uma outra solução "consagrada", o APUF, a abordagem de assumir distribuições estatísticas simplificadas, mas em condições iguais para ambas abordagens, produzirá resultados consistentes.

Assim, para cada modelo de PUF é determinada uma configuração de todos os atrasos existentes através deste modelo de simulação. Desta forma, por exemplo, para 10000 indivíduos gerados de um determinado modelo de PUF, ou seja, 10000 combinações de atrasos, cada trecho do caminho é gerado de forma aleatória dentro da distribuição gaussiana.

Para a simulação do APUF são utilizadas duas gaussianas semelhantes à Figura 4.1 para determinação dos atrasos. As Figuras 4.2 e 4.3 destacam a representação de cada trecho do caminho.

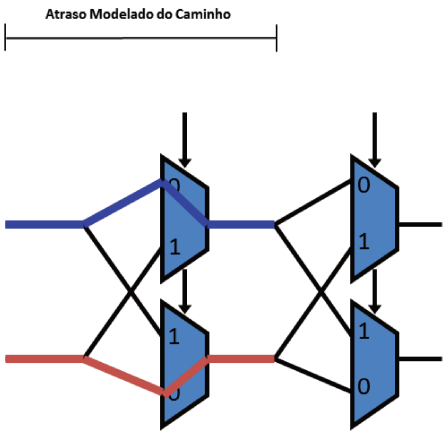


Figura 4.2: Destaque do Atraso dos Trechos Menores de um Estágio do APUF.

A Figura 4.2 destaca os trechos menores de um estágio do APUF. Cada um deles é representado na simulação por um atraso gerado por uma gaussiana. Esses trechos são percorridos simultaneamente pelo pulso quando o bit do desafio é igual a '0'.

Já a Figura 4.3 destaca os trechos maiores de um estágio do APUF. Cada um deles é representado na simulação por um atraso gerado por outra gaussiana. Esses trechos são percorridos simultaneamente pelo pulso quando o bit do desafio é igual a '1'.

Assim, os dois trechos destacados na Figura 4.3 são determinados por uma gaussiana com valor médio maior que o determinado na Figura 4.2.

Tanto os atrasos para o CRPUF quanto para o APUF possuem a mesma variação σ de $0,5$, como se fossem fabricados numa mesma tecnologia.

A Figura 4.4 destaca a representação do atraso em um trecho do caminho do CRPUF.

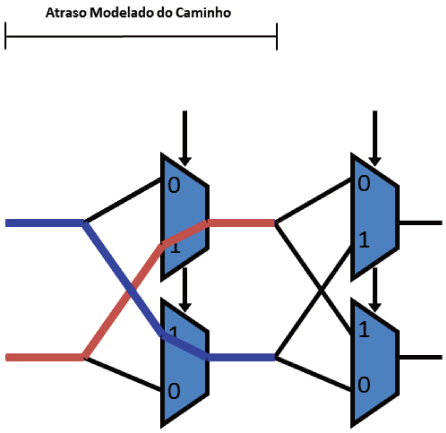


Figura 4.3: Destaque do Atraso dos Trechos Maiores de um Estágio do APUF.

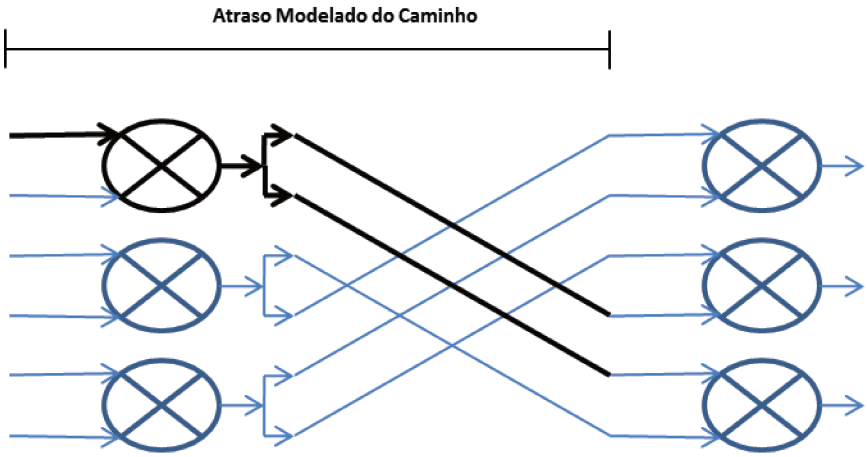


Figura 4.4: Destaque do Atraso do CRPUF

Assim, para o CRPUF, um atraso gerado representa cada caminho do CRPUF, como por exemplo, o trecho destacado na Figura 4.4. Desta forma, os atrasos representam o caminho de uma entrada de um estágio à saída dele que se propaga por dois outros caminhos para o estágio seguinte.

Os valores dos atrasos são simulados através do comando ***after*** em VHDL. Cada entrada das células funciona de acordo com o valor do atraso carregado como apresenta o trecho do código em VHDL a seguir:

$$\text{A} := \text{B } \textbf{xor } \text{C } \textbf{after } \text{delay[i]} \cdot 1 \text{ fs;}$$

Esse trecho de código simula o funcionamento de uma porta lógica XOR, onde B e C são as entradas e A é a saída. Assim, o valor inteiro da posição i do vetor de atrasos *delay* é multiplicado por $1fs$ (um femtosegundo), representando o atraso no PUF. Desta forma foi possível simular os atrasos das células dos PUFs.

Os árbitros (*flipflops*) dos PUFs simulados são considerados como ideais. Qualquer diferença de atraso entre os sinais que são as entradas do *flipflops* é detectável pelo *flipflop*. Desta forma, o CRPUF e o APUF são colocados nas mesmas condições. O resultado da resposta com todos os sinais já estabilizados é capturado de forma síncrona.

4.1.2 Geração dos Desafios

Os desafios são obtidos através da geração de números aleatórios. Os números são gerados para o tamanho do desafio utilizado pelo PUF. Para a simulação, os desafios são gerados de forma que eles não sejam repetidos para evitar que um único desafio seja utilizado mais de uma vez.

Os PUFs recebem a mesma sequência de desafios aplicados, ou seja, todos os PUFs do mesmo modelo são estimulados pelo mesmo conjunto de desafios. A sequência dos desafios aplicados é determinada aleatoriamente na própria geração dos desafios.

4.1.3 Cálculo da Cobertura

Para calcular a cobertura é utilizado um procedimento que identifica quais as respostas são distintas. As respostas são agrupadas e comparadas com todas as outras com o mesmo agrupamento das respostas. Por exemplo, como apresenta a Figura 4.5(a), suponha a existência de 6 indivíduos do $CRPUF_1(3x3)$ ($P1$, $P2$, $P3$, $P4$, $P5$, e $P6$) que geram respostas de 3 bits quando aplicado um desafio. Nestes 6 PUFs são aplicados 5 desafios ($D1$, $D2$, $D3$, $D4$, $D5$). O cálculo da cobertura C é apresentado na sequência da Figura 4.5.

Como apresenta a Figura 4.5(b), na aplicação do primeiro desafio ($D1$), todas as respostas do conjunto das respostas destacadas no quadrado são comparadas entre si,

	D1	D2	D3	D4	D5
P1	101	011	001	110	001
P2	000	101	100	000	101
P3	000	101	111	000	001
P4	011	001	011	101	011
P5	101	001	101	111	010
P6	000	101	111	000	011
C					

(a) Respostas.

	D1	D2	D3	D4	D5
P1	101	011	001	110	001
P2	000	101	100	000	101
P3	000	101	111	000	001
P4	011	001	011	101	011
P5	101	001	101	111	010
P6	000	101	111	000	011
C	1				

(b) Cobertura no Primeiro Desafio.

	D1	D2	D3	D4	D5
P1	101	011	001	110	001
P2	000	101	100	000	101
P3	000	101	111	000	001
P4	011	001	011	101	011
P5	101	001	101	111	010
P6	000	101	111	000	011
C	1	3			

(c) Cobertura no Segundo desafio.

	D1	D2	D3	D4	D5
P1	101	011	001	110	001
P2	000	101	100	000	101
P3	000	101	111	000	001
P4	011	001	011	101	011
P5	101	001	101	111	010
P6	000	101	111	000	011
C	1	3	4		

(d) Cobertura no Terceiro Desafio.

	D1	D2	D3	D4	D5
P1	101	011	001	110	001
P2	000	101	100	000	101
P3	000	101	111	000	001
P4	011	001	011	101	011
P5	101	001	101	111	010
P6	000	101	111	000	011
C	1	3	4	4	

(e) Cobertura no Quarto desafio.

	D1	D2	D3	D4	D5
P1	101	011	001	110	001
P2	000	101	100	000	101
P3	000	101	111	000	001
P4	011	001	011	101	011
P5	101	001	101	111	010
P6	000	101	111	000	011
C	1	3	4	4	6

(f) Cobertura no Quinto Desafio.

Figura 4.5: Como Determinar a Cobertura para 6 PUFs com 3 bits de resposta.

onde apenas a resposta do PUF $P4$ é distinta. Desta forma o PUF $P4$ já é identificado na aplicação do primeiro desafio. A cobertura C, portanto, é igual a 1, pois existe apenas uma resposta "011". Da mesma forma, na aplicação do segundo desafio $D2$ (Figura 4.5(b)), a cobertura (C) é igual a 3, pois os PUFs $P1$, $P4$ e $P5$ possuem respostas distintas, iguais a "101011", "011001" e "101001" respectivamente. Na Figura 4.5(d) os PUFs $P1$, $P2$, $P4$ e $P5$ são identificados ($C = 4$). Já na aplicação do desafio $D4$ (Figura 4.5(e)) nenhum novo PUF é identificado ($C = 4$). E por fim, quando aplicado o desafio $D5$ (Figura 4.5(f)) todos os PUFs são identificados, resultando numa cobertura de 100% ($C = 6$).

4.1.4 Ferramentas

Os estudos e experimentos apresentados neste trabalho foram realizados com o auxílio do simulador GHDL. Este é um simulador de VHDL, que usa tecnologia GCC. Ele implementa a linguagem VHDL de acordo com os padrões *IEEE 1076-1987* ou *IEEE 1076-1993*. Assim, ele compila os arquivos VHDL e cria um binário que simula (ou executa) o projeto.

Além disso, foi utilizado código em JAVA para a geração dos atrasos dos indivíduos e

para a geração dos desafios em arquivo de texto.

4.1.5 Infraestrutura Utilizada

Durante este trabalho utilizou-se a infraestrutura do Laboratório de Sistemas da Computação - LSC do Instituto de Computação da UNICAMP. Foi utilizada uma máquina pessoal com processador Intel i5 e 4GB de RAM para a geração *offline* dos arquivos que alimentam a simulação do circuito em VHDL e também para a análise dos dados dos resultados das simulações.

Na infraestrutura do LSC, o tempo de simulação médio para um conjunto de 10000 PUFs foi de aproximadamente 12 minutos. Entretanto, foram realizadas simulações de centenas de conjuntos de 10000 PUFs.

A máquina utilizada para a simulação dos PUFs possui processador Intel Xeon E5645 2.40 GHz com 2 CPUs e 6 núcleos cada e ainda 32 GB de RAM, 1740 GB (RAID-0) e Ubuntu Server 10.04.3 LTS. Assim, foram simuladas diversas configurações tanto para o CRPUF quanto para o APUF.

4.1.6 Descrição dos Arquivos Utilizados

Para a simulação foram utilizados quatro arquivos para cada configuração dos PUFs:

1. Arquivo com vários indivíduos representados por um conjunto de atrasos, onde cada linha do arquivo possui um conjunto de atrasos que representa um indivíduo;
2. Arquivo contendo os desafios a serem aplicados;
3. Arquivos VHDL com a descrição do modelo de PUF simulado;
4. Arquivo de saída com as respostas dos PUFs aos desafios aplicados.

Os arquivos dos PUFs são descritos em VHDL estrutural, ou seja, instanciando todos os componentes e conectando-os.

Dependendo da quantidade de indivíduos e também do tamanho do PUF, os arquivos de atrasos, apesar de ser no formato texto, em alguns casos, chega a ter 300 MB cada. Por isso, este fator limitou a quantidade de indivíduos das simulações e o tamanho de cada um deles.

4.2 Experimentos

Esta seção descreve e apresenta os resultados dos experimentos utilizados para avaliar o CRPUF e o APUF sob vários aspectos abordados.

4.2.1 Descrição

Com o objetivo de facilitar a descrição dos experimentos, considera-se a Equação 4.1 que descreve os parâmetros dos quais depende a cobertura para identificação de um PUF. Estes parâmetros experimentais são descritos a seguir, onde a cobertura representa a relação entre o número de indivíduos de um determinado modelo K de PUF identificados com relação a um número P de indivíduos daquele modelo de PUF.

$$C = f(K, L, W, H, P) \tag{4.1}$$

Onde:

C - cobertura de PUFs identificados;

K - modelo de PUF utilizado;

L - número de desafios aplicados;

W - tamanho do desafio em bits;

H - quantidade aproximada do hardware total utilizado em número de transistores;

P - número de indivíduos estimulados do modelo de PUF.

Assim, dado um modelo de PUF K, que tenha um desafio de tamanho W e consequentemente uma quantidade de hardware H, a cobertura C daquele modelo de PUF é dada pela porcentagem de PUFs identificados dentro de um conjunto P destes após a aplicação de L desafios.

Desta forma, a cobertura do PUF será tanto melhor quanto menor for o número de desafios necessários à detecção do conjunto de PUFs. Por exemplo, considerando K_1 e K_2 como modelos hipotéticos e diferentes de PUFs, se diante de um conjunto de 20 PUFs do modelo K_1 forem identificados 16 deles com 5 desafios e também para 20 PUFs do modelo K_2 forem identificados 16 deles com 4 desafios, a cobertura do modelo de PUF K_2 será melhor, pois com menos desafios alcançou-se a mesma cobertura que o modelo K_1. Além disso, caso os outros 4 PUFs restantes sejam todos com conjunto de desafios e respostas iguais ou dois pares com conjunto de desafios e respostas iguais, a cobertura total do modelo será de 80% ($C = \frac{16}{20}$), pois os PUF seriam considerados iguais e não haveria como identificá-los com a aplicação de novos desafios. Outra representação do conceito de cobertura é apresentada pela equação:

$$C = 1 - t_A \tag{4.2}$$

Onde t_A significa a taxa de *aliasing*. Ela representa o número de sinônimos, ou seja, o número de PUFs com conjunto de desafios e respostas iguais dentro de um conjunto de PUFs. Desta forma, a soma da cobertura (C) com a taxa de *aliasing* (t_A) é igual a 100%. Para o exemplo citado no parágrafo anterior, a taxa de *aliasing* (t_A) seria de 20% ($\frac{4}{20}$), ou seja, 4 PUFs de um total de 20 do modelo K_1 hipotético utilizado.

Quando um indivíduo é identificado significa que ele foi identificado dentro de um conjunto de PUFs do mesmo modelo e tamanho, ou seja, são PUFs diferentes apenas devido à variação no processo de fabricação, pois o projeto é o mesmo.

Os métodos aplicados e a sequência da realização dos experimentos são os seguintes:

1. Caracterização do APUF (seção 4.2.2);
2. Caracterização do CRPUF (seção 4.2.3);
3. Comparação APUF x CRPUF (seção 4.2.4).

Assim são realizadas duas comparações nos experimentos. São utilizadas duas abordagens em cada comparação que visam identificar a cobertura (C) do CRPUF e do APUF em relação à quantidade de hardware (H) e aos desafios aplicados (L). Entretanto, antes de apresentar as comparações, o APUF e o CRPUF são caracterizados separadamente e em seguida são realizadas as comparações dos resultados.

Desta forma, foram realizadas as seguintes comparações:

- **Experimento com APUF como Referencial** (seção 4.2.4.1)
  - *Medida C**x**L para* ***Hardware Igual*** (seção 4.2.4.1.1): para um mesmo número (P) de PUFs, busca identificar, a partir da quantidade de hardware do APUF (H_{APUF}), CRPUFs com hardware equivalente (H_{CRPUF}) e em seguida comparar as coberturas.
  - *Medida C**x**L para* ***Cobertura Igual*** (seção 4.2.4.1.2): para um mesmo número (P) de PUFs, busca identificar, a partir da cobertura obtida pelo APUF (C_{APUF}), o CRPUF com cobertura equivalente (C_{CRPUF}) e em seguida comparar os hardwares.
- **Experimento com CRPUF como Referencial** (seção 4.2.4.2)
  - *Medida C**x**L para* ***Hardware Igual*** (seção 4.2.4.2.1): para um mesmo número (P) de PUFs, busca identificar, a partir da quantidade de hardware do CRPUF (H_{CRPUF}), o APUF com hardware equivalente (H_{APUF}) e em seguida comparar as coberturas.

– *Medida C**x**L para **Cobertura Igual*** (seção 4.2.4.2.2): para um mesmo número (P) de PUFs, busca identificar, a partir da cobertura obtida pelo CRPUF (C_{CRPUF}), o APUF com cobertura equivalente (C_{APUF}) e em seguida comparar os hardwares.

Para a realização dos experimentos acima descreve-se como calcular a quantidade de hardware em número de transistores para o APUF e CRPUF na subseção 4.3.1. Além disso, na subseção 4.3.2 é apresentada uma análise de limite superior do número de respostas do APUF e CRPUF, visando identificar o potencial máximo em função do tamanho do PUF.

O mesmo número (P) de PUFs é utilizado nos experimentos para que os resultados possam ser obtidos em condições semelhantes. Diferentes quantidades de indivíduos para cada modelo de PUF podem distorcer o resultado da análise devido a variações estatísticas.

Tanto nos resultados experimentais do APUF quanto do CRPUF são utilizados 10000 indivíduos para determinação da cobertura. E a partir destes dados é possível ter um perfil do comportamento dos dois modelos.

4.2.2 Caracterização do APUF

Nesta subseção, investiga-se o comportamento da cobertura do APUF com relação à variação do tamanho do desafio e do número de APUFs em paralelo (Figura 4.6).

Quando descrito que o APUF tem apenas uma dimensão, é porque possui apenas um APUF de M bits ($APUF(Mx1)$) e, portanto, um bit na resposta. No entanto, quando apresentado que possui várias dimensões (cópias em paralelo), N APUFs de M bits são agrupados ($APUF(MxN)$) e submetidos ao mesmo desafio de acordo com a organização apresentada na Figura 4.6.

Para analisar o comportamento do APUF e verificar o funcionamento do APUF, foram simuladas as configurações do APUF com $M \in \{12, 16, 32, 64\}$ bits e $N \in \{1, 2, 3, 4, 6, 8\}$. Onde M representa o tamanho do desafio em bits, já o N representa o número de cópias em paralelo do APUF. Em [33], utilizou-se apenas o APUF 64x1 nos experimentos. Entretanto, as demais configurações foram utilizadas para que o comportamento do APUF fosse identificado em outros tamanhos.

Para a caracterização do APUF foram realizados os seguintes experimentos:

1. cada valor de N com todos os 4 valores de M;
2. cada valor de M com todos os 6 valores de N (apenas uma organização diferente dos resultados do experimento acima);
3. tabela identificando o número de desafios necessários para o APUF obter cobertura de 95%.

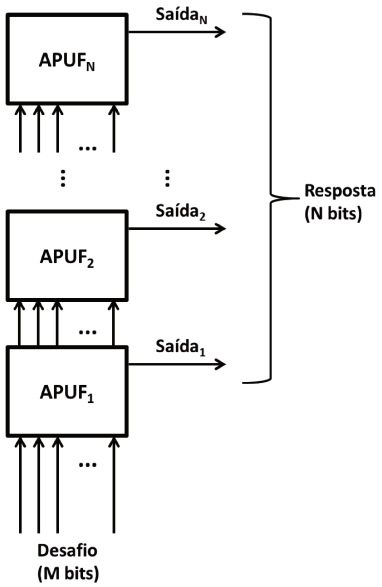


Figura 4.6: Organização do APUF(MxN).

Por existirem várias combinações de simulações do APUF e também para reduzir o número de gráficos com comportamentos semelhantes, nesta subseção são apresentados apenas resultados de algumas combinações de tamanho do APUF.

No entanto, os resultados de todas as combinações são apresentados no Apêndice A. Os resultados apresentados nesta subseção tem objetivo de descrever de forma geral todas demais combinações análogas apresentadas no Apêndice A sobre o APUF.

O resultado da cobertura para o APUF com $N = 1$ é apresentado na Figura 4.7.

Como se observa na Figura 4.7, as coberturas dos APUFs ($APUF(12x1)$, $APUF(16x1)$, $APUF(32x1)$ e $APUF(64x1)$) apresentam comportamentos semelhantes. Também apresenta que o APUF com 64 bits de desafio possui melhor cobertura que o APUF com 12 bits de desafio.

Entretanto, como a saída dos APUFs da Figura 4.7 para a aplicação de um desafio possui sempre 1 bit ($APUF(Mx1)$), observa-se que desafios maiores não melhoram significativamente a cobertura do APUF, pois a cobertura máxima é limitada pelo número de bits da reposta. Assim, por exemplo, com 3 desafios aplicados ao APUF com $N = 1$ não é possível distinguir 9 indivíduos, mesmo que seja um $APUF(1024x1)$. A capacidade máxima da cobertura seria de 2^3 possibilidades.

O resultado da cobertura para o APUF com $N = 8$ APUFs é apresentado na Figura 4.8.

Como se observa na Figura 4.8, a cobertura para a combinação de 8 APUFs ($N = 8$) apresenta comportamentos semelhantes (sobrepostos) para todos os valores de M.

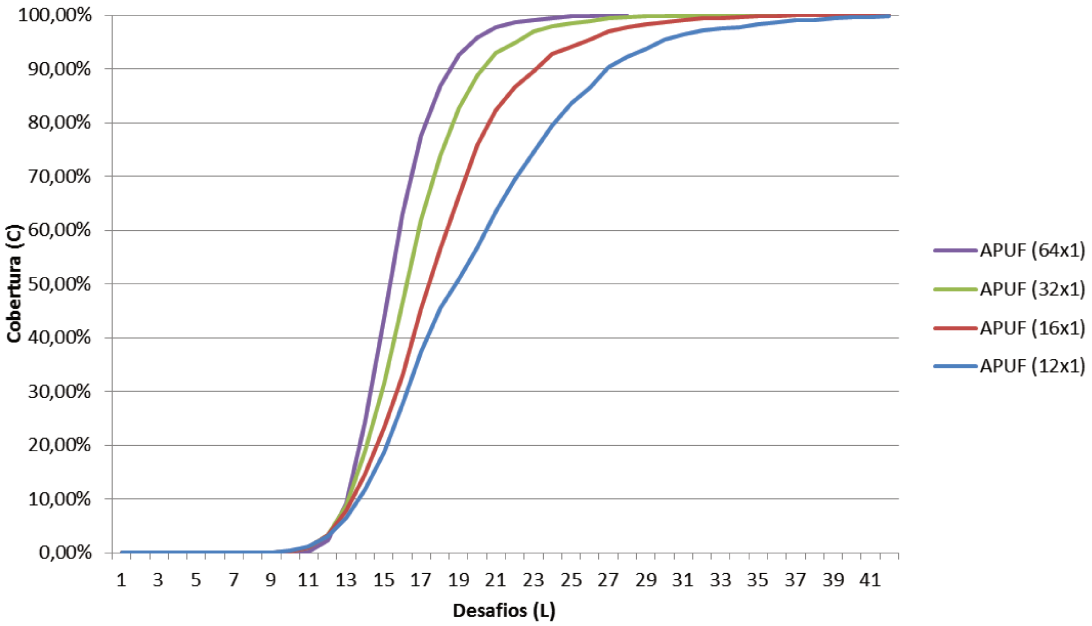


Figura 4.7: Cobertura para o APUF(Mx1).

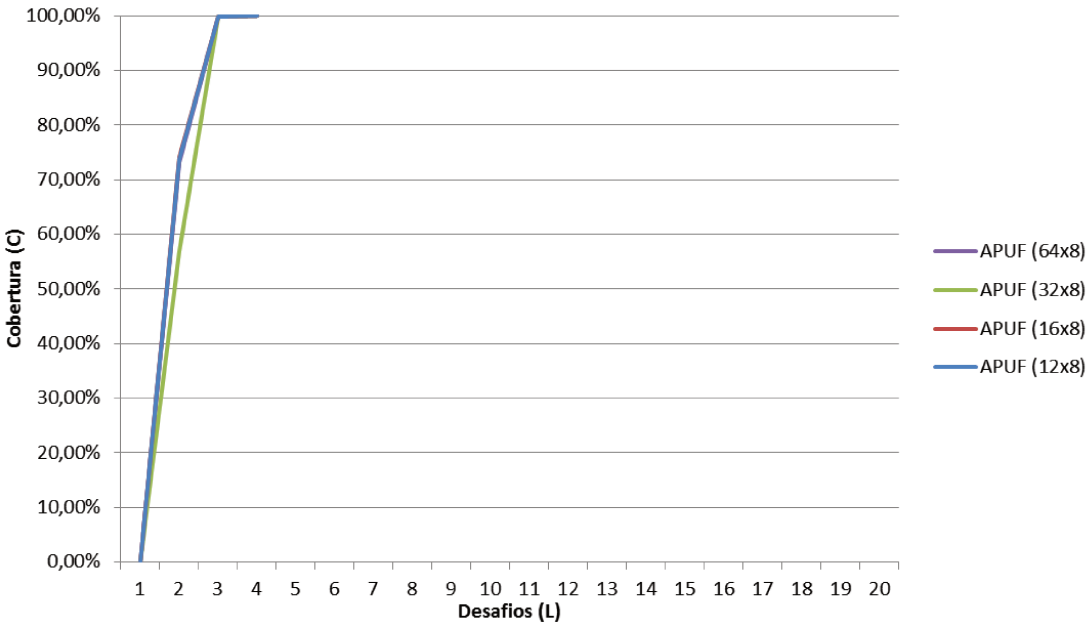


Figura 4.8: Cobertura para o APUF(Mx8).

Visando sintetizar os resultados da cobertura para o APUF de forma simplificada de todas as combinações de APUFs simuladas (apresentadas no Apêndice A), a Tabela 4.1 apresenta o número de desafios necessários para que o APUF alcance uma cobertura de

95% . Esta cobertura de 95% é utilizada como valor referencial de cobertura para ambos os modelos.

Tabela 4.1: Número de Desafios para o APUF Alcançar a Cobertura de 95%.

Dimensão (N)	**Estágios** M			
	12	**16**	**32**	**64**
1	30	26	23	20
2	12	11	11	10
3	7	7	7	7
4	5	5	6	5
6	4	4	4	4
8	3	3	3	3

Na Tabela 4.1 observa-se que para N maior ou igual a 2, a cobertura de 95% é alcançada por um número de desafios aplicados semelhantes para qualquer valor de M (número de estágios do APUF).

Os resultados mostram que a cobertura do APUF não melhora significativamente com o aumento de M. O número de desafios aplicados para alcançar uma cobertura específica é sempre semelhante para o mesmo valor de M. Entretanto, o aumento do número de cópias (N) melhora significativamente a cobertura do APUF. Observa-se ainda que a ordem dos desafios aplicados ao APUF não afeta o resultado geral da cobertura, pois não há dependência de um desafio com o anterior.

4.2.3 Caracterização do CRPUF

Nesta subseção, investiga-se o comportamento da cobertura do $CRPUF_1$ e do $CRPUF_2$ com relação ao seu tamanho (número de linhas e de estágios ou colunas).

Busca-se um valor referencial em que a relação cobertura/hardware seja satisfatória. Para isso foram simuladas as configurações do CRPUF(MxN) para $M \in \{3, 5, 7, 9\}$ e $N \in \{2, 3, 4, 5, 6, 7, 8, 9\}$, onde o M representa o número de linhas e o N representa o número de colunas.

Para o CRPUF foram realizadas as seguintes simulações:

- cada valor de M com todos os oito valores de N;
- cada valor de N com todos os quatro valores de M (apenas uma organização diferente dos resultados do experimento acima);
- tabela identificando o número de desafios necessários para obter 95% de cobertura.

A partir destes dados é possível determinar uma configuração do tamanho do CRPUF com melhor cobertura e menor hardware.

Por também existirem várias combinações de simulações do CRPUF e também para reduzir o número de gráficos com comportamentos semelhantes nessa subseção, são apresentados apenas resultados de algumas combinações de tamanho do CRPUF.

No entanto, os resultados de todas as combinações são apresentados no Apêndice B. Os resultados apresentados nesta subseção tem objetivo de descrever de forma geral todas demais combinações análogas apresentadas no Apêndice B para o CRPUF.

Inicialmente são apresentados os resultados das simulações para $M = 3$ linhas e cada um dos valores de $N \in \{2, 3, 4, 5, 6, 7, 8, 9\}$ colunas. Os dois resultados que mostram a cobertura para o CRPUF_1 e CRPUF_2 com $M = 3$ linhas à medida que os desafios são aplicados são apresentados no gráfico da Figura 4.9 (resultados para o $CRPUF_1$) e no gráfico da Figura 4.10 (resultados para o $CRPUF_2$).

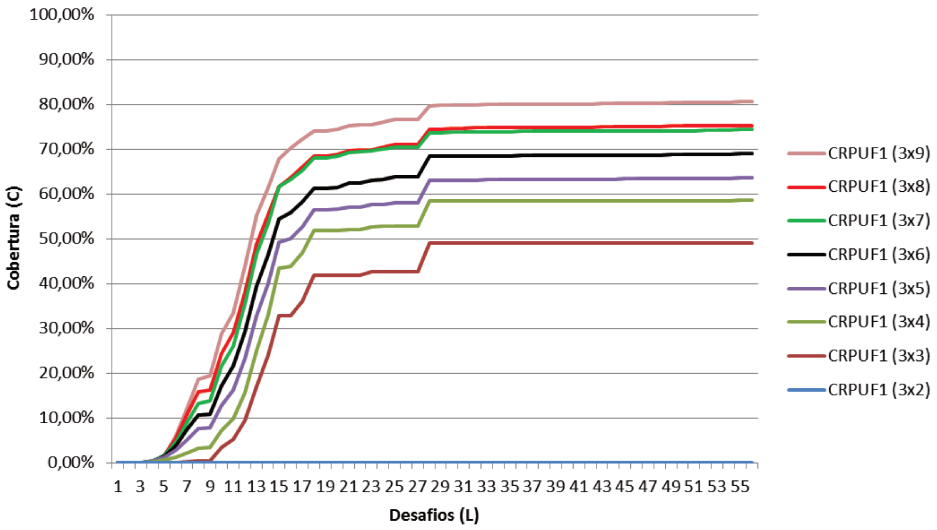


Figura 4.9: Cobertura para o $CRPUF_1$ com 3 linhas.

O gráfico da Figura 4.9 apresenta que o $CRPUF_1$ estabiliza a sua cobertura por volta do desafio 28. A sequência de comportamentos de crescimento semelhantes acontece nas curvas porque são aplicados os mesmos desafios na entrada que é tamanho fixo. O mesmo comportamento acontece no CRPUF_2. O gráfico da Figura 4.10 apresenta que o $CRPUF_2$ estabiliza um pouco antes (desafio 14) que o CRPUF_1 por representar melhor as diferenças das corridas nos PUFs.

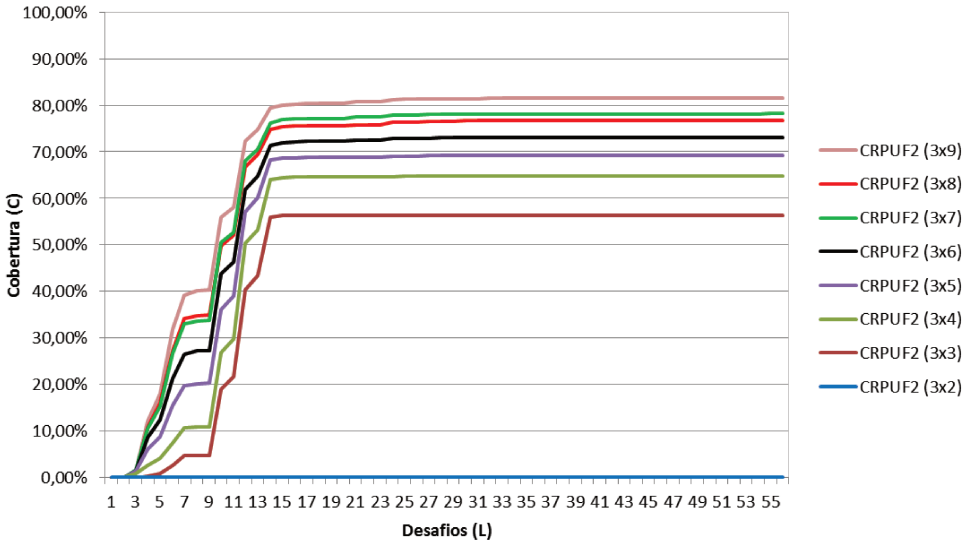


Figura 4.10: Cobertura para o $CRPUF_2$ com 3 linhas.

Os gráficos das Figuras 4.9 e 4.10 apresentam que nenhum dos CRPUFs com 3 linhas conseguem identificar todos os 10000 indivíduos. Esse resultado experimental confirma a previsão teórica do limite superior da cobertura apresentada na seção 4.3.2.1 (ver Tabela 4.10, pág. 80).

Na sequência são apresentados os resultados da cobertura para o CRPUF com $M = 5$ linhas. No gráfico da Figura 4.11 são apresentados os resultados para o $CRPUF_1(5xN)$.

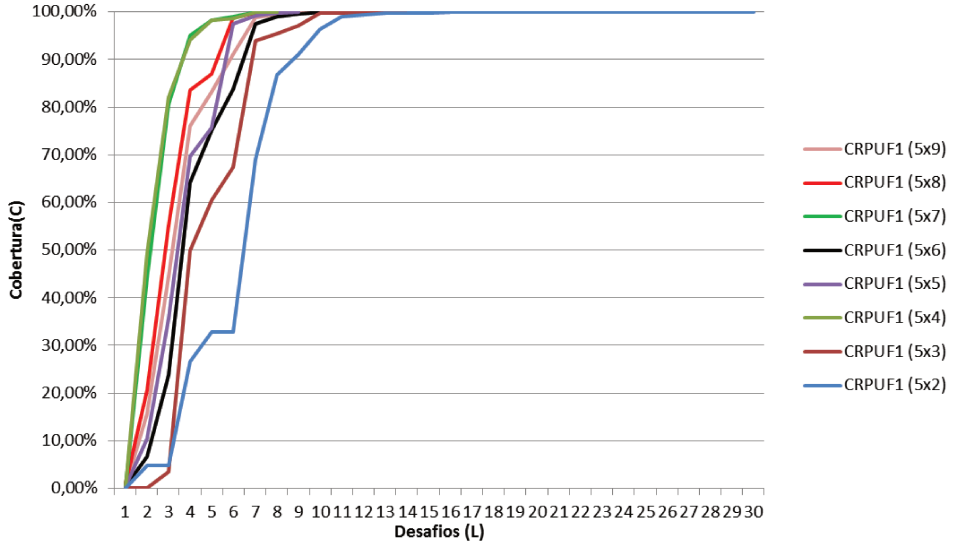


Figura 4.11: Cobertura para o $CRPUF_1$ com 5 linhas.

Como se observa na Figura 4.11, a cobertura para o $CRPUF_1(5xN)$ $(M = 5)$ apresenta comportamento semelhante para todos os valores de N. No entanto, o $CRPUF_1(5x2)$ é o único que não alcança a cobertura de 100% dos indivíduos no décimo sétimo desafio aplicado.

No gráfico da Figura 4.12 são apresentados os resultados para o $CRPUF_2(5xN)$.

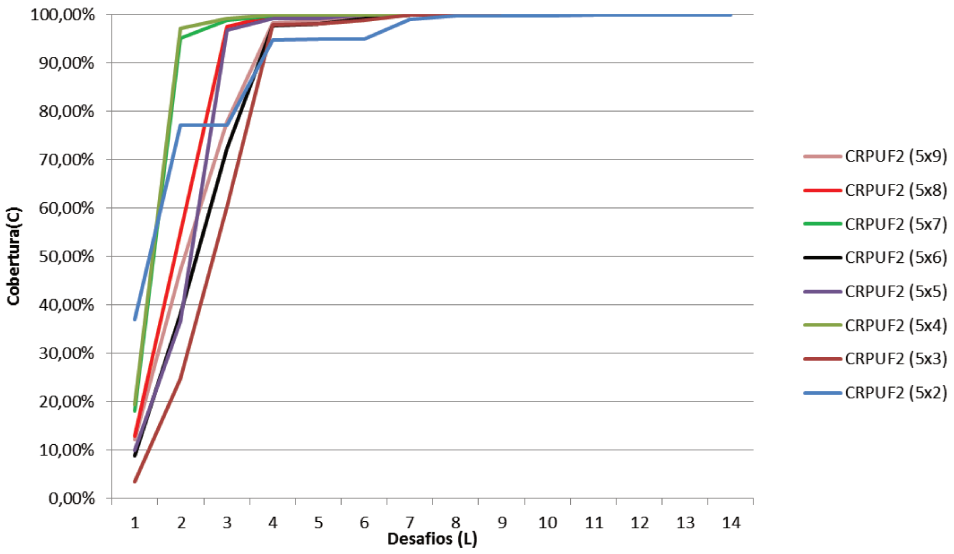


Figura 4.12: Cobertura para o $CRPUF_2$ com 5 linhas.

Na Figura 4.11, a cobertura para o $CRPUF_2(5xN)$ $(M = 5)$ apresenta comportamento ainda mais semelhante para todos os valores de N do que para o $CRPUF_1(5xN)$. Isso porque o $CRPUF_2$ possui mais bits de saída, neste caso para M igual a 5 são 20 bits de saída por desafio aplicado.

Também foram realizados experimentos fixando a quantidade de colunas (N) em combinação com todos os valores de M. Esses gráficos são utilizados para verificar o desempenho com relação ao tamanho do CRPUF com relação à quantidade de linhas (M) para uma mesma quantidade de colunas (N).

Os resultados apresentados para o $CRPUF_1$ com $N = 2$ estágios são apresentados na Figura 4.13 (resultados para o $CRPUF_1(Mx2)$).

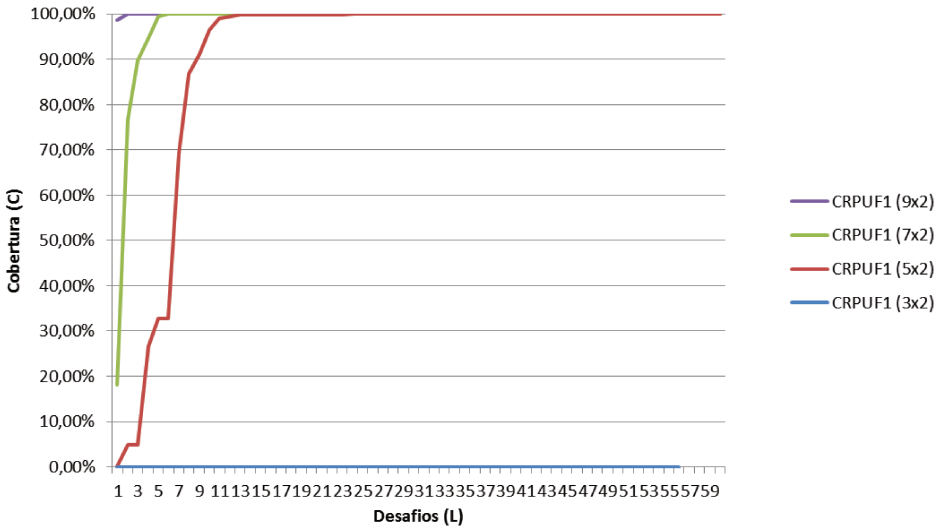


Figura 4.13: Cobertura Para o $CRPUF_1$ com 2 Estágios.

Na Figura 4.14 são apresentados os resultados para o $CRPUF_2(Mx2)$.

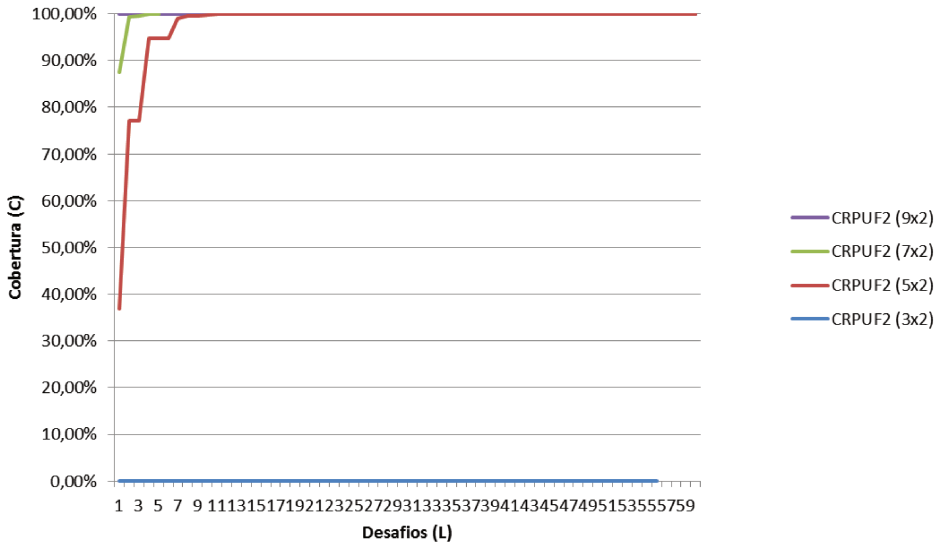


Figura 4.14: Cobertura Para o $CRPUF_2$ com 2 Estágios.

Os resultados dos gráficos da Figura 4.13 e da Figura 4.14 mostram que a cobertura melhora à medida que aumenta o número de linhas do CRPUF. O CRPUF com maior número de linhas possui maior cobertura para um mesmo número de desafios. Um dos motivos é que o maior número de linhas aumenta o número de caminhos e também o número de bits da resposta.

Os demais resultados para os outros tamanhos de N do CRPUF são apresentados no Apêndice B. Estes resultados apresentam resultados semelhantes aos apresentados na Figura 4.13 e na Figura 4.14.

Os resultados apresentados também permitem avaliar o número mínimo de desafios para que seja atingida uma determinada cobertura. Os dados da Tabela 4.2 apresentam a quantidade de desafios necessários para cada dimensão de modo que o modelo CRPUF alcance cobertura de 95%.

Tabela 4.2: Número de Desafios para o CRPUF Alcançar a Cobertura de 95%.

Linhas	**Tipo**	**Estágios**							
		2	**3**	**4**	**5**	**6**	**7**	**8**	**9**
3	$CRPUF_1$	∞ [1]	∞	∞	∞	∞	∞	∞	∞
	$CRPUF_2$	∞	∞	∞	∞	∞	∞	∞	∞
5	$CRPUF_1$	10	8	5	6	7	5	6	7
	$CRPUF_2$	7	4	2	3	4	3	3	4
7	$CRPUF_1$	5	2	3	2	3	3	3	3
	$CRPUF_2$	2	1	2	1	2	2	2	2
9	$CRPUF_1$	1	1	1	2	1	2	1	2
	$CRPUF_2$	1	1	1	1	1	1	1	1

Os dados da Tabela 4.2 indicam que para alcançar 95% de cobertura são necessários a aplicação de 10 desafios para o $CRPUF_1(5x2)$ e 7 desafios para o $CRPUF_2(5x2)$.

A representação ∞ indica que não foi possível que o CRPUF alcançasse a cobertura de 95% para o número de desafios aplicados e o número de PUFs a serem identificados (10000). Observa-se que este resultado experimental confirma a previsão teórica do limite superior de respostas. A cobertura de 95% foi utilizada por representar um valor satisfatório próximo de uma cobertura total dos indivíduos. Este valor foi definido e determinado neste trabalho.

O objetivo dos experimentos com o CRPUF apresentados nesta seção é o de analisar o perfil e escolher um bom candidato para ser comparado com o APUF.

Com base nos resultados da cobertura do CRPUF apresentados nesta seção e no Apêndice B, foram analisados as várias configurações do CRPUF e assim foi escolhido o CRPUF com dimensões $M = 5$ e $N = 3$ para ser comparado com o APUF, pois com apenas 8 desafios o $CRPUF_1(5x3)$ e $CRPUF_2(5x3)$ já alcançam o mínimo de 95% de cobertura.

4.2.4 Comparações APUF x CRPUF

Após apresentar os resultados da caracterização da cobertura do APUF (seção 4.2.2) e do CRPUF (subseção 4.2.3), esta subseção apresenta resultados das comparações entre o APUF e o CRPUF.

A seguir são apresentados os dois experimentos que comparam estes dois PUFs.

4.2.4.1 Experimento com APUF como Referencial

4.2.4.1.1 Medida CxL para Hardware Igual - o objetivo deste experimento é, dado um APUF, encontrar CRPUFs com **hardware igual**/próximo e a partir daí identificar qual o PUF que alcança melhor cobertura com menor número de desafios aplicados.

Neste experimento é utilizado como PUF referencial o APUF de 64 bits, pois em [33] o autor do APUF recomenda APUFs de tamanho mínimo de 64 bits nos experimentos para diminuir a ocorrência de *aliasing*. Assim, a partir do APUF de 64 bits são encontrados CRPUFs com **hardware semelhante** para que sejam comparadas as coberturas.

O hardware do APUF de 64 bits - APUF(64x1) - em número de transistores foi calculado pela Equação 4.7 (pág. 76). Ele possui 1418 transistores (ver Tabela 4.8, pág. 77).

Em seguida foi calculado o número de transistores de $CRPUFs(MxN)$ que tenham o hardware próximo do total de 1418 transistores. Para isso foram utilizados CRPUFs com menores tamanhos de M (5, 7 e 9). Não foi utilizado M igual a 3 porque pela sua própria característica física do limite superior de respostas, que é descrito na subseção 4.3.2, não foi possível obter nos experimentos cobertura para 10000 indivíduos.

As dimensões encontradas para o CRPUF com hardware equivalente ao APUF de 64 bits são apresentadas na Tabela 4.3.

Tabela 4.3: Dimensões do CRPUF para Hardware do APUF(64x1).

M	**CRPUF**	**N**	**PUF**	**Hardware Total em Transistores**
5	$CRPUF_1$	20	$CRPUF_1(5x20)$	1300
	$CRPUF_2$	20	$CRPUF_2(5x20)$	1400
7	$CRPUF_1$	12	$CRPUF_1(7x12)$	1218
	$CRPUF_2$	12	$CRPUF_2(7x12)$	1428
9	$CRPUF_1$	9	$CRPUF_1(9x9)$	1332
	$CRPUF_2$	6	$CRPUF_2(9x6)$	1318

Como apresentado na Tabela 4.3, são utilizados nas simulações os seguintes CRPUFs: $CRPUF_1(5x20)$, $CRPUF_1(7x12)$, $CRPUF_1(9x9)$, $CRPUF_2(5x20)$, $CRPUF_2(7x12)$ e $CRPUF_2(9x6)$.

Desta forma, os resultados das simulações para o APUF(64x1) e os CRPUFs da Tabela 4.3 são apresentados na Figura 4.15 (APUF(64x1) x $CRPUF_1$) e na Figura 4.16 (APUF(64x1) x $CRPUF_2$).

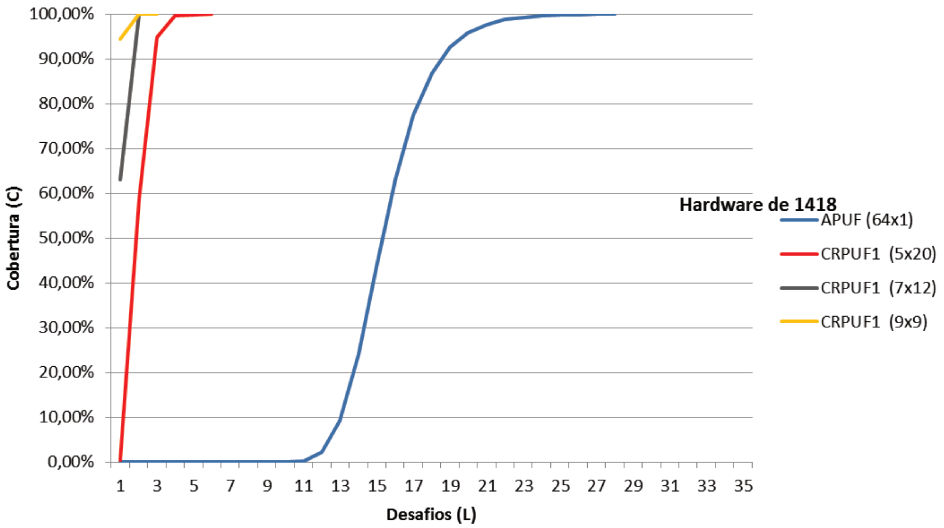


Figura 4.15: APUF(64x1) x $CRPUF_1$ com Hardware Semelhante.

Observa-se na Figura 4.15 que todos os PUFs são identificados já no sétimo desafio aplicado para o $CRPUF_1(5x20)$. Já o APUF(64x1) somente começa a identificar os PUFs a partir do nono desafio aplicado.

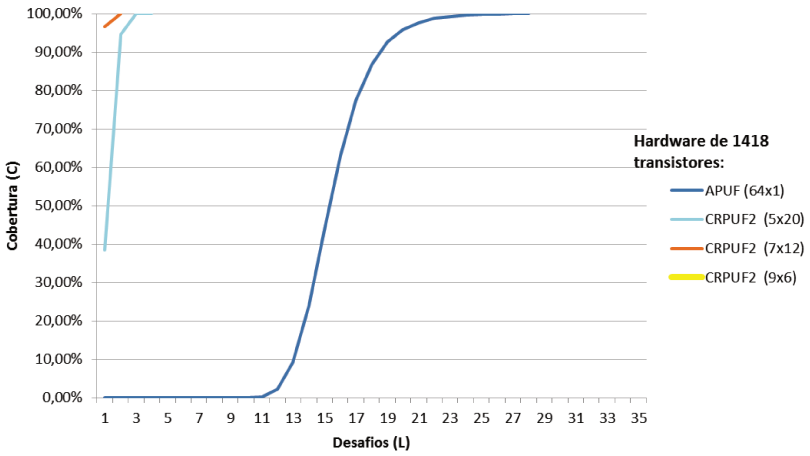


Figura 4.16: APUF(64x1) x $CRPUF_2$ com Hardware Semelhante.

A Figura 4.16 mostra que todos os indivíduos para o $CRPUF_2(9x6)$ são identificados

no primeiro desafio, ou seja, representa apenas um ponto no gráfico no primeiro desafio aplicado.

Como se observa nas Figuras 4.15 e 4.16 a cobertura do APUF(64x1) é menor que a cobertura dos menores $CRPUFs_1$ e a do $CRPUFs_2$ para uma mesma quantidade de hardware em transistores. De forma geral, é possível observar que o CRPUF com pior cobertura, o $CRPUF_1(5x20)$, possui uma cobertura ainda muito superior que a cobertura do APUF de 64 bits.

Até mesmo o $CRPUF_1(5x2)$ que é apresentado na Figura B.3 (Apêndice B) possui cobertura superior ao APUF(64x1). Desta forma não há como igualar a cobertura do APUF de 64 bits com a do CRPUF, ainda que este tenha hardware menor.

4.2.4.1.2 Medida CxL para Cobertura Igual - como forma de fazer o APUF(64x1) alcançar uma **mesma cobertura** que o CRPUF com a mesma quantidade de hardware, o hardware total do APUF(64x1) é dividido em APUFs(MxN) de mesmo tamanho (M) para que alcancem a cobertura do CRPUF através de múltiplos (N) APUFs em paralelo.

Desta forma, considerando pela Equação 4.7 que o hardware do APUF de 64 bits é igual a 1418 transistores, foram simulados vários APUFs em grupos de mesmo tamanho (M) que o total seja igual ou próximo de 1418 transistores.

A Tabela 4.4 apresenta as combinações das dimensões M e N dos APUFs encontradas com o número de transistores próximo a 1418.

Tabela 4.4: Conjunto de APUFs(MxN) com Hardware Total Igual a 1418 Transistores.

Estágios(M)	**Hardware**	**Dimensão**(N)	**Hardware Total em Transistores**
64	1418	1	1418
31	670	2	1400
21	472	3	1416
15	340	4	1260
12	274	5	1350
10	230	6	1380
6	142	10	1420

Assim, como apresenta a Tabela 4.4, foram encontradas 6 configurações de APUFs com hardware equivalente ao APUF(64x1) (1418 transistores). Por exemplo, como também se observa na Tabela 4.4, um APUF(21x3) utiliza 1416 transistores, pois cada APUF(21x1) possui 472 transistores.

Definidas as configurações dos APUFs (APUF(31x2), APUF(21x2), APUF(15x3), APUF(12x5), APUF(10x6) e APUF(6x10)) com quantidade de hardware semelhante ao

APUF(64x1), a Figura 4.17 apresenta o resultado da cobertura para cada uma destas configurações dos APUFs.

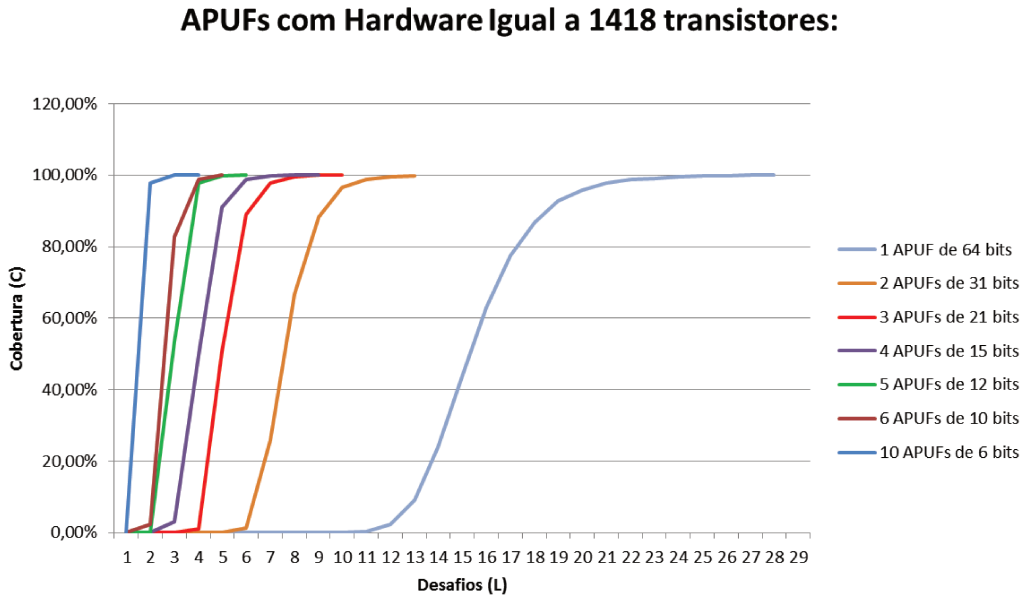


Figura 4.17: APUFs(Mxn) x com Hardware Total Próximo a 1418 Transistores.

Como se observa na Figura 4.17, à medida que aumenta a dimensão N a cobertura do APUF com hardware total aproximado ao hardware do APUF(64x1) melhora.

A cobertura dessas configurações encontradas do APUF com número total de transistores próximo de 1418 são comparadas com cada modelo do CRPUF que também foram comparados com o APUF(64x1) nas Figuras 4.15 e 4.16.

Na Figura 4.18 são apresentados os resultados da comparação (APUF(MxN) X $CRPUF_1(MxN)$) com hardware total próximo de 1418 transistores.

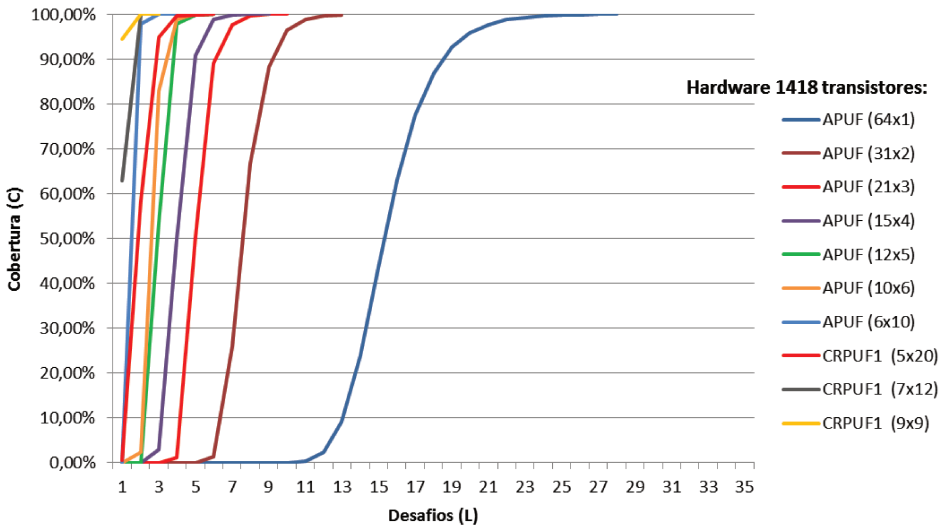


Figura 4.18: APUF(MxN) x $CRPUF_1(MxN)$ com mesmo Hardware do APUF(64x1).

Os resultados mostrado na Figura 4.18 são discutidos em conjunto com os resultados da Figura 4.19. Estes apresentam os resultados da comparação (APUF(MxN) X $CRPUF_2(MxN)$) com hardware total próximo de 1418 transistores.

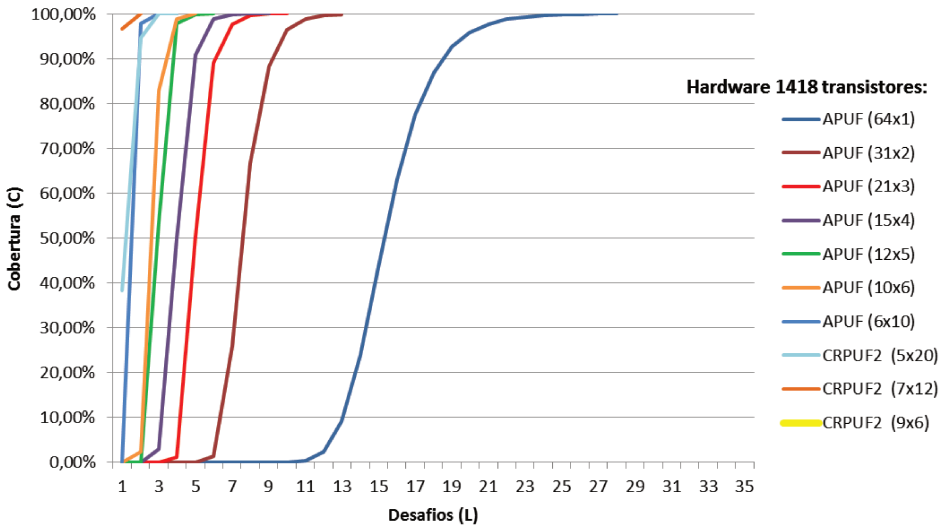


Figura 4.19: APUF(MxN) x $CRPUF_2(MxN)$ com mesmo Hardware do APUF(64x1).

Assim, os resultados dos experimentos apresentados nas Figuras 4.18 e 4.19 mostram que apenas a configuração APUF(10x6) possui cobertura pouco melhor que o $CRPUF_1(5x20)$ (Figura 4.18) e que ele possui também cobertura equivalente ao $CRPUF_2(5x20)$ (Figura

4.19). Todas as demais configurações de APUFs ainda possuem cobertura inferior aos CRPUFs apresentados.

Além disso, deve ser considerado que um APUF(6x10) pode receber apenas 64 desafios. E com a aplicação de 64 desafios é possível identificar todo o conjunto de pares de desafio-resposta de cada um dos dez APUFs de 6 bits. Essa característica torna o APUF com M pequeno fácil de serem mapeados todos os conjuntos de CRPs tanto em bancada, como por modelagem matemática.

4.2.4.2 Experimento Com CRPUF Como Referencial

4.2.4.2.1 Medida CxL para Hardware Igual - o objetivo deste experimento é o de, dado um CRPUF, encontrar APUFs com **hardware igual**/próximo e a partir daí identificar qual o PUF que alcança melhor cobertura com menor número de desafios aplicados.

Com base nos resultados da cobertura do CRPUF apresentados na subseção 4.2.3, foram analisados as várias configurações do CRPUF e assim foi escolhido o CRPUF com dimensões $M = 5$ e $N = 3$ para a realização dos experimentos, pois possui um pequeno hardware e com apenas 8 desafios já alcança uma cobertura próxima de 95%. Estes CRPUFs têm 280 ($CRPUF_1(5x3)$) e 380 ($CRPUF_2(5x3)$) transistores (ver Tabela 4.7).

Os CRPUFs utilizados nesta primeira comparação dos experimentos são: $CRPUF_1(5x3)$ e $CRPUF_2(5x3)$.

O hardware total do $CRPUF_1(5x3)$ e do $CRPUF_2(5x3)$ foi calculado através das Equações 4.4 (pág. 74) e 4.6 (pág. 74) respectivamente. A partir do número de transisores de cada um dos CRPUFs foi obtido o APUF com número de transistores próximo de acordo com Equação 4.7.

O APUF que possui o número de transistores próximos ao $CRPUF_1(5x3)$ e ao $CRPUF_2(5x3)$ é o APUF de 16 bits (ver Tabela 4.8, pág. 77), eles possuem 280, 380 e 362 transistores respectivamente. O $CRPUF_1(5x3)$ possui menor número de transistores, entretanto ele possui as mesmas dimensões do $CRPUF_2(5x3)$. Desta forma, na sequência são apresentados os resultados para a cobertura do APUF de 16 bits quando comparado com o $CRPUF_1(5x3)$ e com o $CRPUF_2(5x3)$.

Nessa sequência, a Figura 4.20 apresenta o crescimento da cobertura à medida que são aplicados desafios ao APUF com 16 bits - APUF(16x1) -, ao $CRPUF_1(5x3)$ e ao $CRPUF_2(5x3)$.

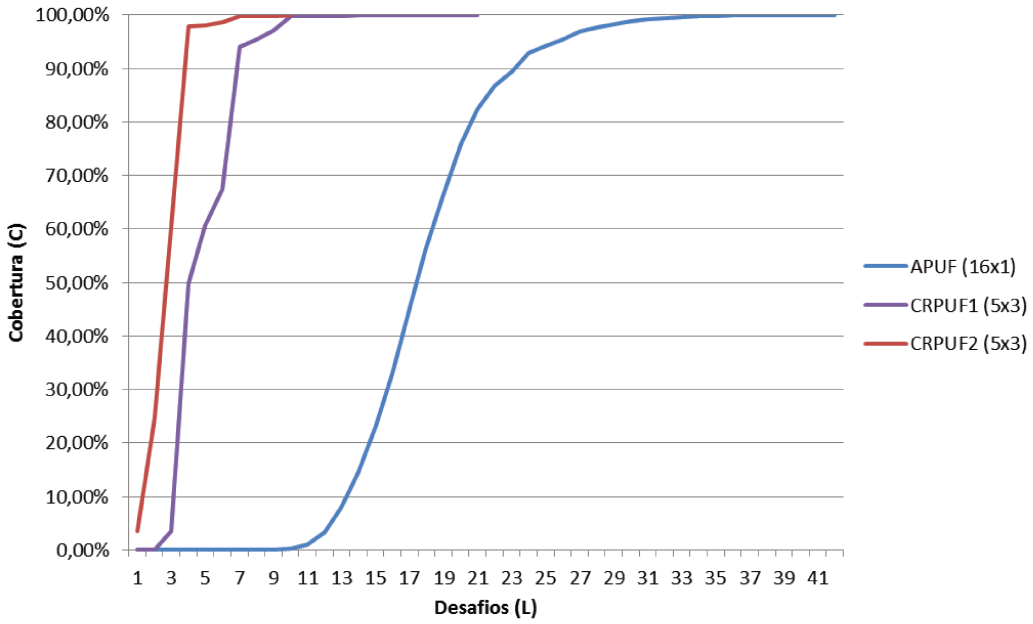


Figura 4.20: Comparação APUF(16x1) e $CRPUF(5x3)$.

O resultado do experimento na Figura 4.20 monstra que a cobertura do APUF cresce a partir do décimo desafio aplicado. No entanto, durante o décimo desafio, o $CRPUF_1(5x3)$ já apresenta uma cobertura próxima dos 100%. O CRPUF alcança os 100% no vigésimo primeiro desafio, já o APUF de 16 bits somente alcança no quadragésimo segundo desafio aplicado. Além disso, o $CRPUF_2(5x3)$ apresenta uma cobertura próxima de 100% no sétimo desafio. Já o APUF alcança os 100% no quadragésimo segundo desafio aplicado.

Desta forma, pode-se concluir que tanto o $CRPUF_1$ quanto o $CRPUF_2$ para os dados simulados possuem coberturas melhores que o APUF com hardware próximos. Um dos motivos seria porque o CRPUF possui uma maior quantidade de bits por resposta. Isso é mostrado na próxima comparação.

Assim, a primeira comparação mostrou que o CRPUF possui melhor cobertura que o APUF quando os dois possuem quantidade de hardware igual para os dados gerados para a simulação.

4.2.4.2.2 Medida CxL para Cobertura Igual - a segunda comparação é feita utilizando-se ainda o $CRPUF_1(5x3)$ e o $CRPUF_2(5x3)$ e a configuração do APUF que possui **cobertura igual** aos CRPUFs. Assim, busca-se identificar a quantidade de hardware necessária para que o APUF tenha cobertura semelhante à obtida pelo $CRPUF_1(5x3)$ e pelo $CRPUF_2(5x3)$.

Entretanto, antes de dar sequência à análise do experimento, é necessário destacar que uma conclusão obtida através da análise do APUF é que a sua cobertura não melhora

significativamente com o aumento do tamanho do desafio. Assim, como apresenta a Figura 4.21, tanto o APUF com 42 bits quanto o APUF com 262 bits possuem cobertura semelhantes.

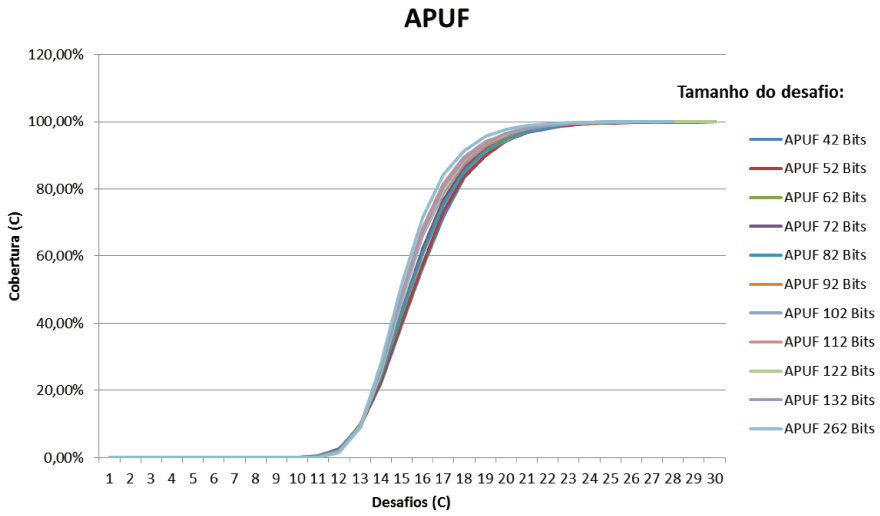


Figura 4.21: Cobertura do APUF para Vários Tamanhos.

Assim, não há grande variação da cobertura para diferentes tamanhos do desafio, pois a cobertura de todos os APUFs da Figura 4.21 são muito próximas. Desta forma, não é viável aumentar o tamanho do APUF através do aumento do desafio.

Uma das explicações para isso é que um único APUF, independente do tamanho do desafio, sempre retorna apenas um bit na saída. E com apenas um bit, por exemplo, não é possível distinguir nem 3 indivíduos, pois para estes seriam necessários a aplicação de no mínimo 2 desafios. Assim, para identificar 10000 indivíduos é necessária a aplicação de pelo menos 14 desafios, pois esse poderia identificar 16384 (2^{14}) indivíduos. Com apenas 13 desafios, no melhor caso, seriam identificados apenas 8192 (2^{13}) indivíduos.

Além disso, dividir o hardware do APUF em dois APUFs com tamanhos M iguais - APUF(Mx2)- de tal forma que o M dos APUFs resultantes seja menor que 16 bits tornaria o APUF com pequena quantidade de desafios. Isso facilita o mapeamento de cada bit de resposta com a aplicação dos poucos desafios.

Considerando-se a Equação 4.7 em que o hardware do APUF de 16 bits é igual a 362 transistores, foram simulados vários APUFs em grupos (valores de N) de tamanhos Ms de forma que o total do número de transistores do conjunto seja próximo de 362 transistores. As dimensões e o número de APUFs em paralelo encontrados - APUF(MxN) - são apresentados na Tabela 4.5.

Tabela 4.5: Conjunto de APUFs(MxN) com Hardware Total Próximo a 362 Transistores.

Estágios (M)	**Hardware**	**Dimensão** (N)	**Hardware Total**
16	362	1	362
8	186	2	372
5	120	3	360
3	76	5	380
2	54	7	378

Na Tabela 4.5, o número de APUFs no conjunto é chamada de dimensões (N). Desta forma, APUF de 8 bits com 2 dimensões - APUF(8x2)- significa que possui 2 APUFs em paralelo de 8 bits estimulados pelo mesmo desafio e, portanto, este PUF retorna uma resposta de 2 bits.

Desta forma, o resultado da simulação para os APUFs da Tabela 4.5 são apresentados na Figura 4.22 para o $CRPUF_1(5x3)$ e na Figura 4.23 para o $CRPUF_2(5x3)$.

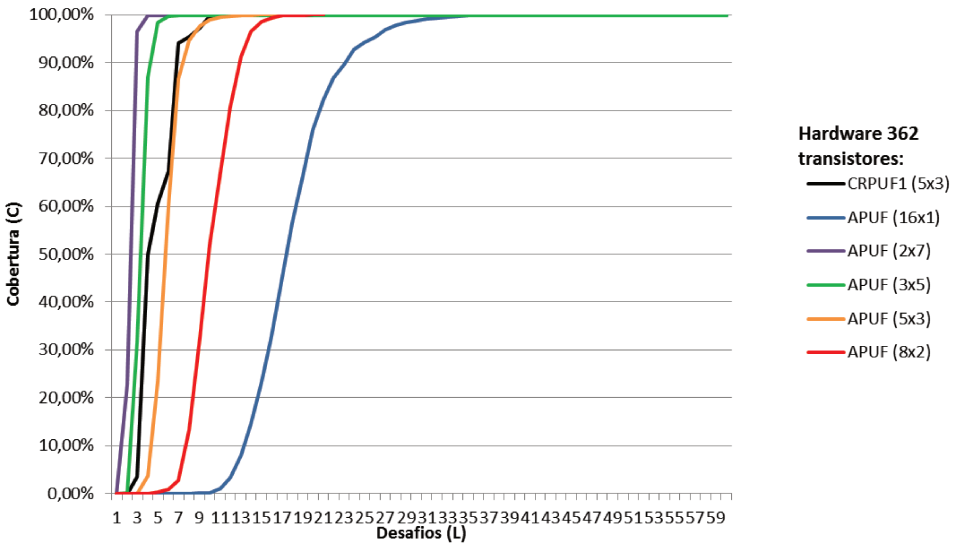


Figura 4.22: APUFs(MxN) para mesmo Hardware do $CRPUF_1(5x3)$.

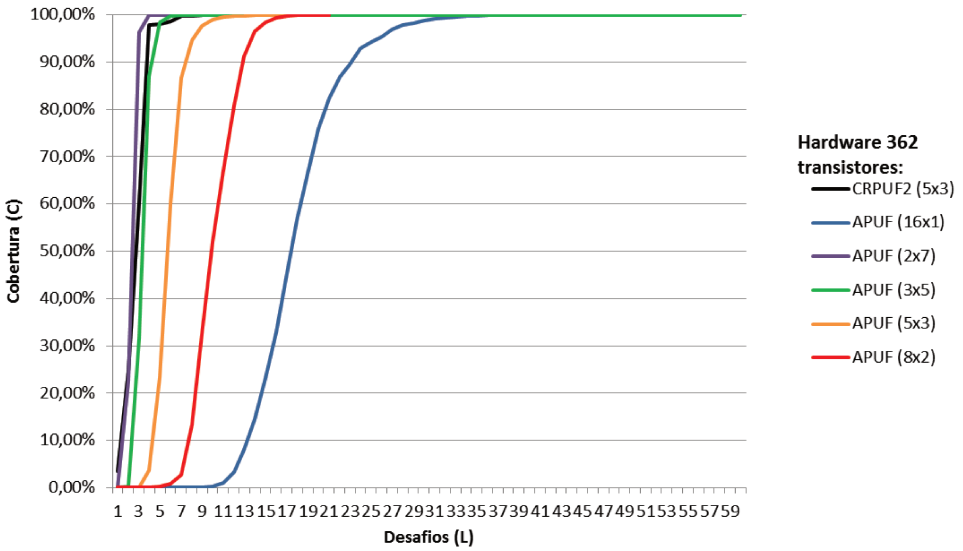


Figura 4.23: APUFs(MxN) para mesmo Hardware do $CRPUF_2(5x3)$.

Assim, como se observa nas Figuras 4.22 e 4.23, a cobertura para quase todas as combinações de APUFs com hardware total próximo a 362 transistores são menores que a cobertura do $CRPUF_1(5x3)$ e a do $CRPUF_2(5x3)$. Apenas na Figura 4.22 as combinações com 7 APUFs de 2 bits e 5 APUFs de 3 bits possuem cobertura melhor que o $CRPUF_1(5x3)$. Entretanto, como discutido, o APUF de 2 pode ter apenas 4 desafios aplicados, e o APUF de 3 bits apenas 8 desafios aplicados. Desta forma, é possível identificar todos os conjuntos de pares de desafio-resposta destes APUFs com poucos desafios.

A segunda comparação busca identificar a quantidade de hardware necessária para obter um APUF com cobertura semelhante ao CRPUF, o APUF de 16 bits também é replicado várias vezes (APUF(16xN), como apresentado pela Figura 2.7, para que a sua cobertura seja próxima à obtida pelo CRPUF_1(5x3) ou CRPUF_2(5x3).

A replicação do APUF de 16 bits N vezes foi utilizada pois é possível obter diretamente um fator multiplicador através da quantidade de replicas (N) do APUF. Isso significa que o APUF(16x4) possui quatro vezes a quantidade de hardware do APUF(16x1) e, portanto, também do CRPUF(5x3).

Desta forma, na Figura 4.24, o $CRPUF_1(5x3)$ possui cobertura próxima à do APUF(16x4). Já o $CRPUF_2(5x3)$ possui cobertura intermediária entre o APUF(16x5) e o APUF(16x6).

Assim, para o APUF obter uma cobertura aproximada à do $CRPUF_1$ foi necessária quatro vezes a quantidade de hardware do $CRPUF_1$, ou seja, o APUF foi replicado quatro vezes. E ainda para o APUF obter uma cobertura aproximada à do $CRPUF_2$ foi necessária pelo menos cinco vezes a quantidade de hardware do $CRPUF_2$.

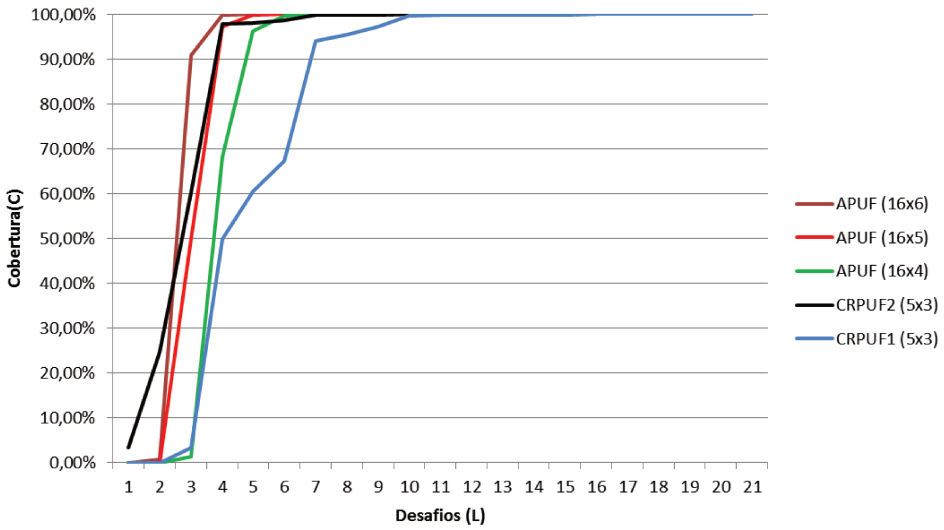


Figura 4.24: APUF 16 bits x $CRPUFs(5x3)$.

4.3 Análise Quantitativa Teórica

Esta seção descreve uma avaliação analítica de como determinar a quantidade de hardware (seção 4.3.1)e o limite superior de respostas do APUF e do CRPUF (subseção 4.3.2).

4.3.1 Cálculo da Quantidade de Hardware em Transistores

O cálculo da quantidade de hardware em número de transistores para a realização dos experimentos é feito utilizando o número de transistores de cada célula que compõe os modelos APUF e CRPUF. Para cada uma das células da arquitetura dos PUFs é determinada o número de transitores, como apresentado na Tabela 4.6.

Tabela 4.6: Quantidade de Transistores CMOS por Bloco/Porta Lógica.

Célula	**Combinação**	**Quantidade Total de Transistores**
XOR	AOI + 2 Inversores	12
Flip-Flop	-	10
MUX2:1	AOI + 1 Inversor	11

Os blocos básicos e portas lógicas de composição de cada um dos modelos são apresentados na seção 2.2.3.1 (APUF) e no Capítulo 3 (CRPUF). Assim, esses valores da Tabela 4.6 são utilizados como referencial para a caracterização da quantidade de hardware de cada um dos modelos APUF e CRPUF. Eles são utilizados no decorrer das demais

subseções para cálculo da quantidade de hardware de cada modelo de PUF.

4.3.1.1 CRPUF

A partir do número de transistores por célula definida na Tabela 4.6, foi analisada a quantidade de hardware do CRPUF para vários tamanhos, conforme descrito na seção 4.2.1. Esses dados são necessários para o cálculo do hardware do CRPUF.

Descrito no Capítulo 3, o CRPUF é composto por portas lógicas XOR e *flipflops* utilizados como árbitro para determinar o resultado da corrida.

Para o cálculo da quantidade de hardware do CRPUF de várias dimensões foram utilizados os dados da Tabela 4.6.

A seguir, baseado na composição do CRPUF, são determinadas as equações gerais para o cálculo da quantidade de hardware do $CRPUF_1$ e $CRPUF_2$. Como existem dois tipos de CRPUF com número de transistores diferentes, é calculado inicialmente o hardware total para o $CRPUF_1$ e, em seguida, para o $CRPUF_2$. Desta forma, a expressão geral que determina a quantidade de hardware em transistores de um $CRPUF_1(MxN)$ composto por portas lógicas XOR e *flipflops* é apresentada na Equação 4.3:

$$H_{CRPUF_1} = M \cdot N_1 \cdot H_{XOR} + N_1 \cdot H_{FF} \tag{4.3}$$

Onde:

- H_{CRPUF_1} é o número de transistores do $CRPUF_1$;
- M é o número de linhas do CRPUF;
- N_1 é o número de colunas (estágios) do $CRPUF_1$;
- H_{XOR} é o número de transistores da porta lógica XOR (ver Tabela 4.6);
- H_{FF} é o número de transistores do *flipflop* (ver Tabela 4.6).

Substituindo N_1 na Equação 4.3 pelo número de *flipflops* do $CRPUF_1$ determinado na Equação 3.2, obtêm-se a seguinte equação:

$$H_{CRPUF_1} = M \cdot N_1 \cdot H_{XOR} + \frac{M \cdot (M-1)}{2} \cdot H_{FF} \tag{4.4}$$

Para o $CRPUF_2$ há uma pequena variação no número de transistores com relação à Equação 4.3 devido ao número maior de *flipflops*. Assim, a Equação 4.5 determina a expressão geral para a quantidade de hardware em transistores de um $CRPUF_2(MxN)$:

$$H_{CRPUF_2} = M \cdot N_2 \cdot H_{XOR} + N_2 \cdot H_{FF} \tag{4.5}$$

Onde:

- H_{CRPUF_2} é o número de transistores do $CRPUF_2$;
- N_2 é o número de colunas (estágios) do $CRPUF_2$.

Substituindo N_2 na Equação 4.5 pelo número de *flipflops* do $CRPUF_2$ determinada na Equação 3.3, obtêm-se a seguinte equação:

$$H_{CRPUF_2} = M \cdot N_2 \cdot H_{XOR} + M \cdot (M-1) \cdot H_{FF} \tag{4.6}$$

A número de transistores utilizado para alguns tamanhos de cada um dos tipos do $CRPUF$ obedecendo as Equações 4.4 e 4.6 é apresentada na Tabela 4.7.

Tabela 4.7: Dimensão e Número de Transistores do $CRPUF$.

M	**N**	$CRPUF_1$	$CRPUF_2$
3	8	318	348
	10	390	420
	20	750	780
	30	1110	1140
5	3	280	380
	8	580	680
	10	700	800
	20	1300	1400
	30	1900	2000
7	8	882	1092
	10	1050	1260
	20	1890	2100
	30	2730	2940
9	8	1224	1584
	10	1440	1800
	20	2520	2880
	30	3600	3960

Observa-se que o aumento da dimensão para os dois tipos do CRPUF possui um crescimento linear no número de transistores com relação ao tamanho do desafio. Desta forma, o CRPUF possui comportamento linear com relação à escalabilidade e à quantidade de hardware.

4.3.1.2 APUF

A partir do número de transistores por célula mostradas na Tabela 4.6, foi analisada a quantidade de hardware do $APUF$ para vários tamanhos de desafios, conforme descrito na seção 4.2.1. Esses dados são necessários para a realização dos experimentos que obtêm a cobertura do APUF.

Como apresentado na seção 2.2.3.1, o APUF é composto por dois multiplexadores por estágio, ou bit de desafio, e um *flipflop* utilizado como árbitro para determinar o resultado da corrida.

A partir dos valores definidos na Tabela 4.6, é possível obter o número de transistores utilizado pelo APUF usando a Equação 4.7:

$$H_{APUF} = 2 \cdot M \cdot H_{MUX2:1} + H_{FF} \tag{4.7}$$

Onde:

- H_{APUF} é o número de transistores do $APUF$;
- M é o número de estágios do APUF ou tamanho do desafio;
- $H_{MUX2:1}$ é o número de transistores do multiplexador de 2 para 1 (ver Tabela 4.6);
- H_{FF} é o número de transistores do *flipflop* (ver Tabela 4.6).

Seguindo a Equação 4.7, a Tabela 4.8 apresenta o crescimento do número de transistores para o APUF com vários tamanhos de desafio.

Tabela 4.8: Dimensão e Quantidade de Transistores do APUF.

Tamanho do Desafio em Bits	**Hardware Total em Transistores**
16	362
32	714
33	736
34	758
35	780
36	802
37	824
38	846
39	868
40	890
64	1418
128	2826
256	5642
512	11274
768	16906
1024	22538

Observa-se que, igualmente ao CRPUF, o aumento da dimensão para o APUF também possui um crescimento linear no número de transistores com relação ao tamanho do desafio. Desta forma, tanto o CRPUF quanto o APUF possuem comportamento semelhantes com relação à escalabilidade e à quantidade de hardware.

4.3.2 Análise de Limite Superior de Respostas

Esta subseção realiza uma abordagem analítica do limite superior de respostas (resultados possíveis) para os modelos CRPUF (subseção 4.3.2.1) e o APUF (subseção 4.3.2.2).

A análise do limite superior de respostas identifica qual o limite teórico da diversidade de comportamentos de um PUF em função de suas dimensões. Através dela é possível

identificar qual a capacidade máxima de PUFs (com comportamentos distintos) em função do tamanho do PUF que pode existir.

Antes de iniciar a análise do limite superior de respostas do APUF e do CRPUF em função de suas dimensões, é preciso especificar que ele é obtido através do número máximo de respostas obtidas por cada desafio e do número máximo de desafios possíveis na entrada. Nas subseções a seguir são apresentados os resultados para cada um destes dois modelos.

4.3.2.1 CRPUF

Para o CRPUF, o número de *flipflops* e a combinação dos pares de sinais da saída do CRPUF (entradas dos *flipflops*) são responsáveis por determinar a assinatura da saída. Assim, a Equação 4.8 determina o **Número de Respostas Por Desafio** - $NRPD$ - em função do número de *flipflops* (N_{FF}) na saída do CRPUF.

$$NRPD = 2^{N_{FF}} \tag{4.8}$$

Substituindo N_{FF} na Equação 4.8 pelo número de *flipflops* de cada um dos tipos $CRPUF_1$ (N_1) e $CRPUF_2$ (N_2) determinados com as Equações 3.2 e 3.3 respectivamente, determina-se o ($NRPD$) para cada um dos CRPUFs.

Assim, para o $CRPUF_1$ o número de respostas por desafio ($NRPD_1$) é determinado pela Equação 4.9:

$$NRPD_1 = 2^{M \cdot \frac{(M-1)}{2}} \tag{4.9}$$

Onde M é o número de linhas do CRPUF.

Para o $CRPUF_2$ o número de respostas por desafio ($NRPD_2$) é determinado pela Equação 4.10:

$$NRPD_2 = 2^{M \cdot (M-1)} \tag{4.10}$$

O **Número Total de Comportamentos Possíveis** - $NTCP$ - é determinado na Equação 4.11 pela potência do Número de Respostas Possíveis Por Desafio - **NRDP** - pelo Número de Desafios Possíveis - **NDP** - da Equação 3.1 de cada um dos tipos de CRPUF determinados pelas Equações 4.9 e 4.10.

$$NTCP = NRPD^{NDP} \tag{4.11}$$

Como existem NDP desafios em que cada um pode resultar em $NRPD$ respostas, o $NTCP$ representa o número máximo total de respostas dos PUFs. Assim, o NTCP para o $CRPUF_1$ ($NTCP_1$) é determinado pela Equação 4.12:

$$NTCP_1 = 2^{\left(\frac{M\cdot(M-1)}{2}\right)\cdot\left(2^{2M}-2^M\right)} \tag{4.12}$$

Já o NTCP para o $CRPUF_2$, o $NTCP_2$, é determinado pela Equação 4.13:

$$NTCP_2 = 2^{M\cdot(M-1)\cdot\left(2^{2M}-2^M\right)} \tag{4.13}$$

Assim, as equações gerais do $CRPUF_1$ e do $CRPUF_2$ são utilizadas na subseção 4.3.2.3 e comparadas com os resultados do APUF que são obtidas na subseção 4.3.2.2.

4.3.2.2 APUF

Para o APUF são utilizados os mesmos critérios que os utilizados para determinação do limite superior do CRPUF. Desta forma, substituindo os parâmetro do APUF na Equação 4.11, onde $NTCP = NRPD^{NDP}$, obtém-se o $NTCP_{APUF}$ que é determinado pela Equação 4.14:

$$NTCP_{APUF} = 2^{2^M} \tag{4.14}$$

Para o APUF, o NDP é igual a 2^M, onde M representa o número de estágios do APUF, e o $NRPD$ é igual a 2, pois só há um *flipflop*, que só pode ter o resultado '1' ou '0'.

Um exemplo hipotético seria o $APUF(2x1)$, que possui dois bits de desafio ($NDP = 2^2$) e apenas um bit de resposta ($NRPD = 2$). De acordo com a Equação 4.14, o $NTCP$ desse APUF é igual a 2^{2^2}. Desta forma são no total de 16 comportamentos possíveis como apresentados na Tabela 4.9.

Desafios Aplicados				Comportamento
00	01	10	11	
0	0	0	0	1
0	0	0	1	2
0	0	1	0	3
0	0	1	1	4
0	1	0	0	5
0	1	0	1	6
0	1	1	0	7
0	1	1	1	8
1	0	0	0	9
1	0	0	1	10
1	0	1	0	11
1	0	1	1	12
1	1	0	0	13
1	1	0	1	14
1	1	1	0	15
1	1	1	1	16

Tabela 4.9: Número Total de Comportamentos Possíveis para o APUF(2x1).

Como mostra a Tabela 4.9, é possível aplicar no máximo 4 desafios distintos no APUF(2x1), ou seja, 4 bits de respostas (1 para cada desafio aplicado).

Baseada nas equações gerais obtidas, a subseção 4.3.2.3 apresenta as comparações entre o limite superior do APUF e do CRPUF.

4.3.2.3 CRPUF x APUF

Utilizando as equações 4.12 e 4.13 para o CRPUF e a equação 4.14 para o APUF é possível comparar os limites máximos de respostas do CRPUF e do APUF, como apresentado na Tabela 4.10.

Tabela 4.10: Resumo da Limite Superior de Respostas do $CRPUF$ e do $APUF$.

M	NDP	$CRPUF_1$		$CRPUF_2$		APUF
		$2^{\frac{M\cdot(M-1)}{2}}$	$2^{\left(\frac{M\cdot(M-1)}{2}\right)\cdot\left(2^{2M}-2^{M}\right)}$	$2^{M\cdot(M-1)}$	$2^{(M\cdot(M-1))\cdot\left(2^{2M}-2^{M}\right)}$	2^{2^M}
		$NRPD_1$	$NTCP_1$	$NRPD_2$	$NTCP_2$	$NTCP_{APUF}$
3	56	8	8^{56}	64	64^{56}	2^8
5	992	1024	1024^{992}	1048576	1048576^{992}	2^{32}
7	16256	2097152	2097152^{16256}	4398046511104	4398046511104^{16256}	2^{128}
9	261632	68719476736	68719476736^{261632}	$4,7223 \cdot 10^{21}$	$(4,7223 \cdot 10^{21})^{261632}$	2^{512}

A dimensão M para os dois modelos identifica o número de estágios (APUF) e o número de células por estágio (CRPUF). Esta análise representa o potencial máximo de respostas para cada um dos modelos de acordo com o tamanho de M.

Observa-se que o crescimento do limite superior de respostas do CRPUF é exponencial. Isto se deve ao fato de que o CRPUF possui várias saídas. Já o crescimento do número de transistores do CRPUF é linear em relação ao aumento do número de linhas (ver Tabela 4.7, pág. 76).

4.4 Considerações Finais

Foram apresentados os experimentos e as análises realizadas tanto com o CRPUF quanto com o APUF. Na seção 4.2 são apresentados o planejamento e a execução dos experimentos. Na seção 4.3 são apresentadas as avaliações realizadas no CRPUF e no APUF.

Na seção 4.3 foi mostrada uma avaliação analítica dos dois modelos de PUFs identificando o número de transistores e o limite superior do número de resultados possíveis de cada tamanho do CRPUF e do APUF. Na seção 4.2 são apresentados os experimentos realizados e suas descrições.

Os experimentos mostraram separadamente o comportamento do CRPUF e do APUF com relação à cobertura para várias configurações de tamanho de cada um deles. Em seguida foram apresentadas no primeiro experimento comparações com base no $CRPUF(5x3)$ escolhido e, no segundo experimento, são apresentadas comparações com base no APUF(64x1).

Assim, foi mostrado pelos resultados dos experimentos realizados que o $CRPUF(5x3)$ possui aproximadamente 4 a 5 vezes melhor resultado de cobertura que o APUF.

O segundo experimento manteve resultados semelhantes aos do primeiro experimento, pois somente aumentando a dimensão do APUF (N) foi possível obter uma cobertura equivalente à do CRPUF.

Através das análises realizadas foi possível concluir que as respostas do CRPUF possuem crescimento do número de respostas exponencial com relação ao tamanho do desafio e, além disso, o crescimento do hardware do CRPUF possui comportamento linear. Os dados determinados por [25] da Tabela 4.11 são apresentados com a inclusão dos resultados do CRPUF obtidos nesta Dissertação.

Tabela 4.11: Imprevisibilidade X Tamanho da Implementação em Relação ao Tamanho do Desafio com CRPUF.

Modelo	**Imprevisibilidade das Respostas**	**Tamanho da Implementação**
Arbiter PUF	Linear	Linear
Basic RO PUF	Linear	Linear
Optical PUF	Superlinear	Linear
Coating PUF	Exponencial	Exponencial
SRAM/FF PUF	Exponencial	Exponencial
Latch/Butterfly PUF	Exponencial	Exponencial
Paired RO PUF	Exponencial	Exponencial
CRPUF	**Exponencial**	**Linear**

Desta forma, o CRPUF responde à questão aberta proposta pelos autores Maes e Verbauwhede em [25].

Questionaram em [25] se seria possível desenvolver algum modelo de PUF em que o aumento do tamanho do desafio implicasse num crescimento da imprevisibilidade das respostas de forma exponencial, mas em um acréscimo linear no tamanho da implementação. Como demonstrado nos resultados experimentais, através das análises realizadas, a imprevisibilidade das respostas do CRPUF crescem exponencialmente com o aumento do desafio, mas mantendo um crescimento linear no número de transistores necessários.

Capítulo 5

Conclusões

Esta Dissertação propõe um novo modelo de PUF (*Physical Unclonable Functions*)baseado em atrasos, contribuindo assim para a área de segurança da informação.

De acordo com os resultados esperados neste trabalho, foi demonstrado que o CRPUF (Cylindrical Reconvergence PUF):

1. apresenta melhor desempenho de cobertura que o APUF quando ambos possuem a mesma quantidade de hardware (número aproximado de transistores);

2. possui, com relação ao seu tamanho, aumento da imprevisibilidade das respostas exponencial com aumento de hardware linear.

Um dos primeiros desafios deste trabalho foi entender as características consideradas mais importantes na construção dos diversos modelos de PUFs elétricos. As características dos principais modelos de PUFs elétricos foram estudadas.

Dentre os vários modelos existentes e discutidos, o pioneiro APUF (*Arbiter PUF*), que trouxe os principais conceitos sobre os PUFs, despertou maior interesse porque utiliza pouco hardware, sendo de fácil implementação.

Durante o desenvolvimento do trabalho desta Dissertação outros novos modelos de PUFs foram pesquisados, simulados e até testados em FPGA através de uma plataforma de teste automático para os PUFs em FPGA que foi desenvolvida especificamente para a finalidade deste trabalho. Entretanto os resultados obtidos dos outros modelos pesquisados não foram suficientes para a caracterização de um novo modelo de PUF. Os resultados obtidos apresentaram baixa cobertura, possibilitando a existência de alta taxa de *aliasing*. Apesar disso, a experiência acumulada ao longo do desenvolvimento destes outros modelos de PUFs gerou aprendizado sobre as dificuldades de implementação e detalhes de funcionamento dos PUFs.

Apesar de a simulação dos experimentos desta dissertação utilizar uma faixa hipotética para o atraso dos PUFs, simulando variações específicas do projeto, tanto o CR-PUF

quanto o APUF foram colocados nas mesmas condições. As pequenas variações na ordem de um femtosegundo, que permite uma variação máxima entre dois atrasos da ordem de um nanosegundo para os caminhos dos PUFs, e a hipótese de *flipflops* ideais permitem considerar os resultados obtidos para os dois modelos. Assim, foi utilizada uma distribuição gaussiana normalizada para gerar cada atraso dentro de uma faixa de atrasos máxima de um nanosegundo simulando uma variação física de um mesmo processo de fabricação para os PUFs.

Durante o desenvolvimento dos trabalhos buscou-se a utilizar de uma variação máxima determinada pelas ferramentas de EDA (*Electronic Design Automation*) da *Cadence Design Systems* para a tecnologia de $0,35\mu m$ da AMS (*Austria Micro Systems*). Entretanto, devido a alguns imprevistos e prazo para conclusão deste trabalho, esta atividade foi abandonada.

Os resultados dos experimentos demonstraram que o CRPUF possui desempenho cinco vezes melhor da cobertura de indivíduos com relação à quantidade de hardware quando comparado com o APUF, ou seja, possui maior imprevisibilidade das respostas. Assim, os resultados experimentais indicam a eficiência do modelo CR-PUF desenvolvido, bem como a capacidade do mesmo cumprir os objetivos definidos para a autenticação do circuito integrado.

5.1 Contribuições da Dissertação

Além desse trabalho outros dois trabalhos em nível de mestrado e de iniciação científica estão sendo desenvolvidos a partir de problemas investigados inicialmente através desta Dissertação.

Este é um trabalho de pesquisa pioneiro relacionado com PUFs no Instituto de Computação da UNICAMP que trouxe várias questões ainda abertas que devem ser investigadas.

Justamente por abordar várias atividades, as contribuições desta Dissertação podem ser elencadas da seguinte forma:

- A proposição de um novo modelo de PUF elétrico baseado em atrasos para autenticação de circuitos integrados. O novo modelo CRPUF (Cylindrical Reconvergence PUF) possui capacidade de identificar um maior número de indivíduos que o APUF para uma mesma quantidade de hardware e usando o mesmo número de desafios;

- A comprovação experimental que o CRPUF possui cobertura equivalente a um APUF que tem cinco vezes mais hardware;

- A comprovação experimental que o aumento das dimensões do CRPUF aumenta a cobertura, ou seja, diminui a taxa de *aliasing*;
- A comprovação experimental que o aumento do tamanho do desafio do APUF não aumenta significativamente a sua cobertura, ou seja, não diminui a taxa de *aliasing*;
- A comprovação que o CRPUF supre a lacuna como modelo de PUF que apresenta crescimento da imprevisibilidade das respostas aos desafios exponencial, mas com aumento do hardware linear;
- A identificação de propriedade específicas para PUFs baseados em atrasos.

Assim, a busca constante em divulgar os resultados obtidos permeou durante todo o desenvolvimento deste trabalho. Entretanto, as condições provenientes de um trabalho pioneiro e que trouxe um resultado inovador, dificultou a geração de resultados rápidos que pudessem ser publicados ainda durante o desenvolvimento deste trabalho.

Os resultados deste trabalho estão sendo consolidados em um artigo que apresenta o CRPUF e suas vantagens e que será submetido a uma conferência internacional.

5.2 Propostas de Trabalhos Futuros

Uma das características marcantes desta Dissertação foi a abertura de trabalhos em várias linhas de pesquisa. Diante desse contexto e com a complexidade inerente à pesquisa envolvendo a proposição de novos modelos de PUFs, há diversas possibilidades de extensão deste trabalho. A lista a seguir, fornece algumas sugestões para desenvolvimento de pesquisas futuras:

- Implementar num circuito integrado e identificar as propriedades do CRPUF através dos seus parâmetros. O intuito aqui é implementar o CRPUF e caracterizar as suas principais propriedades, podendo sugerir mudanças e utilizá-lo num processo de autenticação de circuito integrado.
- Implementar o CRPUF num circuito integrado e compará-lo com outros modelos de PUFs.
- Utilizar outras células para composição do CRPUF e identificar as propriedades.
- Simular o funcionamento do CRPUF com a variação de atrasos provenientes de ferramentas de EDA utilizando várias tecnologias para caracterizar a sua cobertura.

Referências Bibliográficas

[1] D. W. Bauder. An Anti - Counterfeiting Concept for Currency Systems. 1983.

[2] James D. R. Buchanan, Russell P. Cowburn, Ana-Vanessa Jausovec, Dorothée Petit, Peter Seem, Gang Xiong, Del Atkinson, Kate Fenton, Dan A. Allwood, and Matthew T. Bryan. Forgery: 'fingerprinting' documents and packaging. *Nature*, 436(7050):475, July 2005.

[3] Commission on Engineering Committee on Next-Generation Currency Design and National Research Council Technical Systems. *Counterfeit Deterrent Features for the Next-Generation Currency Design*. The National Academies Press, 1993.

[4] D. Lim and J. W. Lee and B. Gassend and G. E. Suh and M. van Dijk and S. Devadas. Extracting Secret Keys From Integrated Circuits. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 13(10):1200–1205, October 2005.

[5] Gerald Dejean and Darko Kirovski. Rf-dna: Radio-frequency certificates of authenticity. In *Proceedings of the 9th International Workshop on Cryptographic Hardware and Embedded Systems*, CHES '07, pages 346–363, Berlin, Heidelberg, 2007. Springer-Verlag.

[6] Blaise Gassend. Physical Random Functions. Master's thesis, Massachusetts Institute of Technology, January 2003.

[7] Blaise Gassend, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas. Controlled Physical Random Functions. In *Proc. of the 18th Annual Computer Security Applications Conference*, December 2002.

[8] Blaise Gassend, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas. Controlled physical random functions. In *ACSAC '02: Proceedings of the 18th Annual Computer Security Applications Conference*, page 149, Washington, DC, USA, 2002. IEEE Computer Society.

[9] Blaise Gassend, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas. Silicon physical random functions. In *ACM Conference on Computer and Communications Security*, pages 148–160, New York, NY, USA, 2002. ACM Press.

[10] Blaise Gassend, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas. Silicon Physical Random Functions. In *Proc. of the* 9^{th} *ACM Conference on Computer and Communications Security*, November 2002.

[11] Blaise Gassend, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas. Delay-based circuit authentication and applications. In *SAC '03: Proceedings of the 2003 ACM Symposium on Applied Computing*, pages 294–301, New York, NY, USA, 2003. ACM.

[12] Blaise Gassend, Marten Van Dijk, Dwaine Clarke, Emina Torlak, Srinivas Devadas, and Pim Tuyls. Controlled physical random functions and applications. *ACM Trans. Inf. Syst. Secur.*, 10(4):1–22, 2008.

[13] Blaise Gassend, Daihyun Lim, Dwaine Clarke, Srinivas Devadas, and Marten van Dijk. Identification and authentication of integrated circuits. *Concurrency and Computation: Practice and Experience*, 16(11):1077–1098, September 2004.

[14] Blaise Gassend, Marten van Dijk, Dwaine Clarke, Emina Torlak, Pim Tuyls, and Srinivas Devadas. Controlled physical random functions and applications. *ACM Transactions on Information and System Security*, 10(4), January 2008.

[15] Ghaith Hammouri, Aykutlu Dana, and Berk Sunar. Cds have fingerprints too. In Christophe Clavier and Kris Gaj, editors, *Cryptographic Hardware and Embedded Systems - CHES 2009*, volume 5747 of *Lecture Notes in Computer Science*, pages 348–362. Springer Berlin / Heidelberg, 2009. 10.1007/978-3-642-04138-9_25.

[16] Ryan Helinski, Dhruva Acharyya, and Jim Plusquellic. A physical unclonable function defined using power distribution system equivalent resistance variations. In *DAC '09: Proceedings of the 46th Annual Design Automation Conference*, pages 676–681, New York, NY, USA, 2009. ACM.

[17] Daniel E. Holcomb, Wayne P. Burleson, and Kevin Fu. Initial SRAM state as a fingerprint and source of true random numbers for RFID tags. In *Proceedings of the Conference on RFID Security*, July 2007.

[18] Ronald S. Indeck and Marcel W. Muller. Method and apparatus for fingerprinting magnetic media. Number 5365586, November 1994.

[19] Aswin Krishna, Seetharam Narasimhan, Xinmu Wang, and Swarup Bhunia. Mecca: A robust low-overhead puf using embedded memory array. In Bart Preneel and Tsuyoshi Takagi, editors, *Cryptographic Hardware and Embedded Systems - CHES 2011*, volume 6917 of *Lecture Notes in Computer Science*, pages 407–420. Springer Berlin / Heidelberg, 2011. 10.1007/978-3-642-23951-9_27.

[20] S.S. Kumar, J. Guajardo, R. Maes, G.-J. Schrijen, and P. Tuyls. Extended abstract: The butterfly puf protecting ip on every fpga. In *Hardware-Oriented Security and Trust, 2008. HOST 2008. IEEE International Workshop on*, pages 67–70, June 2008.

[21] Daihyun Lim. Extracting Secret Keys from Integrated Circuits. Master's thesis, MIT, USA, 2004.

[22] Lang Lin, Dan Holcomb, Dilip Kumar Krishnappa, Prasad Shabadi, and Wayne Burleson. Design optimization and security validation of sub-threshold pufs. In *SECSI-2010*, 2010.

[23] Lang Lin, Dan Holcomb, Dilip Kumar Krishnappa, Prasad Shabadi, and Wayne Burleson. Low-power sub-threshold design of secure physical unclonable functions. In *Proceedings of the 16th ACM/IEEE International Symposium on Low Power Electronics and Design*, ISLPED 2010, pages 43–48, New York, NY, USA, 2010. ACM.

[24] Keith Lofstrom, W. Robert Daasch, and Donald Taylor. Ic identification circuit using device mismatch. In *In Proceedings of ISSCC 2000*, pages 372–373, 2000.

[25] R. Maes and I. Verbauwhede. A discussion on the properties of physically unclonable functions. In *TRUST - Workshop on Security Hardware*, 2010.

[26] Roel Maes and Ingrid Verbauwhede. Physically unclonable functions: A study on the state of the art and future research directions. In Ahmad-Reza Sadeghi and David Naccache, editors, *Towards Hardware-Intrinsic Security*, Information Security and Cryptography, pages 3–37. Springer Berlin Heidelberg, 2010. 10.1007//978-3-642-14452-3_1.

[27] Abhranil Maiti, Vikash Gunreddy, and Patrick Schaumont. A systematic method to evaluate and compare the performance of physical unclonable functions. *IACR Cryptology ePrint Archive*, 2011:657, 2011.

[28] E. Ozturk, G. Hammouri, and B. Sunar. Physical unclonable function with tristate buffers. In *Circuits and Systems, 2008. ISCAS 2008. IEEE International Symposium on Circuits and Systems*, pages 3194 –3197, 18-21 2008.

[29] Ravikanth S. Pappu. *Physical One-Way Functions*. PhD thesis, Massachusetts Institute of Technology, March 2001.

[30] Ravikanth S. Pappu, Ben Recht, Jason Taylor, and Niel Gershenfeld. Physical one-way functions. *Science*, 297:2026–2030, 2002.

[31] Ulrich Rührmair, Frank Sehnke, Jan Sölter, Gideon Dror, Srinivas Devadas, and Jürgen Schmidhuber. Modeling attacks on physical unclonable functions. Cryptology ePrint Archive, Report 2010/251, 2010. `http://eprint.iacr.org/`.

[32] Ying Su, Jeremy Holleman, and Brian P. Otis. A 1.6pJ/bit 96% Stable Chip-ID Generating Circuit using Process Variations. In *International Solid-State Circuits Conference (ISSCC)*, pages 406–611. IEEE, February 2007.

[33] G. Edward Suh and Srinivas Devadas. Physical unclonable functions for device authentication and secret key generation. In *DAC '07: Proceedings of the 44th Annual Design Automation Conference*, pages 9–14, New York, NY, USA, 2007. ACM.

[34] Daisuke Suzuki and Koichi Shimizu. The glitch puf: A new delay-puf architecture exploiting glitch shapes. In Stefan Mangard and François-Xavier Standaert, editors, *Cryptographic Hardware and Embedded Systems, CHES 2010*, volume 6225 of *Lecture Notes in Computer Science*, pages 366–382. Springer Berlin / Heidelberg, 2011. 10.1007/978-3-642-15031-9_25.

[35] K.M. Tolk. Reflective Particle Technology for Identification of Critical Components. Technical report, sand–92-1676c; conf-9207102–26, Sandia National Labs., Albuquerque, NM, USA, August 1992.

[36] Pim Tuyls, Geert-Jan Schrijen, Boris Škorić, Jan van Geloven, Nynke Verhaegh, and Rob Wolters. Read-proof hardware from protective coatings. In *Proceedings of the 8th International Conference on Cryptographic Hardware and Embedded Systems*, CHES'06, pages 369–383, Berlin, Heidelberg, 2006. Springer-Verlag.

[37] Lina Wang, Xiaqiu Jiang, Shiguo Lian, Donghui Hu, and Dengpan Ye. Image authentication based on perceptual hash using gabor filters. *Soft Comput.*, 15(3):493–504, March 2011.

[38] Neil Weste and David Harris. *CMOS VLSI Design: A Circuits and Systems Perspective*. Addison-Wesley Publishing Company, USA, 4th edition, 2010.

Apêndice A

Avaliação do APUF

Resultados experimentais para a caracterização do APUF (MxN) referentes ao Capítulo 4 (seção 4.2.2).

Os resultados são apresentados em ordem crescente com relação às dimensões M e N do APUF. Foram simuladas as configurações do APUF com $M \in \{12, 16, 32, 64\}$ bits e $N \in \{1, 2, 3, 4, 6, 8\}$. Onde M representa o tamanho do desafio em bits, já o N representa o número de cópias em paralelo do APUF. Os resultados são apresentados na mesma escala.

Inicialmente são apresentados os resultados das simulações do APUF para variações de N.

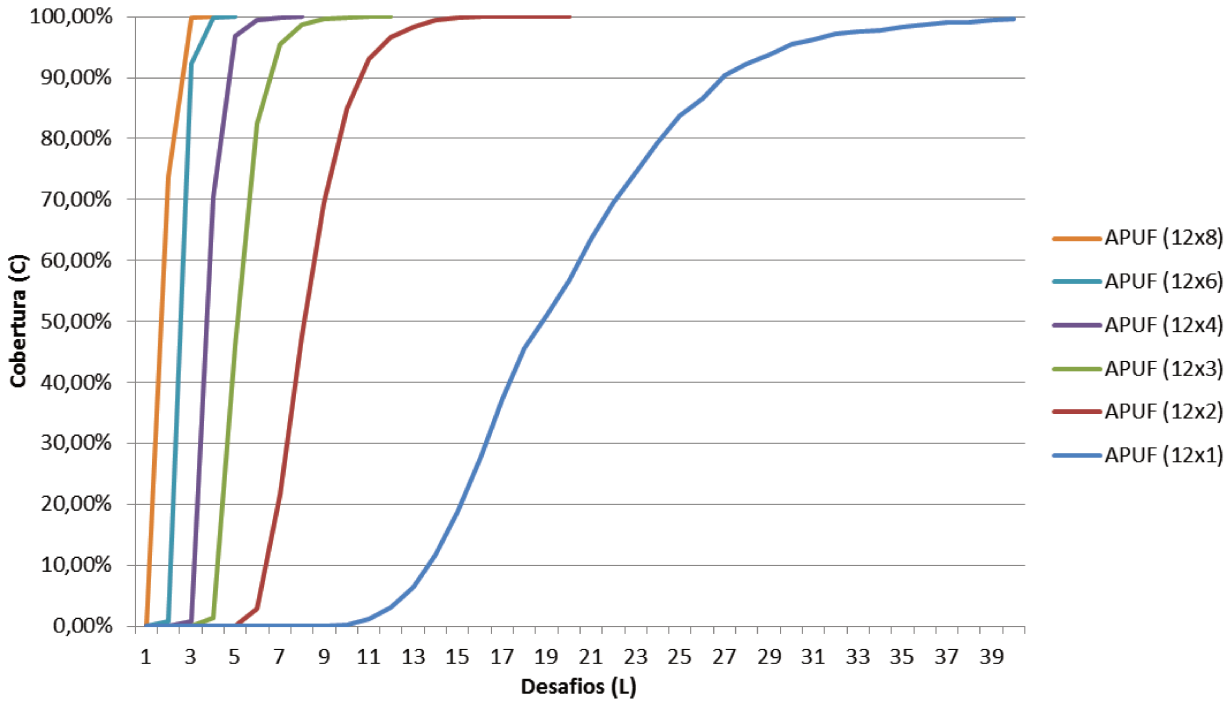


Figura A.1: Cobertura para o APUF(12xN).

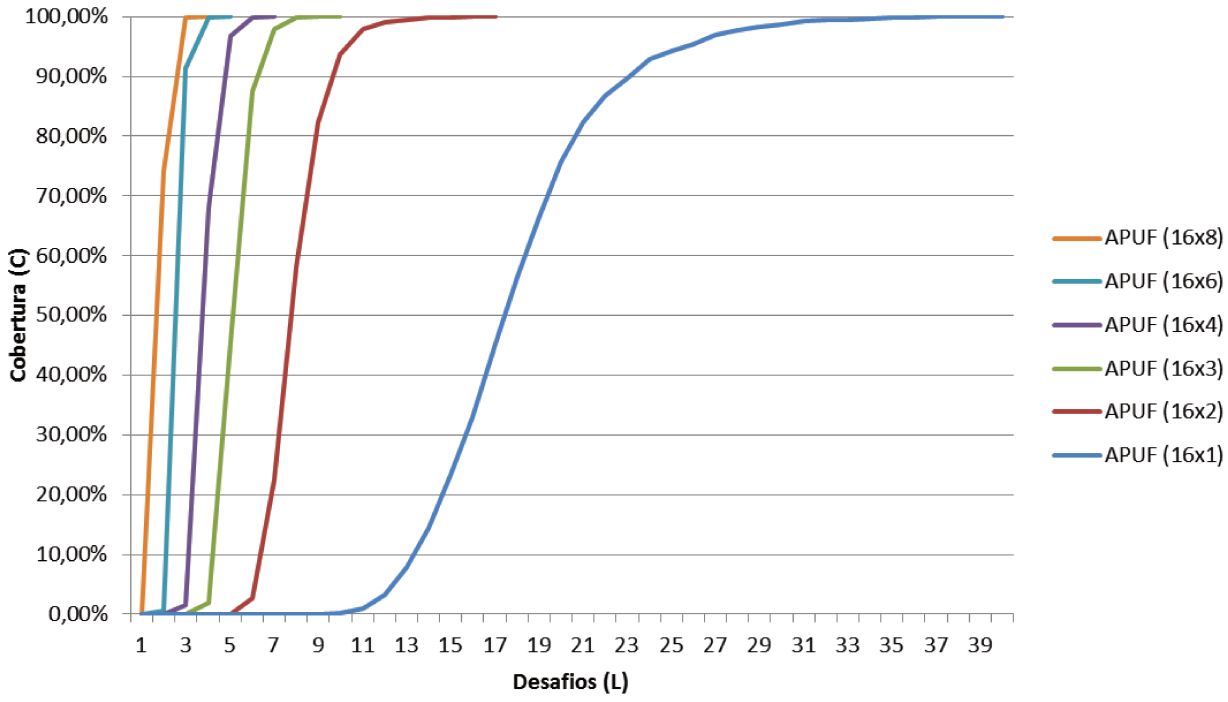


Figura A.2: Cobertura para o APUF(16xN).

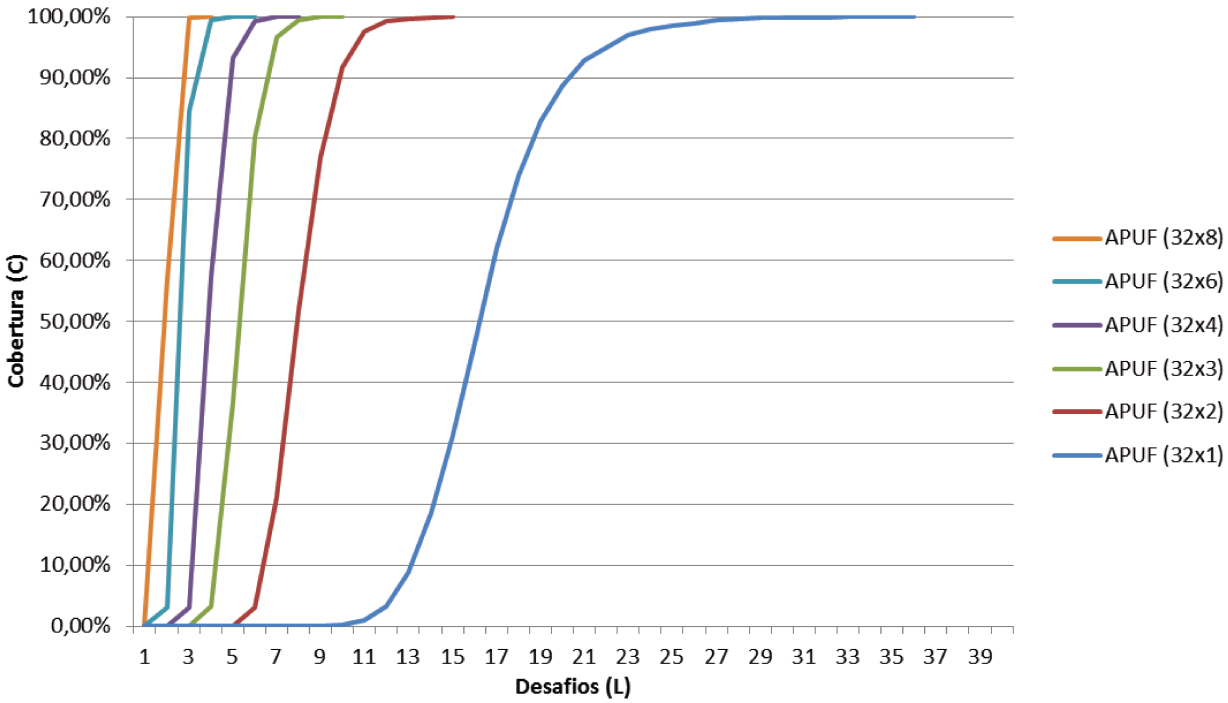


Figura A.3: Cobertura para o APUF(32xN).

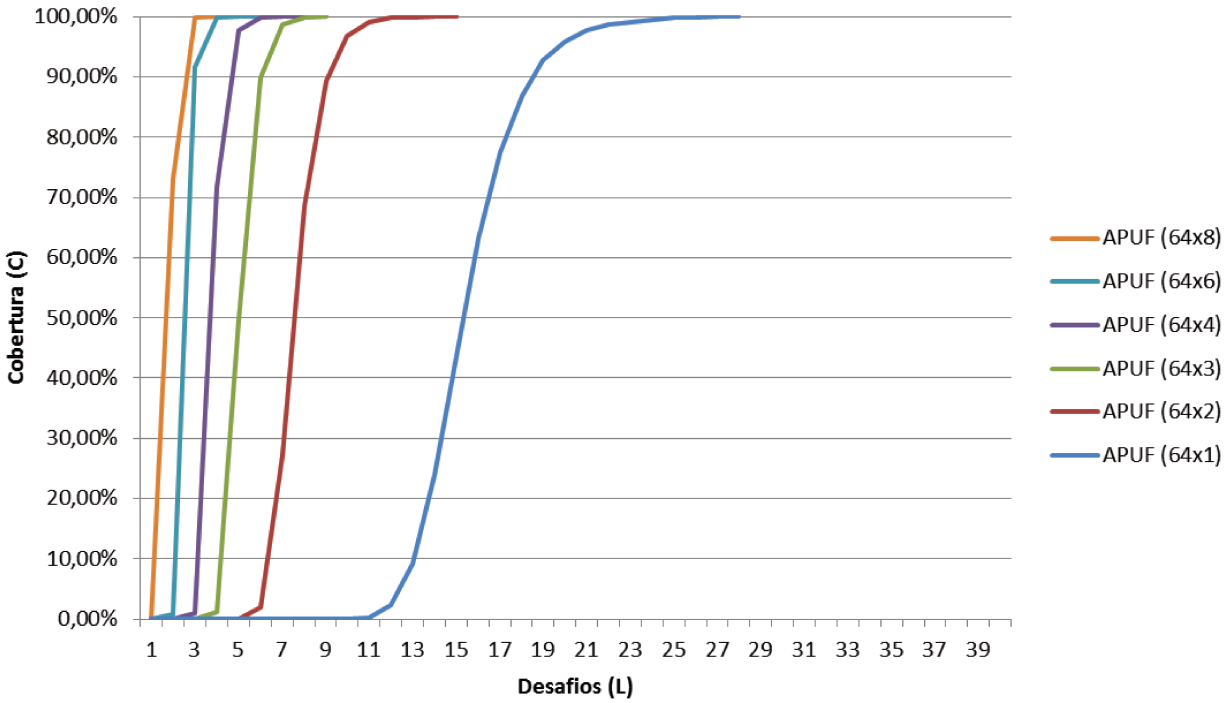


Figura A.4: Cobertura para o APUF(64xN).

Na sequência são apresentados os resultados das simulações do APUF para variações de M.

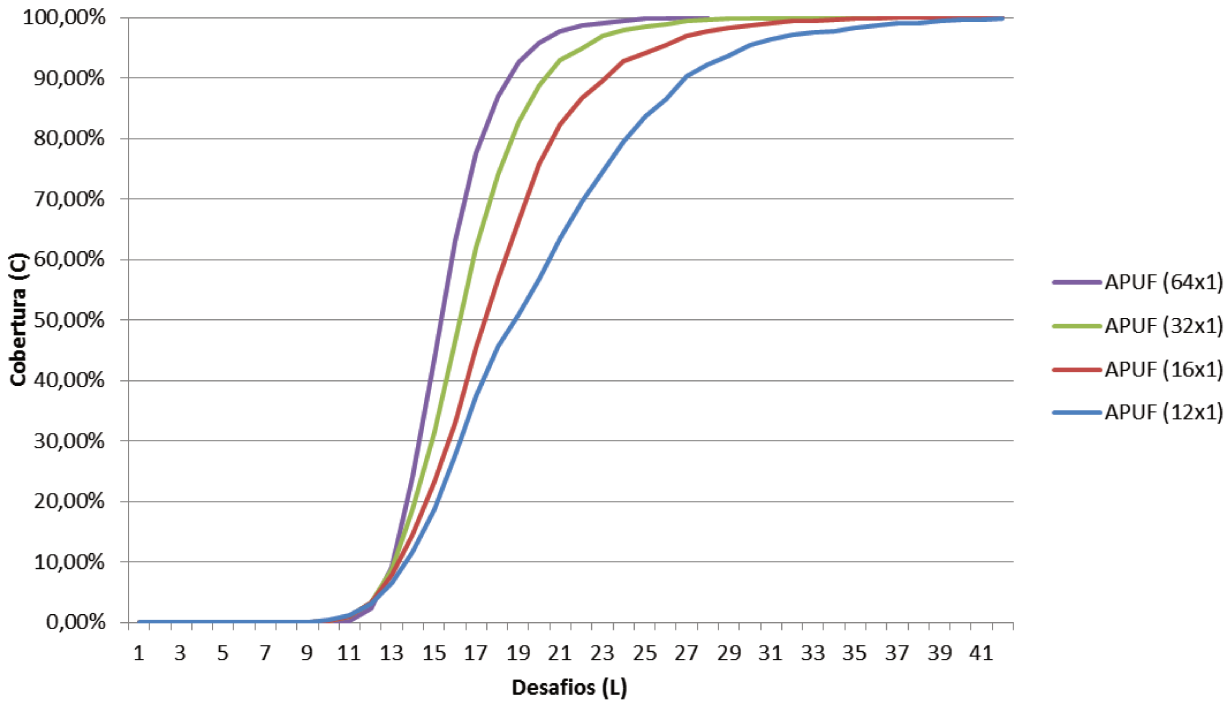


Figura A.5: Cobertura para o APUF(Mx1).

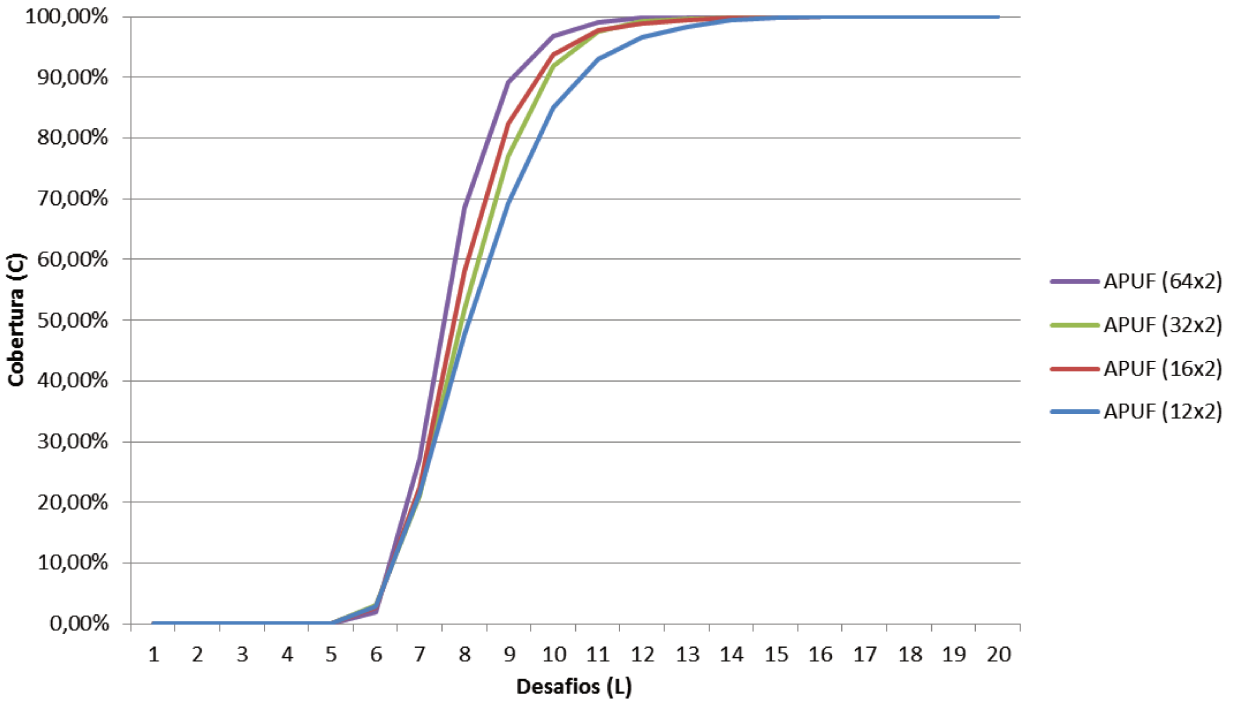


Figura A.6: Cobertura para o APUF(Mx2).

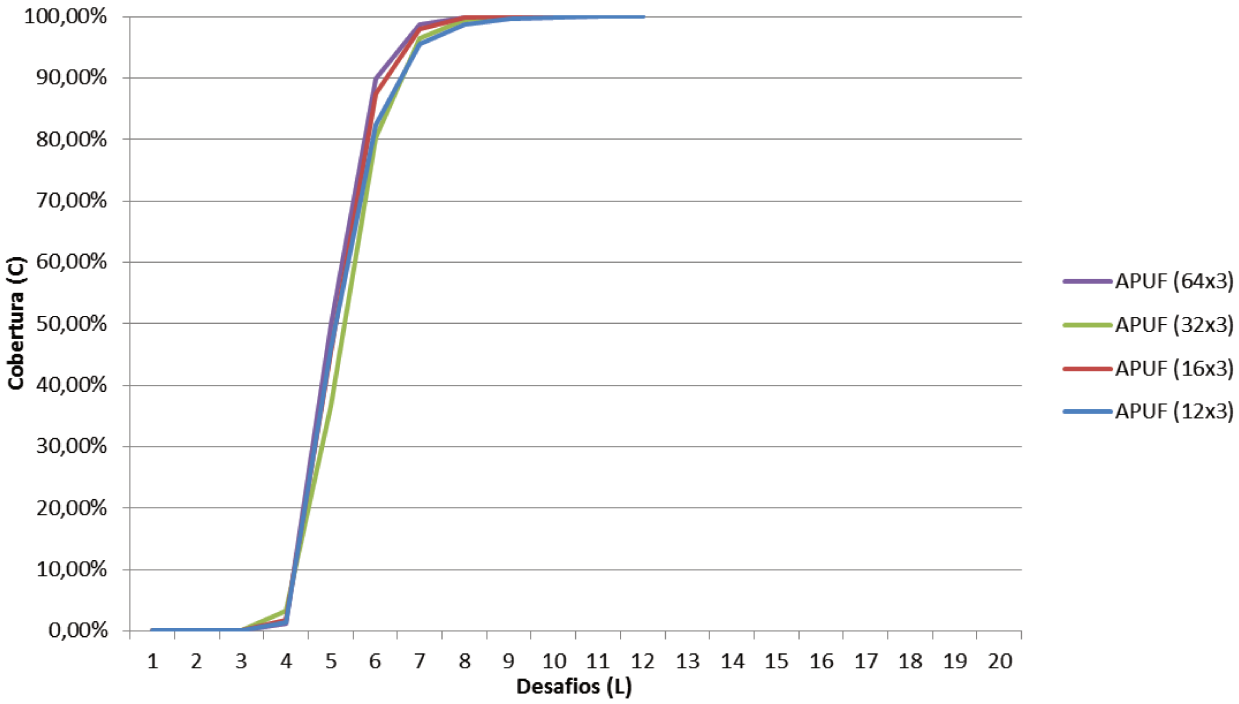


Figura A.7: Cobertura para o APUF(Mx3).

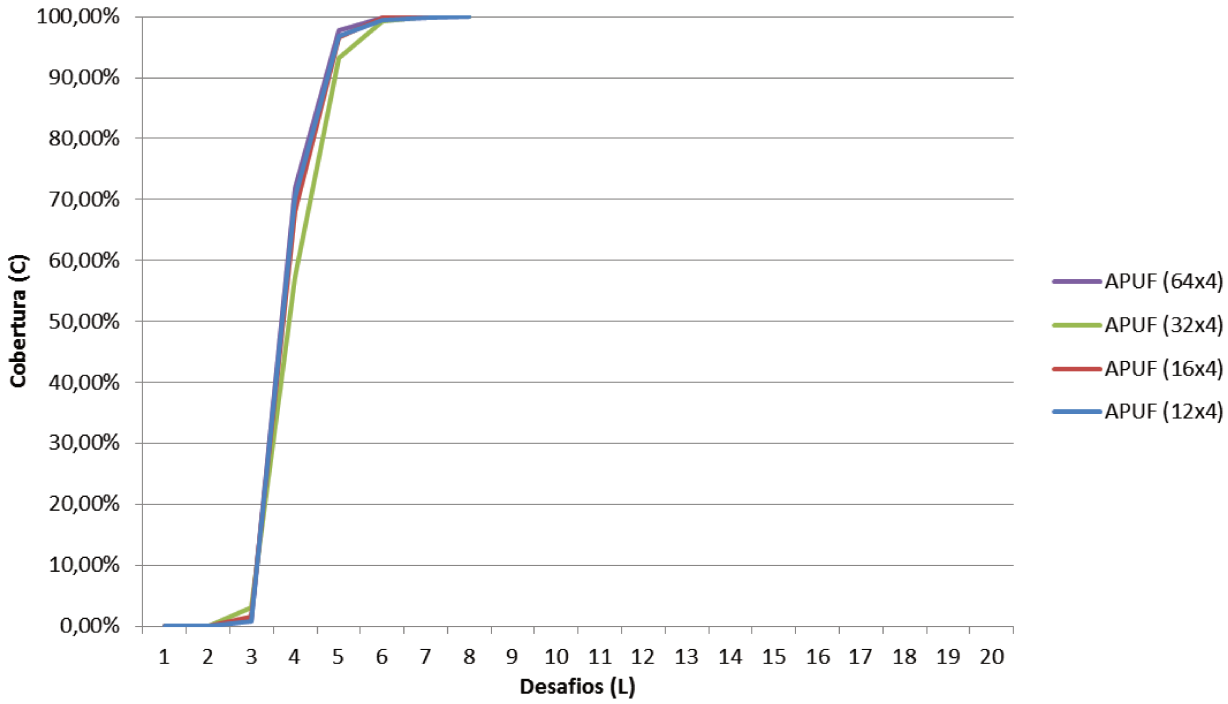


Figura A.8: Cobertura para o APUF(Mx4).

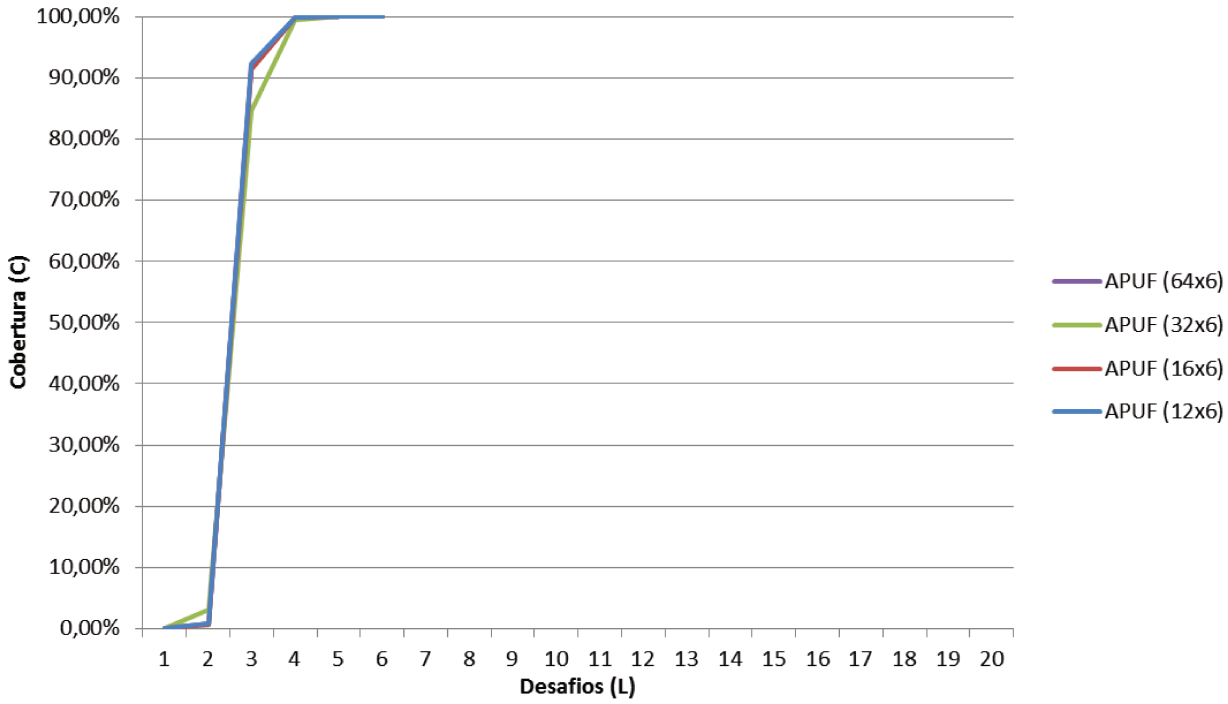


Figura A.9: Cobertura para o APUF(Mx6).

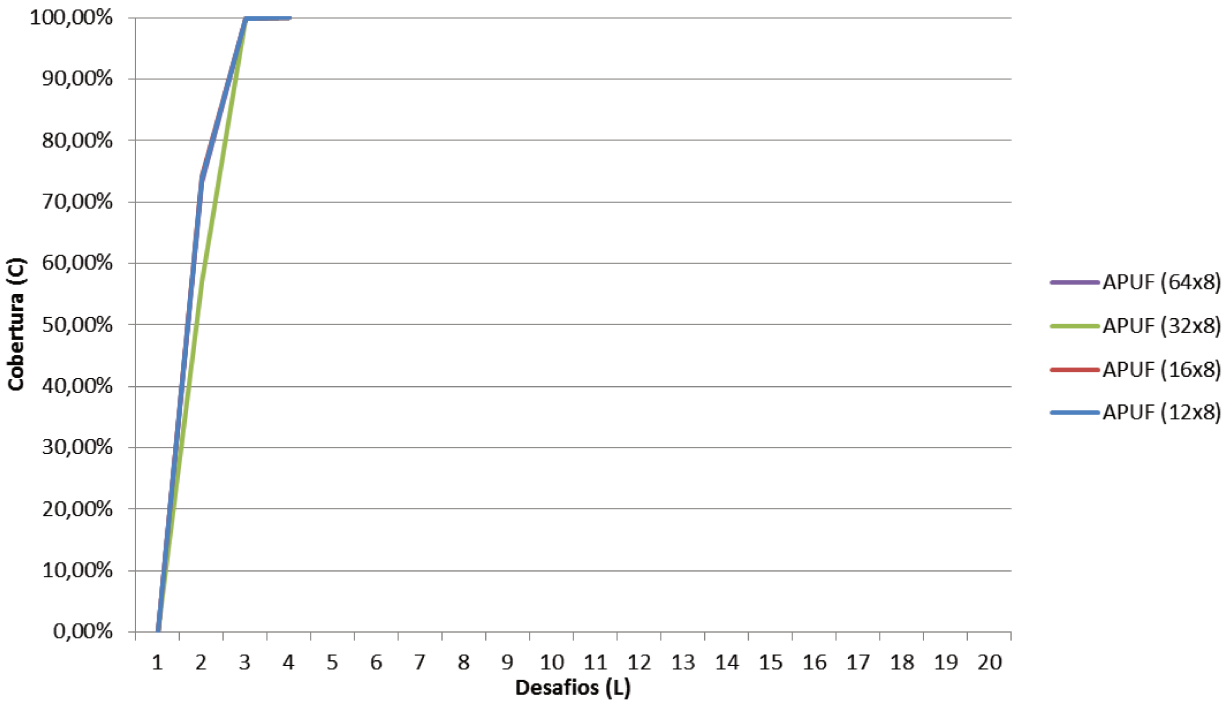


Figura A.10: Cobertura para o APUF(Mx8).

Apêndice B

Avaliação do CRPUF

Os resultados são apresentados em ordem crescente com relação à dimensão destacada de M e N nos gráficos. E em todos os gráficos são utilizados 10000 indivíduos para cálculo da cobertura. Além disso, alguns resultados são apresentados em escala diferentes para destacar a diferença entre cada uma das configurações no mesmo gráfico.

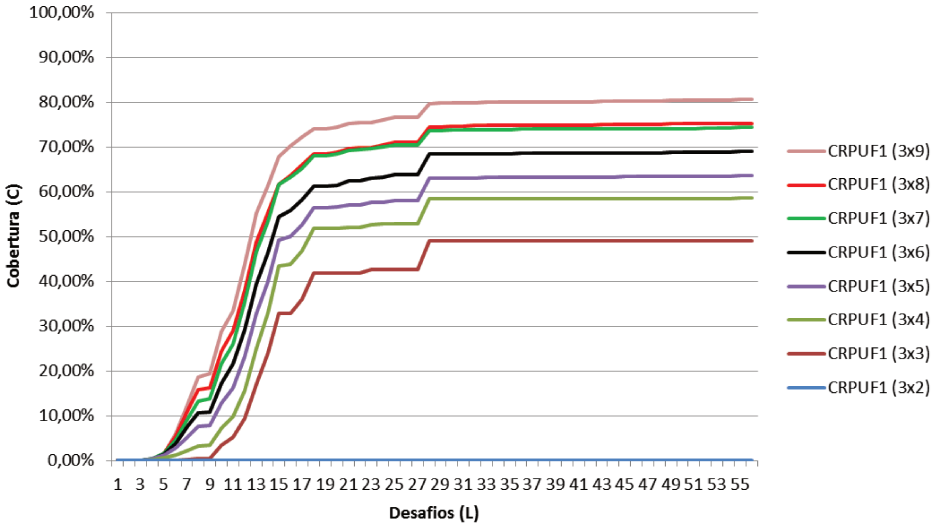


Figura B.1: Cobertura para o $CRPUF_1$ com 3 linhas.

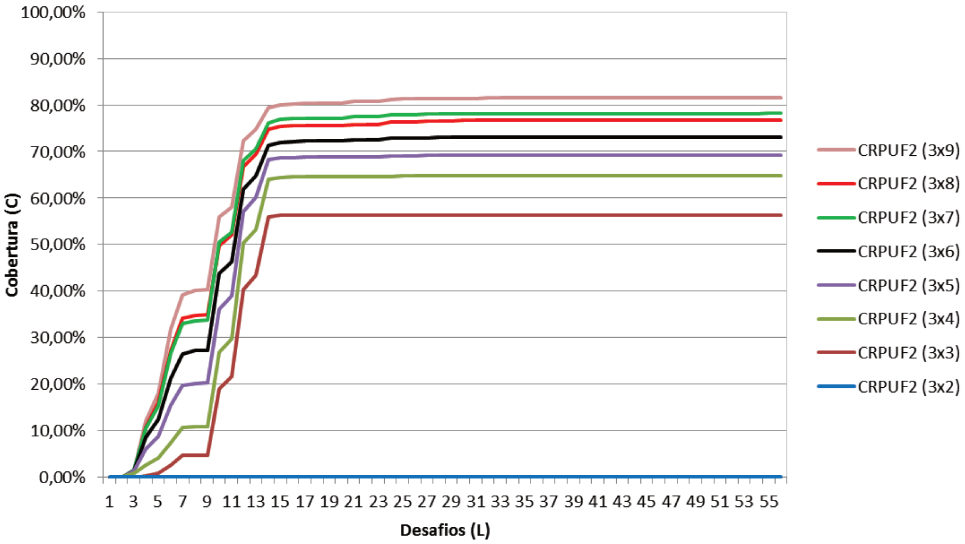


Figura B.2: Cobertura para o $CRPUF_2$ com 3 linhas.

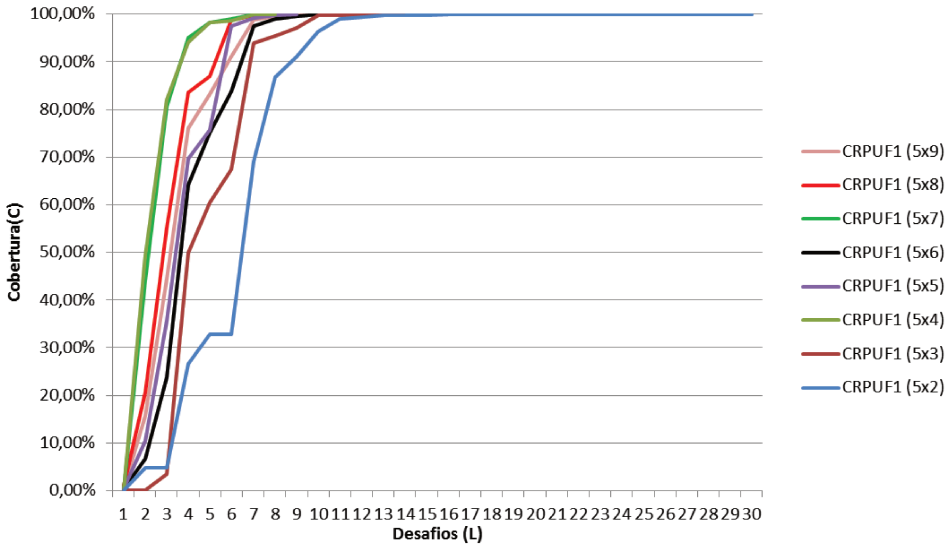


Figura B.3: Cobertura para o $CRPUF_1$ com 5 linhas.

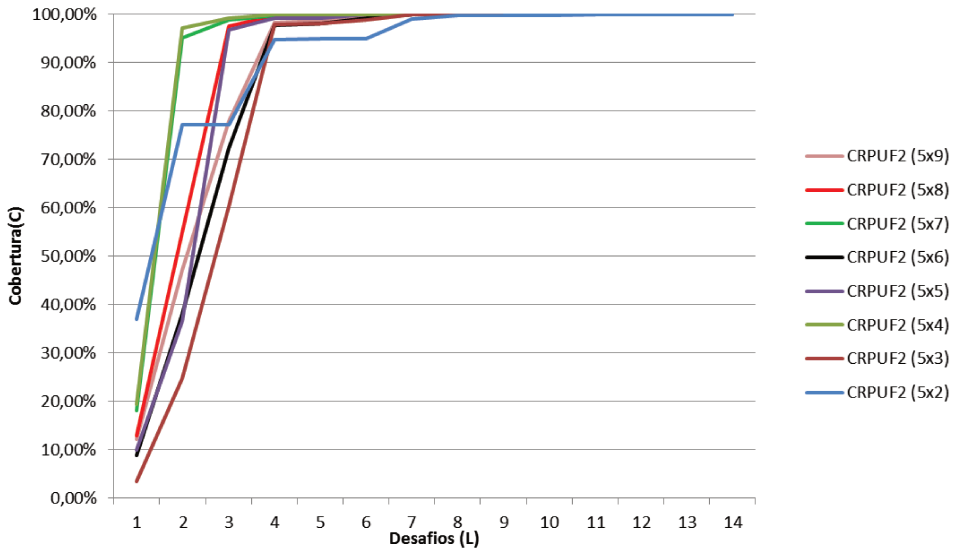


Figura B.4: Cobertura para o $CRPUF_2$ com 5 linhas.

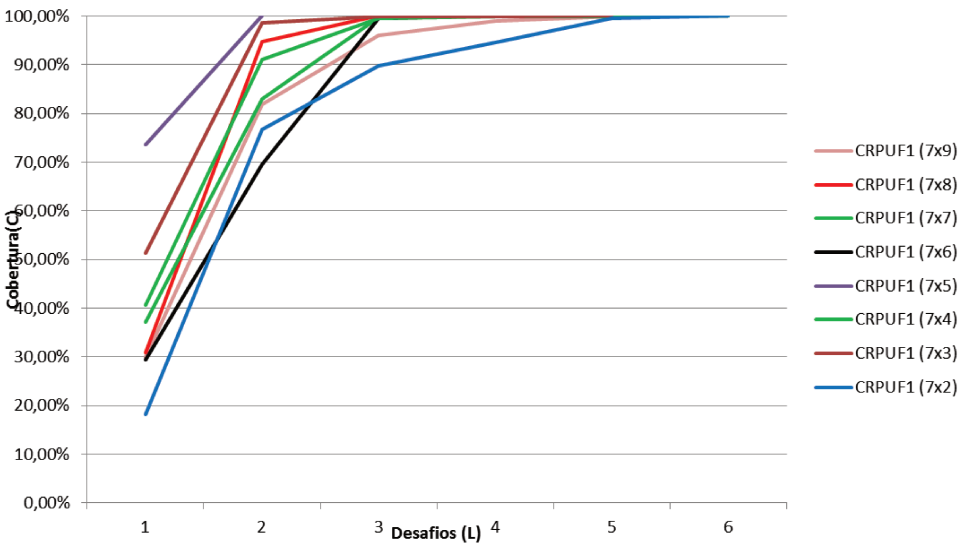


Figura B.5: Cobertura para o $CRPUF_1$ com 7 linhas.

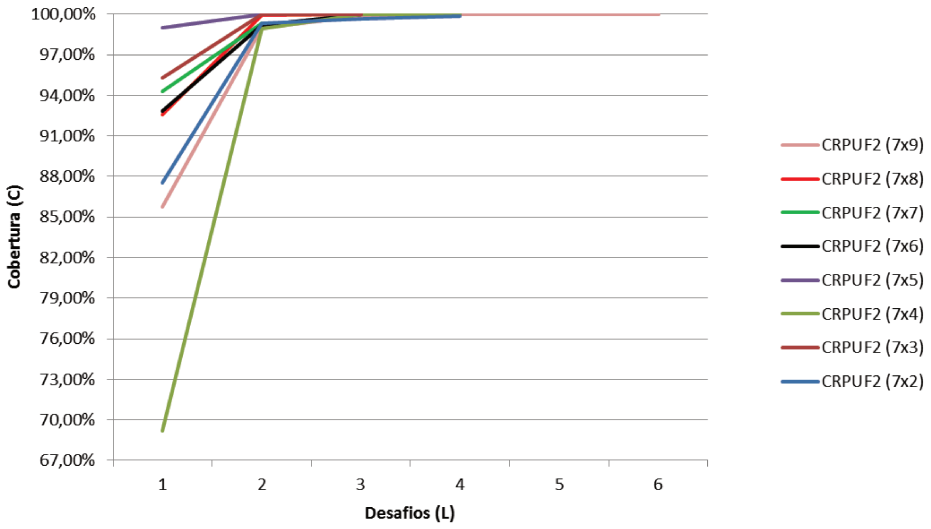


Figura B.6: Cobertura para o $CRPUF_2$ com 7 linhas.

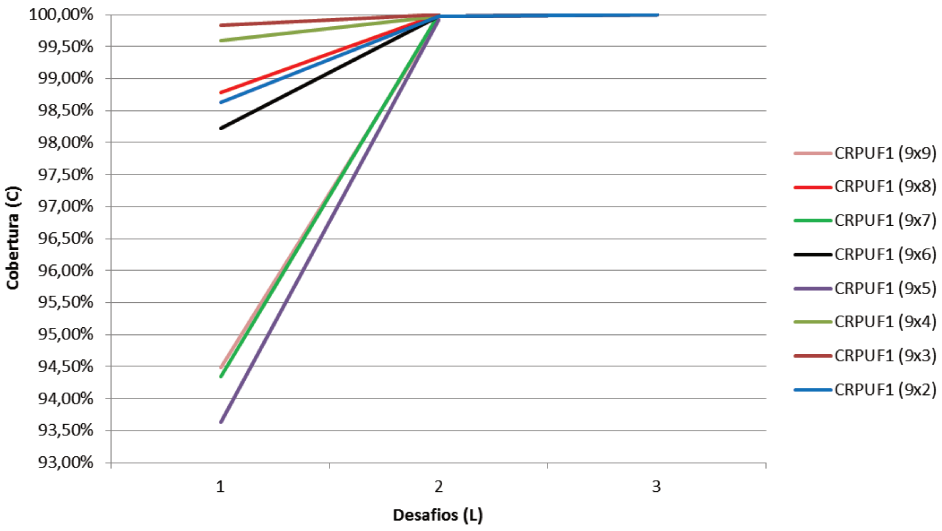


Figura B.7: Cobertura para o $CRPUF_1$ com 9 linhas.

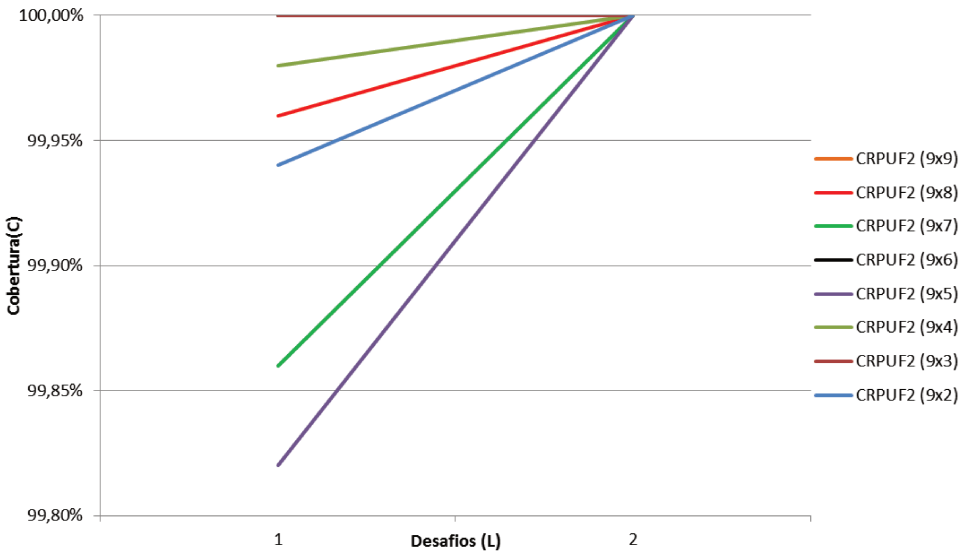


Figura B.8: Cobertura para o $CRPUF_2$ com 9 linhas.

Os resultados a seguir são organizados por números de estágios do CRPUF ($N \in \{2, 3, 4, 5, 6, 7, 8, 9\}$).

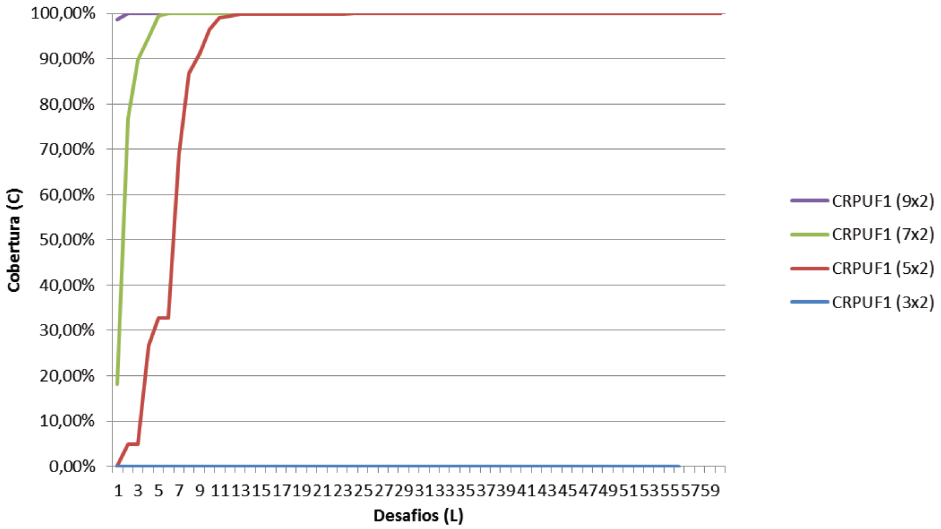


Figura B.9: Cobertura para o $CRPUF_1$ com 2 Estágios.

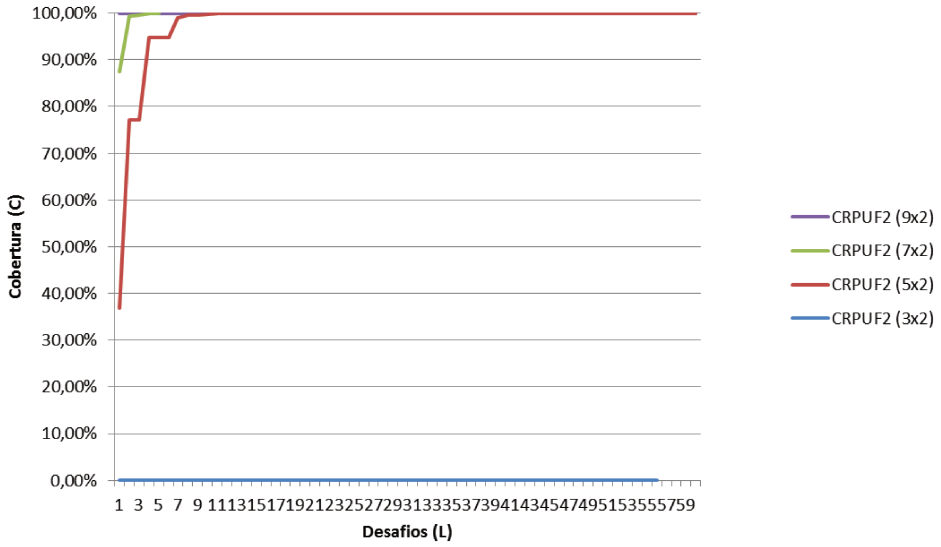


Figura B.10: Cobertura para o $CRPUF_2$ com 2 Estágios.

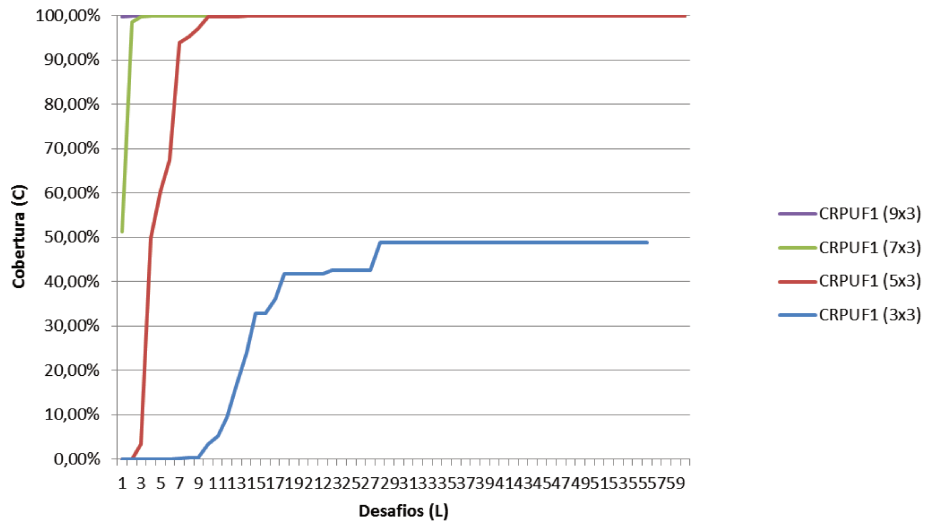


Figura B.11: Cobertura para o $CRPUF_1$ com 3 Estágios.

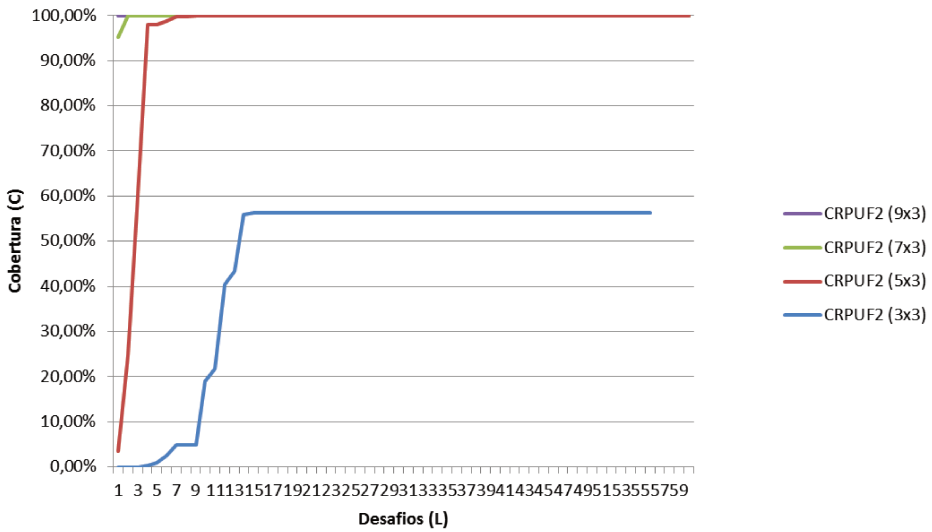


Figura B.12: Cobertura para o $CRPUF_2$ com 3 Estágios.

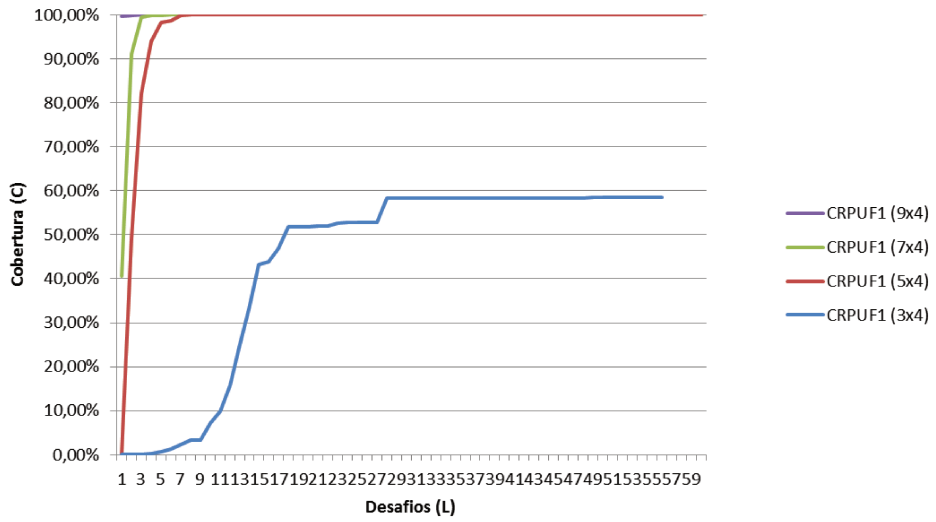


Figura B.13: Cobertura para o $CRPUF_1$ com 4 Estágios.

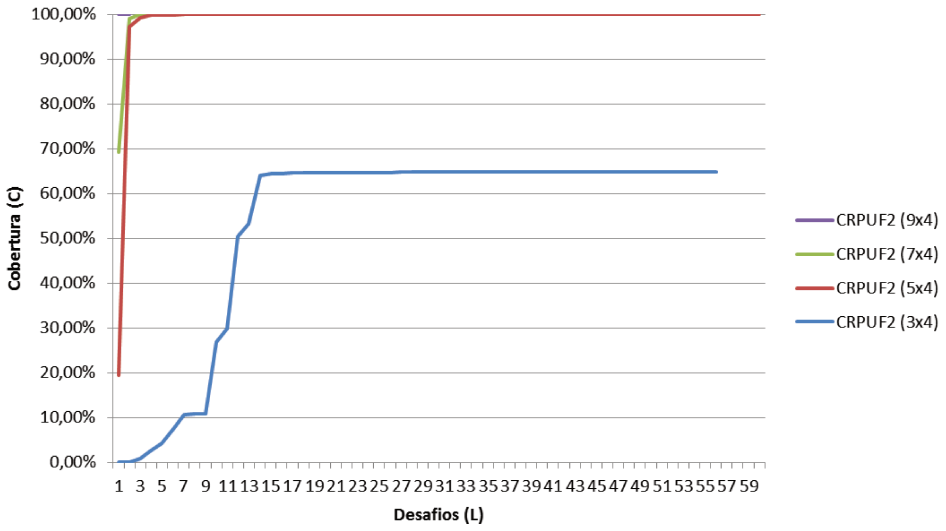


Figura B.14: Cobertura para o $CRPUF_2$ com 4 Estágios.

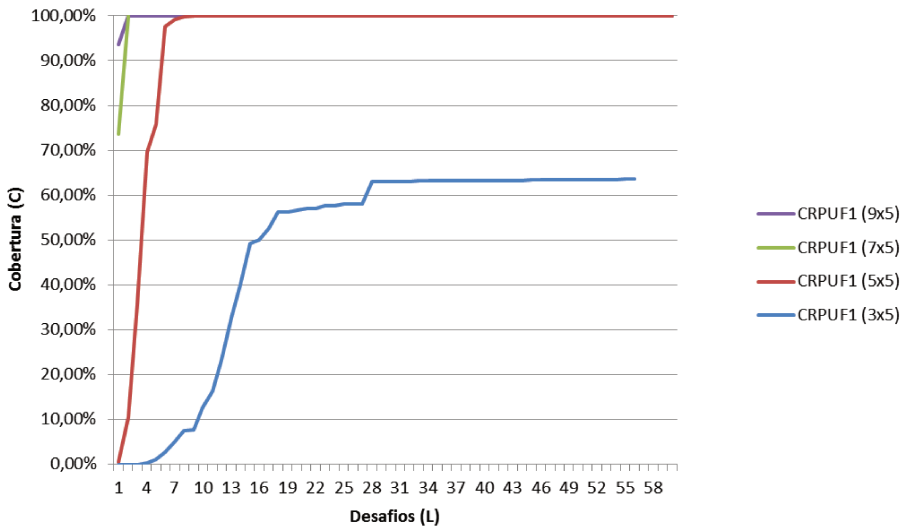


Figura B.15: Cobertura para o $CRPUF_1$ com 5 Estágios.

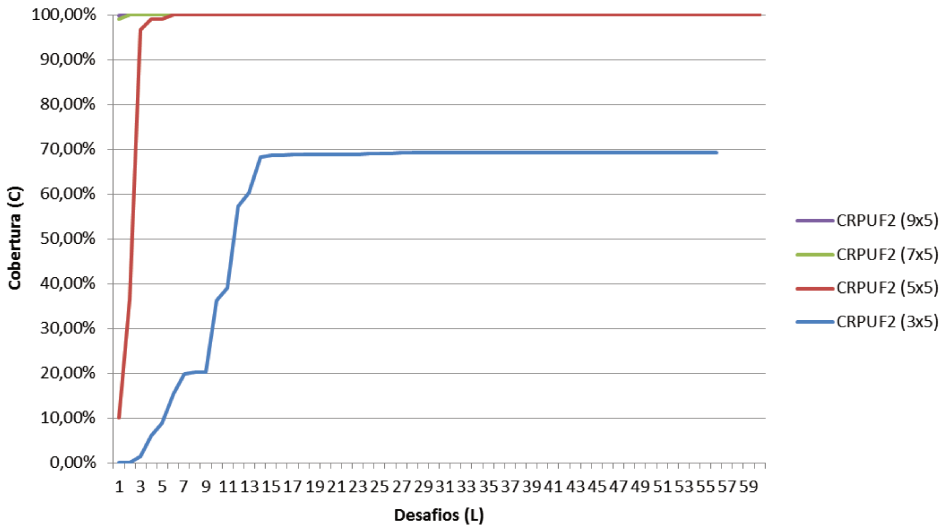


Figura B.16: Cobertura para o $CRPUF_2$ com 5 Estágios.

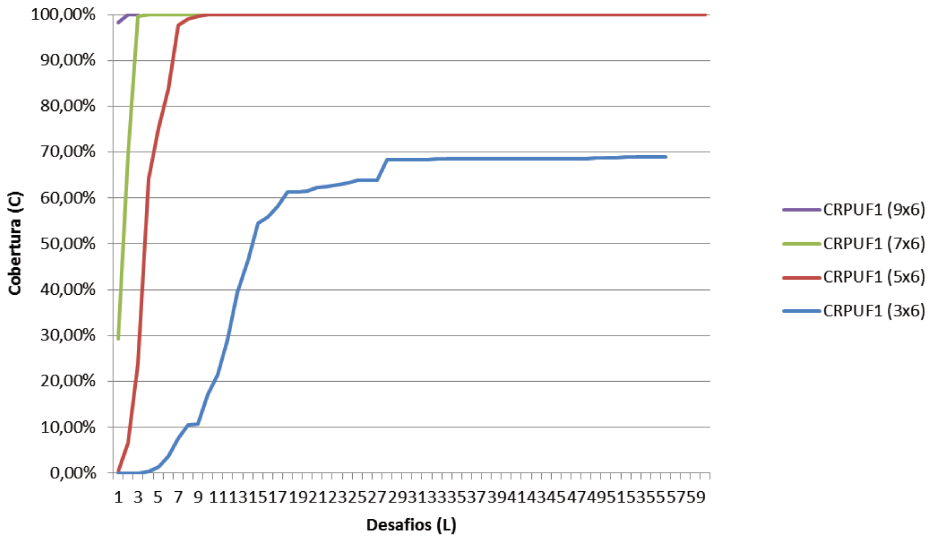


Figura B.17: Cobertura para o $CRPUF_1$ com 6 Estágios.

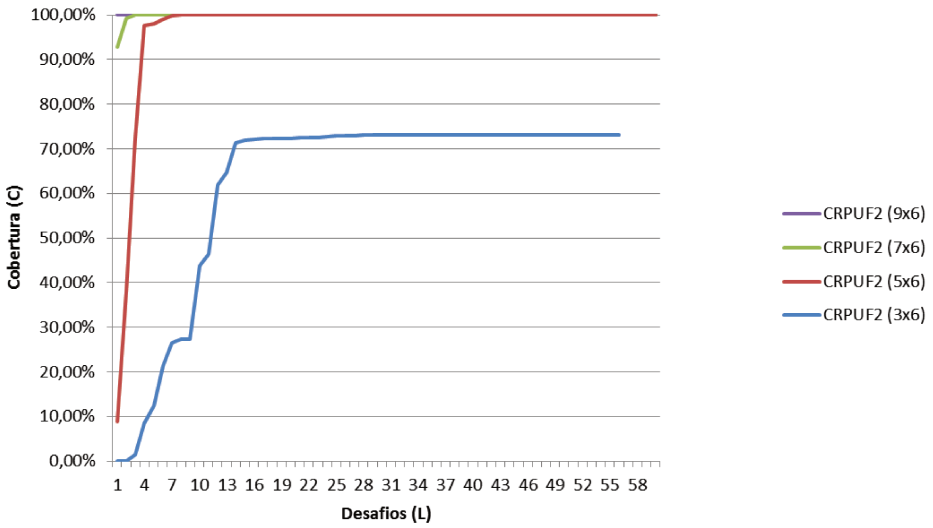


Figura B.18: Cobertura para o $CRPUF_2$ com 6 Estágios.

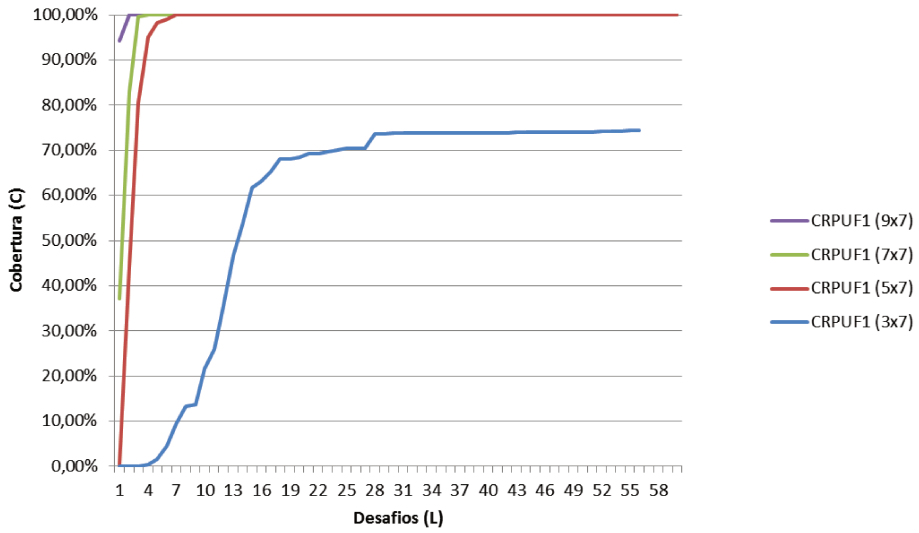


Figura B.19: Cobertura para o $CRPUF_1$ com 7 Estágios.

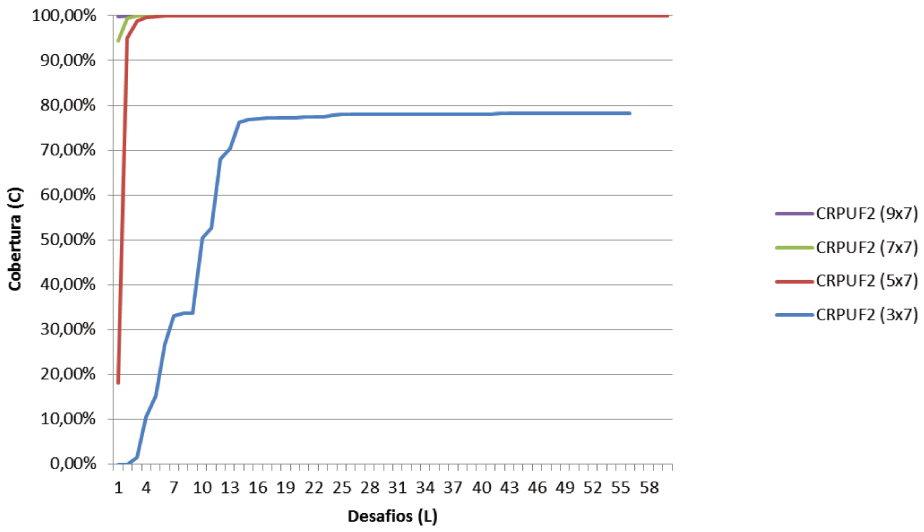


Figura B.20: Cobertura para o $CRPUF_2$ com 7 Estágios.

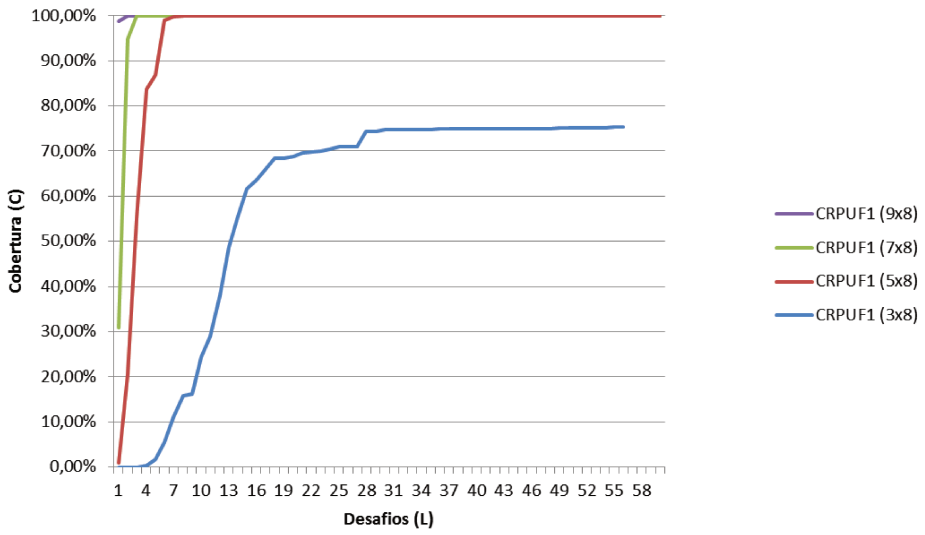


Figura B.21: Cobertura para o $CRPUF_1$ com 8 Estágios.

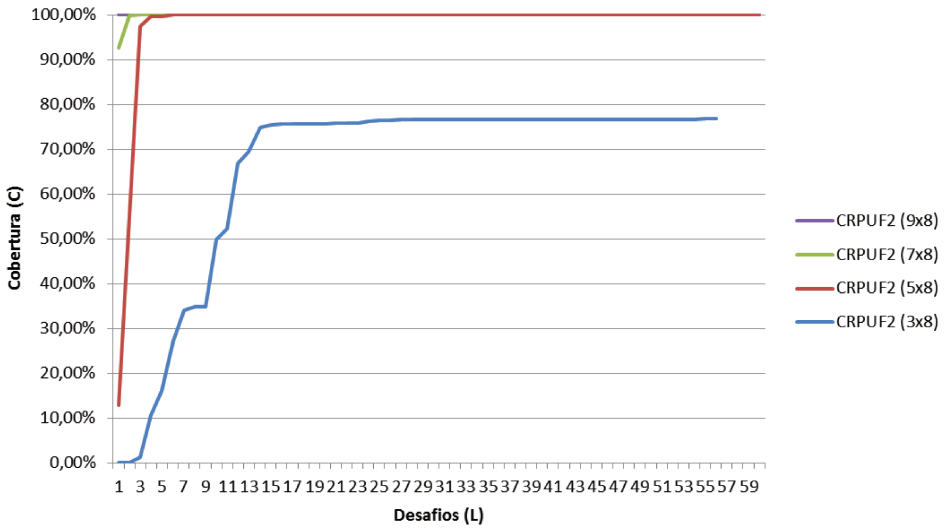


Figura B.22: Cobertura para o $CRPUF_2$ com 8 Estágios.

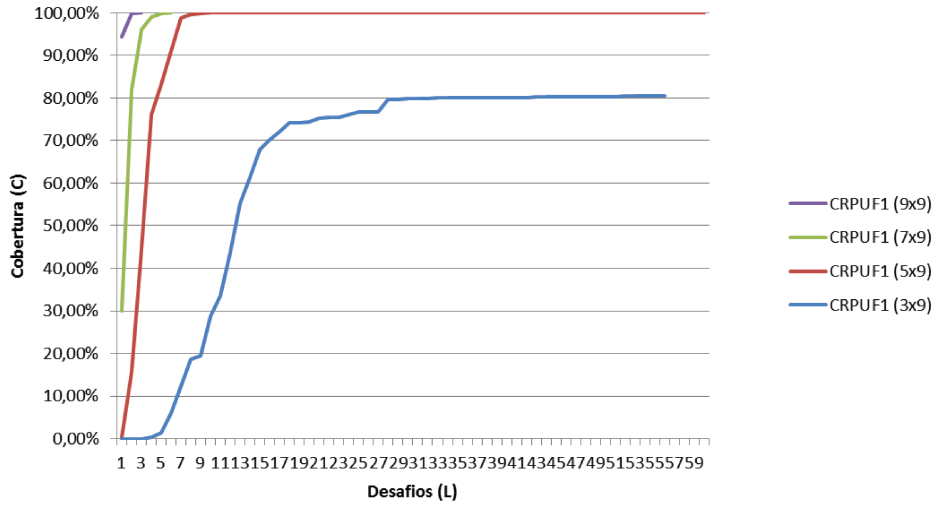


Figura B.23: Cobertura para o $CRPUF_1$ com 9 Estágios.

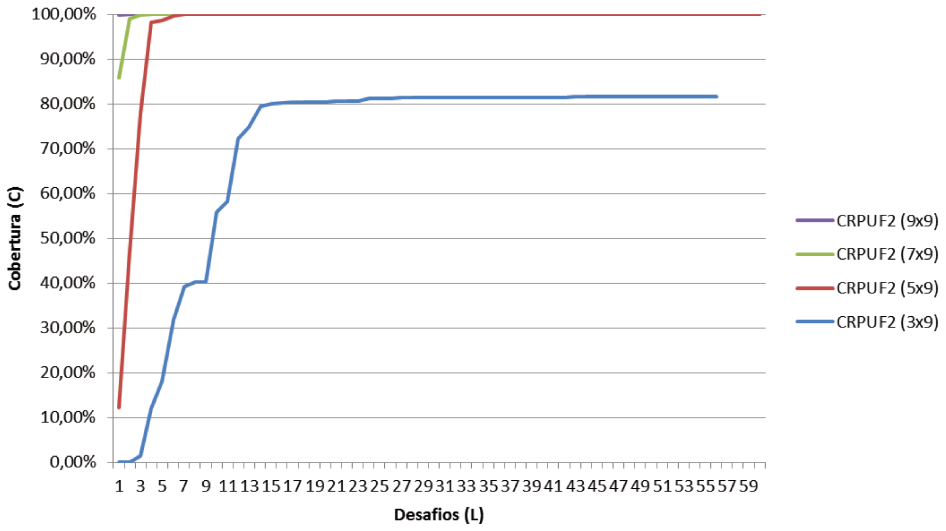


Figura B.24: Cobertura para o $CRPUF_2$ com 9 Estágios.