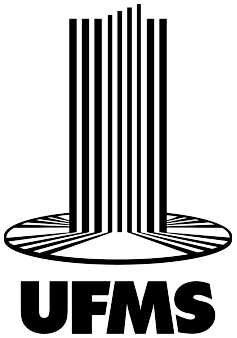
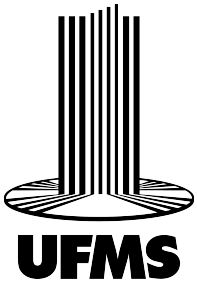


UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
FACULDADE DE ENGENHARIAS, ARQUITETURA E URBANISMO E GEOGRAFIA
CURSO DE ENGENHARIA ELÉTRICA

Função Fisicamente Inclonável como algoritmo de cibersegurança embarcado em FPGA

Lucas Yoshihiro Okano

Campo Grande - MS
8 de Maio de 2025



UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
FACULDADE DE ENGENHARIAS, ARQUITETURA E URBANISMO E GEOGRAFIA
CURSO DE ENGENHARIA ELÉTRICA

Função Fisicamente Inclonável como algoritmo de cibersegurança embarcado em FPGA

Lucas Yoshihiro Okano

Trabalho de Conclusão de Curso apresentado como exigência para obtenção do grau de Bacharel em Engenharia Elétrica da Universidade Federal de Mato Grosso do Sul – UFMS.

Orientador: Prof. Dr. Edson Antonio Batista

Campo Grande - MS
8 de Maio de 2025

Função Fisicamente Inclonável como algoritmo de cibersegurança embarcado em FPGA

Trabalho de Conclusão de Curso apresentado como exigência para obtenção do grau de Bacharel em Engenharia Elétrica da Universidade Federal de Mato Grosso do Sul – UFMS.

Banca Examinadora:

Documento assinado digitalmente
gov.br
EDSON ANTONIO BATISTA
Data: 02/07/2025 18:56:01-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Edson Antonio Batista

Documento assinado digitalmente
gov.br
BRUNO SERENI
Data: 02/07/2025 16:14:35-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Bruno Sereni

Documento assinado digitalmente
gov.br
ALLAN DOUGLAS BENTO DA COSTA
Data: 02/07/2025 16:07:18-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Allan Douglas Bento da Costa

Campo Grande - MS

8 de Maio de 2025

Agradecimentos

Dedico este trabalho aos meus familiares, que me apoiaram ao longo desta longa jornada, aos meus amigos que me auxiliaram nos momentos mais desafiadores e aos meus professores por me guiarem por este caminho.

Resumo

Este trabalho apresenta a implementação de um algoritmo do tipo *Arbiter Physically Unclonable Function* (Arbiter PUF ou APUF) em um sistema embarcado baseado em *Field-Programmable Gate Array* (FPGA). Com o crescimento da *Internet of Things* (IoT) e a ampla incorporação de técnicas de Inteligência Artificial (IA) em diferentes áreas tecnológicas, torna-se cada vez mais necessário desenvolver mecanismos de segurança digital compatíveis com as restrições de desempenho e recursos desses sistemas. Neste contexto, adotou-se como principal referência o artigo [1], que propõe uma implementação de um XOR-APUF em FPGA.

Para o desenvolvimento, foram utilizados os softwares **Quartus II 13.0** e **ModelSim**, além do kit de desenvolvimento **Altera DE2**, o que permitiu a realização de testes práticos e simulações do funcionamento do algoritmo. Durante a implementação, foi necessário adaptar diversas seções do material de referência, devido a incompatibilidades entre o ambiente descrito no artigo e os recursos específicos dos softwares e hardwares utilizados.

Este trabalho também descreve detalhadamente o funcionamento do APUF em FPGA, servindo como material didático e técnico para interessados na área. Como contribuição principal, busca-se demonstrar o potencial de uso de FPGAs como plataforma para protocolos de segurança baseados em PUFs em aplicações IoT, além de promover a formação de recursos humanos qualificados e fomentar o desenvolvimento de novos projetos de pesquisa na área de cibersegurança embarcada.

Palavras-chave: Função Físicamente Inclonável, APUF, FPGA, Cybersegurança.

Abstract

This work presents the implementation of an Arbiter Physically Unclonable Function (Arbiter PUF or APUF) algorithm on a system based on a Field-Programmable Gate Array (FPGA). With the rise of the Internet of Things (IoT) and the widespread adoption of Artificial Intelligence (AI) across various technological domains, there is an increasing demand for digital security mechanisms tailored to the performance and resource constraints of such systems. In this context, the project was primarily based on the methodology described in [1], which outlines an implementation of an XOR-APUF on FPGA.

The development was carried out using **Quartus II 13.0** and **ModelSim** software tools, along with the **Altera DE2** development board, enabling both simulation and practical validation of the proposed algorithm. During the implementation process, several adaptations to the original reference were required due to hardware and software differences that prevented direct replication.

This work provides a detailed explanation of the operation of an APUF embedded in FPGA, serving as a technical and educational reference for researchers and developers. As a key contribution, the project demonstrates the feasibility of using FPGAs to deploy PUF-based security protocols in IoT applications, while also contributing to the training of qualified personnel and promoting research initiatives in the field of embedded cybersecurity.

Keywords: Physical Unclonable Function, APUF, FPGA, Cybersecurity.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

PUF	Physical Unclonable Function (Função Fisicamente Inclonável)
APUF	Arbiter PUF
XOR-APUF	Exclusive-OR Arbiter PUF
QPUF	Quantum Physical Unclonable Function (Funções Físicas Inclonáveis Quânticas)
FPGA	Field Programmable Gate Array (Arranjo de Portas Programáveis em Campo)
IoT	Internet of Things (Internet das Coisas)
QIoT	Quantum Internet of Things (Internet Quântica das Coisas)
IIoT	Industrial Internet of Things(Internet das Coisas Industrial)
IA	Inteligência Artificial
EEPROM	Electrically-Erasable Programmable Read-Only Memory (Memória Somente Leitura Programável e Apagável Eletricamente)
CFTV	Circuito Fechado de Televisão
SoC	System On Chip
CAD	Computer-Aided Design (Desenho Assistido por Computador)
CC	Corrente Contínua
OBC	On-Board Charger
MOSFET	Metal Oxide Semiconductor Field Effect Transistor (Transistor de Efeito de Campo Semicondutor de Óxido Metálico)
FIR	Finite Impulse Response
DPA	Differential Power Analysis
CRP	Challenge Response Pairs
ML	Machine Learning (Aprendizado de Máquina)
LUT	Look-Up Tables

PDL	Programmable Delay Lines
TVM	Temporal Majority Voting
LR	Logistic Regression (Regressão Logística)
CMA-ES	Covariance Matrix Adaptation Evolution Strategy (Estratégia Evolutiva com Adaptação de Matriz de Covariância)
SVM	Support Vector Machine (Máquinas de Vetores de Suporte)
ANN	Artificial Neural Network (Rede Neural Artificial)

Sumário

1 Introdução

1.1 Contextualização

O mercado de *Field-Programmable Gate Array* (FPGA) tem ganhado atenção no cenário tecnológico, sendo esperado que sua receita alcance 11 bilhões de dólares até 2025[3]. Um grande impulsionador desta tecnologia é a popularização do conceito de *Internet of Things* (IoT), que consiste em conectar diversos dispositivos a uma rede ou nuvem, de forma que os dispositivos possam se comunicar entre si, a fim de observar, controlar, automatizar ou facilitar algum processo. Os FPGAs possuem algumas propriedades que lhes dão vantagens no cenário IoT, como o baixo custo no desenvolvimento de algoritmos e a rapidez com que os produtos chegam ao mercado. Por isso, o FPGA está presente em várias aplicações de múltiplas áreas, como sensoriamento, sistemas militares e aeroespaciais, controle industrial, em automóveis e eletrodomésticos.

No entanto, também há algumas desvantagens. Com algumas exceções, a maioria dos FPGAs possui memória volátil, ou seja, perde sua programação quando desenergizados, sendo necessário transferir de volta seus dados a cada inicialização. Portanto, os dados da programação precisam ser salvos em uma memória não volátil separada, geralmente em uma EEPROM ou memória flash acoplada na mesma placa de circuito do FPGA.

Isto os torna vulneráveis à clonagem, pois os dados da programação podem ser facilmente copiados para outro dispositivo. Existem técnicas que visam combater tal vulnerabilidade, como criptografar os dados da programação, mas esta solução também apresenta problemas, pois traz a necessidade de armazenar uma chave digital para descriptografar em algum lugar da placa, aumentando o tamanho e custo geral do produto, além de não solucionar de vez o problema original, pois este método também tem suas próprias vulnerabilidades que precisam ser consideradas em sua implementação.

Este trabalho propõe o uso de PUFs (Physical Unclonable Functions) embarcados em um FPGA, como uma alternativa capaz de identificar e autenticar o dispositivo, enquanto inibe a possibilidade de clonagem. PUFs são circuitos de hardware cuja operação não pode ser replicada de maneira idêntica, mesmo que se conheça o algoritmo e o projeto do hardware. Tais dispositivos utilizam pequenas variações incontroláveis que ocorrem na fabricação de um chip semicondutor, junto a um algoritmo capaz de identificar essas variações. Dessa forma, é possível garantir que o algoritmo esteja sendo executado no dispositivo original, sem depender de chaves criptográficas.

1.2 Objetivos

1.2.1 Objetivo Geral

Este trabalho tem como objetivo principal estudar e implementar um algoritmo de segurança baseado em Physically Unclonable Functions (PUFs), especificamente do tipo Arbiter PUF (APUF), em uma plataforma embarcada com FPGA. O foco está na análise da viabilidade técnica de se utilizar esse tipo de abordagem como alternativa segura para autenticação e proteção de dispositivos IoT.

Entre os objetivos específicos, destacam-se:

1.2.2 Objetivos Específicos

- Investigar os fundamentos teóricos das Physically Unclonable Functions (PUFs) e sua aplicação em protocolos de segurança digital.
- Analisar os principais desafios de segurança em dispositivos IoT e o papel dos PUFs como mecanismo de autenticação leve e confiável.
- Implementar uma versão do algoritmo XOR-APUF em uma plataforma FPGA, utilizando ferramentas como Quartus II e ModelSim.
- Realizar simulações e testes práticos utilizando a placa Altera DE2 para validar a funcionalidade e robustez do algoritmo embarcado.
- Adaptar e documentar o processo de implementação do APUF considerando as limitações de hardware e software encontradas durante o desenvolvimento.
- Avaliar o potencial de integração de PUFs como elemento de segurança em sistemas embarcados aplicados à IoT, considerando aspectos de desempenho, escalabilidade e confiabilidade.

1.3 Organização do Trabalho

- Capítulo 1 – Introdução: Apresenta o contexto geral da pesquisa, destacando a relevância do uso de Physically Unclonable Functions (PUFs) como mecanismos de segurança em sistemas embarcados para IoT. Define o problema, os objetivos (geral e específicos), a justificativa e a metodologia empregada, além da organização do documento.
- Capítulo 2 – Fundamentação Teórica: Realiza uma revisão bibliográfica abrangente sobre segurança em dispositivos IoT, abordando os principais riscos, requisitos de

proteção e limitações técnicas. São apresentados os conceitos fundamentais de PUFs, especialmente do tipo Arbiter PUF (APUF), suas aplicações em autenticação e geração de chaves, bem como a viabilidade de sua implementação em FPGAs. Também são descritas as ferramentas utilizadas no projeto, como Quartus II, ModelSim e o kit Altera DE2.

- Capítulo 3 – Metodologia de Implementação: Detalha o processo de desenvolvimento do sistema proposto, incluindo a descrição do algoritmo XOR-APUF, a arquitetura de hardware desenvolvida, os módulos VHDL utilizados, a estrutura de testes simulados e os critérios adotados para validação do sistema. São discutidas ainda as adaptações necessárias em relação ao artigo de referência, considerando as especificidades do ambiente de desenvolvimento utilizado.

- Capítulo 4 – Resultados e Discussões: Apresenta os resultados obtidos com a implementação prática do APUF na plataforma FPGA, incluindo diagramas, cronogramas de sinais, resultados de simulações e testes funcionais realizados com a placa Altera DE2. São discutidos os principais desafios enfrentados, limitações encontradas e a eficácia da solução quanto à autenticidade e à aleatoriedade das respostas geradas pelo PUF.

- Capítulo 5 – Conclusão e Trabalhos Futuros: Resume os principais achados do trabalho, avaliando o cumprimento dos objetivos propostos.

2 Revisão Bibliográfica

2.1 Fundamentação Teórica

O crescente mercado de dispositivos de *Internet of Things* (IoT) tem gerado novas modalidades de comunicação e interação com o ambiente, permitindo tomada de decisão remota e contribuindo para a melhoria da qualidade de vida[4].

Assim como toda inovação tecnológica, o mercado de dispositivos IoT traz diversos desafios na implementação e potenciais riscos associados. As aplicações de IoT envolvem diferentes áreas, como telecomunicação, processamento de sinais, sensoriamento, segurança de dados, entre outras. Neste aspecto, pode-se imaginar que uma falha de segurança pode trazer consequências desastrosas, como perdas financeiras ou até mesmo riscos à segurança pública. Por esse motivo, várias linhas de pesquisa referentes à segurança de sistemas IoT têm sido desenvolvidas, a fim de prever e mitigar tais riscos.

O artigo [5], publicado em julho de 2021, aborda a necessidade do uso de ferramentas que permitam a criptografia e autenticação de dados em aplicações de IoT, citando alguns ataques em redes *smart city* já documentados, que tiveram grandes repercussões.

Em seguida, propõe o uso do método de criptografia Ascon (método selecionado pelo Instituto Nacional de Padrões e Tecnologia dos EUA como o padrão para criptografias leves) implementado em FPGA, considerando as limitações e necessidades de diversas aplicações. Por exemplo, aplicações que exigem processamento de vídeo em tempo real, como câmeras de segurança, priorizam a velocidade no processo de criptografia; já dispositivos móveis tendem a priorizar a economia de energia.

Considerando esses fatores, foi realizada a análise da eficiência de diferentes formas de implementação do Ascon, comparando-as quanto à área utilizada, velocidade de processamento, relação entre área e desempenho, potência elétrica consumida e energia gasta por bit processado.

Com base nos resultados obtidos, o artigo conclui com sugestões sobre quais projetos seriam mais adequados para aplicações como Circuito Fechado de Televisão (CFTV), tráfego inteligente de veículos, gerenciamento de resíduos, monitoramento de consumo elétrico, entre outros. No entanto, o estudo não considera eventuais vulnerabilidades nos projetos de FPGA ou na própria criptografia Ascon, concentrando-se principalmente em sua aplicabilidade prática.

Algumas vulnerabilidades dos FPGAs foram discutidas no artigo [6], publicado em agosto de 2019, que aborda o sistema de segurança ARM TrustZone, utilizado em SoCs

baseados em FPGA. O TrustZone é uma extensão de hardware que divide os recursos, tanto de software quanto de hardware, em dois ambientes distintos: um seguro e outro não seguro. Apesar de oferecer certo nível de proteção, ainda existem diversas formas de atacar os FPGAs.

Ao explorar vulnerabilidades intrínsecas desses sistemas, o artigo cita diversos estudos nos quais pesquisadores conseguiram burlar suas proteções. O funcionamento da tecnologia ARM TrustZone é explicado brevemente, com o objetivo de fornecer o embasamento teórico necessário para compreender como cada ataque foi realizado.

O artigo também apresenta formas de interferência no TrustZone que podem ser exploradas em ataques de negação de serviço e escalonamento de privilégios, além de descrever como essas ações podem ocorrer. Por exemplo, isso pode acontecer caso as ferramentas de Desenho Assistido por Computador (CAD), utilizadas no desenvolvimento dos SoCs, estejam comprometidas por malware ou manipuladas por um desenvolvedor com intenções maliciosas.

Outro tipo de ataque abordado ocorre por meio do acesso direto à memória externa; método que permite ignorar quase todas as medidas de segurança, dando ao atacante a capacidade de ler dados sigilosos, roubar chaves criptográficas e instalar malwares. Por fim, são mencionadas algumas estratégias para evitar essas falhas de segurança, como a metodologia IDF, que permite isolar a operação de certos componentes no FPGA; os controladores TrustZone, que detectam violações das regras de segurança; e a verificação, por parte do CAD, que compara o código-fonte em VHDL com as conexões internas dos componentes do SoC. Conclui-se que o sistema de segurança ARM TrustZone apresenta diversas vulnerabilidades, por isso, é fundamental priorizar aspectos de proteção desde o projeto dos SoCs e manter atenção constante ao cenário atual de ameaças.

O artigo [7], publicado em dezembro de 2020, analisa os impactos de ciberataques em equipamentos de eletrônica de potência, presentes em carros elétricos e seus carregadores. Os cenários testados incluem interferência no controlador FPGA do carregador, estabelecimento de comunicação falsa entre o carregador e outras unidades de controle e interferência no sistema de gerenciamento das baterias. Esses cenários foram simulados no ambiente MATLAB/Simulink, e o sistema de carregamento testado utiliza um conversor CC-CC de alta frequência com potência de 6,6 kW.

O artigo apresenta alguns casos relatados de ciberataques em carros elétricos e discute os perigos associados à falta de segurança em redes de comunicação implementadas em sistemas de IoT. Além disso, explica brevemente o funcionamento de um sistema de carregamento de carros elétricos, incluindo os sensores e suas conexões. Em seguida, são mostradas diversas topologias de conversores utilizados em *On-Board Chargers* (OBCs), assim como seus benefícios e limitações, e os esquemas de controle implementados no Simulink. Com isso, são obtidas as formas de onda de tensão e corrente do conversor em

operação normal, além de outros dados como eficiência e fator de potência.

Posteriormente no artigo, é citada maneiras que um agente malicioso possa interferir no OBC, caso obtenha acesso aos dados dos controladores ou sensores, entre elas a modificação de sinais (em que se altera o ganho), a interferência (poluindo os dados gerados pelos sensores com ruído), a interrupção (remoção do sinal) e a interceptação (obtenção de dados por parte de um terceiro); e com base nisso, foram modelados os ataques para a simulação.

Os ataques testados, se utilizados na prática, seriam capazes de danificar permanentemente os circuitos de hardware ou reduzir sua vida útil. Isso pode ocorrer por meio da geração de picos de tensão ou corrente, da criação de instabilidades em sistemas de controle ou da operação em níveis de tensão superiores aos projetados.

As contramedidas para esse tipo de ataque apresentam certas limitações devido à baixa velocidade com que o controlador consegue transmitir dados; com apenas 500 kbps, é desafiador implementar medidas robustas de criptografia e autenticação, já que essas exigem grande capacidade de processamento e largura de banda. Portanto, a contramedida analisada utiliza filtros lógicos capazes de detectar sinais que possam causar algum tipo de mau funcionamento no controlador e, assim, rejeitá-los.

Esta solução pode ser implementada tanto em software quanto em hardware, oferecendo proteção em diversos níveis. O artigo cita três formas de proteção via hardware. A primeira tem a finalidade de evitar curtos-circuitos no conversor, que podem ocorrer caso os MOSFETs sejam acionados incorretamente, e propõe um circuito lógico digital adicional para garantir que isso não aconteça. A segunda visa neutralizar ataques que utilizam ruídos, propondo o uso de um filtro digital *Finite Impulse Response* (FIR) em tempo discreto para remover ruídos de alta frequência e mitigar seus efeitos no laço de controle. A terceira busca detectar ataques de modificação de sinal, por meio de um algoritmo que identifica mudanças abruptas nos sinais, considerando que, em operação normal, os sinais dos sensores se alteram gradualmente.

As três contramedidas foram simuladas e todas apresentaram resultados satisfatórios, evitando que o conversor opere de forma destrutiva. Também são mencionadas algumas contramedidas para ataques que utilizam DPA (Differential Power Analysis), nos quais se monitora a potência consumida e o tempo necessário para processar os dados, a fim de obter as chaves criptográficas. No entanto, tais contramedidas não foram testadas neste artigo.

Em [8], publicado em dezembro de 2022, analisa o uso de PUFs (Physical Unclonable Functions) como uma solução promissora na segurança para autenticação e identificação em IoT. O artigo apresenta uma implementação de XOR Arbiter PUF (XOR-APUF) em FPGA, incorporada em outros métodos e algoritmos para aumentar a segurança. Além

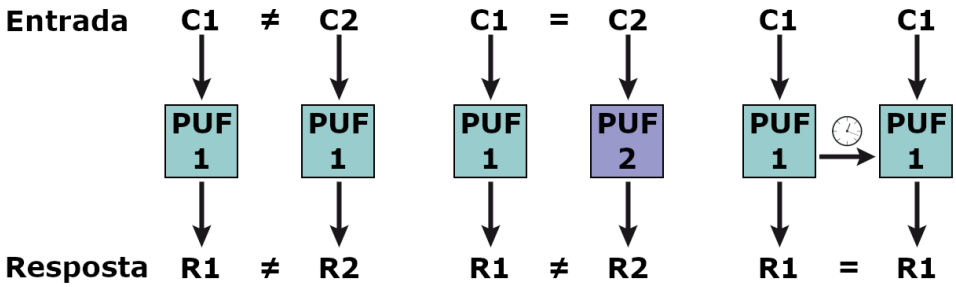


Figura 1 – Propriedades do PUF ideal. Adaptado de [1]

disso, também foi investigada a resistência a ataques no modelo proposto. O conceito de PUF se dá por um algoritmo cuja resposta depende de qual dispositivo ele foi executado, mesmo que a topologia usada e a entrada sejam idênticas. Por consequência, a segurança de um dispositivo PUF independe de chaves criptográficas, retirando a necessidade de algoritmos computacionalmente intensivos.

A Figura 1 ilustra as propriedades de um PUF ideal:

1. Cada entrada gera uma única saída. Os pares de entrada e saída são chamados de *Challenge Response Pairs* (CRPs)
2. Dois PUFs diferentes devem produzir CRPs diferentes, e devem ser impossíveis de serem previstos antes de sua fabricação.
3. O PUF deve operar de maneira estável. Os CRPs não podem se alterar dependendo das condições em que o dispositivo se encontra, como temperatura e tensão de operação.

No caso de um APUF, sua estrutura consiste em vários multiplexadores ligados em sequência, que geram dois pulsos percorrendo caminhos diferentes, dependendo da entrada. A resposta é decidida ao se comparar qual caminho é mais rápido, sendo o número total de CRPs proporcional a 2^n, em que n representa o número de multiplexadores. Como o número de CRPs aumenta exponencialmente com a quantidade de multiplexadores, é possível gerar uma grande quantidade de CRPs com uso relativamente baixo de hardware, o que torna viável não repetir os pares utilizados, dificultando um ataque bem-sucedido.

É possível simular um modelo APUF caso se conheçam os atrasos (delays) de cada componente lógico individual, resumindo o funcionamento do APUF a uma expressão matemática. Essa propriedade permite economizar memória ao substituir o armazenamento dos CRPs por um modelo simulado. No entanto, isso também revela uma vulnerabilidade a ataques baseados em aprendizado de máquina (Machine Learning), já que tais algoritmos são capazes de inferir o modelo matemático de um APUF com alta taxa de acerto, mesmo a partir de uma pequena quantidade de CRPs.

O artigo propõe o XOR-APUF como uma contramedida para aumentar a resistência a ataques baseados em aprendizado de *Machine Learning* (ML). A estrutura básica utiliza múltiplas células APUF em paralelo, cuja saída é formada por uma operação XOR entre todas as saídas das unidades APUF, resultando em 0 ou 1. Essa operação reduz a linearidade em relação ao APUF convencional, dificultando sua modelagem por algoritmos de ML.

A implementação do trabalho foi realizada em um FPGA Xilinx Artix-7. Para controlar o atraso dos componentes do APUF, foi utilizada uma propriedade das *Look-Up Tables* (LUTs), segundo a qual a entrada interfere no tempo necessário para finalizar cada operação, permitindo sua aplicação como *Programmable Delay Lines* (PDLs) para aumentar a estabilidade do APUF. Além disso, foi aplicado o conceito de *Temporal Majority Voting* (TMV), no qual cada APUF é repetido uma certa quantidade de vezes (15, neste caso), com a mesma entrada, utilizando-se a resposta majoritária na operação XOR.

Em seguida, o sistema foi avaliado em termos de *Uniqueness* (UQ), *Uniformity* (UF) e *Reliability* (RE). A variável UQ representa a porcentagem de quão independente é a resposta de dois PUFs para uma mesma entrada. Como existem apenas duas respostas possíveis (0 e 1), seu valor ideal é 50%. A UF verifica se há algum viés na resposta de um PUF; seu valor ideal também é 50%, pois a chance de a resposta ser 0 deve ser a mesma de ser 1. Já a RE demonstra a capacidade de um PUF gerar o mesmo resultado para uma mesma entrada, sendo seu valor ideal 100%.

Os resultados obtidos experimentalmente para o XOR-APUF proposto foram: 48,69% em UQ, 50,73% em UF e 99,41% em RE. Observa-se que os três parâmetros apresentaram valores próximos dos ideais, sendo que, em termos de UQ e UF, os resultados foram superiores aos obtidos com um APUF simples — embora o valor de RE tenha sido um pouco inferior (99,41% para o XOR-APUF, contra 99,55% para o APUF). Também foram analisados a imprevisibilidade (0,64, sendo o valor ideal igual a 1) e o consumo de hardware.

Por fim, foi testada a resistência a ataques por ML, sendo considerados quatro algoritmos distintos: LR, CMA-ES, SVM e ANN (a Figura 2 ilustra graficamente a representação desses algoritmos de ML). Os modelos foram alimentados com 40.000 CRPs e obtiveram as seguintes taxas de previsão: 58,2% para LR; 59,8% para CMA-ES; 61,5% para SVM; e 64,9% para ANN. Os resultados indicam que o modelo XOR-APUF proposto apresenta uma elevada resistência a ataques por ML, apesar de existirem outros modelos com desempenho ligeiramente superior nesse aspecto. O artigo conclui que o modelo de PUF proposto é um potencial candidato para autenticação via CRP no contexto da IoT.

Até o presente momento, não há qualquer estudo publicado envolvendo uma implementação de um algoritmo APUF em um FPGA da família Cyclone II. Entretanto, para esta família de FPGAs, existem alguns artigos que descrevem a implementação de

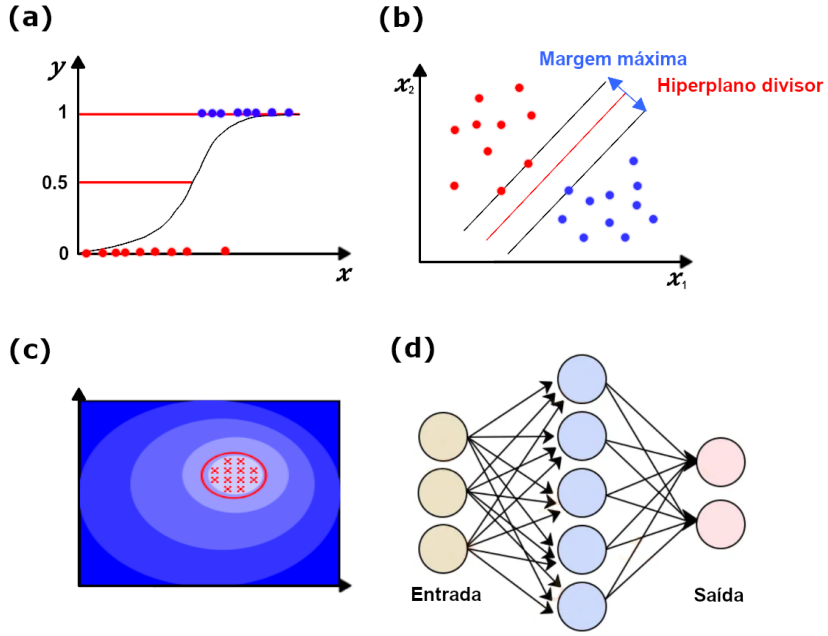


Figura 2 – Representação de algorítmos Machine Learning: LR (a), SVM (b), CMA-ES (c) e ANN (d). Adaptado de [2]

outros tipos de PUF, como o Ring-Oscillator PUF[9] e o Loop PUF[10].

2.2 Estado da Arte

O artigo [11] propõe um novo paradigma de segurança para a Internet das Coisas Industrial (IIoT), fundamentado em conceitos da computação quântica. A proposta central consiste na utilização de Quantum Physically Unclonable Functions (QPUFs), que exploram propriedades fundamentais da mecânica quântica, como superposição, emaranhamento e a impossibilidade de clonagem de estados quânticos para gerar identificadores únicos e invioláveis para cada dispositivo. As QPUFs configuram-se como uma solução promissora para proteção de dispositivos IIoT contra clonagem e ataques cibernéticos, antecipando os requisitos de segurança previstos para a futura Quantum Internet of Things (QIoT)

No contexto das PUFs embarcadas, o artigo [12] apresenta uma revisão abrangente sobre as arquiteturas de Physically Unclonable Functions implementadas em FPGAs. São discutidos os princípios de funcionamento, categorias, aplicações, métricas de desempenho, vantagens, limitações e perspectivas de pesquisa. O estudo compara diferentes abordagens de implementação em FPGA, evidenciando a relevância dessas soluções como mecanismos de segurança de hardware aplicáveis em autenticação de dispositivos, geração de chaves criptográficas, proteção contra clonagem e resistência a ataques de engenharia reversa em sistemas IoT.

3 Metodologia

Dentre as topologias de PUF que podem ser aplicadas em FPGAs, duas se destacam. A primeira é o Ring-Oscillator PUF, que funciona utilizando laços de portas lógicas NOT e um contador para determinar a frequência do sinal no laço, verificando, assim, qual é o mais rápido. A segunda é o Arbiter PUF (APUF), que utiliza vários multiplexadores em sequência para controlar o caminho percorrido pelo sinal, além de um árbitro que identifica qual trajeto foi mais rápido.

Neste trabalho, a topologia a ser estudada será o APUF, e o software escolhido para o desenvolvimento do projeto é o Quartus II.

Para criar o efeito de atraso do sinal necessário ao funcionamento do APUF, utilizou-se o conceito de PDL, produzido por meio de Look-Up Tables. Isso foi obtido através de vários multiplexadores conectados em hierarquia, formando uma estrutura em árvore binária. O bit enviado aos multiplexadores na primeira camada controla o bit que chega à saída, mas com inversão, como em uma porta lógica NOT; já os bits enviados às camadas seguintes alteram o caminho percorrido, modificando levemente a velocidade de operação. O número de multiplexadores necessário para cada PDL corresponde a $2^n - 1$, sendo n o número de camadas. Como o crescimento é exponencial, deve-se escolher um número de camadas razoável, de forma que cada PDL não ocupe muitas unidades lógicas ao ser embarcado no FPGA.

Ao criar duas linhas distintas de PDLs conectadas em sequência, obtém-se o primeiro requisito necessário para o funcionamento de um APUF: uma forma de gerar dois sinais com diferentes velocidades de propagação — imprevisíveis, mas não aleatórios — e facilmente expansíveis. Trata-se de uma topologia em que não há cruzamento de sinais: os bits alteram a velocidade dos caminhos de forma independente entre si. No entanto, vale ressaltar que essa não é a única abordagem possível. Algumas topologias de APUFs utilizam multiplexadores que alternam o caminho dos sinais.

A próxima etapa é desenvolver o árbitro. Esta parte é responsável por identificar qual dos sinais é mais rápido, sendo essencial para o funcionamento do APUF. Existem duas topologias de árbitro frequentemente utilizadas para esta função:

A primeira é constituída por duas portas lógicas NAND, cujas saídas são retroalimentadas a uma das entradas da outra porta NAND. A detecção do sinal ocorre porque a recepção de um sinal em uma das portas impede a outra de operar. Dessa forma, apenas a NAND que recebeu o sinal primeiro altera sua saída de 1 para 0.

Já a segunda topologia é constituída por um flip-flop do tipo D, com a entrada A

conectada tanto ao ponto D quanto ao pino de clear (invertido), enquanto a entrada B é conectada ao clock do flip-flop. A detecção do sinal ocorre porque a saída do flip-flop só se altera caso o sinal A chegue antes de B. No caso de B chegar antes de A, nada acontece, pois o clock será ativado antes de o ponto D estar em nível alto.

Vale ressaltar que as duas topologias geram sinais diferentes na saída, não sendo imediatamente substituíveis entre si. Portanto, os circuitos posteriores a esta parte devem ser projetados levando esta diferença em conta. Outro ponto a se levantar é que, como os árbitros precisam operar em tempos inferiores a 1 ciclo de clock, sua operação pode-se diferir entre um modelo simulado e sua execução embarcada em um FPGA, além de poder variar até mesmo entre diferentes versões de FPGA, sendo muito importante testá-lo em um ambiente real, a fim de certificar que a topologia do árbitro é compatível com o FPGA a ser utilizado.

As próximas partes não são obrigatórias para o funcionamento do PUF, mas auxiliam a manter a estabilidade da operação, de forma que os CRPs sejam mais consistentes. Ela é feita utilizando o conceito de Temporal Majority Voting (TVM), em que o mesmo challenge é repetido várias vezes, e salvando a resposta de cada uma em um registro. Com isso, é obtida a saída após n tentativas, de acordo com a maioria das respostas.

4 Resultados

No software Quartus II, foi elaborado o multiplexador através do diagrama de blocos (Figura 3). Para tal, foi feito um circuito lógico combinacional.

Em seguida, foi desenvolvido um PDL com 5 camadas de multiplexadores em cascata, sendo uma para inversão, e as outras 4 para controle, resultando em 2^4 combinações diferentes de entrada para cada PDL (Figura 4).

O funcionamento do PDL é simulado no software ModelSim, para a obtenção das formas de onda. Dado o circuito, deve agir como uma porta lógica NOT para o primeiro bit de entrada, o que é confirmado na simulação (Figura 5). No entanto, a mesma não é capaz de aferir o efeito de delay causado pelas diferentes rotas dos sinais internos do PDL.

Para o árbitro, inicialmente foi escolhido o método baseado em duas portas NAND com as saídas retroalimentadas (Figura 6). O motivo pela escolha desta topologia se dá pois sua operação é mais rápida do que a outra alternativa, que utiliza o D Flip Flop, já que este precisa de um pulso de clock para alterar sua saída. Como resultado, a saída do Flip Flop fica um pulso de clock atrasado em relação à entrada. Em comparação, o árbitro por NAND altera sua saída no mesmo pulso de clock em que recebe os sinais.

O funcionamento do árbitro foi testado no ambiente ModelSim. O efeito de delay foi simulado utilizando dois sinais de clock com a mesma frequência, porém com duty cycles diferentes (Figura 7). A simulação não permite que ambos os sinais ocorram exatamente no mesmo instante, pois a ferramenta interpreta esse caso como um loop infinito — embora isso não ocorra em uma placa real. A detecção do sinal ocorre de forma inversa: a saída é alta para o sinal mais lento e baixa para o sinal mais rápido.

As próximas etapas consistem no desenvolvimento de um registro de 15 bits, responsável por armazenar as 15 respostas anteriores (Figura 8), e do esquema de *Tempo-*

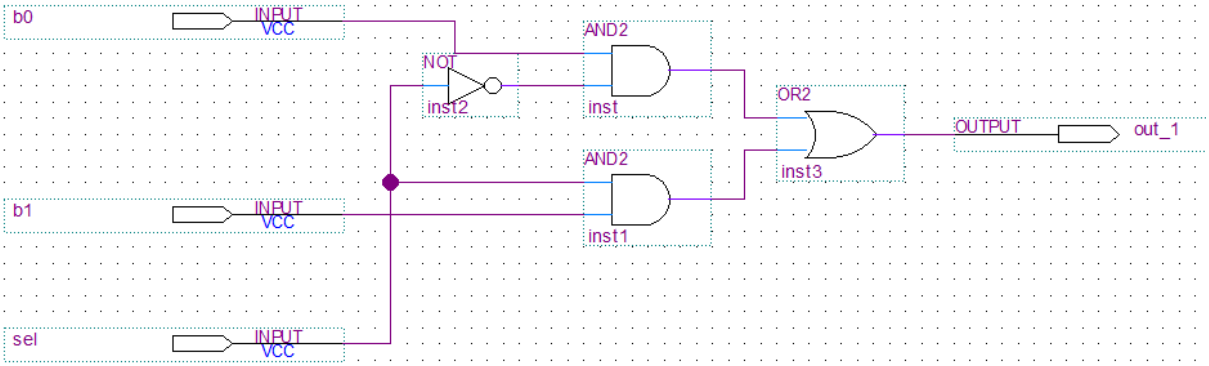


Figura 3 – Diagrama de blocos do multiplexador de 1 bit.

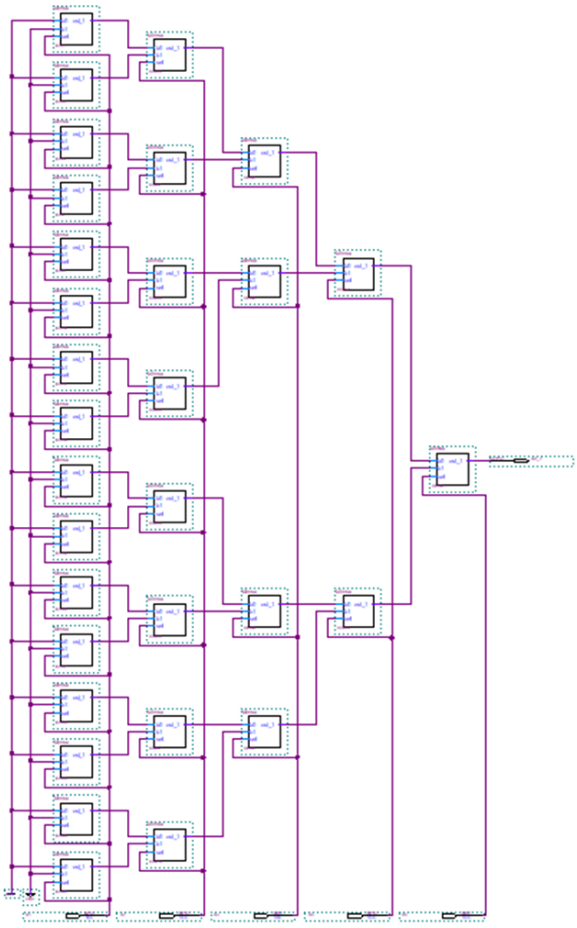


Figura 4 – Diagrama de blocos do PDL de 5 camadas.

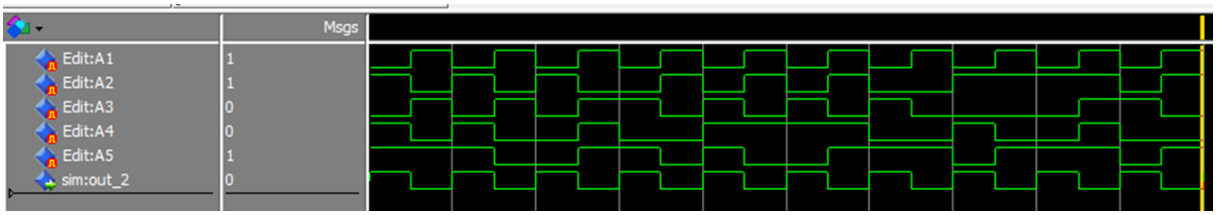


Figura 5 – Simulação do PDL no Modelsim. Este componente age como uma porta NOT para a entrada A1.

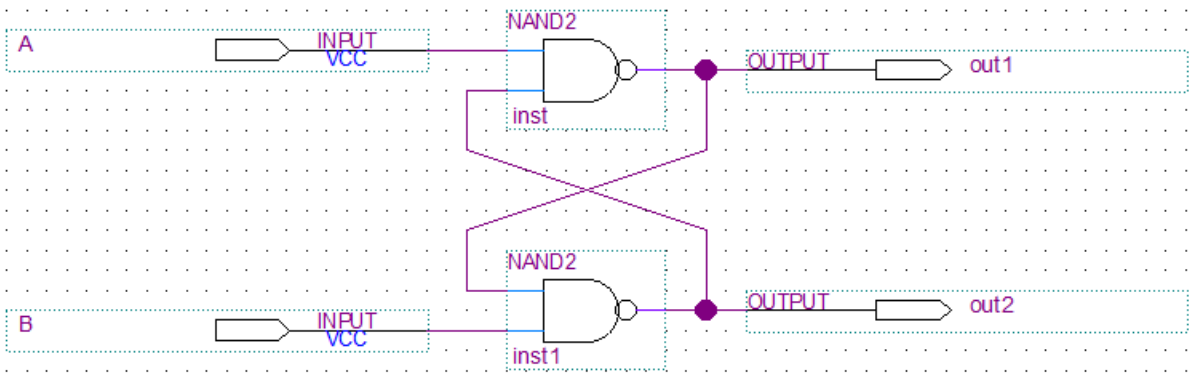


Figura 6 – Diagrama de blocos do árbitro, feito com 2 NANDs.

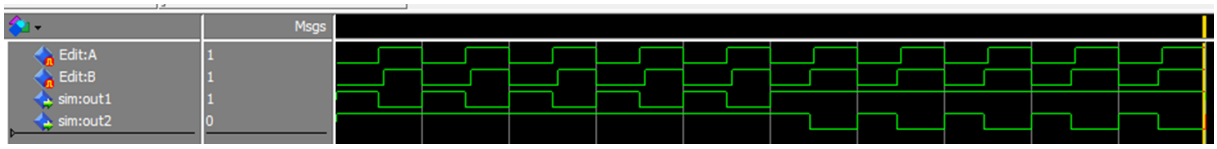


Figura 7 – Simulação do arbitro. O duty-cycle do sinal B foi definido em 0,45 na primeira metade, e 0,55 na segunda metade.

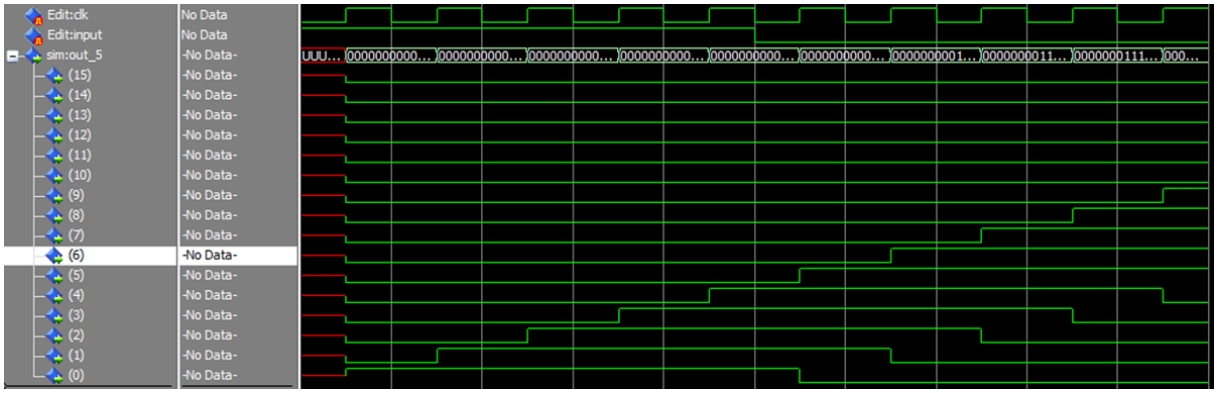


Figura 8 – Simulação do registro de 15 bits. Possui entrada serial, e saída paralelo.

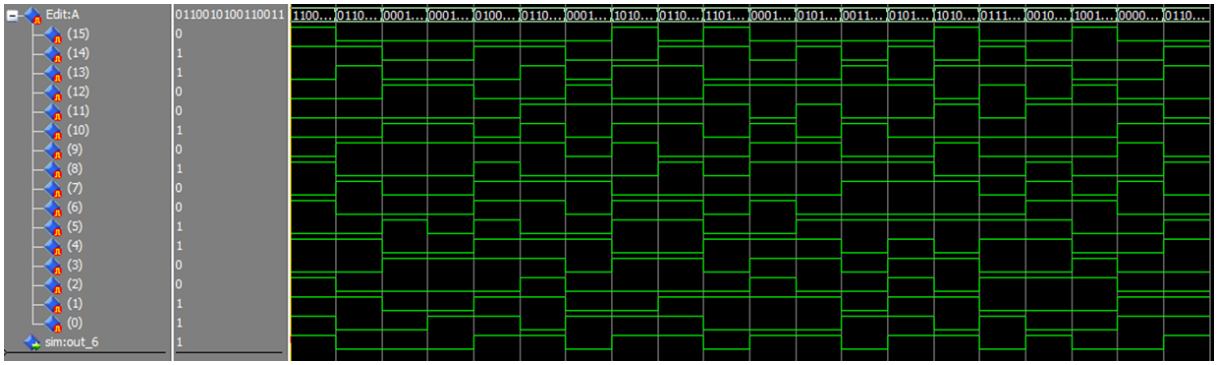


Figura 9 – Simulação do TVM. Os bits de entrada foram gerados aleatoriamente. A saída é 1, caso 8 ou mais bits também sejam 1.

ral Majority Voting (TMV), encarregado de identificar qual resposta ocorre com maior frequência (Figura 9). Diferentemente das etapas anteriores, que foram sintetizadas por meio de diagramas de blocos, essas foram implementadas diretamente em VHDL, utilizando o Quartus II. Embora o árbitro possua duas saídas (uma para cada caminho), o registro necessita apenas de uma delas. A detecção do caminho secundário é inferida quando o pulso de clock ocorre sem a presença do sinal principal.

Em seguida, cada componente foi testado individualmente em uma placa de testes Altera DE2, equipada com um FPGA Cyclone II EP2C35F672C6. As entradas e saídas de cada componente foram conectadas a interruptores e LEDs, respectivamente, sendo acionadas manualmente. Nesta etapa, o registro de 15 bits apresentou problemas de compilação. O código original considerava as duas saídas do árbitro, o que não é permitido pelo compilador. No entanto, o problema foi resolvido rapidamente com ajustes no código: passou-se a utilizar apenas uma das saídas, e a outra foi inferida com base em um pulso

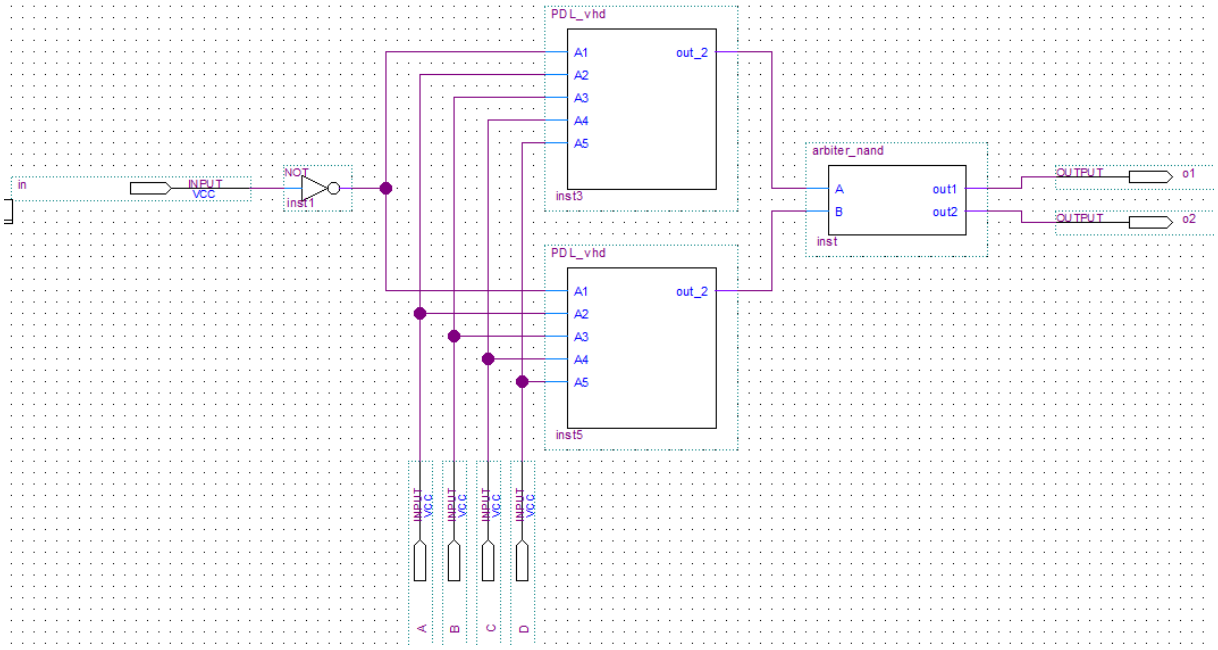


Figura 10 – Circuito para teste do arbitro.

de clock isolado. Os demais componentes funcionaram conforme o esperado, de acordo com os resultados obtidos na simulação.

Para verificar o funcionamento do árbitro, foi desenvolvido um circuito PUF simplificado, contendo apenas um PDL em cada linha, ambos conectados à mesma entrada (Figura 10). O objetivo era verificar se a saída do árbitro se alterava conforme a configuração dos PDLs. No entanto, o árbitro não foi capaz de detectar qualquer variação significativa na propagação do sinal. Independentemente do estado dos PDLs, o circuito sempre detectava primeiro a entrada B. O resultado permaneceu inalterado mesmo após a remoção de qualquer elemento lógico na linha A que pudesse introduzir atraso, com o intuito de favorecer essa entrada.

Tais resultados indicam que esse design de árbitro apresenta uma tendência a favorecer um dos lados, o que é incompatível com o funcionamento adequado do PUF. Utilizando a ferramenta *RTL Viewer*, disponível no Quartus II, foi possível identificar uma possível causa para esse comportamento (Figura 11): o compilador adiciona automaticamente um buffer na saída do NAND conectado à entrada A. Essa modificação introduz um atraso adicional no sinal, o que pode explicar a preferência sistemática pela entrada B, já que o sinal da entrada A chega ao árbitro com maior latência.

Visto que esse tipo de árbitro não atende aos requisitos do projeto, ele foi substituído por outro esquema: um flip-flop do tipo D, com as entradas D e CLRN conectadas. No entanto, ao repetir o teste anterior, o resultado permaneceu o mesmo, o árbitro continuou detectando o caminho inferior como o mais rápido, independentemente da configuração dos PDLs.

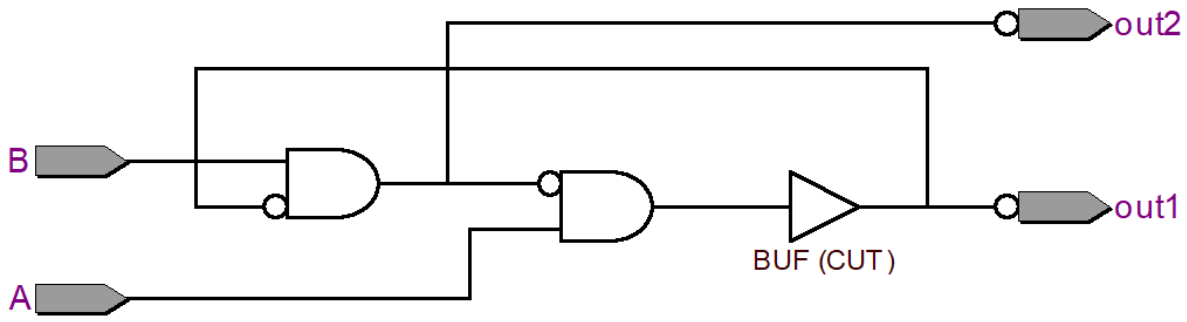


Figura 11 – Visualização do circuito do arbitro, com a ferramenta RTL Viewer do Quartus II.

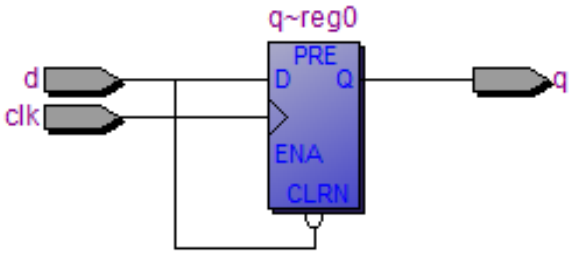


Figura 12 – Circuito do árbitro por D Flip Flop.

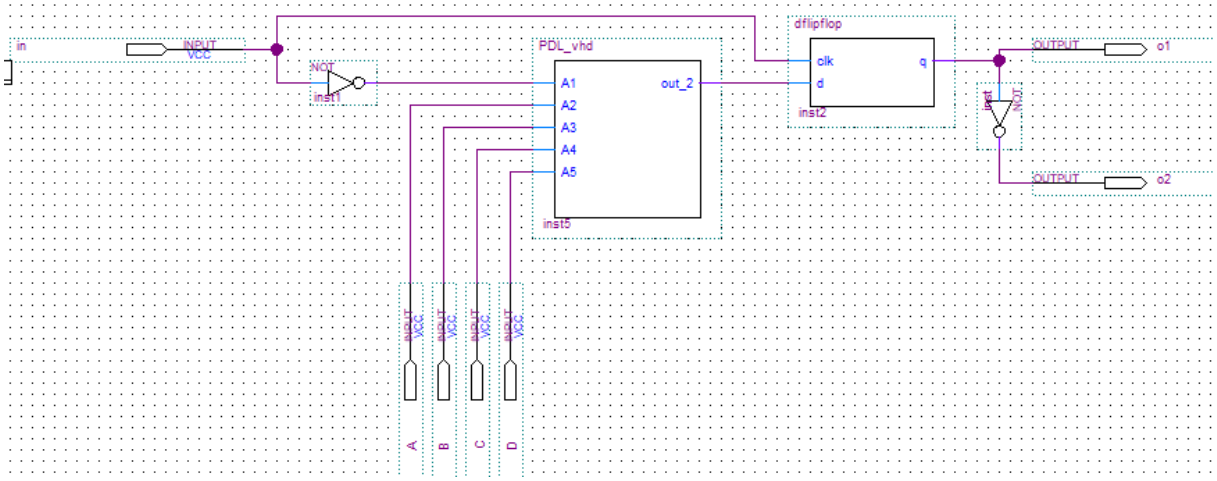


Figura 13 – Circuito para teste do árbitro por D Flip Flop, com o caminho superior livre.

Entretanto, em um outro posterior, no qual um dos caminhos foi mantido livre de qualquer elemento que causasse atraso, esta nova topologia de árbitro funcionou conforme o esperado, identificando corretamente o caminho mais rápido.

Os resultados indicam que o D Flip-Flop é capaz de atuar como árbitro, apresentando uma tendência interna significativamente menor do que a observada no árbitro baseado em portas NAND. No entanto, os testes também revelam indícios de que o design de PDL utilizado é compilado de forma determinística e sequencial pelo FPGA, o que compromete sua aplicação em um PUF.

Elemento	**Função**	**Implementação**	**Resultado na simulação**	**Resultado embarcado**
Multiplexador	Bifurcar o sinal.	Diagrama de Blocos	Funcionou	Funcionou
PDL	Gerar um atraso controlável do sinal.	Diagrama de Blocos	Não foi possível aferir o efeito de atraso na simulação.	Não aparentou ter a aleatoriedade necessária.
Arbitro NAND	Discernir as velocidades dos sinais.	Diagrama de Blocos	Funcionou ao alterar o duty cycle do sinal.	Não foi capaz de identificar o sinal com atraso.
Arbitro DFF	Discernir as velocidades dos sinais.	Diagrama de Blocos	Funcionou ao alterar o duty cycle do sinal.	Identificou o sinal com atraso no teste com um dos caminhos livres.
Registro de 15 bits	Armazenar as 15 ultimas respostas emitidas pelo arbitro.	VHDL	Funcionou	Funcionou, mas não foi testado no circuito completo.
Esquema TVM	Identificar a maioria as respostas obtidas.	VHDL	Funcionou	Funcionou, mas não foi testado no circuito completo.

Tabela 1 – Resumo dos resultados das partes individuais do APUF.

Conclusão

Este trabalho apresentou o estudo e a implementação de um algoritmo do tipo Arbiter Physically Unclonable Function (APUF) em uma plataforma FPGA, com foco na viabilidade de sua aplicação como mecanismo de segurança para dispositivos IoT. A partir da revisão teórica e do desenvolvimento prático com ferramentas como Quartus II, ModelSim e a placa Altera DE2, foi possível validar, por meio de simulações e testes em hardware, o funcionamento do algoritmo XOR-APUF, destacando sua capacidade de gerar respostas únicas e não clonáveis.

Durante a implementação, foram identificadas e superadas limitações relacionadas à compatibilidade entre os recursos descritos na literatura e o ambiente de desenvolvimento adotado. A experiência prática reforçou o potencial dos PUFs embarcados como solução eficiente e de baixo custo computacional para autenticação e geração de chaves criptográficas em sistemas embarcados.

Como contribuição adicional, este trabalho também pode servir como referência técnica para outros projetos relacionados à segurança de hardware, especialmente no contexto da Internet das Coisas. A exploração de PUFs em FPGA representa uma alternativa estratégica frente aos crescentes desafios de cibersegurança em ambientes distribuídos e com restrições de recursos.

Trabalhos Futuros

Com base nos resultados obtidos e nas tendências atuais em cibersegurança embarcada, destacam-se as seguintes direções para trabalhos futuros:

- Avanços Técnicos: Integração do APUF com protocolos criptográficos clássicos e pós-quânticos, visando maior robustez em ambientes hostis.Implementação de controladores adaptativos baseados em Lógica Fuzzy ou Redes Neurais para calibragem dinâmica da resposta do PUF frente a variações ambientais (temperatura, tensão etc.).Desenvolvimento de IP-cores otimizados de PUFs para aplicações System-on-Chip (SoC) com suporte a interfaces padrão (AXI/AMBA).Avaliação da aleatoriedade e unicidade utilizando métricas estatísticas padrão (Hamming Distance, Uniformity, Reliability), com base em conjuntos de dados reais de larga escala.

- Aplicações Práticas: Aplicação do PUF como módulo de autenticação e anti-clonagem em sistemas IoT industriais (IIoT), sensores remotos e automação de processos.Emprego em ambientes militares, como módulos de segurança embarcados em equipamentos de comunicações criptografadas, drones e sistemas C4ISR.Utilização em sistemas de telecomunicações, como identificação segura de estações rádio-base e terminais em redes 5G/6G.Implementação em dispositivos de consumo (smart wearables, automação residencial) como alternativa leve e eficiente a elementos criptográficos convencionais.

- Extensão Acadêmica e Científica: Consolidação de uma linha de pesquisa em “Segurança de Hardware e Sistemas Embarcados” no âmbito da pós-graduação, com foco em PUFs, FPGAs, criptografia leve e confiável.Publicação de artigos técnicos em periódicos e conferências especializados, como IEEE Transactions on VLSI Systems, ACM TECS, IEEE Access, entre outros.

Referências

1 MESARITAKIS, C. et al. Physical unclonable function based on a multi-mode optical waveguide. *Scientific Reports*, v. 8, n. 1, Jun 2018.

2 ZHANG, Y. *New advances in machine learning*. [S.l.]: InTech, 2010.

3 ANANDAKUMAR, N. N.; HASHMI, M. S.; TEHRANIPOOR, M. Fpga-based physical unclonable functions: A comprehensive overview of theory and architectures. *Integration*, v. 81, p. 175–194, Nov 2021.

4 LU, Y.; XU, L. D. Internet of things (iot) cybersecurity research: A review of current research topics. *IEEE Internet of Things Journal*, v. 6, n. 2, p. 2103–2115, Apr 2019.

5 KHAN, S.; LEE, W.-K.; HWANG, S. O. Scalable and efficient hardware architectures for authenticated encryption in iot applications. *IEEE Internet of Things Journal*, v. 8, n. 14, p. 11260–11275, Jul 2021.

6 BENHANI, E. M.; BOSSUET, L.; AUBERT, A. The security of arm trustzone in a fpga-based soc. *IEEE Transactions on Computers*, v. 68, n. 8, p. 1238–1248, Aug 2019.

7 CHANDWANI, A.; DEY, S.; MALLIK, A. Cybersecurity of onboard charging systems for electric vehicles—review, challenges and countermeasures. *IEEE Access*, v. 8, p. 226982–226998, Dec 2020.

8 ANANDAKUMAR, N. N.; HASHMI, M. S.; CHAUDHARY, M. A. Implementation of efficient xor arbiter puf on fpga with enhanced uniqueness and security. *IEEE Access*, v. 10, p. 129832–129842, Dec 2022.

9 FEITEN, L. et al. Implementation and analysis of ring oscillator pufs on 60 nm altera cyclone fpgas. *Information Security Journal: A Global Perspective*, v. 22, n. 5–6, p. 265–273, Nov 2013.

10 CHERIF, Z. et al. An easy-to-design puf based on a single oscillator: The loop puf. *2012 15th Euromicro Conference on Digital System Design*, p. 156–162, Sep 2012.

11 BATHALAPALLI, V. K. et al. Qpuf: Quantum physical unclonable functions for security-by-design of industrial internet-of-things. *Cryptography*, v. 9, n. 2, p. 34, May 2025.

12 LATA, K.; CENKERAMADDI, L. R. Fpga-based puf designs: A comprehensive review and comparative analysis. *Cryptography*, v. 7, n. 4, p. 55, Nov 2023.