UNIVERSIDADE FEDERAL DO PARANÁ
SETOR DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
CURSO DE ENGENHARIA ELÉTRICA

Leonardo de Andrade Santos

Projeto de um circuito integrado de um Pré-Distorcedor Digital baseado em polinômio de memória

Curitiba
2024

Leonardo de Andrade Santos

Projeto de um circuito integrado de um Pré-Distorcedor Digital baseado em polinômio de memória

Trabalho de conclusão de curso do Curso de Graduação em Engenharia Elétrica da Universidade Federal do Paraná, como exigência parcial para obtenção do grau de Bacharel em Engenharia Elétrica.

Orientadora: Sibilla Batista da Luz França

Coorientador: Eduardo Gonçalves de Lima

Curitiba
2024

Resumo

A evolução dos sistemas de comunicação sem fio acarretou na implementação de diversas aplicações móveis e sem fio como desenvolvimento web, aplicação IoT, entre outros. Neste cenário, melhorar a eficiência energética se torna uma alternativa desejável tanto para os dispositivos móveis que buscam melhorar a autonomia das suas baterias, quanto para as estações de rádio base, que buscam reduzir seus desperdício em perdas de calor. No entanto, uma melhor eficiência energética implica em uma menor linearidade nos sistemas de amplificação de sinais, presentes nos sistemas transmissores de sinais de rádio. Isto é importante de ser ressaltado, pois a banda reservada para aplicações móveis é reduzida, de forma que para se alcançar maiores taxas de transmissão é necessário alternar estratégias de modulação tanto da fase, quanto da amplitude da onda portadora. E essas duas condições são conflitosas, já que a modulação AM é sensível a linearidade de forma que quanto mais linear um sistema ocorrem menos erros de transmissão. Sendo assim, uma alternativa para contornar esse obstáculo, que é implementar um sistema, eficiente energeticamente e linear é a implementação de um DPD em cascata com um PA. Portanto, o objetivo deste trabalho de conclusão de curso é o design de um circuito integrado dedicado de um DPD. Para atingir esse objetivo, esse projeto foi dividido em quatro etapas: o estudo e modelagem dos DPDs, modelagem do DPD em software, implementação do DPD em FPGA e finalmente o design do circuito integrado do DPD. Para a modelagem do DPD foi utilizada a métrica do NMSE; nela quanto menor o NMSE encontrado mais fiel é o modelo com a realidade. Sendo assim, na etapa de modelagem do PA alcançou-se um NMSE de -23,57 dB. Em seguida, foi feito o levantamento do número de bits necessários para a realização desses cálculos de forma a minimizar o NMSE. Para isso foi verificado que com apenas 8 bits de resolução do sinal já foi possível alcançar um NMSE próximo do valor alcançado em vírgula flutuante. Após feito esses levantamentos foi feita a implementação do circuito em VHDL e validação em FPGA Virtex5 XC5VLX50T, que utilizou um total de 150 registradores, 692 LUTs e 4 unidades DSP48E, operando a uma frequência de 61,5 MHz. Em seguida seguiu-se para a etapa de síntese lógica a qual resultou em um circuito com 1567 células lógicas, com uma área total de 28116 μm^2 e um consumo de energia de 1.6 mW, atuando a uma taxa de operação de 33,34 MHz.

Palavras-chave: VHDL, FPGA, DPD .

Abstract

The evolution of wireless communication systems has led to the implementation of various mobile and wireless applications, such as web development and IoT applications, among others. In this context, improving energy efficiency becomes a desirable alternative both for mobile devices aiming to enhance battery autonomy and for base radio stations seeking to reduce heat loss waste. However, better energy efficiency implies lower linearity in the signal amplification systems present in radio signal transmitters. This is significant because the bandwidth allocated for mobile applications is limited, meaning that achieving higher transmission rates requires alternating modulation strategies for both the phase and amplitude of the carrier wave. These two conditions are conflicting since AM modulation is sensitive to linearity, and the more linear a system is, the fewer transmission errors occur. Thus, an alternative to overcoming this challenge—implementing a system that is both energy-efficient and linear—is the use of a Digital Predistortion (DPD) system in cascade with a Power Amplifier (PA). Therefore, the goal of this undergraduate thesis is the design of a dedicated integrated circuit for a DPD. To achieve this goal, the project was divided into four stages: studying and modeling DPDs, DPD modeling in software, DPD implementation on FPGA, and finally, designing the DPD integrated circuit. For DPD modeling, the NMSE (Normalized Mean Square Error) metric was used, where a lower NMSE indicates a model that is more faithful to reality. In the PA modeling stage, an NMSE of -23,57 dB was achieved. Next, the number of bits required to perform these calculations while minimizing NMSE was determined. It was found that with only 8 bits of signal resolution, it was already possible to achieve an NMSE close to the floating-point value. After this analysis, the circuit was implemented in VHDL and validated on an FPGA Virtex5 XC5VLX50T, using a total of 150 registers, 692 LUTs, and 4 DSP48E units, operating at a frequency of 61,5 MHz. Subsequently, the logical synthesis stage was carried out, resulting in a circuit with 1,567 logic cells, a total area of 28,116 um^2, and power consumption of 1,6 mW, operating at a frequency of 33,34 MHz.

Palavras-chave: VHDL, FPGA, DPD

Lista de abreviaturas e siglas

DPD	Pré-Distorcedor Digital
FPGA	Matriz de Portas Programáveis em Campo
PA	Amplificador de Potência
RF	Rádio Frequência
PARF	Amplificador de Potência de Rádio Frequência
HDL	Linguagem de Descrição de software
VHSIC	Circuito integrado de Velocidade Muito Elevada
VHDL	VHSIC Hardware Description Language
LUT	Look Up Table
SOP	Soma de Produtos
LAB	Logic Array Block
ALM	Adaptive Logic Module
LE	Logic Element
HEMT	transistor de efeito de campo de heterojunção
VSA	analisador de sinal vetorial
NMSE	Erro Médio Quadrado Normalizado

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

A evolução dos sistemas de comunicação móveis, impulsionada pela crescente demanda por comunicações mais rápidas e eficientes, tem levado à implementação de uma variedade de serviços, incluindo aplicações multimídia, desenvolvimento web e aplicações IoT [1]. No entanto, essa evolução também trouxe desafios significativos, como a necessidade de melhorar a eficiência energética, tanto para dispositivos móveis, visando aumentar a autonomia da bateria, quanto para estações de rádio base, visando reduzir o consumo de energia devido às perdas de calor. Para atender a essas demandas, estratégias de modulação que alteram tanto a fase quanto a amplitude de ondas portadoras em radiofrequência se tornaram essenciais [2]. Além disso, a modulação na amplitude requer linearidade na transmissão para evitar erros e interferências na comunicação entre usuários vizinhos [3]. Essa complexa tarefa recai sobre o projetista do PARF (Amplificador de Potência de Rádio Frequência), que enfrenta o desafio de desenvolver um hardware eficiente em termos energéticos e linear ao mesmo tempo, uma vez que esses dois objetivos podem entrar em conflito [4]. Uma solução para contornar esse desafio é a implementação de um pré-distorcedor de Sinais Digital em Banda Base, que visa compensar a distorção causada pelo PARF [3]. O DPD (Pré-distorcedor Digital) é conectado em cascata ao PARF e requer um modelo de alta precisão e baixa complexidade computacional para representar as características de transferência direta e inversa do PARF. Existem duas abordagens para modelar o PARF: modelos físicos, que são detalhados e computacionalmente complexos, e modelos empíricos, que se baseiam em medições de entrada e saída do PARF, com menor complexidade computacional, mas com uma possível diminuição da precisão. Devido às exigências rigorosas de frequência de operação, a paralelização das operações torna-se essencial, sendo assim, as FPGAs (Matriz de Portas Programáveis em Campo) emergem como uma alternativa viável para a implementação de circuitos pré-distorcedores [5]. As FPGAs são dispositivos lógicos programáveis que permitem a reconfiguração física de componentes de eletrônica digital, acelerando processos e suportando operações paralelas e sequenciais. Neste contexto esse projeto foi planejado com os objetivos detalhados a seguir.

1.1 Objetivo Geral

O desenvolvimento de um circuito integrado dedicado para um pré-distorcedor digital na tecnologia BiCMOS 130 nm 8HP.

1.2 Objetivos Específicos

Para alcançar o objetivo geral, este trabalho foi desenvolvido com base nos seguintes objetivos específicos:

1. Modelar com precisão o PA em software;
2. Modelar o DPD em software a partir da modelagem do PA;
3. Implementar o DPD em hardware (FPGA) utilizando uma HDL;
4. Desenvolver a síntese.

2 Revisão de Literatura

A evolução dos sistemas de comunicações sem fio fomentou a implementação de diversas aplicações móveis. Neste contexto, melhorar a eficiência energética desses sistemas se torna uma característica desejável, tanto para os dispositivos móveis que buscam melhorar a autonomia das baterias quando para os sistemas de comunicação rádio base que tendem a perder energia em forma de calor. Sendo assim, o sistema de comunicação pode ser dividido em 3 sub-sistemas principais: Meio transmissor, Receptor e o Transmissor, conforme argumentado por [7]. No entanto, este trabalho foca exclusivamente no sistema transmissor, ilustrado pela figura 1, em que observa-se diversos componentes que compõem o transmissor de sinal. Entre esses elementos o amplificador de potência é o componente de maior demanda energética, por se tratar do componente que converte a energia da fonte em energia irradiada pela antena de transmissão. Portanto, a eficiência do sistema de transmissão depende diretamente do desempenho do transmissor.

Fonte: [7]

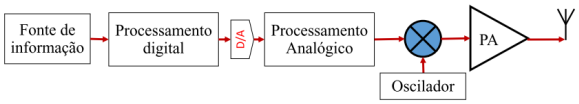

Figura 1 – Sistema de transmissão simplificado

Considerando-se também que a largura de banda reservada para sistemas de comunicação sem fio é reduzida, torna-se desejável utiliza-la da maneira mais eficiente o possível. Diante desse cenário, segundo [2], só é possível alcançar as maiores taxas utilizando estratégias de modulações, que alterem tanto a fase quanto a amplitude de uma onda portadora em rádio frequência. Ainda segundo [2], a modulação pela amplitude exige linearidade na transmissão para evitar erros e interferência na comunicação entre os usuários vizinhos. Ante esse panorama, o projetista do PARF se depara com esse desafio, que é desenvolver um hardware eficiente energeticamente e com uma boa linearidade, o que é um compromisso conflitante, conforme descrito por [3]. Esse comportamento se deve, pois um PARF atua de forma eficiente, ou seja, com baixo consumo de energia, na área próxima à de saturação, que é a região em que opera em regimes não lineares, conforme ilustrado pela figura 2.

A fim de contornar esse obstáculo foi adicionada à cadeia de transmissão um método de equalização de sinais, conforme argumentado por [2]. Um exemplo de técnica de linearização de sinais é a implementação de um pré-distorcedor de sinais digitais em banda base, o qual apresenta um melhor custo-benefício [2]. Essa técnica consiste em distorcer o sinal de entrada utilizando técnicas de processamento digital, antes que esse module uma

Fonte: [4]

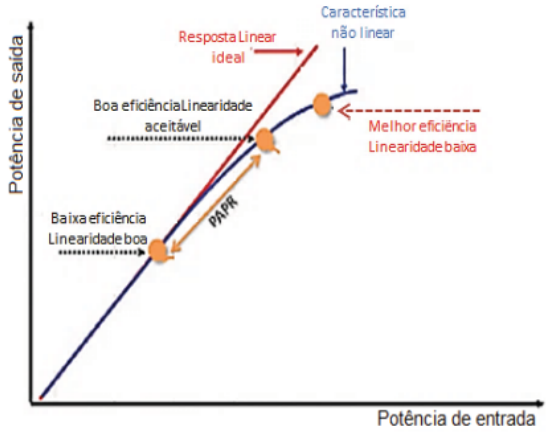

Figura 2 – Curva de saída do amplificador

portadora, de forma compensativa à distorção causada pelo PARF. De maneira sucinta, o DPD é conectado em cascata ao PARF e é projetado de forma que apresenta a função transferência inversa ao PARF. Para isso, é necessário um modelo de alta precisão e baixa complexidade computacional, capaz de representar as características de transferência direta e inversa de um PARF. Isso significa modelar o seu comportamento real utilizando um software. A figura 3 ilustra o processo do um pré-distorcedor digital.

Fonte: [4]

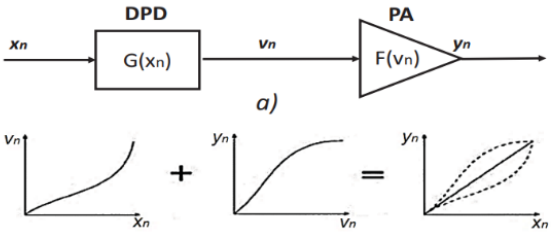

Figura 3 – ilustração do pré-distorcedor em cascata

Segundo [1], existem duas técnicas utilizadas para fazer essa modelagem. Uma consiste na descrição detalhada do PARF, que implica em uma maior complexidade computacional; esses modelos são conhecidos como modelos físicos. A outra abordagem é conhecida como modelo empírico; este modelo consiste em coletar amostras na entrada e na saída do PARF em domínio temporal, e através destes dados simular um modelo matemático do sistema. Uma das vantagens desse método é sua independência em relação ao conhecimento prévio da estrutura do PARF, além de apresentar baixa complexidade computacional. Mesmo que todos os parâmetros do circuito fossem conhecidos e seu equacionamento completo fosse realizado, a obtenção de uma função inversa seria possível, embora provavelmente mais complexa do que as séries de Volterra. No entanto, sua precisão pode ser ligeiramente afetada pelo modelo adotado. Sendo assim, como a proposta do

projeto é a implementação de um DPD em hardware, se faz necessário que o circuito apresente a menor complexidade possível, torna-se mais viável fazer a implementação utilizando modelagem matemática.

2.1 Modelagens Matemáticas

Séries de Volterra

Segundo [6] série de Volterra pode ser vista como uma extensão multidimensional da série de Taylor para sistemas dinâmicos. A modelagem começa com a representação do sistema através de uma série infinita de integrais convolucionais, em que cada termo da série corresponde a uma ordem de não linearidade e memória.

A saída $y(t)$ de um sistema pode ser expressa pela equação 2.1:

$$y(t) = h_0 + \sum_{n=1}^{\infty} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(\tau_1, \tau_2, \ldots, \tau_n) \prod_{i=1}^{n} x(t - \tau_i)\, d\tau_i \tag{2.1}$$

onde h_n são os núcleos de Volterra, que caracterizam a resposta do sistema para a n-ésima ordem de não linearidade e $x(t)$ é a entrada do sistema.

Os núcleos de Volterra h_n são funções de várias variáveis que capturam a dinâmica do sistema em diferentes ordens. Para a maioria das aplicações práticas, a série é truncada para incluir apenas um número finito de termos, já que a identificação de todos os núcleos de uma série infinita é impraticável.

Polinômio de memória

Um modelo simples, utilizado na modelagem comportamental simplifificada das séries de Volterra, considerando apenas componentes unidimensionais[1], é o MP, que é um modelo compacto, de baixo custo computacional e linear em seus parâmetros. O MP gera baixo erro quando aplicado à PAs que apresentam pouco efeito de memória. O DPD e pós distorsor apresentam uma característica de transferência inversa a do PA [7], portanto o mesmo modelo pode ser utilizado. A equação 2.2 apresenta o MP conforme apresenta [7]:

$$y(n) = \sum_{p=1}^{P} \sum_{m=0}^{M} h_{p,m} x(n-m)\, |x(n-m)|^{p-1} \tag{2.2}$$

Como a proposta do trabalho é a implementação em hardware desse modelo, torna-se necessário paralelizar operações aritméticas de forma a alcançar uma taxa de operação que satisfaça à norma regulamentadora. Nesse contexto, as FPGAs apresentam-se como uma alternativa viável para a implementação de circuitos pré-distorcedores

[1] Cada termo do somatório é composto por amostras no mesmo instante, por exemplo: $x(n)|x(n)|, x(n-1)|x(n-1)|$; termos bidimensionais são compostos por amostras em instantes de tempos distintos, como por exemplo: $x(n)|x(n-1)|$

2.2 FPGAs

Como descrito em [5], FPGAs são uma classe de dispositivos lógicos programáveis que permitem a reconfiguração física de seus componentes de eletrônica digital por meio de uma linguagem de descrição de hardware. Basicamente, as FPGAs consistem em um conjunto de subcircuitos digitais interconectados, capazes de realizar diversas funções comuns enquanto oferecem um alto nível de flexibilidade. Devido a essas características, FPGAs podem ser utilizadas para aplicações como processamento de imagem em tempo real e aprendizado de máquina.

FPGAs têm a capacidade de sintetizar arquiteturas complexas de eletrônica digital, resultando em um funcionamento altamente paralelizado que permite um processamento rápido com várias portas de entrada e saída. Além disso, elas também suportam o desenvolvimento de códigos sequenciais.

A estrutura interna de uma FPGA é composta fundamentalmente por blocos lógicos interligados, organizados em uma matriz. Cada bloco é formado por diversos sub-blocos, que por sua vez contêm os componentes mais básicos da hierarquia. FPGAs da Intel e da Xilinx possuem nomenclaturas e organizações diferentes para esses blocos e sub-blocos. Isso é ilustrado na Figura 4 e na Figura 5, que mostram as FPGAs Intel Stratix X e Xilinx Ultrascale+, respectivamente. Embora as arquiteturas sejam fundamentalmente semelhantes, com a disposição em matriz dos blocos e funcionalidades dos componentes fundamentais sendo universais, os blocos lógicos são denominados LAB nas FPGAs da Intel e CLB nas FPGAs da Xilinx. Os sub-blocos são chamados de ALM ou LE, dependendo da FPGA da Intel, e de Slices nas FPGAs da Xilinx.

Fonte: [5]

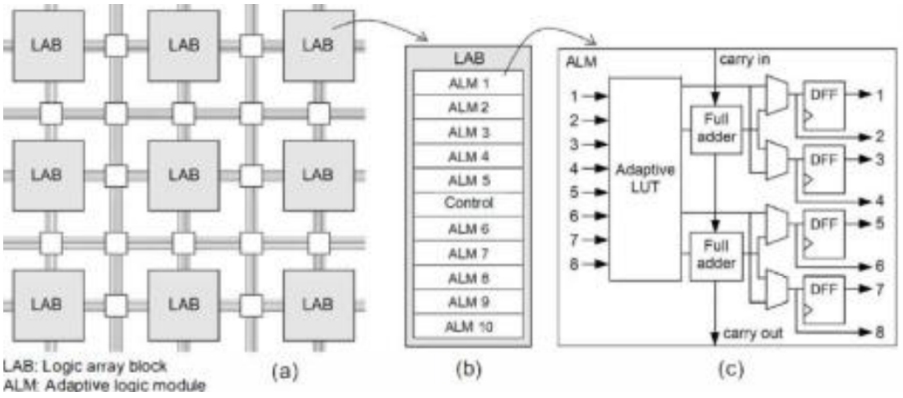

Figura 4 – Estrutura Interna da FPGA Stratix X da Intel

Os sub-blocos das FPGAs são compostos por LUTs e registradores. As LUTs são compostas por uma árvore binária de multiplexadores 2:1, permitindo o armazenamento de uma função lógica na forma de SOP. Os registradores são os componentes síncronos dos sub-blocos. Além da estrutura mencionada, FPGAs comumente possuem diversos

Fonte: [5]

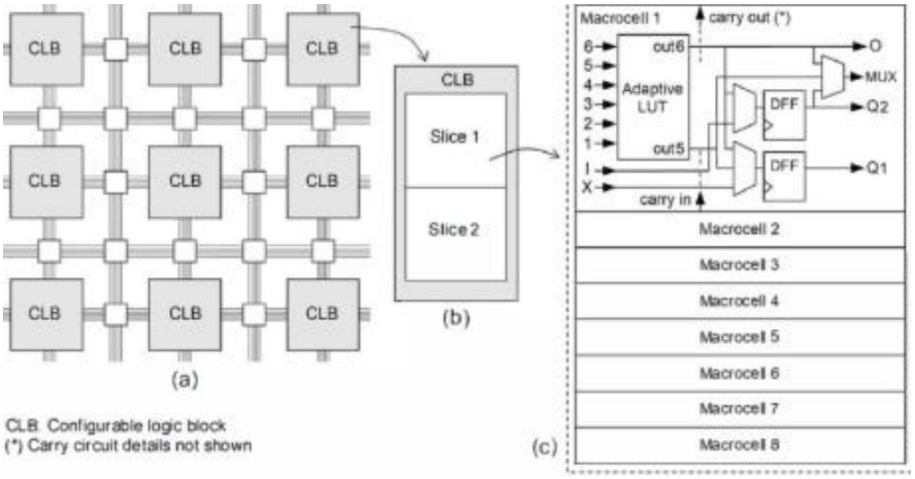

Figura 5 – Estrutura Interna da FPGA Ultrascale+

módulos integrados, como CPUs, DSPs, memória Flash, PLLs, que aumentam ainda mais as capacidades do FPGA.

As FPGAs são programadas utilizando uma linguagem de descrição de hardware, sendo o VHDL uma das mais comuns para a síntese de circuitos integrados de alta velocidade. Criada por uma iniciativa financiada pelo Departamento de Defesa dos Estados Unidos em meados dos anos 80, o VHDL foi a primeira linguagem de descrição de hardware padronizada pela IEEE.

A estrutura de um código VHDL consiste em três partes principais: declaração de bibliotecas/pacotes, entidade e arquitetura. Na primeira parte, são listadas as bibliotecas e pacotes necessários para o projeto. As bibliotecas padrão incluem a ‘std‘ e a ‘work‘. A entidade, que é a interface do sistema, descreve as entradas e saídas e é dividida em duas partes: parâmetros e conexões. Os parâmetros são valores constantes, como a largura de um barramento, que são declarados como genéricos. As conexões, por sua vez, definem a transferência de informações e correspondem aos pinos de entrada e saída do circuito. Já a arquitetura é a parte principal do sistema, na qual o circuito é descrito. Nessa seção, são definidas as atribuições, operações lógicas e aritméticas, comparações, entre outros. Há também uma parte declarativa da sintaxe, que apresenta uma ampla variedade de declarações possíveis.

Dessa forma, circuitos digitais para processamento de sinais em tempo real são muito utilizados em sistemas de comunicações sem fio. Um exemplo de aplicação são os DPDs para transmissores sem fio. Os DPDs são baseados em operações matemáticas que envolvem uma grande quantidade de somas, produtos e tabelas de busca. Devido às rigorosas exigências de frequência de operação, torna-se fundamental a paralelização das operações necessárias. Nesse contexto, as FPGAs se mostram uma alternativa viável para a implementação de circuitos DPDs, especialmente devido à sua capacidade de

paralelização. Considerando que a paralelização em FPGAs pode alcançar uma taxa de operação adequada para o uso pretendido, a implementação desse hardware em um circuito lógico dedicado pode oferecer resultados ainda mais eficientes para essa aplicação, dado o potencial de otimização específica e o desempenho superior de circuitos dedicados em relação a arquiteturas reconfiguráveis.

2.3 Síntese com as células da tecnologia

A concepção do circuito lógico do DPD segue o fluxo de projeto VLSI para design de um circuito integrado de aplicação específica, inclui a descrição do circuito em VHDL, síntese lógica utilizando as células padrão da tecnologia, PAR (*place and route*) e simulações comportamentais e temporais. O diagrama do fluxo VLSI que é ilustrado pela figura 6.

Fonte: [13]

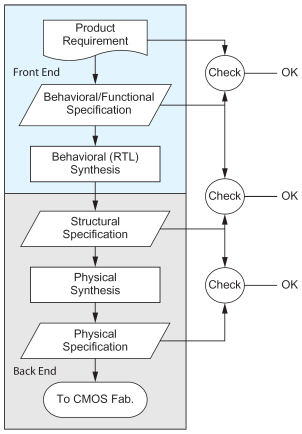

Figura 6 – Fluxo de projeto VLSI.

No desenvolvimento do circuito, várias etapas são seguidas. Inicialmente, realiza-se a simulação comportamental para assegurar que o circuito descrito em VHDL cumpre os requisitos esperados, utilizando um *testbench* em VHDL e a ferramenta Cadence NCLaunch. Posteriormente, ocorre a síntese lógica a partir do modelo comportamental, empregando a ferramenta Genus para gerar um modelo RTL com células padrão de uma tecnologia específica, levando em conta restrições de área, frequência e consumo de energia. Essa síntese resulta em dois arquivos: um em Verilog, contendo componentes e conexões, e outro com informações de atraso no formato SDF. A simulação pós-síntese é então executada para validar o netlist gerado, utilizando o mesmo *testbench* da simulação comportamental. Na etapa de PAR, o layout é desenvolvido posicionando as células e estabelecendo as conexões entre elas, com o uso da ferramenta Innovus. Finalmente, na simulação pós-PAR, o circuito é avaliado considerando as resistências e capacitâncias parasitas. Cada etapa é crucial para assegurar o funcionamento correto do circuito.

3 Material e Métodos

Como mencionado anteriormente, este trabalho teve como objetivo desenvolver o design de um circuito integrado para um DPD, partindo de um modelo previamente validado tanto em software quanto em hardware, especificamente em FPGA. O projeto foi dividido em quatro etapas principais:

- Estudo sobre PA e modelagem matemática;
- Implementação em software do PA e do DPD;
- Implementação do DPD em FPGA;
- Design e validação do circuito implementado com a tecnologia de 8HP 130nm.

3.1 Estudo sobre PA e modelagem matemática

A etapa consistiu no estudo de modelagens de Amplificadores de potência para posteriormente fazer a modelagem do DPD, conforme apresentado no Capítulo 2, na qual foi feito todo o levantamento sobre os tipos de modelagem dos DPDs. O objetivo deste estudo é entender as diferentes abordagens de modelagem, avaliar seus desempenhos e identificar as mais adequadas para a aplicação em amplificadores de potência.

3.2 Implementação em software

Nesta etapa, implementou-se o modelo DPD em software, utilizando a linguagem de programação Python. Esta linguagem amigável é amplamente difundida na comunidade acadêmica.

Para essa modelagem, foram coletados sinais de entrada e saída de um amplificador de potência classe AB, HEMT fabricado com tecnologia GaN. O amplificador foi excitado por um sinal portador de frequência de 900 MHz, modulado por um sinal de envelope WCDMA 3GPP com aproximadamente 3,84 MHz de largura de banda. Os dados de entrada e saída do amplificador de potência foram medidos usando um VSA Rohde & Schwarz com uma taxa de amostragem de 61,44 MHz, conforme disponível em [8].

Em seguida, realizou-se o cálculo da estimativa do sinal utilizando números com vírgula fixa. Para verificar a precisão dessa estimativa em relação ao sinal original, calculou-se o NMSE. Para essa validação, os dados foram inicialmente divididos em conjuntos de extração e validação. A matriz de extração foi calculada com os dados de extração, utilizando o código disponível no anexo 6. Esse cálculo é essencial para a extração dos

coeficientes do polinômio de memória. Após a extração dos coeficientes, calculou-se o modelo do PA, que foi então validado com os dados de validação. O NMSE obtido para um polinômio de 2° grau com uma amostra memorizada foi de -23,57 dB.

Em seguida, o algoritmo foi ajustado para operar com números em vírgula fixa e o número total de bits foi reajustado para atingir a menor resolução possível, buscando o menor NMSE simulado, conforme ilustrado pelo anexo 6. Por ser tratar de um cálculo em vírgula fixa, fez-se necessário uma readequação do resultado obtido entre cada multiplicação de forma a manter a resolução inicial.

3.3 Implementação em FPGA

Essa etapa implementou-se o DPD em FPGA, o que exige a paralelização das operações aritméticas. Em cada ciclo de clock, três operações são realizadas simultaneamente: o sinal atual é elevado ao quadrado, armazenado em um registrador de deslocamento dentro de uma matriz de extração [1], o cálculo do produto de todos os elementos da matriz de extração e a soma dos produtos entre os sinais do mesmo instante e seus respectivos coeficientes. Esse processo se repete P vezes, correspondendo ao grau P do polinômio de memória. Como consequência, a saída do DPD estará incompleta durante os primeiros P ciclos de clock, pois, nesse intervalo, o cálculo depende de amostras de sinais anteriores que ainda não foram processadas, resultando em uma saída parcial. A Figura 7 ilustra como esse processo está dividido entre cada ciclo de clock.

Fonte: Autor

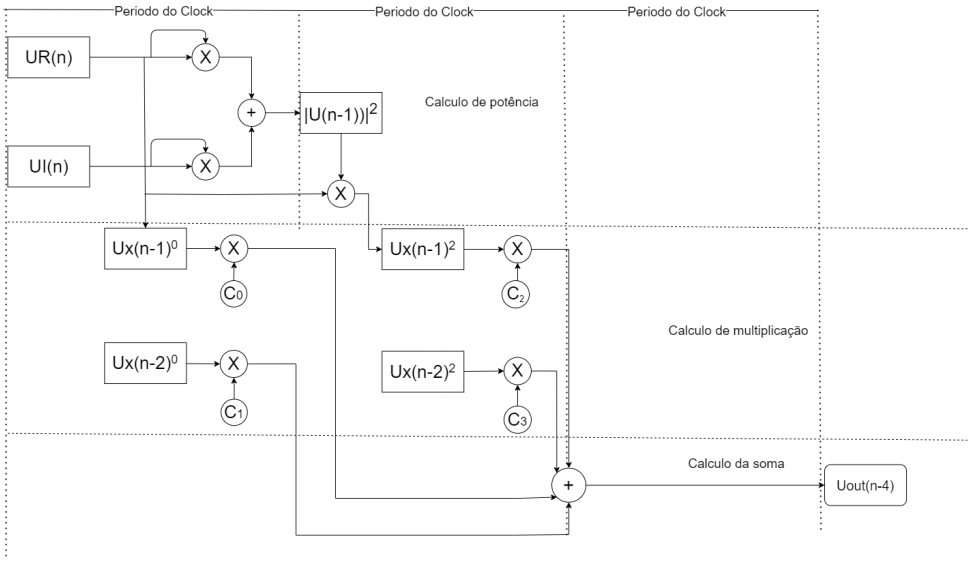

Figura 7 – Processo de cálculo da saída

Conforme exibido no diagrama, cada etapa do processo fornece os dados necessários para a próxima fase do cálculo com um atraso de um ciclo de clock. Contudo, é importante

[1] matriz que contém todas as potências e amostras anteriores necessárias para o cálculo da saída

destacar que o processo completo demanda ciclos adicionais, uma vez que o sinal de saída só é registrado na borda de subida seguinte do clock, garantindo a sincronização adequada no fluxo de dados.

3.3.1 Design e Validação do Circuito Lógico

O design da síntese lógica do circuito DPD foi realizado seguindo o fluxo VLSI, utilizando a tecnologia BiCMOS 130 nm 8HP. O processo abrangeu desde a descrição em alto nível até a validação final do circuito.

Inicialmente, a descrição em VHDL foi estruturada com base em um modelo comportamental que prioriza a eficiência computacional e a paralelização das operações. Essa abordagem assegura que o design inicial seja compatível com as restrições impostas pelas etapas subsequentes de síntese e layout.

A síntese lógica foi conduzida na ferramenta Cadence Genus, utilizando células padrão otimizadas para a tecnologia alvo. Durante essa etapa, foram exploradas configurações alternativas de pipeline e estratégias de paralelização para minimizar atrasos críticos e melhorar a taxa de transferência.

Além disso, relatórios detalhados contendo informações sobre área ocupada, consumo de energia e número de células lógicas utilizadas foram gerados ao final da síntese lógica. Esses relatórios serviram como base para avaliar a eficiência do circuito e orientar ajustes nas etapas subsequentes, garantindo que o design final atendesse às especificações do projeto de forma otimizada.

Após a síntese, os arquivos Verilog e SDF gerados foram analisados em simulações pós-síntese com a ferramenta Cadence NCLaunch. A principal preocupação foi verificar o impacto do atraso nas operações críticas do DPD, validando o funcionamento lógico sob as condições especificadas de temporização.

4 Resultados e Discussão

Conforme mencionado no capítulo 3, o desenvolvimento deste trabalho foi dividido em quatro etapas. A primeira etapa envolveu o estudo dos DPDs e dos métodos de modelagem associados. Na segunda etapa, essa modelagem foi implementada em software utilizando a linguagem Python. A terceira etapa consistiu na implementação do modelo de DPD selecionado em hardware, empregando a linguagem VHDL. Por fim, na quarta etapa, foi realizada a síntese lógica para o design do circuito integrado. Este capítulo apresenta os resultados obtidos ao longo do desenvolvimento do projeto.

4.1 Modelagem do PA

Para fazer a modelagem em software foi utilizada a linguagem de programação Python. Para isso, separou-se os dados citados na seção 3.2 do capítulo 3, em dados de extração e dados de validação, os quais são utilizados para extração dos coeficientes do modelo do MP e validação do modelo encontrado, respectivamente. Para fazer a validação do modelo utilizou-se a métrica do NMSE, que consiste em calcular o erro quadrático médio do valor medido pelo VSA para o valor calculado pelo modelo. Portanto, quanto menor o NMSE mais fiel é o modelo do PA. Nesta etapa obteve-se um NMSE de -23.57 dB, para cálculos em vírgula flutuante, cujo o resultado está presente no gráfico da figura 8.

4.2 Apuração dos números de bits e resolução do sinal

Após concluída a modelagem matemática, foi feita a modelagem do DPD para então ser feito o levantamento da quantidade de bits necessários para a implementação do DPD em hardware minimizando os erros de quantização. Para isso foi necessário refazer a extração dos coeficientes, mas desta vez com os dados normalizados para valores de 0 a 2^{bits}. O resultado desse levantamento está presente no gráfico na figura 9.

Neste gráfico observa-se dois conjuntos de amostras, onde as amostras em azul apresenta a quantidade total de bits contando com os bits de overflow necessários para as operações de multiplicação, enquanto a curva em vermelho representa a quantidade de bits de resolução do sinal. Analisando este gráfico observou-se que não existem ganhos significativos no erro a partir de 7 bits, portanto foi feita a modelagem do PA utilizando uma resolução de 8 bits, cujo o resultado alcançado está ilustrado pela figura 10.

Fonte: Autor

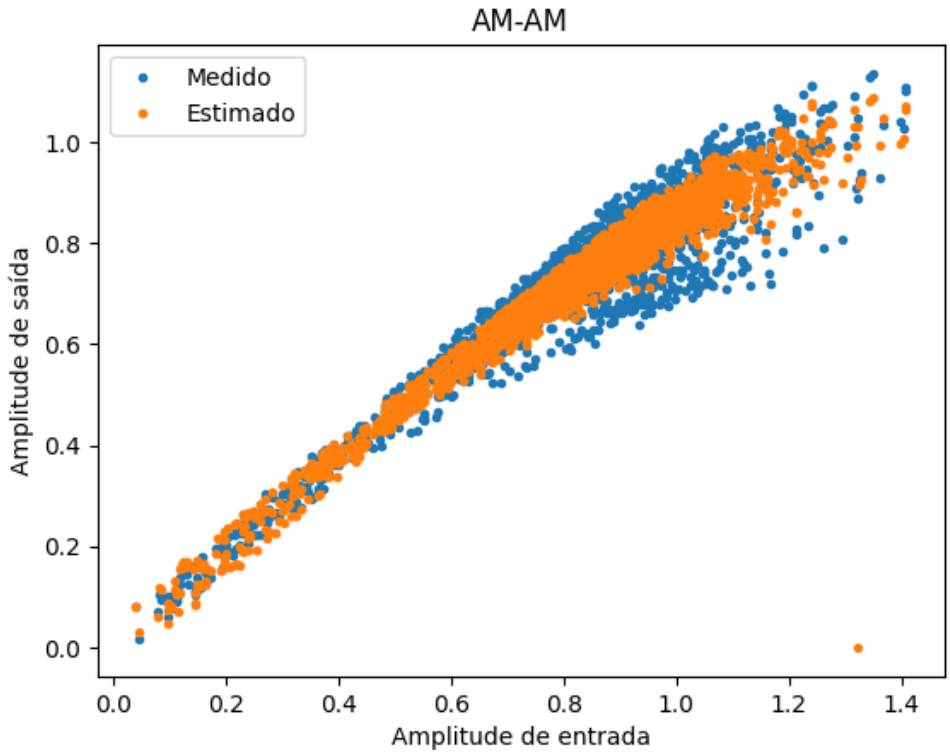

Figura 8 – Modelo do PA em vírgula flutuante

Fonte: Autor

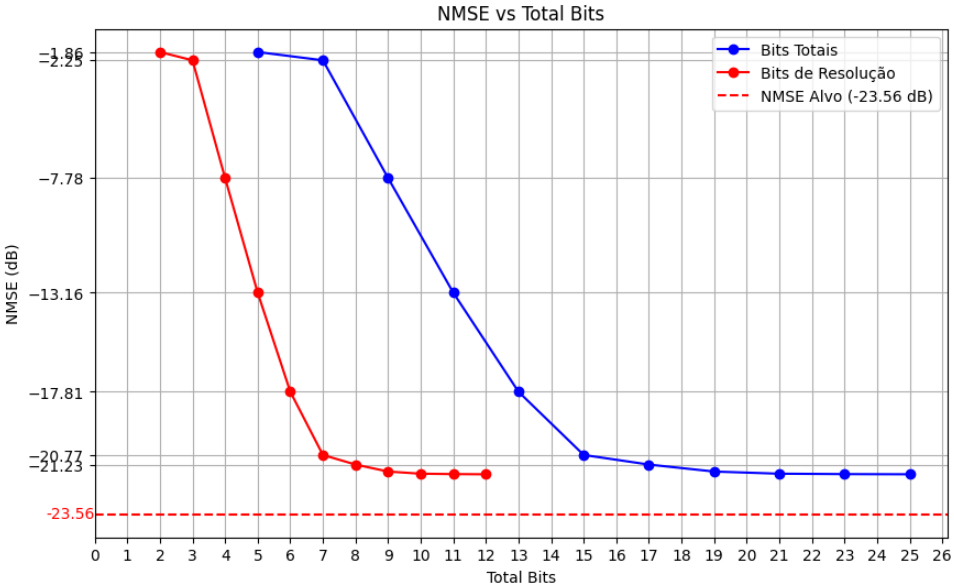

Figura 9 – Gráfico Número de bits x NMSE

Fonte: Autor

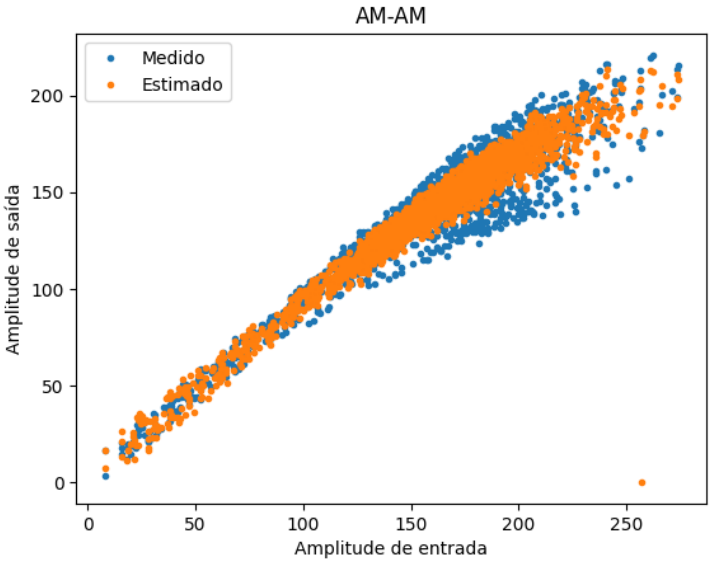

Figura 10 – Modelo do PA em vírgula fixa

4.3 Modelagem do DPD

A partir dos resultados obtidos foi possível fazer a modelagem do DPD; para isso foi feito o mesmo processo de modelagem do PA, porém para alcançar a característica de transferência inversa PA foi invertida a ordem dos dados de entrada e saída para extração dos coeficientes do DPD. O resultado desta modelagem está ilustrado pela figura 11.

Fonte: Autor

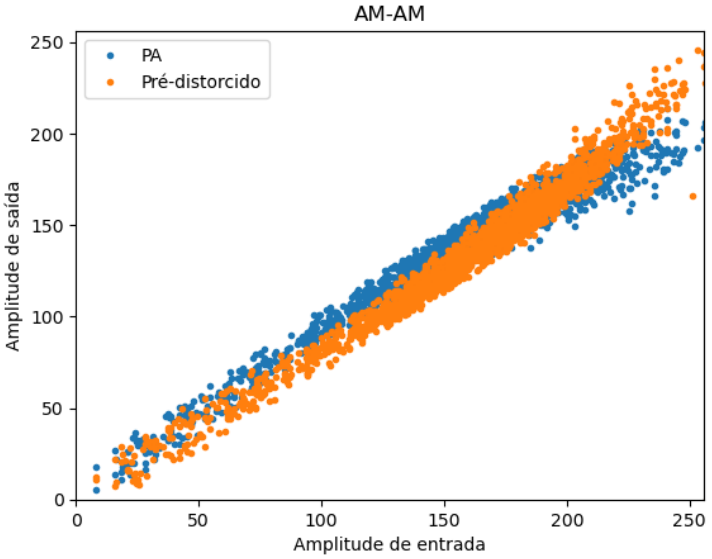

Figura 11 – Modelo do DPD em vírgula fixa

4.4 Implementação em FPGA

Posteriormente, foi desenvolvido o código em VHDL para a implementação em FPGA; nessa implementação, cada operação aritmética é realizada de maneira síncrona, e o fluxo dos cálculos desse processo está ilustrado no diagrama da figura 12.

Fonte: Autor

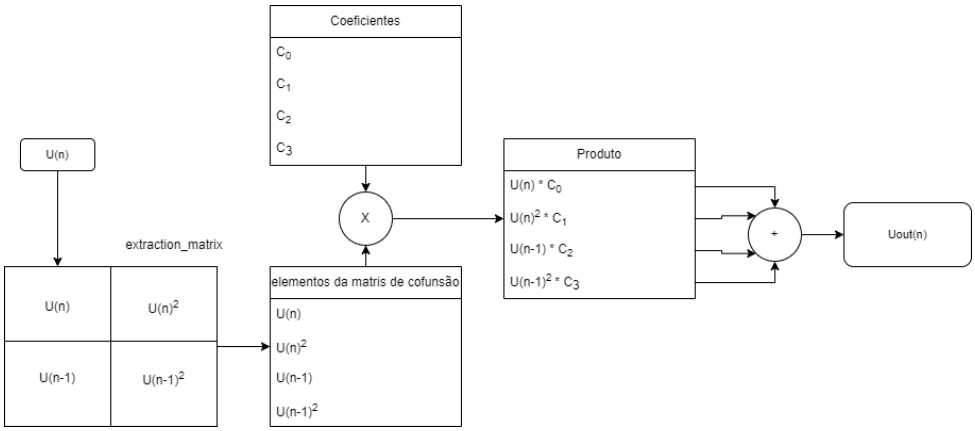

Figura 12 – Fluxo de cálculo FPGA

No primeiro ciclo de clock, o sinal de entrada é registrado e, em seguida, elevado ao quadrado em P^1 graus do polinômio. Após essa etapa, o resultado é adicionado a um buffer de matriz de extração, que processa todos os sinais da amostra. Por fim, cada valor é multiplicado pelos seus respectivos coeficientes e somado, compondo o sinal de saída.

Essa descrição de hardware foi implementada FPGA Virtex5 XC5VLX50T, operando a uma frequência de 62,5 MHz, cujo os recursos lógicos estão sendo mostrados na tabela 1 a seguir.

Recursos	Unidade	Percentual
Registradores	150	1%
LUTs	692	2%
DSP48E	4	8%

Tabela 1 – Utilização dos recursos do FPGA no projeto analisado.

Para fazer essa simulação foi utilizado um *testbench* com as mesmas entradas simuladas no python. Essa simulação foi feita no Xilinx ISE cujo o resultado está ilustrado na Figura 13.

O *testbench* empregado nesta simulação gera um arquivo de texto contendo os sinais de saída, permitindo a validação desses sinais calculados pela FPGA simulados no ISE em comparação com os resultados calculados em Python. A diferença entre os sinais foi avaliada utilizando a métrica NMSE (-16,77 dB). A figura 14 ilustra o resultado dessa

Fonte: Autor

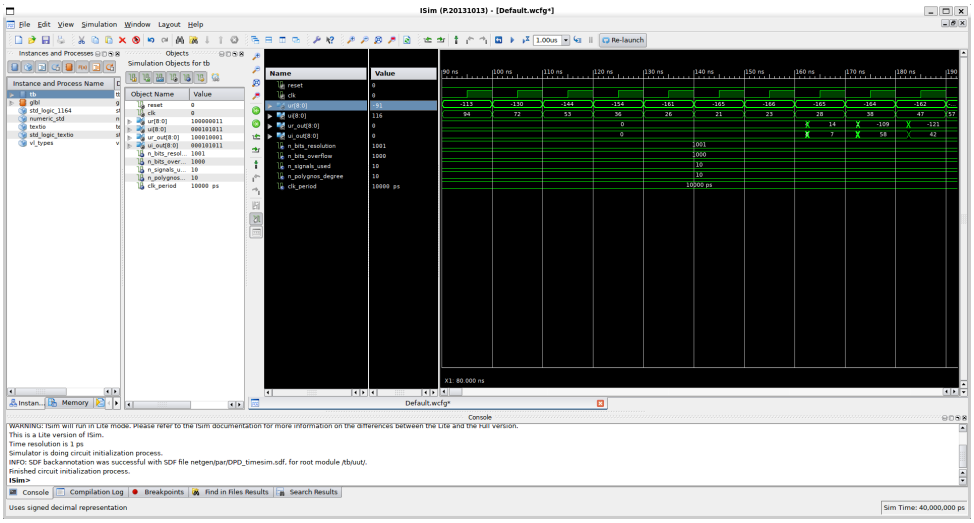

Figura 13 – Simulação ISE

Fonte: Autor

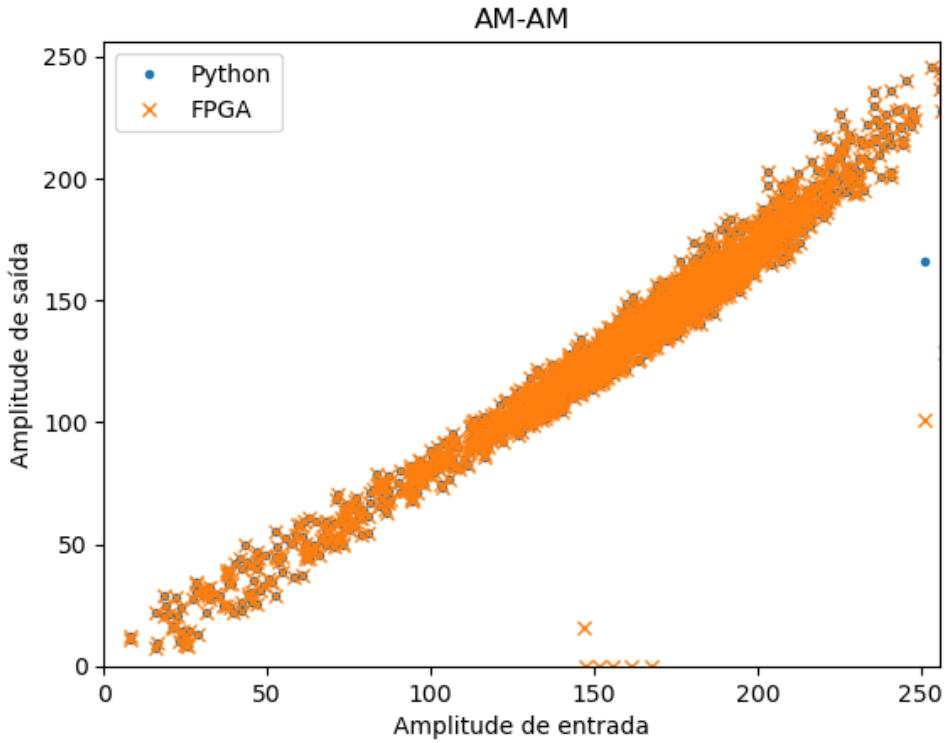

Figura 14 – Simulação FPGA

comparação.

4.5 Síntese lógica

Por fim foi feito a síntese lógica do circuito e a simulação pós síntese lógica. A primeira parte consistiu na síntese lógica no Genus cujo o resultado está disponível na Figura 15. Já os relatórios de consumo, área e afins estão disponíveis na tabela 2.

Fonte: Autor

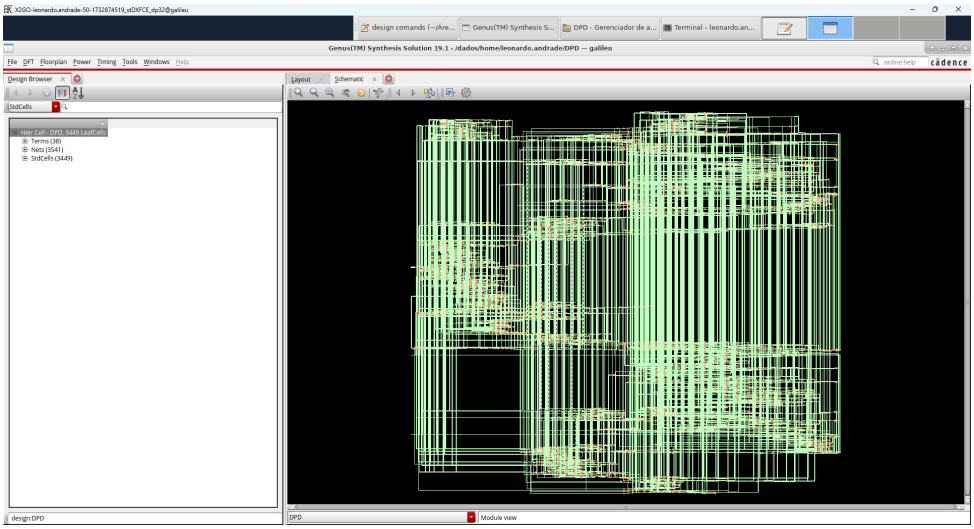

Figura 15 – Circuito lógico

Recursos	Unidade
Células lógica	1567
Consumo	1.6 mW
Área	28116 μm^2

Tabela 2 – Utilização dos recursos de Células Lógicas.

Esse circuito foi simulado utilizando o mesmo *testbench* empregado na etapa anterior de implementação na FPGA, porém desta vez no ambiente NcLaunch do Cadence, conforme ilustrado na figura 16. A simulação gerou um arquivo de texto com os sinais de saída, permitindo a comparação e validação dos resultados calculados em Python e simulados na FPGA pelo ISE. A métrica NMSE também foi aplicada nessa simulação para validar os dados obtidos no NcLaunch em relação aos calculados em Python, resultando em um NMSE de -17,53 dB. A figura 17 apresenta o resultado dessa comparação.

Esse circuito foi simulado atuando a uma taxa de operação de 33,34 MHz, ou seja, a síntese lógica apresentou um desempenho pior que o apresentado pela FPGA.

[1] P representa o grau máximo do polinômio utilizado no modelo matemático, que define o número de termos não lineares considerados no processamento do sinal.

Fonte: Autor

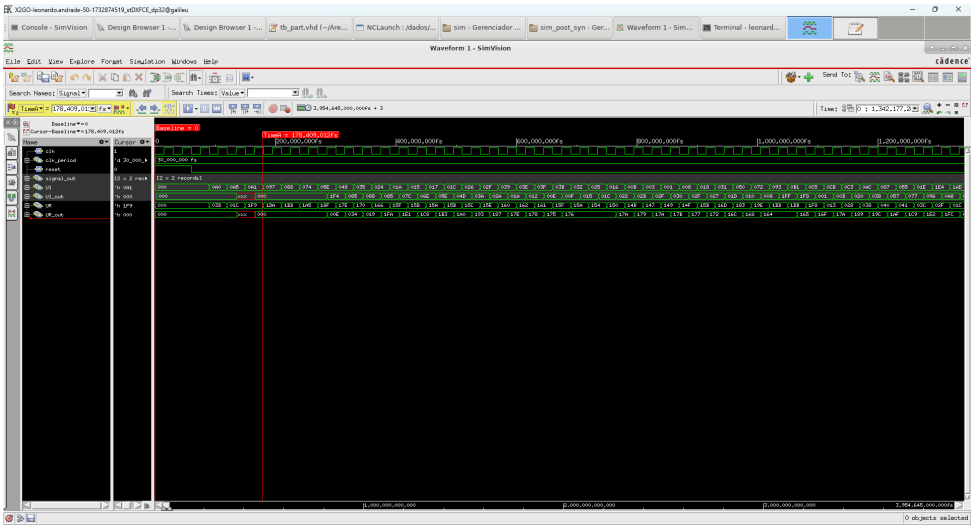

Figura 16 – Simulação NcLaunch

Fonte: Autor

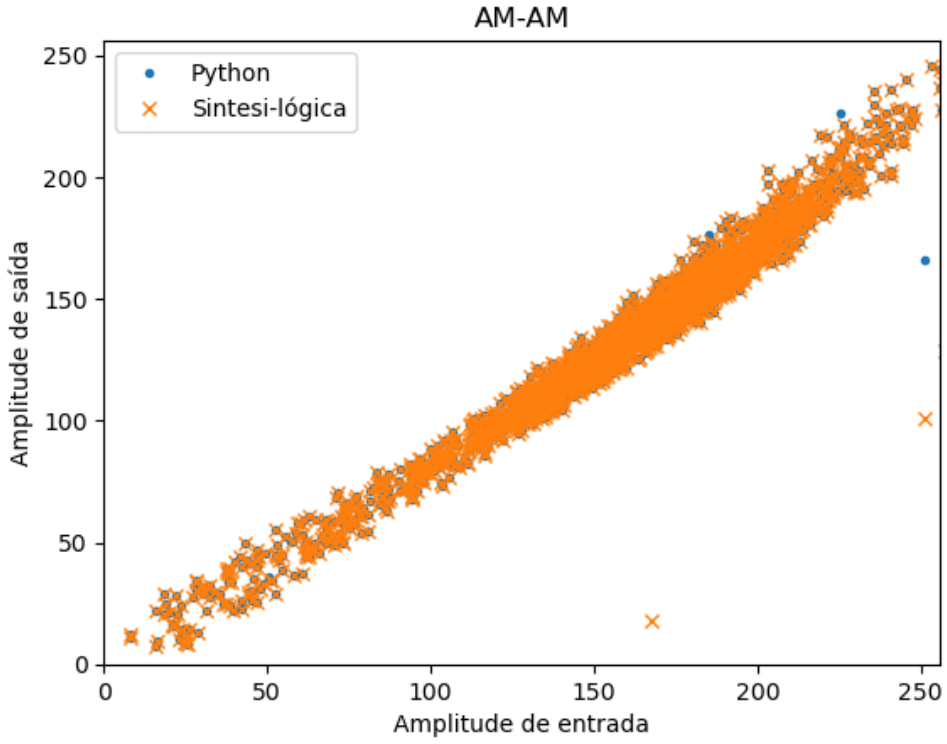

Figura 17 – Comparação dos sinais calculados em Python e no NcLaunch

5 Conclusão

A evolução dos sistemas de comunicação sem fio tem promovido a implementação de diversos serviços móveis, tornando essencial que esses sistemas operem com máxima eficiência. Nesse cenário, a implementação de um DPD em cascata com o PA surge como uma alternativa de baixo custo e interessante para melhorar o desempenho desses sistemas. O objetivo deste trabalho de conclusão de curso foi implementar em hardware um DPD baseado no modelo de Polinômio de Memória. Para isso, o projeto foi dividido em quatro etapas: estudo do DPD e da modelagem matemática, modelagem do DPD em software, implementação do DPD em hardware e, finalmente, design do circuito integrado. Sendo assim a primeira etapa de desenvolvimento do projeto foi a modelagem do PA em vírgula flutuante, utilizando o método do MP, para fazer essa modelagem utilizou-se um polinômio de 2° grau com uma amostra de memória, para fazer a validação dessa modelagem utilizou-se a métrica do NMSE. Nesta etapa obteve-se um NMSE de -23,57 dB, a próxima etapa consistiu em otimizar a quantidade de células lógicas utilizadas no processo limitando o número de bits utilizados. Nesta etapa observou-se que a partir de 8 bits, não havia melhora expressiva no NMSE, assim, essa foi a resolução em bits utilizadas para a amostragem de sinais. Em seguida foi feita a modelagem do DPD em software o qual apresentou um comportamento inverso em relação ao do PA. Após isso foi feita a implementação do circuito em VHDL e validação em FPGA Virtex5 XC5VLX50T, que utilizou um total de 150 registradores, 692 LUTs e 4 unidades DSP48E, operando a uma frequência de 62,5 MHz. Em seguida seguiu-se para a etapa de síntese lógica a qual resultou em um circuito com 1567 células lógicas, com uma area total de 28116 μm^2 e um consumo de energia de 1.6 mW, atuando a uma taxa de operação de 33,34 MHz. Conclui-se, portanto, que o projeto alcançou os resultados esperados, com uma implementação eficaz do DPD em hardware, exceto pela taxa de operação da síntese lógica esperou-se uma melhor performance, o que não foi observado nas simulações.

6 REFERÊNCIAS

1 Elton John, “Modelagem comportamental de amplificadores de potência de radiofrequência usando termos unidimensionais e bidimensionais de séries de Volterra", 2016.

2 Peter Kenington, “High Linearity RF Amplifier Design", 2000.

3 Steve Cripps, “RF Power Amplifiers for Wireless Communications", 2006.

4 Joel Huanca Chavez, “Estudo comparativo entre as arquiteturas de identificação de pré-distorcedores digitais através das aprendizagens direta e indireta", 2018.

5 Volnei Pedroni, “Eletrônica Digital e VHDL ", 2010.

6 Eduardo Gonçalves de Lima and Giovanni Ghione, “Behavioral modeling and digital base-band predistortion of RF power amplifiers", 2009.

7 Luis Schuartz and Eduardo Lima, “Polinômios com Memória de Complexidade Reduzida e sua Aplicação na Pré-distorção Digital de Amplificadores de Potência", 2017.

8 Elton J Bonfim and Eduardo G De Lima, “A Modified Two Dimensional Volterra-Based Series for the Low-Pass Equivalent Behavioral Modeling of RF Power Amplifiers", vol. 47, pp. 27-35, 2016.

9 Wayne Wolf, “Modern VLSI Design: IP-Based Design, Fourth Edition", Prentice Hall Modern Semiconductor Design Series.

10 Dipankar Raychaudhuri and Narayan B. Mandayam, “Frontiers of Wireless and Mobile Communications", *Proceedings of the IEEE*, vol. 100, no. 4, pp. 824-840, April 2012, doi: 10.1109/JPROC.2011.2182095.

11 Pedro Silva, “Combinação entre pré-distorção digital e redução de fator de crista para linearização de amplificadores de potência para sistemas de telecomunicações móveis", 2013.

12 Eduardo Gonçalves de Lima and Giovanni Ghione, “Behavioral modeling and digital base-band predistortion of RF power amplifiers", 2009.

13 Neil H.E.Weste and David Money Harris, “CMOS VLSI Design: A Circuits and Systems Perspective (4th Edition)", 2010.

14 Volnei Pedroni, “Circuit design with VHDL", 2020.

Anexos

ANEXO A - função que calcula matriz de extração em vírgula flutuante

```
def mp(P, M, xn):
    L = xn.shape
    XX = np.zeros((L[0] - M, P * (M+1)), dtype=np.complex128)
    for l in range(M+1, L[0]):
        for p in range(1, P+1):
            for m in range(0, M+1):
                XX[l-M-1, ((p-1)*(M+1))+m] = (np.abs(xn[l-m])
                   **(2*p-2)*(xn[l-m]))[0]
    return XX

XX_ext = mp(P, M, in_data_ext)
coefficients, _, _, _ = np.linalg.lstsq(XX_ext, out_data_ext[M:],
    rcond=None)
predicted_val = XX_val @ coefficients
```

ANEXO B - Função que calcula matriz de confusão em vírgula fixa

readeq = lambda val, precision: np.floor(val / (2 ** precision))

```
1   def mp_int(P, M, xn, bits):
2   L = xn.shape
3   XX = np.zeros((L[0] - M, P * (M+1)), dtype=np.complex128)
4   for l in range(M+1, L[0]):
5       for p in range(1, P+1):
6           for m in range(0, M+1):
7                   A = np.real(xn[l-m])[0]
8                   B = np.imag(xn[l-m])[0]
9                   modulo_power = 2**bits
10                  modulo_square = readeq(A ** 2 + B ** 2, bits)
11                  for _ in range(1, p):
12                      modulo_power = readeq(modulo_power *
                            modulo_square, bits)
13                  real_part = readeq(A * modulo_power,bits)
14                  imag_part = readeq(B * modulo_power,bits)
15                  XX[l-M-1, ((p-1)*(M+1))+m] = complex(
                        real_part,imag_part)
16  return XX
```