Universidade Federal de Juiz de Fora
Faculdade de Engenharia
Programa de Pós-Graduação em Engenharia Elétrica

Luiz Antônio Corrêa Júnior

Técnicas de Modelagem e Pré-Distorção Utilizando FPGA

Juiz de Fora
2018

Luiz Antônio Corrêa Júnior

Técnicas de Modelagem e Pré-Distorção Utilizando FPGA

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Juiz de Fora, na área de concentração Sistemas Eletrônicos , como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Daniel Discini Silveira

Juiz de Fora

2018

Ficha catalográfica elaborada através do Modelo Latex do CDC da UFJF
com os dados fornecidos pelo(a) autor(a)

Corrêa Júnior, Luiz Antônio.
Técnicas de Modelagem e Pré-Distorção Utilizando FPGA / Luiz Antônio Corrêa Júnior. – 2018.
93 f. : il.

Orientador: Daniel Discini Silveira
Dissertação de Mestrado – Universidade Federal de Juiz de Fora, Faculdade de Engenharia. Programa de Pós-Graduação em Engenharia Elétrica, 2018.

1. Identificação de Sistemas. 2. Pré-Distorção. 3. FPGA. I. Discini Silveira, Daniel, Técnicas de Modelagem e Pré-Distorção Utilizando FPGA

Luiz Antônio Corrêa Júnior

Técnicas de Modelagem e Pré-Distorção Utilizando FPGA

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Juiz de Fora, na área de concentração Sistemas Eletrônicos , como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica.

Aprovada em:

BANCA EXAMINADORA

Prof. Dr. Daniel Discini Silveira - Orientador
Universidade Federal de Juiz de Fora

Professor Dr. Marco José da Silva
Universidade Tecnológica Federal do Paraná

Professor Dr. Alvaro Augusto Machado de Medeiros
Universidade Federal de Juiz de Fora

AGRADECIMENTOS

Primeiramente, agradeço a Deus, que pela interseção de Nossa Senhora Aparecida, padroeira do Brasil, me abençoa e abençoou durante essa trajetória.

Aos meus pais, Luiz e Imaculada, todo o meu agradecimento. Eles sempre acreditaram em mim, e durante toda a minha trajetória acadêmica me apoiaram e nunca me deixaram sozinho. Aos meus irmãos, Rodrigo e Lincoln, que sempre me apoiaram nos momentos em que eu precisava.

A minha amada esposa, Ana, que esteve ao meu lado durante este mestrado me dando todo o seu apoio, sem o qual, a minha caminhada seria mais difícil. Ao meu filho, Luiz Eduardo, que eu ainda não conheço, mas já amo demais.

Aos meus companheiros do Laboratório de Instrumentação e Telemetria (LITel) que me apoiaram durante o meu mestrado. Agradeço, também, aos meus colegas da Universidade Federal de Ouro Preto (UFOP) que sempre me ajudaram durante a graduação, e cito nominalmente João Paulo que me acompanhou durante o mestrado na Universidade Federal de Juiz de Fora (UFJF).

Minha gratidão especial ao meu orientador, Prof. Dr. Daniel Discini. Obrigado pela sua dedicação e profissionalismo.

Gostaria de agradecer também ao programa de pós-graduação da Universidade Federal de Juiz de Fora (PPEE/UFJF) pelo apoio para a realização desse projeto.

"O homem precisa de conhecimento, precisa de verdade, porque sem ela não se mantém de pé, não caminha. Sem verdade, a fé não salva, não torna seguros os nossos passos."
(V.S. Papa Francisco, Lumen Fidei, 2013)

RESUMO

Esta dissertação descreve, inicialmente, um sistema implementado em FPGA para a geração de modulações digitais. Posteriormente, descreve uma plataforma capaz de realizar a estimação de modelos com coeficientes lineares e demonstra aplicações de pré-distorção.

O trabalho começa com uma breve visão sobre os conceitos básicos envolvidos na dissertação. É descrito como se comportam os sinais e os sistemas no domínio discreto, é feito um resumo sobre o que é um FPGA e, após isso, são analisadas figuras de mérito, ferramentas matemáticas utilizadas para medir a qualidade dos modelos.

No capítulo 3, é apresentada uma ferramenta em FPGA capaz de gerar sinais com diversas modulações digitais que serão utilizadas na identificação e pré-distorção.

No capítulo 4 é abordado a construção e validação de um algoritmo de estimação eficiente utilizando o método de Mínimos Quadrados. A identificação de um filtro, utilizando um sinal de trem de impulsos, é mostrado na seção 4.2. A resposta do filtro no domínio da frequência é comparada com a resposta medida por um equipamento comercial, onde os resultados mostram que o modelo obtido representa bem um filtro real.

A criação de um modelo de pré-distorção é mostrado no capítulo 5. A qualidade do modelo é avaliada por quatro Figuras de Mérito diferentes.

As conclusões sobre o trabalho e as pesquisas futuras são apresentadas no último capítulo.

Palavras-chave: FPGA. Identificação de Sistemas. Pré-Distorção. Processamento de Sinais.

ABSTRACT

This work describes initially a system implemented in FPGA for the generation of a digital modulations. After, it describes a platform capable of performing the estimation of a device under test. Finally, it shows applications of this platform for pre-distortion purposes.

A brief overview of the basic concepts involved in the dissertation are presented, as signals and systems in the discrete domain, FPGA, and Figures of Merit, tools used to measure the quality of the models.

In chapter 3, a FPGA tool capable of generating signals with various digital modulations is presented. This platform will lately be used for identification and pre-distortion purposes.

Chapter 4 shows the construction of efficient estimation algorithm using the Least Squares method. The identification of a filter using a pulse wave signal is shown in the section 4.2, and the resulting frequency domain curve is compared to that generated by a commercial equipment, where results show that the estimated model represents well a real filter.

A pre-distortion model of a filter is shown in the chapter 5. The quality of the model is evaluated by four different Figures of Merit.

Conclusions about the work and future research are drawn in the last chapter.

Key-words: FPGA. Pre-distortion. Signal Processing. System identification.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

8-PSK	*Eight Phase-Shift Keying*
16-PSK	*Sixteen hase-Shift Keying*
4-QAM	*Four Quadrature Amplitude Modulation*
16-QAM	*Sixteen Quadrature Amplitude Modulation*
32-QAM	*Thirty Two Quadrature Amplitude Modulation*
64-QAM	*Sixty Four Quadrature Amplitude Modulation*
ADC	*Analog-to-Digital Converter*
ASIC	*Aplication Specific Integrated Circuit*
BER	*Bit Error Rate*
BPSK	*Binary Phase-Shift Keying*
DAC	*Digital-to-Analog Converter*
DC	*Direct Current*
DSP	*Digital Signal Processor*
DUT	*Device Under Test*
EVM	*Error Vector Magnitude*
FFT	*Fast Fourier Transform*
FIR	*Finite Impulse Response*
FPGA	*Field Programmable Gate Array*
GPIO	*General Purpose Input/Output*
IIR	*Infinite Impulse Response*
LED	*Light Emitting Diode*
LIT	Linear Invariante no Tempo
LUT	*Look up Table*
MAE	*Mean Absolute Error*
MQ	Mínimos Quadrados

NMSE	*Normalized Mean Square Error*
PN	*Pseudo-Noise*
QPSK	*Quadrature Phase-Shift Keying*
RO	*Roll-Off*
SCPI	*Standard Commands for Programmable Instruments*
XOR	*Exclusive Or*

LISTA DE SÍMBOLOS

Ω	Resistência em Ohm
$\sum$	Somatório
∞	Infinito
$E[\cdot]$	Operador esperança matemática
$max[\cdot]$	Máximo valor
I	Sinal em fase
Q	Sinal em quadratura

SUMÁRIO

1 INTRODUÇÃO

O crescimento da renda da população e a popularização de dispositivos com acesso a internet fizeram com que os sistemas de telecomunicações modernos demandassem cada vez mais velocidade de transmissão de dados [2]. Esta pode ser traduzida em uma maior largura de banda, o que faz o estudo da transmissão de dados em banda larga importante no contexto atual. Para suprir essa demanda de dados, sistemas modernos são projetados para operar com modulações digitais [3].

Toda transmissão de dados é feita através de um canal [4] e dentro desse canal podem estar inseridos dispositivos que alteram o sinal transmitido. Em um sistema de telecomunicações, o canal pode ser o ar, um cabo ou outro meio pelo qual o sinal é enviado. Dispositivos podem ser amplificadores de potência, filtros, transformadores, etc. Esses formam um sistema de transmissão de informação. Esse dispositivo pode ser representado por um modelo discreto que descreverá o sistema real através de uma formulação matemática [5], a qual é totalmente descrita através de seus coeficientes se o sistema for Linear Invariante no Tempo (LIT) [6]. A Figura 1 representa essa modelagem, em que um bom modelo minimiza o erro.

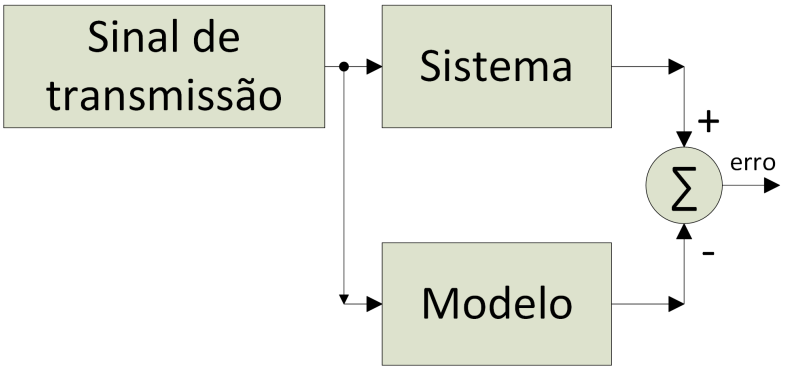

Figura 1 – Sistema genérico de telecomunicações e sua representação por um modelo equivalente.

Este trabalho tem por objetivo apresentar a construção de uma plataforma em *Field Programmable Gate Array* (FPGA) que permita o modelamento de dispositivos utilizados na cadeia de transmissão de dados, tanto passivos (filtros, acopladores) como ativos (amplificadores), e também aplicações de pré-distorção para corrigir as alterações provocadas pelos dispositivos. A construção dessa plataforma pode ser separada em três etapas:

(i) Construção de um gerador de sinais em alta velocidade utilizando o FPGA.

(ii) Criação um algoritmo em MATLAB que permita a identificação de sistemas através de um conjunto de dados.

(iii) Aplicar uma pré-distorção no sinal transmitido pelo FPGA.

Primeiramente, foi desenvolvida uma plataforma capaz de gerar sinais modulados digitalmente em alta velocidade utilizando um kit de desenvolvimento de um FPGA [1], combinado com uma placa para conversão Digital-Analógica (DAC) e Analógica-Digital (ADC) [7]. Esse kit é o DE2-115 da Altera, juntamente com uma placa ADC/DAC proprietária. Em trabalhos anteriores [8] [9], foram criados sistemas para geração de múltiplas modulações e para avaliação da qualidade do sinal transmitido. Nessa dissertação, aumentaram-se as possibilidades de modulações digitais e a largura de banda do sinal transmitido. O desempenho do sinal gerado é comparado com um kit comercial, como demonstrado na Figura 2, utilizando figuras de mérito. Como parâmetro de comparação é utilizado um equipamento de referência. O desempenho do kit FPGA deve ser equivalente ou superior ao desse equipamento.

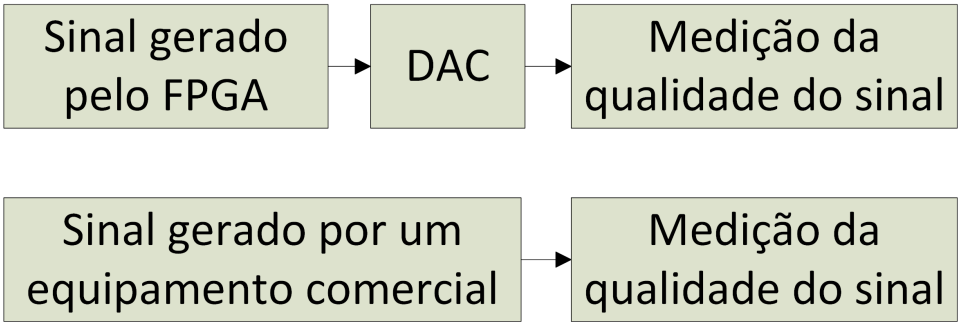

Figura 2 – Medição de desempenho de um sinal gerado pelo FPGA e por um kit comercial.

Como segundo objetivo do trabalho tem-se a criação de uma plataforma para a identificação de sistemas. Pode ser gerado um conjunto de dados em uma tabela , *Look up Table* (LUT) , que representa o modelo do dispositivo a ser identificado [10]. Para isso, o sinal deve ser processado pelo DAC, ser enviado através do canal e dos dispositivos e, após isso, ser recebido no ADC, como mostrado na Figura 3. O sinal enviado e recebido são capturados no FPGA [11], gerando assim uma LUT.

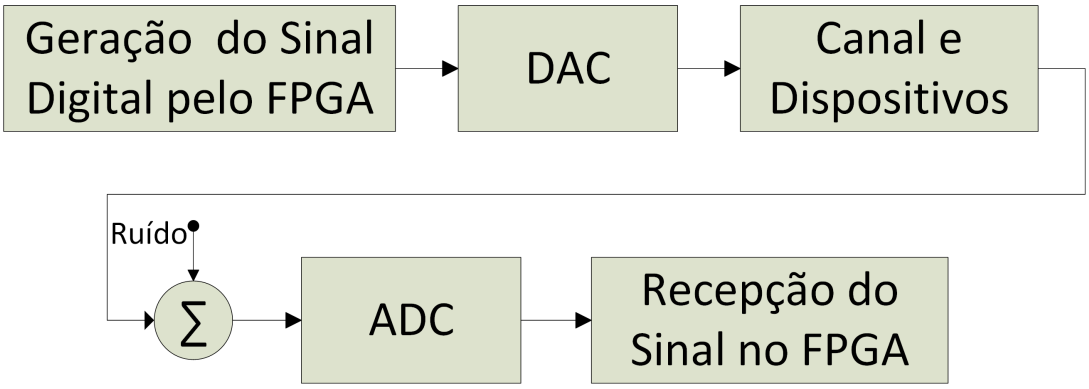

Figura 3 – Setup utilizado para identificação.

Os dados da LUT são processados no MATLAB® para geração de um modelo que deve representar o sistema com fidelidade. Esse modelo é obtido através de uma estimação que utiliza o algoritmo de Mínimos Quadrados (MQ) para a obtenção dos coeficientes.

No terceiro objetivo visa-se a aplicação de uma pré-distorção no sinal transmitido pelo FPGA. Os coeficientes do modelo de pré-distorção são obtidos através do uso de estimação indireta no MATLAB® e são implementados no FPGA. A Figura 4 descreve esse sistema.

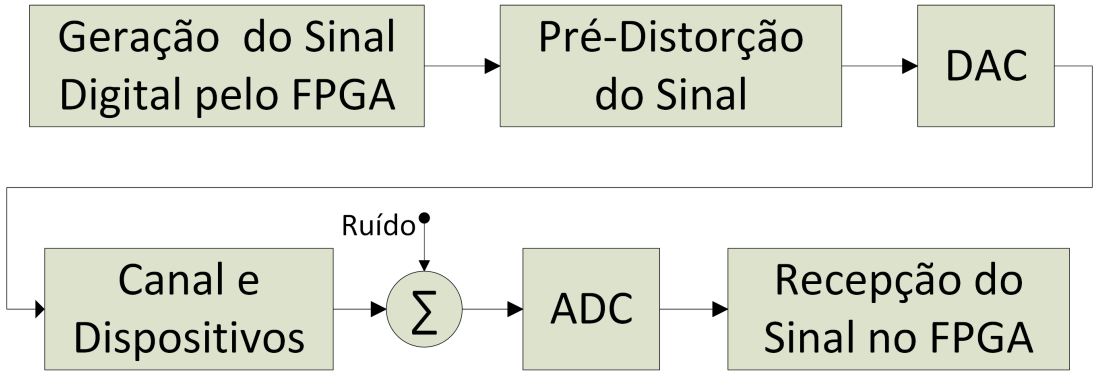

Figura 4 – Setup utilizado para pré-distorção.

A pré-distorção foi implementada no sinal digital antes do DAC e deve compensar as alterações provocadas pelo sistema real. O modelo do sistema deve ser o mais próximo possível do real, para que a pré-distorção seja efetiva no sistema proposto.

1.1 REVISÃO BIBLIOGRÁFICA

Esta seção apresentará algumas referências bibliográficas disponíveis na área de geração de sinais em FPGA, identificação de sistemas e pré-distorção.

O primeiro objetivo é a construção de um gerador de sinais em alta velocidade utilizando o FPGA. Neste trabalho foi gerado um sinal que transmite a uma taxa de 12,5 Mega Símbolos por segundo (MS/s), com 128 possibilidades de modulação digital utilizando essa plataforma.

Na literatura, encontram-se alguns trabalhos que descrevem a implementação de moduladores em FPGA. Em [12], é mostrado um modulador capaz de 3 modulações digitais com uma taxa em kS/s. Em [13], têm-se a geração de um sinal em banda base QPSK com uma taxa de 2,5 MS/s. Uma estrutura com várias modulações transmitidas a uma taxa de 625 S/s é mostrada em [14]. Em [8], o autor construiu uma plataforma que gera sinais cosseno-levantado com até 6,25 MS/s, com 8 possibilidades de taxa de símbolo. Esse trabalho é utilizado como base para criação do gerador de sinais em FPGA.

O segundo objetivo é a criação um algoritmo em MATLAB que permita a identificação de sistemas através de um conjunto de dados. A identificação de sistemas pelo método de Mínimos Quadrados (MQ) é um método comumente utilizado [5] a [10]. Esse pode ser implementado em MATAB [15] para avaliar os dados gerados pelo FPGA [11].

O MQ é utilizado em outros trabalhos para a equalização de canais [16] [17] [18]. Nesse trabalho o MQ é utilizado para a identificação de um filtro real e também para estimação de um modelo inverso desse dispositivo.

O filtro Butterworth de ordem 5 pode ser construído na prática utilizando um filtro protótipo [19]. Essa metodologia permite a obtenção dos valores dos componentes através da frequência de corte desejada. Nessa dissertação, é utilizado o MQ para a identificação do filtro real construído a partir de um filtro protótipo.

O terceiro objetivo é a implementação de um modelo inverso no FPGA para a pré-distorção. Esse será implementado na plataforma através de coeficientes obtidos no MATLAB®.

Métodos de estimação indireta são bastante utilizados para estimação Caixa Preta de dispositivos de telecomunicação [20] [21]. Essa metodologia permite a descrição do modelo inverso através dos dados de entrada e saída do dispositivo a ser identificado. Portanto, pode-se utilizar essa metodologia para se obter os coeficientes de um modelo que compense as alterações provocadas pelo filtro nesse trabalho.

A implementação de coeficientes estimados no FPGA é bastante utilizada em diversas áreas [22] [23] [24] [25]. Portanto os coeficientes obtidos por estimação indireta podem ser implementados na plataforma visando a pré-distorção do sinal.

Figuras de mérito são utilizadas para se verificar a similaridade entre dois conjuntos de dados. Em [9] o autor construiu uma plataforma que utiliza a BER para avaliação de canais. Nesse trabalho são utilizadas 4 Figuras de mérito: NMSE, EVM, MAE e BER para a avaliação da qualidade do sinal antes e depois da pré-distorção [26] [27] [28] [29]. O

modelo de pré-distorção é feito através da estimação indireta, onde, os coeficientes obtidos são implementados no FPGA.

1.2 ESBOÇO DA DISSERTAÇÃO

Essa dissertação é composta de 7 capítulos, conforme descrição a seguir:

Capítulo 1: Introdução ao problema de transmissão de dados, descrição da organização da dissertação, e revisão da bibliografia disponível na área.

Capítulo 2: Conceitos básicos de sistemas discretos e contínuos. Descreve-se o que é uma alteração induzida no canal e como um modelo pode ser representado por um sistema recursivo ou não-recursivo. Relata-se o que é um FPGA, os periféricos utilizados no trabalho, e alguns conceitos sobre a operação com números binários. Além disso, são apresentadas as figuras de mérito utilizadas durante o trabalho.

Capítulo 3: Apresenta um algoritmo de geração de sinais digitais implementado em FPGA e os conceitos envolvidos na modulação digital. São descritos, também, os periféricos responsáveis pela conversão dos dados.

Capítulo 4: A implementação do estimador em MATLAB® é retratada neste capítulo. Primeiramente, conceitua-se o que é um estimador Caixa-Preta, após essa etapa, relata-se a implementação desse estimador para sistemas recursivos e não recursivos. Descreve-se como os dados devem ser organizados dentro do algoritmo e a construção do mesmo para ser utilizado na estimação do modelo. É descrito a identificação da alteração induzida no canal, expondo, também, como os coeficientes estimados para pré-distorção são implementados no FPGA. Por fim, compara-se o resultado medido no domínio da frequência em um equipamento comercial com o modelo criado em MATLAB®.

Capítulo 5: Da mesma forma que se pode identificar os coeficientes do modelo, um modelo inverso pode ser construído para a pré-distorção. Essa implementação e a avaliação dos resultados são descritas por este capítulo.

Capítulo 6: Conclusões e algumas considerações sobre trabalhos futuros.

2 CONCEITOS INICIAIS

Neste capítulo, serão revisados alguns conceitos utilizados para a construção desse trabalho.

2.1 SINAIS NO TEMPO CONTINUO E NO TEMPO DISCRETO

Um sinal pode ser representado no tempo contínuo ou no tempo discreto. Um sinal contínuo possui uma representação infinita no tempo, enquanto o discreto uma representação limitada. A Figura 5 mostra um sinal contínuo $x(t)$ e a sua versão discretizada x[n], onde o sinal discreto é gerado através da multiplicação do sinal contínuo por um trem de impulsos $p(t)$, gerando assim, a versão amostrada do sinal $x_a(t)$. A recriação do sinal contínuo por um sinal digital $x[n]$ pode ser feita por um retentor de ordem zero, criando, assim, uma versão analógica do sinal $x_r(t)$. T é o período de amostragem.

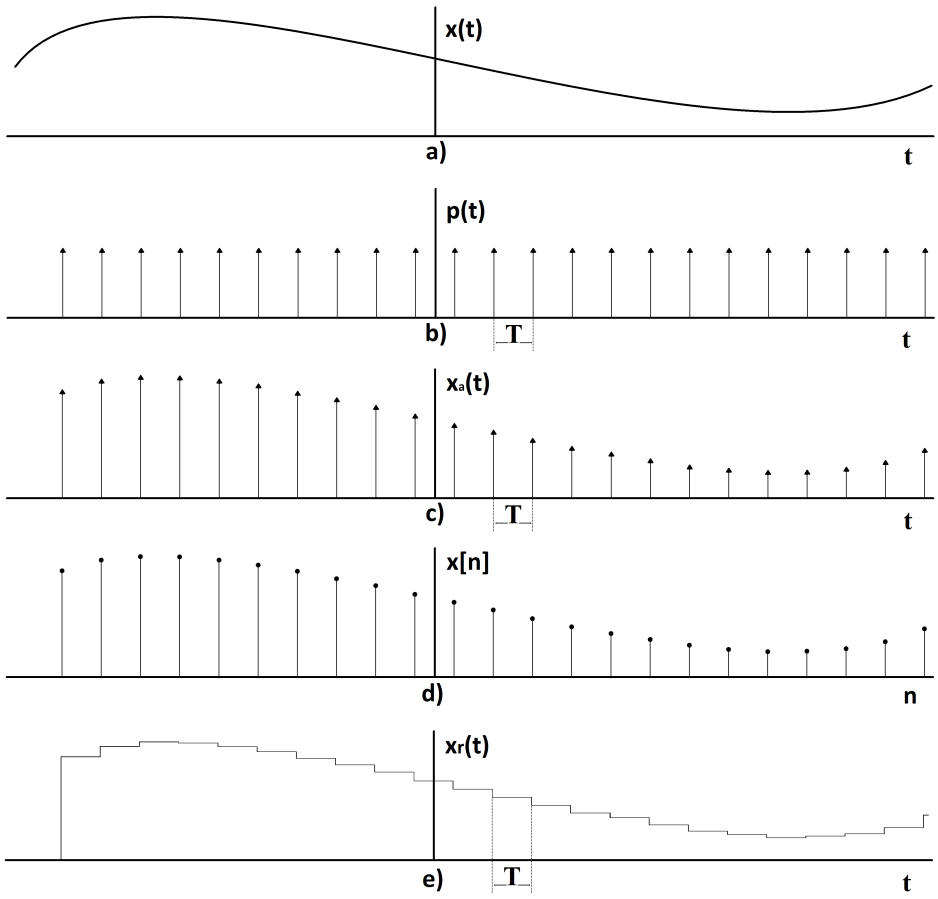

Figura 5 – Representação no tempo do: a) Sinal analógico; b) amostrador; c) sinal amostrado; d) sinal discretizado; e) sinal analógico reconstruído pelo retentor de ordem zero.

A representação de um sinal contínuo por um sinal discreto através da amostragem é bastante útil quando necessita-se processar esse sinal em um processador. Essa represen-

tação não é perfeita, e ocasiona distorções em um sinal. Podem haver ruídos impulsivos, ruídos de fundo, além de erros de quantização.

Uma ferramenta útil para análise de um sinal é a sua representação através da transformada de Fourier. A análise do sinal é facilitada no espectro das frequências. Em um sinal discretizado, em que os dados estão alocados em posições de memória, é possível obter essa representação através de algoritmos de *Fast Fourier Transform* (FFT) implementados em MATLAB® [4]. Para um sinal contínuo, pode-se utilizar equipamentos específicos que fazem a análise do sinal no espectro como o Analisador Vetorial de Sinais Keysight CXA N9000A [30].

A representação discreta de um sinal contínuo pela amostragem ocasiona uma mudança no espectro do sinal no domínio das frequências. Essa deformação acontece devido à amostragem ser basicamente uma multiplicação por um trem de impulsos com intervalos T. Essa multiplicação no tempo ocasiona uma convolução no espectro de frequências por um trem de impulsos separados por w_s.

Na Figura 6 é possível visualizar que, se um sinal tiver largura de banda B maior do que a metade da frequência de amostragem w_s, os limites do espectro de $|X(\omega)|$ interferirão nos limites dos espectros posteriores. Esse fenômeno impossibilita a recuperação do sinal contínuo por um filtro Passa-Baixas e dificulta a análise do sinal amostrado no tempo discreto [4]. Para evitar que isso aconteça, utiliza-se o critério de Nyquist para amostrar um sinal contínuo. Esse critério define que a frequência de amostragem deve ser duas vezes maior que a largura de banda de um sinal contínuo [31].

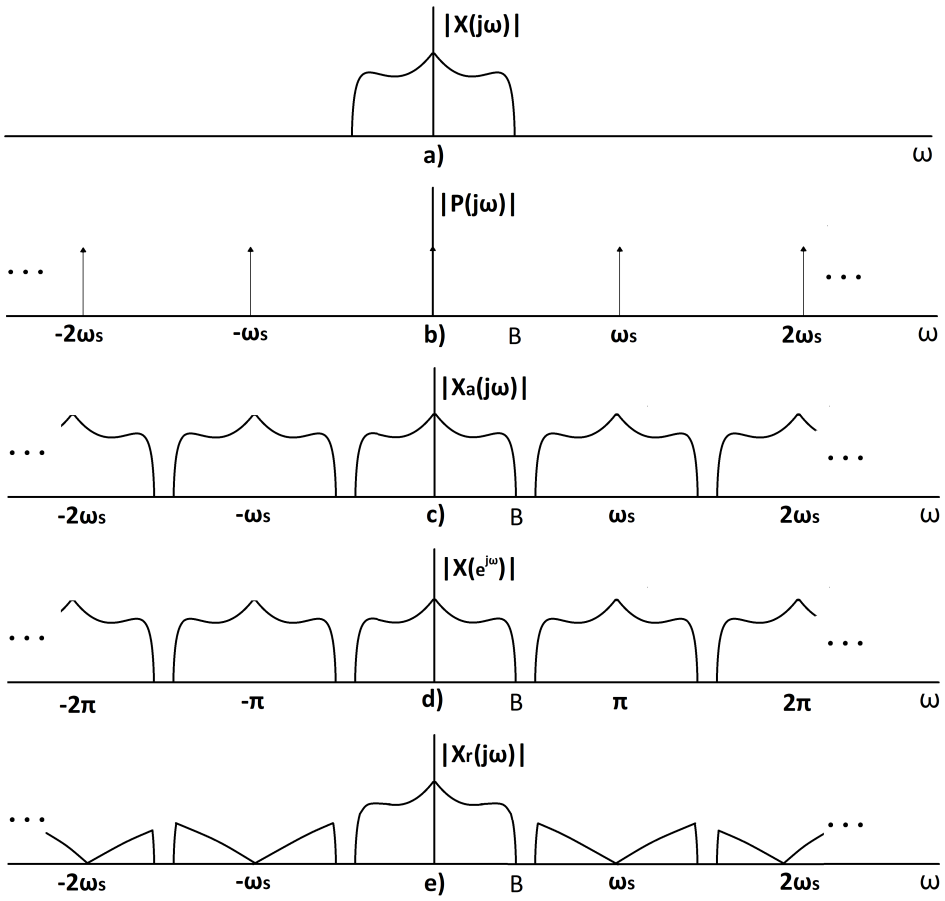

Figura 6 – Representação no espectro do: a) Sinal analógico; b) amostrador; c) sinal amostrado; d) sinal discretizado; e) sinal analógico reconstruído pelo retentor de ordem zero.

A equação que define o critério de Nyquist pode ser vista em (2.1).

$$\omega_s > 2B \tag{2.1}$$

2.2 SISTEMAS LINEARES E INVARIANTES NO TEMPO

Um sistema linear invariante no tempo (LIT) é aquele onde as suas características são fixas no tempo [31]. Na equação (2.2), $h(t)$ é um sistema LIT, $x_1(t)$ representa a entrada e $y_1(t)$ a saída.

$$y_1(t) = x_1(t) * h(t) \tag{2.2}$$

Na equação (2.3), $h(t)$ é um sistema LIT, $x_2(t)$ representa outra entrada e $y_2(t)$ a saída dessa nova entrada submetida ao mesmo sistema LIT.

$$y_2(t) = x_2(t) * h(t) \tag{2.3}$$

Em (2.4), as entradas $x_1(t)$ e $x_2(t)$ são somadas, resultando em $x_3(t)$.

$$x_3(t) = x_1(t) + x_2(t) \tag{2.4}$$

Na equação (2.5), $y_3(t)$ é o resultado da entrada $x_3(t)$ submetida ao sistema LIT $h(t)$.

$$y_3(t) = x_3(t) * h(t) \tag{2.5}$$

Para um sistema LIT o tempo onde a entrada ocorreu não importa, portanto, para esse tipo de sistema, a equação (2.6) é verdadeira.

$$y_3(t) = y_1(t) + y_2(t) \tag{2.6}$$

Ligando-se dois sistemas LITs, $H_1(j\omega)$ e $H_2(j\omega)$, a resposta $H(j\omega)$ será o resultado da multiplicação desses dois sistemas LIT como descrito na equação (2.7).

$$H(j\omega) = H_1(j\omega) \times H_2(j\omega) \tag{2.7}$$

2.3 SISTEMAS RECURSIVOS E NÃO-RECURSIVOS

Existem, basicamente, dois tipos de sistemas LIT: Aqueles que possuem resposta finita ao impulso (FIR) e aqueles que possuem resposta infinita (IIR). A resposta de um sistema digital é obtida pelo sinal da saída resultante de um impulso como o descrito na equação (2.8) [32].

$$\delta(k) = \begin{cases} 1,\ x = 0 \\ 0,\ x \neq 0 \end{cases} \tag{2.8}$$

Os sistemas não-recursivos dependem apenas da entrada atual combinada com constantes e as entradas passadas. A resposta desse tipo de sistema ao impulso é finita e a sua ordem é determinada pelo número de coeficientes m.

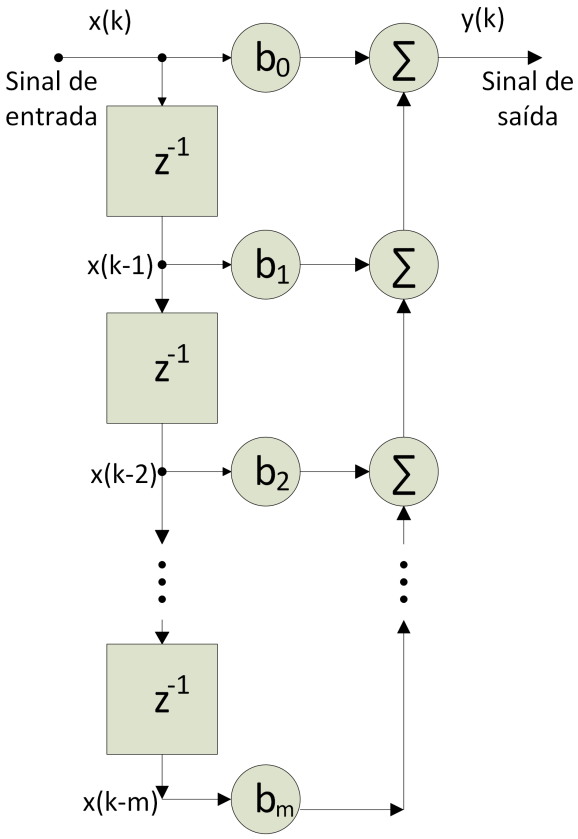

Figura 7 – Estrutura do filtro não-recursivo.

Um sistema com resposta infinita também pode ser chamado de recursivo. A saída de um filtro com resposta infinita resulta da combinação das amostras passadas da entrada digital multiplicadas por constantes a_k com as saídas passadas multiplicadas por constantes b_k. Essa estrutura é mostrada na Figura 8.

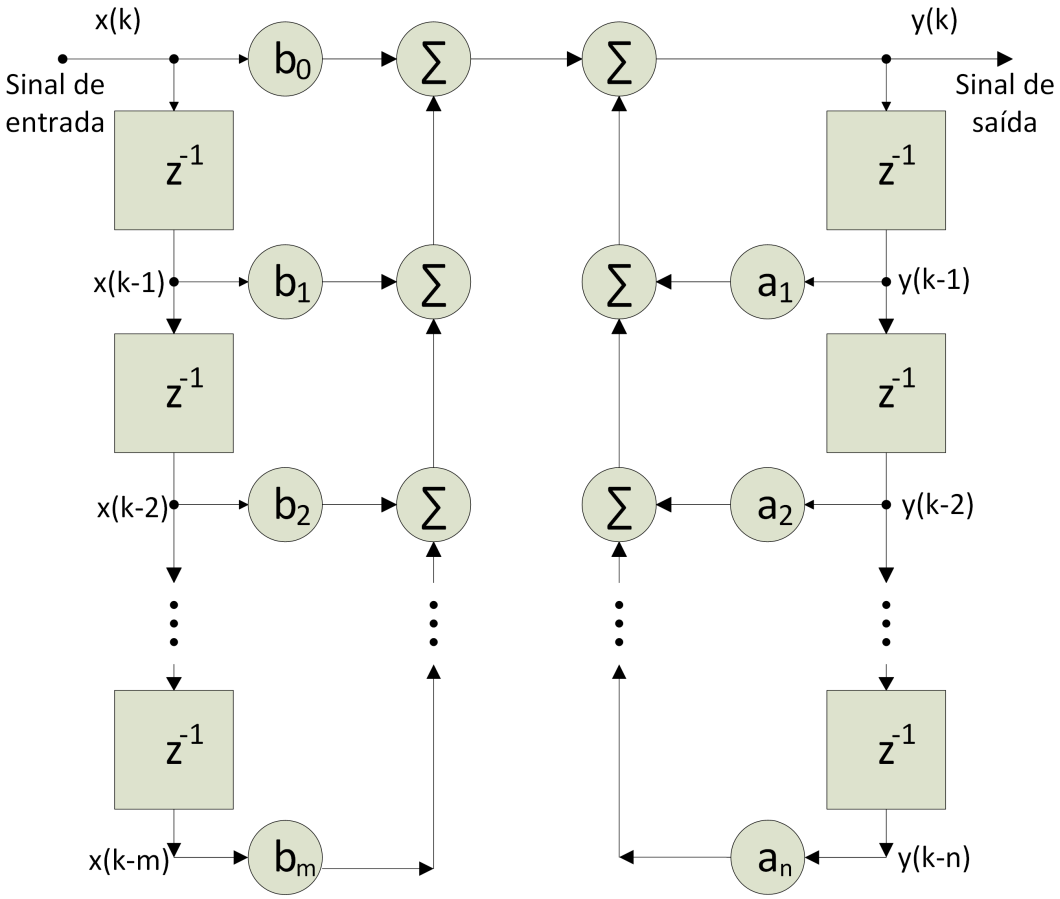

Figura 8 – Estrutura do filtro recursivo.

A ordem do filtro é determinada pelo máximo valor entre o número de coeficientes diretos (m) e realimentados (n). A relação 2.9 demonstra isso.

$$ordem = max[m, n] \tag{2.9}$$

2.4 FPGA

Para construir a ferramenta que torna possível a identificação de sistemas e a pré-distorção, fez-se necessário o uso de uma plataforma que consiga realizar as operações convoluções discretas, e as operações de conversão analógico-digital e digital-analógico. Além disso, a ferramenta selecionada deve possuir uma alta velocidade para processamento de dados, pois necessita-se de um sinal de banda larga.

Uma descrição específica do hardware pode ser realizada através de um *Aplication Specific Integrated Circuit* (ASIC). Porém, esse tipo de plataforma de alto custo só permite a definição dos componentes e conexões antes da confecção do circuito integrado, aumentando o tempo do projeto e reduzindo a sua flexibilidade.

Uma segunda solução seria a utilização de microcontroladores, programados via linguagem C, como o DSP TMS320C6713 e o Tiva TM4C123GH6PM. Essas plataformas conseguem processar sinais digitais com uma frequência de amostragem de 600 kHz e 1 MHz, respectivamente [33] [34], porém esta velocidade ainda seria insuficiente para os requisitos do projeto.

FPGAs podem ser programados a qualquer momento, desde que se utilize uma linguagem apropriada [35]. O kit DE2-115 utilizado pode processar taxas de amostragem de até 100 MHz [7]. O seu custo é moderado, porém ainda acessível. Devido a facilidade da programação, a elevada taxa de amostragem e ao seu custo baixo, o FPGA se mostrou uma ferramenta válida para este trabalho. O FPGA utiliza a descrição de hardware como forma de programação, ou seja, o programador não se utiliza de linguagens de alto nível para a confecção do algoritmo, mas sim de uma descrição do hardware a ser implementado [35]. Essa plataforma é uma solução intermediária entre ASICs e microcontroladores.

A opção por essa plataforma se deu pelo seu baixo custo e possibilidade de operação com um DAC/ADC proprietário e de fácil conexão [7]. Esse kit possui a opção de interface com o usuário através de chaves seletoras, LEDs, Displays, etc. Essas possibilidades facilitam a iteratividade da plataforma com o usuário após a programação [1]. É possível ver estes componentes na Figura 9.

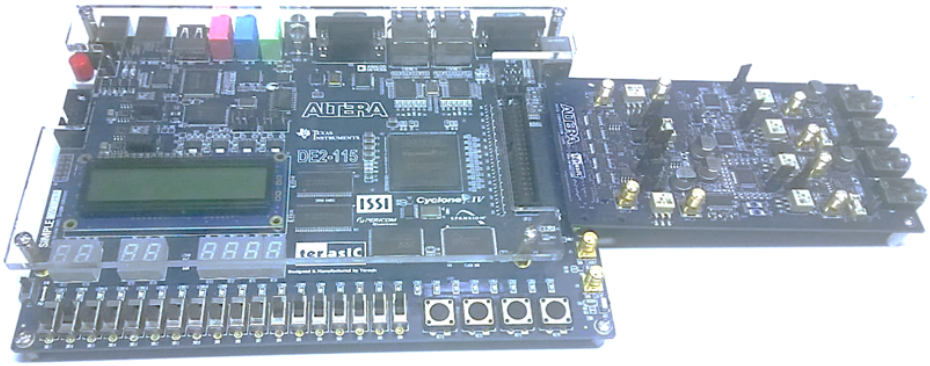

Figura 9 – Altera DE2-115 FPGA kit conectado a uma placa DAC/ADC.

Neste trabalho, são utilizadas as entradas e saídas do GPIO (X36) para a comunicação do Cyclone IV com o ADC/DAC. As chaves seletoras (X18) são utilizadas para a seleção de alguns parâmetros do algorítimo. É possível ver o diagrama de blocos do FPGA na Figura 10.

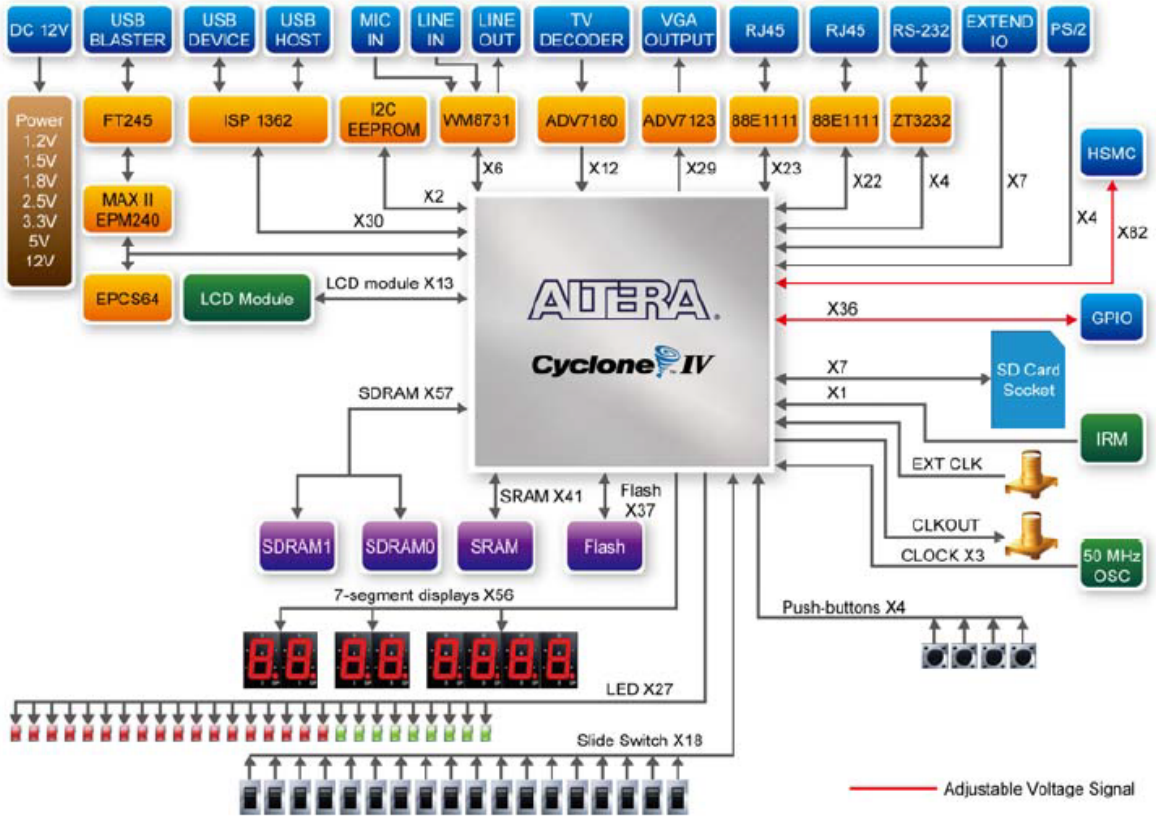

Figura 10 – Diagrama de blocos do kit DE2-115 [1].

Dentro do processador Cyclone IV, o FPGA contém vários blocos de lógica programável. Esses blocos são conectados através das interconexões programáveis para se desenhar o circuito desejado. A Figura 11 demonstra a estrutura básica do FPGA.

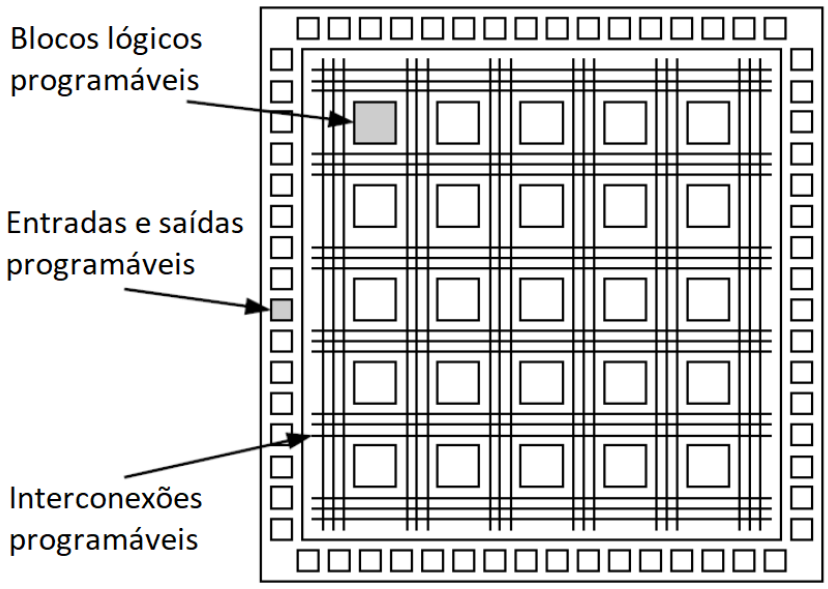

Figura 11 – Estrutura básica do FPGA.

Como forma de programação, optou-se pelo Verilog, que é uma das linguagens de descrição de hardware disponíveis para FPGA [36]. O software utilizado para se programar o FPGA foi o Quartus II, o qual possui uma versão gratuita disponível para download no site da Altera [37].

2.4.1 OPERAÇÃO COM NÚMEROS BINÁRIOS

O Verilog utilizado na programação do FPGA realiza operações com números binários. Esse tipo de implementação em baixo nível torna a construção do código mais trabalhosa do que algoritmos implementados em C, mas possibilita a realização de operações com uma maior velocidade.

Um problema da representação com números binários fracionados é o uso de ponto flutuante [35]. no trabalho são utilizadas palavras de 18 bits para se representar os coeficientes do filtro e os sinais. A multiplicação desses dois gera palavras de 36 bits [38]. São descartados 18 bits após essa operação para que todas as multiplicações realizadas sejam entre números de 18 bits. Esse corte deve obedecer para onde o ponto flutuou, para se evitar operações erradas.

18 bits são suficientes para representar 262144 possibilidades, o que evita problemas de subamostragem nas operações. A saída para o DAC e a entrada ADC possuem 14 bits, portanto, antes da transmissão, 4 bits menos significativos são removidos do sinal.

Todas as variáveis nessa linguagem de descrição de Hardware são declaradas apenas de 2 maneiras:

(i) O valor da operação será um resultado permanente das entradas. As declarações das variáveis são feitas como fios (*wire*).

(ii) O valor da operação será controlada por uma passagem de clock positiva ou negativa. As declarações das variáveis são feitas como registradores (*reg*).

Os registradores só podem realizar operações controlado por um ciclo de clock. Esse tipo de implementação possibilita o uso de paralelismo, pois durante um ciclo de clock pode-se realizar mais de uma operação.

2.4.2 OVERFLOW

O *Overflow* é um efeito indesejado que pode ocorrer durante as operações dentro do FPGA.

Uma palavra de bits tem uma limitação de um valor máximo, por exemplo, uma palavra de 4 bits só pode representar 16 valores [38]. Um problema típico que ocorre em um modulador é quando o valor de amplitude demandado não pode ser representado pelo

número de bits reservado naquele ponto do algoritmo. Esse efeito pode ser chamado de *Overflow*, apresentando uma resposta errada. Essas formas distorcidas na onda geram espalhamento espectral, como pode ser visto na Figura 12.

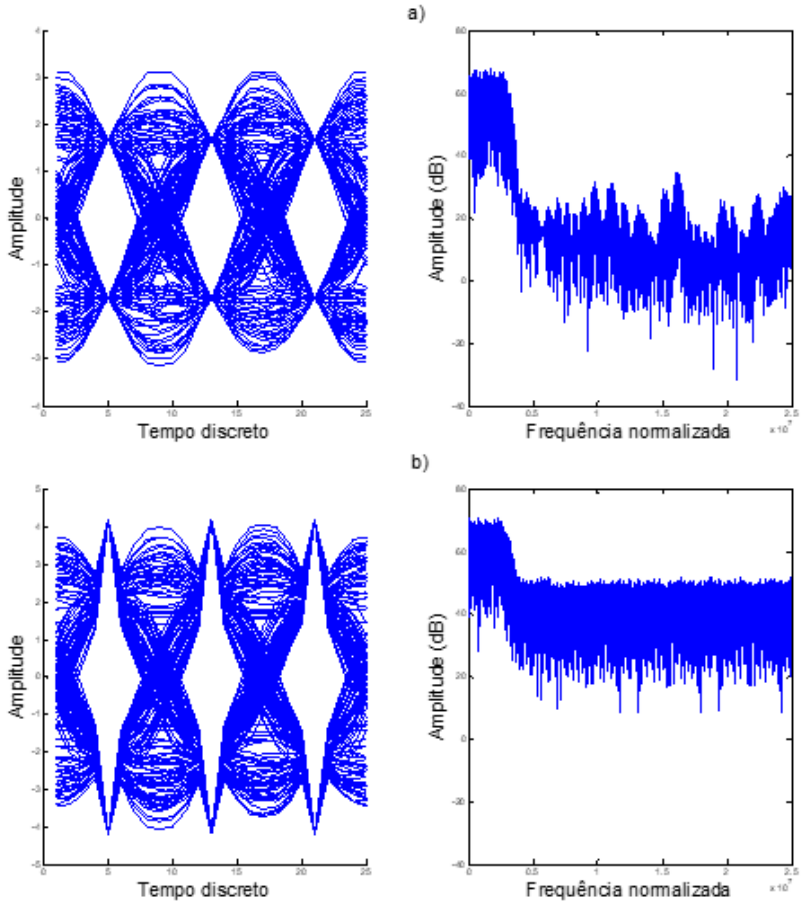

Figura 12 – a) Sinal de banda limitada e b) Sinal de banda limitada com overflow.

Como em toda operação no FPGA é necessário se descrever o tamanho da palavra de bit, é preciso tomar muito cuidado com esse efeito durante a implementação.

2.4.3 CONVERSÃO DIGITAL-ANALÓGICA e ANALÓGICA-DIGITAL

Para conversão Digital-Analógica e Analógica-Digital é utilizado um periférico disponível para placa DE2-115 da FPGA [7]. Essa placa consegue alcançar frequências de amostragem de até 100 MHz e possui capacidade de se comunicar com o FPGA pelas GPIO disponíveis. É possível receber e enviar dados com resolução de 14 bits, ou seja, é possível representar 16384 intervalos de amplitude analógica utilizando essa placa.

Como o periférico foi desenvolvido pela Altera para o kit contendo o FPGA, a comunicação ocorre sem muitos problemas, e a frequência de amostragem pode ser alterada dentro do código em Verilog.

2.5 FIGURAS DE MÉRITO

Figuras de mérito são quantitativos utilizados para se verificar qualidade de um determinado modelo ou sinal. Nesse trabalho, são utilizadas quatro: Erro Quadrático Médio Normalizado (NMSE), Erro Médio Absoluto (MAE), Erro de Magnitude Vetorial (EVM) e a taxa de erro de bit (BER).

2.5.1 NMSE

O NMSE é uma medida bastante utilizada para verificar a similaridade de dois sinais [26] [27]. Nessa medida, a diferença dos dois sinais é calculada ponto a ponto, como demonstrado na equação (2.10).

$$NMSE = 10log\left\{\frac{\sum_{k=1}^{N}|y(k)-\hat{y}(k))|^2}{\sum_{k=1}^{N}|y(k)|^2}\right\} \tag{2.10}$$

O cálculo do NMSE é utilizado como critério de seleção de modelos. É uma ferramenta útil para a avaliação da qualidade do modelo estimado, como também para a avaliação da qualidade da pré distorção. Quanto menor o valor da NMSE maior será a similaridade dos dois sinais, sendo $-\infty$ para dois sinais iguais.

2.5.2 MAE

O MAE, também é uma medida utilizada para se verificar a similaridade de dois sinais [28]. Nesta medida, a diferença dos dois sinais é calculada ponto a ponto de forma absoluta como demonstrado na equação (2.11).

$$MAE = \frac{\sum_{k=1}^{N}|y(k)-\hat{y}(k))|}{N} \tag{2.11}$$

O MAE é utilizado como medida de desempenho nos capítulos 4.2 e 5. Para a comparação de dois sistemas utilizando a MAE é importante que a normalização do sinal seja a mesma, pois essa figura mede o erro absoluto. Quanto menor o valor da MAE maior será a similaridade dos dois sinais.

2.5.3 EVM

O EVM mede a distância, dentro de uma constelação, do símbolo $S_{medido,r}$ para o ponto esperado $S_{ideal,r}$, como descrito na equação (2.12) [39].

$$EVM_{RMS} = \left[\frac{\sum_{r=1}^{N} |S_{ideal,r} - S_{medido,r}|^2}{\frac{1}{N} \sum_{r=1}^{N} |S_{ideal,r}|^2} \right]^{\frac{1}{2}} \tag{2.12}$$

Nesse trabalho, esse valor é medido através do Analisador Vetorial de Sinais Keysight CXA 9000A. Utiliza-se essa medida para se validar o desempenho dos sinais gerados pelo kit FPGA e também para a avaliação da qualidade da pré distorção. Quanto menor o valor percentual da EVM, maior será a similaridade dos dois sinais.

2.5.4 BER

A equação (2.13) a BER . Essa taxa é obtida através da divisão do número total de erros N_E pelo número total de bits N [29].

$$BER = N_E/N \tag{2.13}$$

Quanto menor o número de erros, menor será o valor da taxa na BER.

2.6 RESUMO DO CAPÍTULO

Neste capítulo, foi demonstrado que um sistema discreto é uma representação de um sistema real. Sistemas discretos podem ser separados em FIR e IIR e esses apresentam diferentes características quanto a realimentação. Para a geração dos sinais digitais vetoriais, foi selecionada a plataforma FPGA. Durante o trabalho foi necessária a avaliação da qualidade dos sinais transmitidos e para isso foram utilizadas quatro figuras de mérito: NMSE, MAE, EVM e BER.

3 IMPLEMENTAÇÃO DE UM MODULADOR VETORIAL

O objetivo deste capítulo é a descrição de um gerador de modulações digitais implementado em FPGA. O fluxograma dessa plataforma é mostrado pela Figura 13, em seguida é descrito o funcionamento de cada um dos blocos.

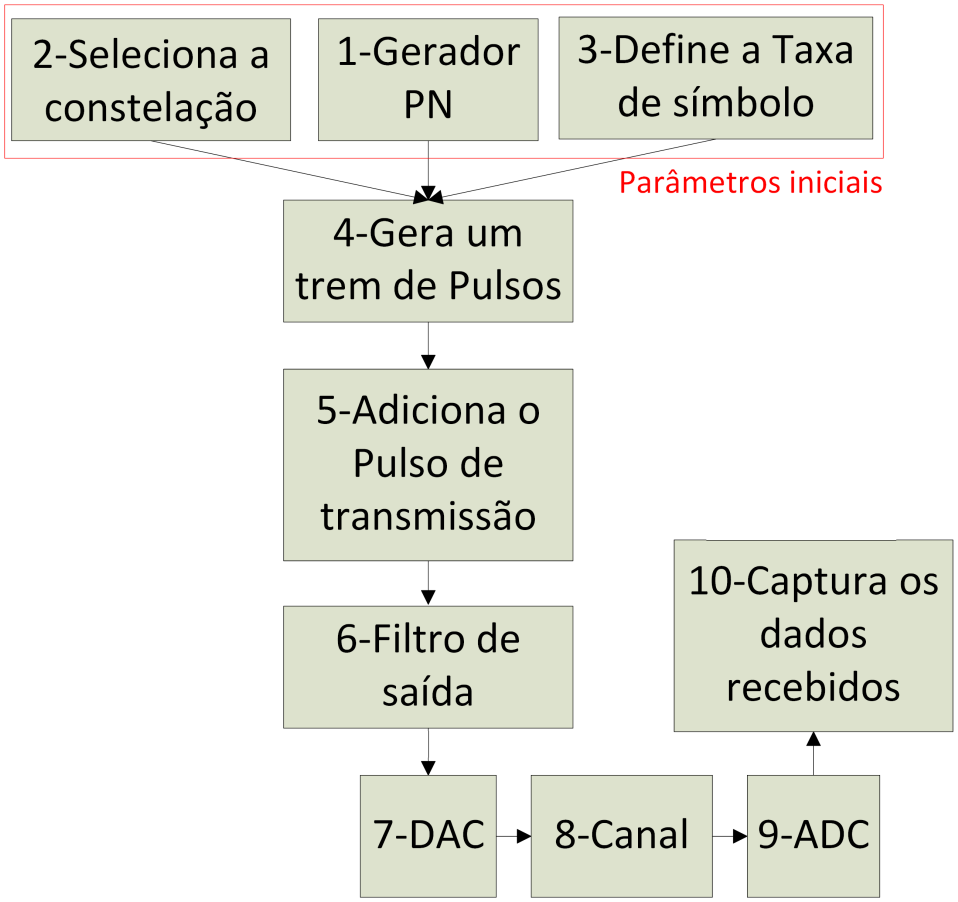

Figura 13 – Fluxograma do algoritmo implementado em Verilog.

1. Gerador PN: utilizado para gerar um conjunto de dados aleatórios que simula uma transmissão real.

2. Seletor de constelações: utilizado para selecionar as constelações BPSK, 4-PSK, 8-PSK, 16-PSK, 4-QAM, 16-QAM, 32-QAM ou 64-QAM.

3. A taxa de símbolo pode ser selecionada de 6,25 MS/s até 12,5 MS/s (Mega símbolos por segundo).

4. A partir dos itens anteriores é gerado um trem de impulsos com uma constelação e uma taxa definidas, transmitindo os dados gerados pelo PN. O processo é descrito na seção 3.2.

5. Adiciona um pulso de transmissão ao trem de impulsos como detalhado na seção 3.3.

6. Pode ser selecionado no FPGA um filtro de saída. Mais detalhes na seção 3.4.
7. O DAC é o responsável pela conversão do sinal digital para o analógico. Esse processo é feito utilizando um periférico disponível para placa DE2-115.
8. O canal utilizado é um cabo coaxial.
9. A conversão Analógico-Digital pode ser feita pelo periférico, mas nessa etapa ela é feita por um equipamento comercial.
10. Os dados capturados são utilizados para medição da (%EVM), avaliação do sinal no tempo e no espectro.

Como parâmetro de comparação, foi utilizado um kit didático comercial da empresa Dreamcatcher, o ME1100, que faz uso de dois Geradores de Funções Keysight 33500B, e também de um software proprietário da empresa, o qual envia os dados através de comandos SCPI para esses geradores, emulando, assim, os sinais em fase I e quadratura Q de um modulador digital padrão.

O osciloscópio Tektronix MSO2012B permite a visualização tanto dos sinais no domínio do tempo como também a constelação do sinal de transmissão em banda básica. Os sinais gerados no FPGA e no kit ME1100 são enviados para um Gerador de RF Keysight N9310A, que translada esse sinal para a frequência de 20 MHz, utilizando seu mixer interno (up-converter), possibilitando a visualização do sinal em banda larga. A largura de banda do sinal transmitido foi verificada pelo Analisador de Espectro CXA N9000A, que permitiu também a análise da porcentagem de erro de magnitude vetorial do sinal (%EVM). O setup completo de testes do gerador de sinais em FPGA pode ser visto na Figura 14.

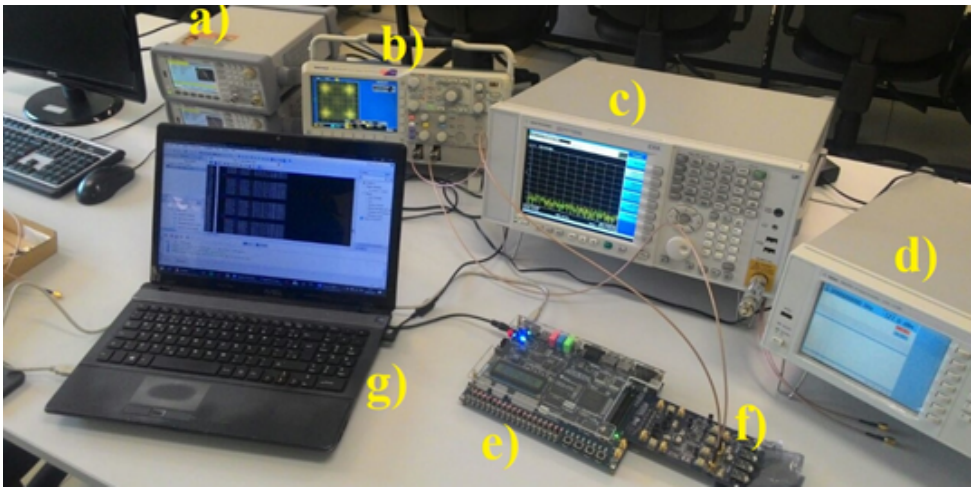

Figura 14 – a) Gerador de Funções Keysight 33500B; b) Tektronix MSO2012B; c) Analisador Vetorial de Sinais Keysight CXA N9000A; d) Gerador de Sinais de RF Keysight N9310A; e) Altera FPGA DE2-115; f) Altera ADC/DAC; g) Notebook.

3.1 GERAÇÃO DE DADOS PARA TRANSMISSÃO

O objetivo do algoritmo implementado no FPGA é a geração de um sinal vetorial capaz de transmitir dados em alta velocidade. Para isso, é necessário que se tenha dados a transmitir. Para simular um sistema de transmissão real, utilizou-se um gerador Pseudo-Aleatório (PN) para se obter uma sequência de dados com uma baixa correlação entre si.

O algoritmo PN utilizado para construir uma sequência PN gera dados muito próximos de sequências puramente aleatórias, e sua implementação em Verilog é relativamente simples. Para o funcionamento, necessita-se de uma sequência inicial de $n + 1$ bits. Esses bits passam por operadores Ou-Exclusivo (XOR), que geram um novo bit b_0. Esse bit é colocado na posição inicial da sequência, enquanto os demais bits são transferidos para próxima posição [40]. A Figura 15 ilustra o funcionamento desse algoritmo.

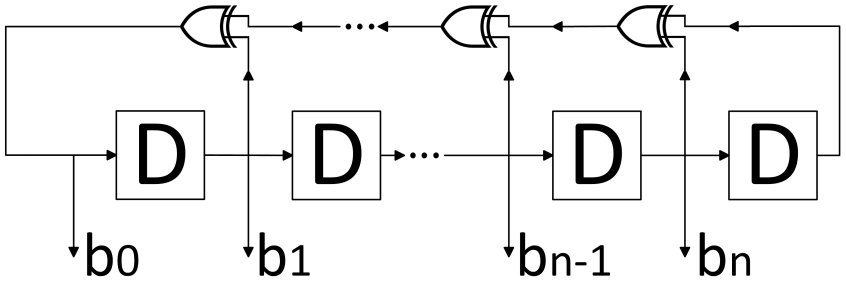

Figura 15 – Fluxograma do Algoritmo Pseudo Aleatório, onde têm-se as portas XOR e os atrasadores D.

A porta XOR é um operador disponível em Verilog através do comando ^, facilitando, assim, a implementação do algoritmo PN no FPGA. Neste trabalho, utilizou-se um PN de 15 bits.

3.2 CONSTELAÇÃO DE BITS

Um sistema digital possui, além de uma representação finita no tempo, uma representação finita na amplitude. A transmissão de dados, geralmente, ocorre no padrão binário (0 ou 1). Esse padrão é simples e possui a vantagem de ter uma grande resistência ao ruído [4].

Uma sinalização comumente utilizada na transmissão de dados é a polar. Nesse tipo de sinalização, o nível lógico 1 é representado como um valor alto (normalmente entre 3 e 5 volts), e o nível lógico 0 é representado por um valor igual e negativo [41]. Essa sinalização possui a vantagem de ter média zero (sem componente DC) [4]. Na Figura 16, é possível ver um sinal uma sinalização polar NRZ.

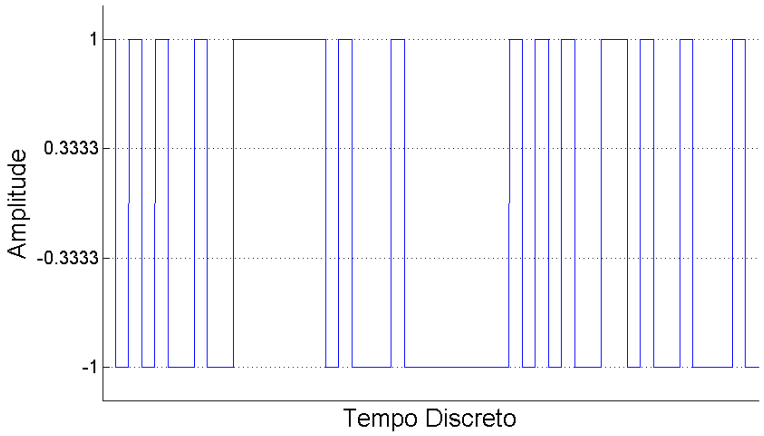

Figura 16 – Sinalização Polar.

Dentro da constelação, o eixo I representa o sinal em fase, enquanto o eixo Q representa o sinal em quadratura [4]. Para o caso anterior, em que a informação está em apenas dois níveis de tensão, somente o eixo I possui valor, como fica ilustrado na Figura 17.

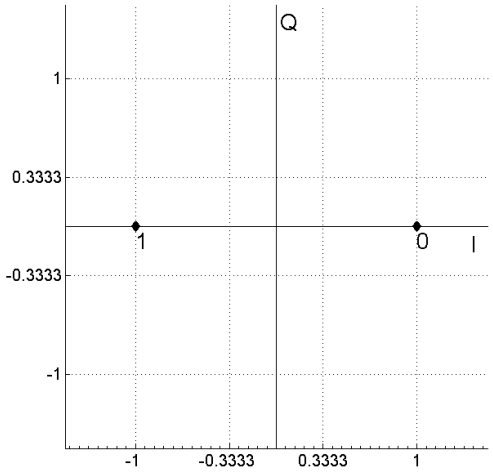

Figura 17 – Constelação para sinalização polar.

Um modo de transmitir os sinais simultâneos é utilizando um sinal em fase e outro em quadratura, ou seja, um sinal estará defasado em noventa graus do outro. Uma maneira bastante simples de se realizar essa operação em um sistema real é a multiplicação dos dois sinais por funções defasadas em noventa graus. Por exemplo, pode se utilizar uma função seno e uma cosseno para se gerar essa defasagem.

Utilizando dois sinais BPSK, pode se gerar uma constelação 4-QAM (ou QPSK). O mapeamento dessa constelação é apresentado na Figura 18, onde é possível ver que

a distribuição dos bits não segue uma codificação tradicional. A codificação utilizada é chamada de codificação Gray, onde cada símbolo varia apenas 1 bit em relação aos símbolos vizinhos. Esse tipo de codificação é utilizada para se reduzir o número de erros de bit (BER) [42], que é uma métrica bastante utilizada em telecomunicações.

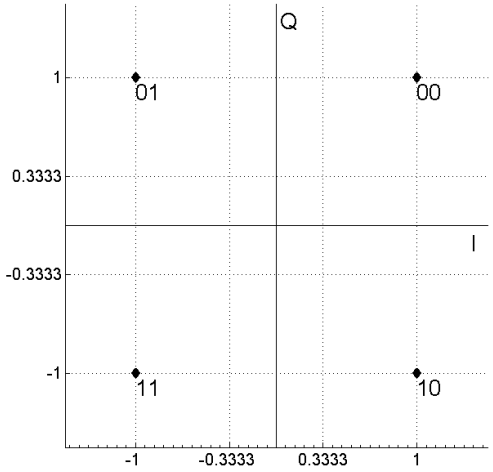

Figura 18 – Constelação QPSK.

Acrescentando um sinal em quadratura é possível transmitir dois bits por símbolo, como visto na Figura 18. Uma metodologia para se aumentar essa quantidade de bits é pela adição de outros níveis de tensão no sinal para a representação de mais possibilidades. Por exemplo, para a codificação 16-QAM, demonstrada na Figura 19, utiliza-se, além dos níveis 1 e -1, os níveis 0.33 e -0.33. Essa adição permite a representação de quatro bits por símbolo.

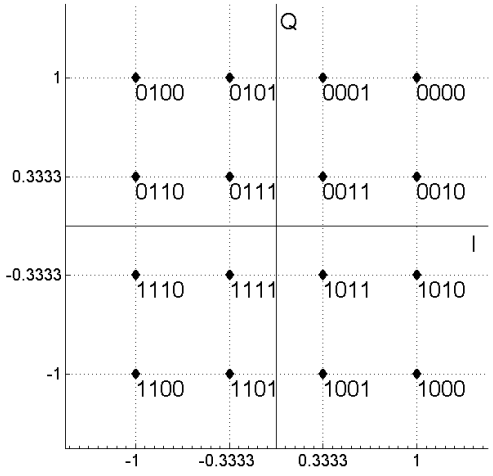

Figura 19 – Constelação 16-QAM.

Os conceitos apresentados anteriormente são aplicados na programação do FPGA. Através da variação das chaves seletoras, o algoritmo implementado atribui diferentes pesos ao sinal I e ao sinal Q, gerando assim 8 constelações: BPSK (1 bit por símbolo), 4-PSK (2 bits por símbolo), 8-PSK (3 bits por símbolo), 16-PSK (4 bits por símbolo), 4-QAM (2 bits por símbolo), 16-QAM (4 bits por símbolo), 32-QAM (5 bits por símbolo) e 64-QAM (6 bits por símbolo).

Como descrito no item 4 da Figura 13, o algoritmo gera primeiramente um trem de impulsos. Esse transmitirá os dados gerados pelo gerador PN, terá as amplitudes dos seus pulsos definidas pela constelação e a distância entre os mesmos estabelecida pela taxa de símbolo. Esse sinal, para o caso em que a informação está em apenas dois níveis de tensão, pode ser visualizado na Figura 20.

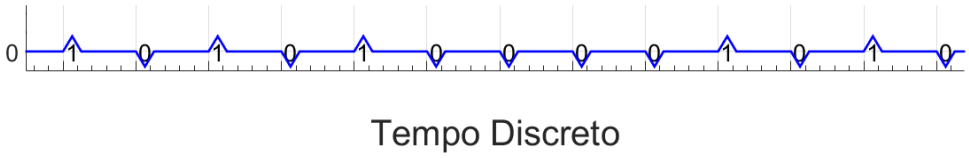

Figura 20 – Sinal sem um pulso de transmissão.

3.3 CODIFICAÇÃO DE PULSO

Um sistema digital possui, além de uma representação finita no tempo, uma representação finita na amplitude. A transmissão de dados, geralmente, ocorre no padrão discreto. Esse possui a vantagem de ter uma grande resistência ao ruído [4].

Como descrito na Figura 13, após a geração do trem de pulsos, é adicionado um pulso de transmissão. A primeira possibilidade é que o algoritmo libere o trem de pulsos como ele é gerado, mas essa abordagem dificulta a visualização do sinal no tempo discreto e a sua constelação no osciloscópio.

O primeiro pulso de transmissão a ser analisado será o pulso quadrado. Esse pulso é constituído de 96 pontos, onde se tem 46 zeros seguidos de 4 uns e 46 zeros novamente. Essa abordagem faz com que durante 4 instantes de tempo discreto o algoritmo permaneça com a informação atual. Como a frequência básica do algoritmo é de 50 MHz, pode se obter a taxa de símbolo pela divisão dessa pelo tempo discreto que o sinal passa com 1 símbolo. Essa divisão de 50 MHz por 4 instantes gera a taxa máxima do algoritmo, que é de 12,5 MS/s. Lembrando que a frequência de operação pode ser reduzida para 25 MHz, gerando uma taxa de 6,25 MS/s. Na Figura 21 é possível se visualizar como a soma funciona para pulsos quadrados. As operações que geram os 4 pontos que transmitem a informação por símbolo são realizadas em paralelo, utilizando, assim, esse recurso disponível no FPGA.

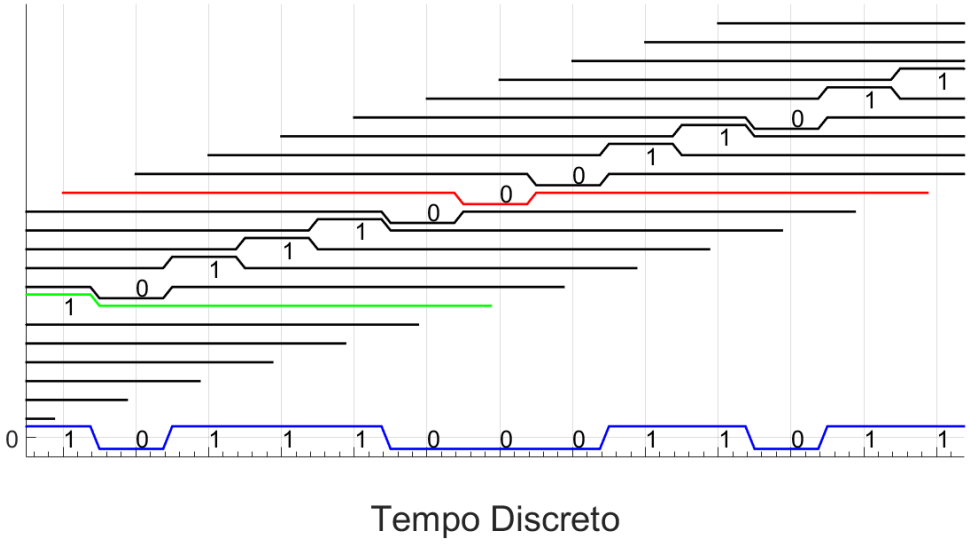

Figura 21 – Sinal com pulso quadrado.

A Figura 22 mostra todas as 8 constelações geradas pelo FPGA com a aplicação de um pulso quadrado observadas no osciloscópio. Essas variações são facilmente selecionadas no FPGA através das chaves seletoras.

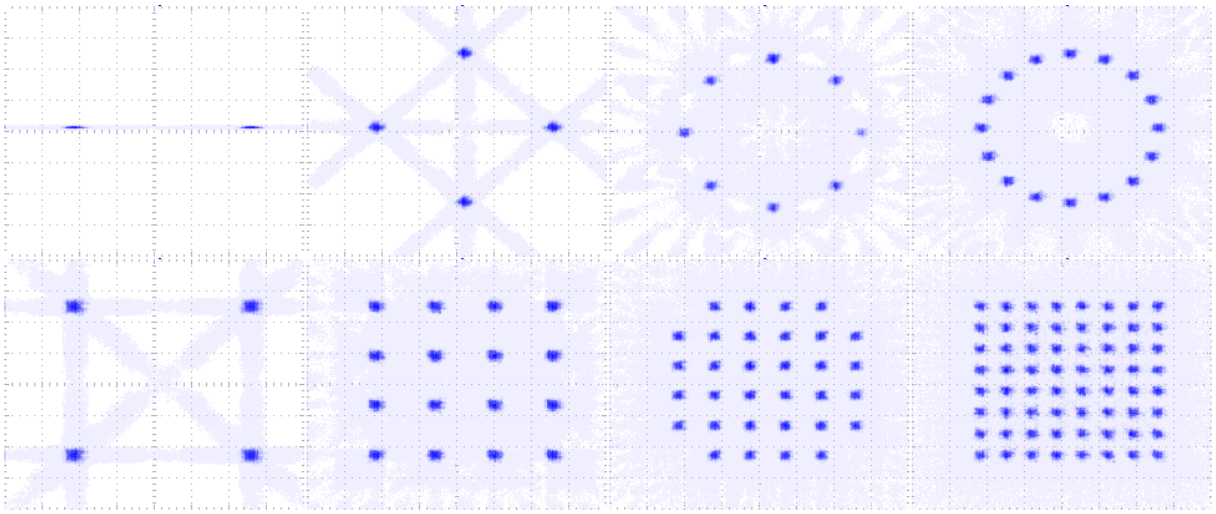

Figura 22 – Constelações BPSK, 4-PSK, 8-PSK, 16-PSK, 4-QAM, 16-QAM, 32-QAM e 64-QAM geradas pelo modulador implementado no FPGA e mostrados no osciloscópio.

A desvantagem do uso de uma sinalização polar pura, ou com retenção de ordem zero, é que o sinal possui um espectro ilimitado no domínio das frequências. Um sinal transmitido deve possuir uma largura de banda limitada, a fim de possibilitar a transmissão de vários sinais separados em canais diferentes. Na literatura, encontra-se que o pulso normalmente utilizado para transmissão de dados com largura de banda limitada é a função cosseno levantado, que é uma variação do pulso sinc. A equação (3.1) define esse

pulso, onde $p(t)$ é o pulso, t é o tempo, T é o período e α é o Roll-Off (RO).

$$p(t) = sinc\left(\frac{t}{T}\right)\left[\frac{cos(\frac{\alpha\pi t}{T})}{1-(\frac{2\alpha t}{T})^2}\right] \tag{3.1}$$

O pulso cosseno levantado varia desde uma sinc pura (RO=0) até um pulso de decaimento rápido (RO=1). Este possui a vantagem de decair mais rápido, facilitando uma melhor representação por uma versão truncada, porém, a largura de banda que ele ocupa é maior. Na Figura 23, é possível visualizar o comportamento destes pulsos através de uma simulação no MATLAB®.

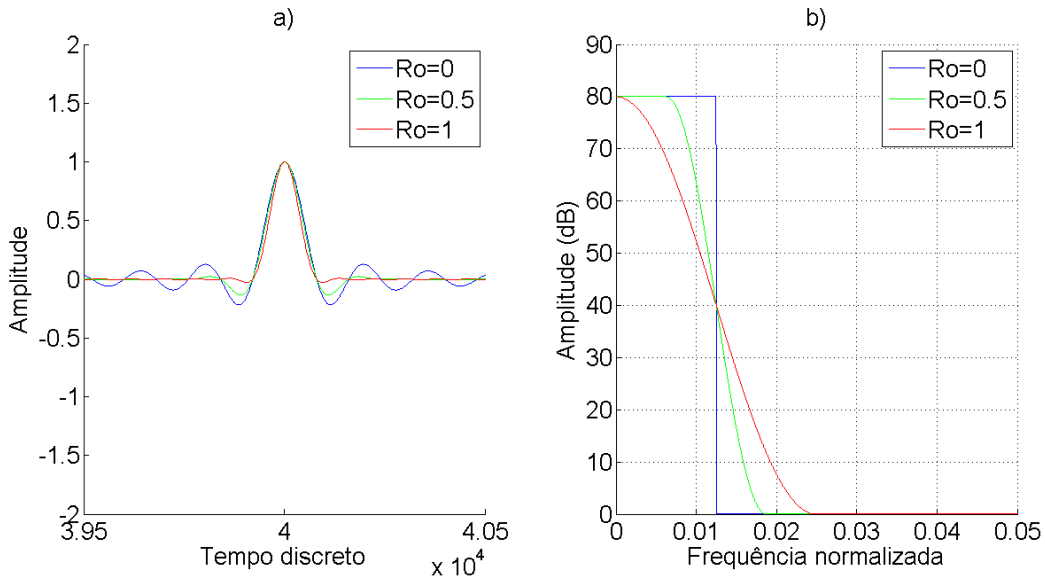

Figura 23 – RO=0 RO=0.5 RO=1.

Como nas Figuras 20 e 21, a Figura 24 define o funcionamento do algoritmo para pulsos cosseno levantado.

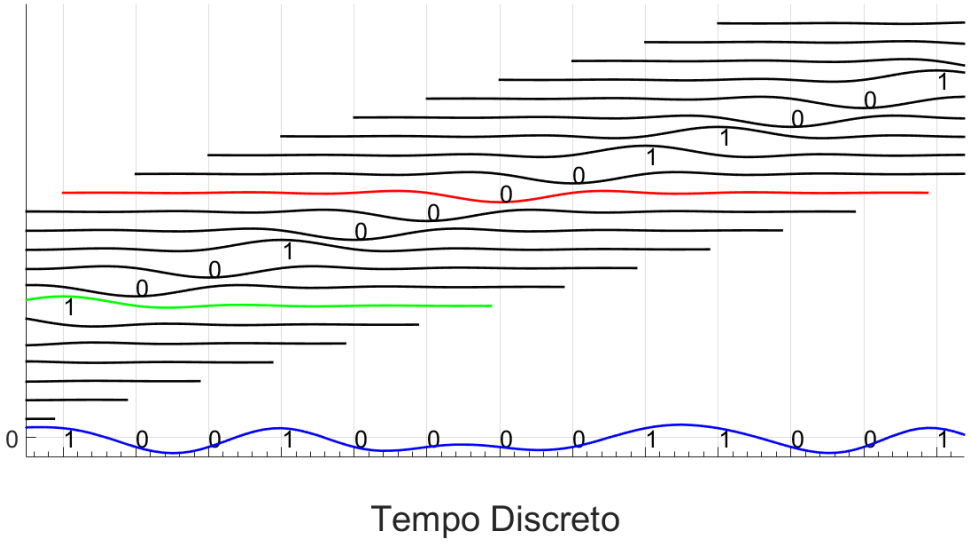

Figura 24 – Sinal com pulso cosseno levantado.

O algoritmo acessa os pulsos de transmissão em uma LUT e os aplica através de uma lógica FIR de ordem 96 no trem de pulsos gerados. Essa ordem é utilizada para que não se perca muita informação em relação ao pulso cosseno-levantado infinito, já que essa versão com 96 pontos é uma versão truncada do pulso original. Esse filtro é aplicado em vários pontos do sinal ao mesmo tempo, valendo-se do paralelismo do FPGA. A Figura 25 mostra a visualização desses sinais no tempo através do Osciloscópio.

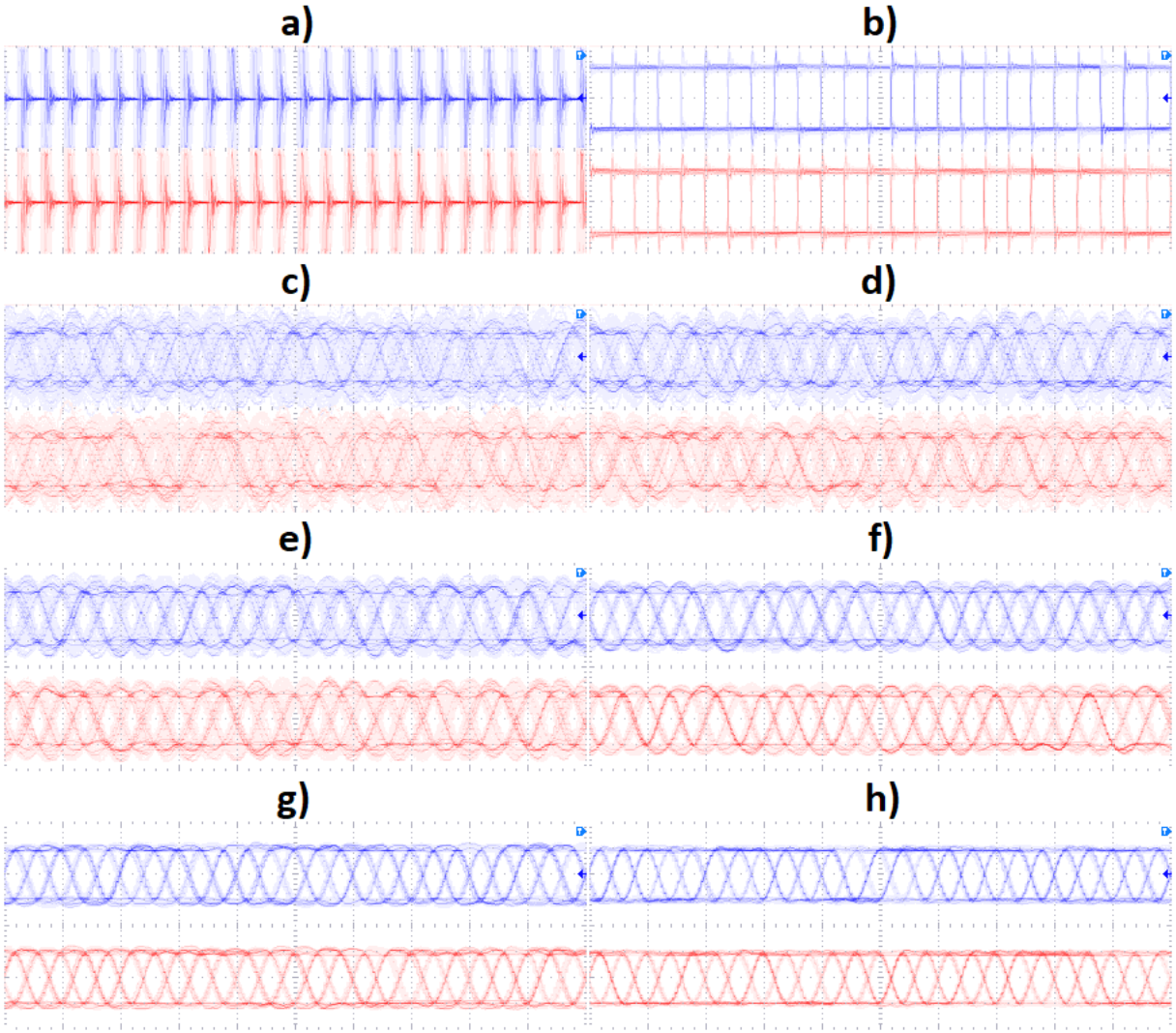

Figura 25 – Todas as possibilidades de pulsos aplicados ao sinal IQ. a) Pulso unitário; b) Pulso quadrado; c) Cosseno levantado com RO=0; d) Cosseno levantado com RO=0,2; e) Cosseno levantado com RO=0,4; f) Cosseno levantado com RO=0,6; f) Cosseno levantado com RO=0,8; g) Cosseno levantado com RO=1.

Da mesma maneira que é possível a visualização do tempo, é possível também ver a constelação no osciloscópio. A Figura 26 mostra a constelação para diferentes pulsos de transmissão.

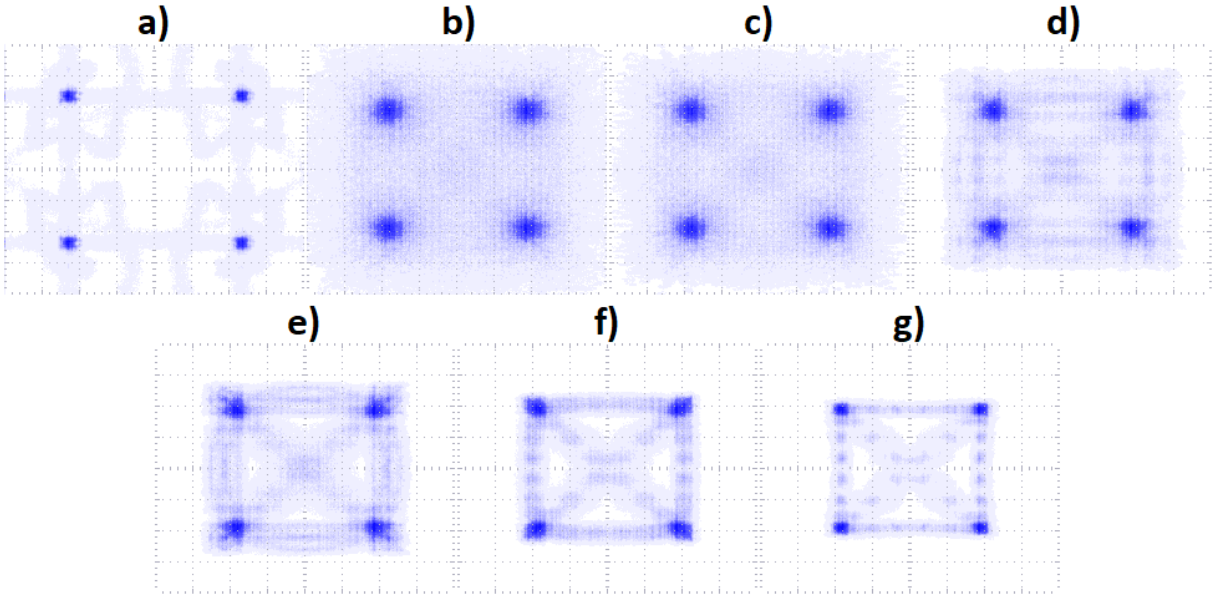

Figura 26 – Constelações para todos os pulsos Cosseno levantado e o pulso quadrado. a) Pulso quadrado; b) RO=0; c) RO=0,2; d) RO=0,4; e) RO=0,6; f)RO=0,8; g) RO=1.

Além do osciloscópio, o setup descrito na Figura 14 mostra um Gerador de RF, um Gerador de Funções em Banda Base e um Analisador Vetorial de Sinais. Para se comprovar o conceito de transmissão com banda limitada, é interessante visualizar o sinal que será transmitido no domínio da frequência. Pode-se então utilizar um Gerador de RF Keysight N9310A. Esse Gerador de RF possui um up-converter interno IQ, o qual translada um sinal direto e um em quadratura para a frequência desejada (neste trabalho 20 MHz). A Figura 27 mostra o resultado da transmissão de um trem de impulsos (sem ser submetido a nenhum filtro de transmissão) e de um pulso que foi submetido a utilização de um filtro de cosseno levantado de RO igual a 0,2 na saída.

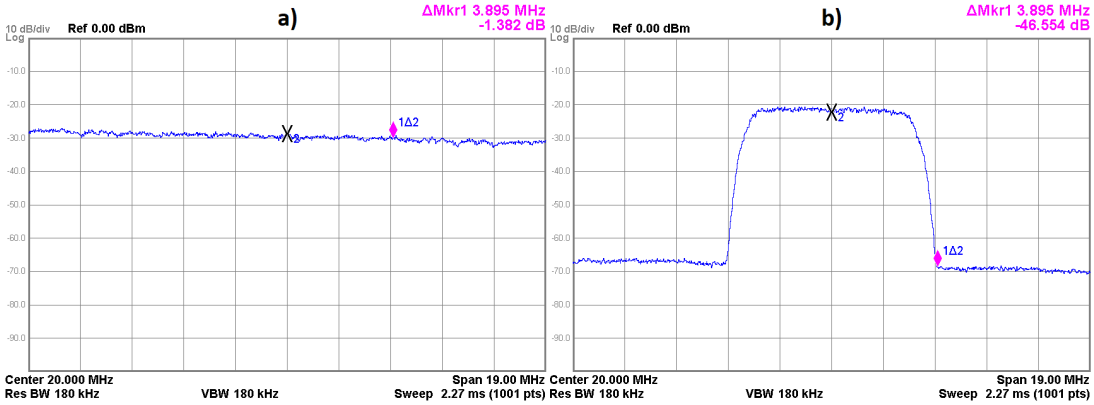

Figura 27 – a) espectro de um trem de impulsos e b) espectro de um trem de impulsos submetido a um filtro cosseno levantado (RO=0,2).

3.4 FILTRO DE SAÍDA

O DAC utilizado nesse trabalho possui uma frequência máxima de amostragem de 100 MHz, enquanto o algoritmo pode trabalhar em uma frequência de 25 ou 50 MHz dependendo da taxa de símbolo desejada. Essa diferença faz com que haja repetição de pontos na saída do DAC, subutilizando a capacidade do equipamento e causando um espalhamento espectral dentro da frequência de 100 MHz.

É possível utilizar uma Mega-função disponível no Quartus Prime 16.0 para se gerar um clock de 100 MHz [43]. Infelizmente, para algumas operações complexas utilizadas nesse trabalho, o uso desse clock não é viável. Porém para a construção de um filtro FIR de ordem 8 na saída, é possível utilizar essa frequência. Assim, o espalhamento espectral é removido pelo filtro, removendo as repetições e linearizando a saída.

3.5 DILATAÇÃO E COMPRESSÃO DE UM SINAL DISCRETO

Uma variação no clock utilizado pode gerar uma variação na frequência do sinal de saída e da sua largura de banda. É importante ressaltar que a dilatação do sinal pela variação da frequência ocasiona um espalhamento espectral. Por isso, o filtro descrito na seção 3.4 é importante. A Figura 28 representa um sinal que consegue transmitir uma taxa de 12,5 MS/s e outro de 6,25 MS/s. Para uma melhor visualização, os sinais são transmitidos utilizando filtros de saída.

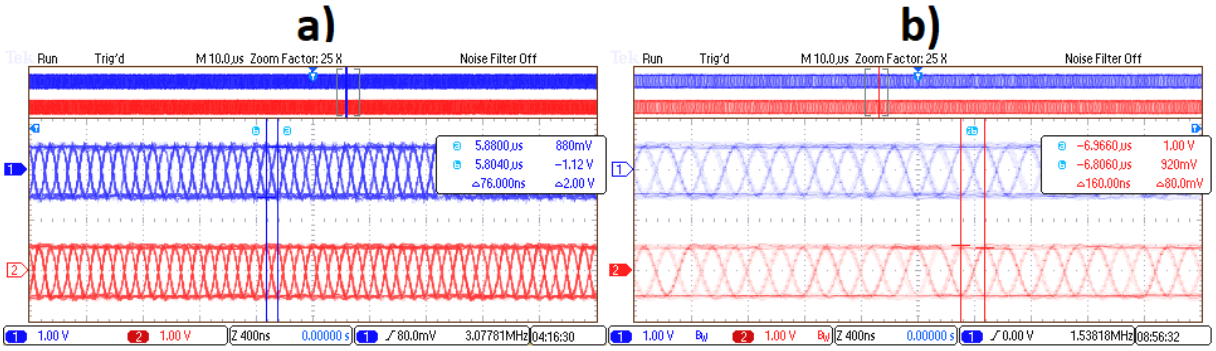

Figura 28 – Diferentes taxa de símbolo observadas no Osciloscópio Tektronix MSO2012B.

Com o objetivo de se visualizar o sinal que será transmitido, pode-se utilizar um Gerador de RF Keysight N9310A. Esse Gerador de RF tem um modulador interno IQ que transmite um sinal direto e um em quadratura, transladando esse sinal para a frequência desejada (nesse caso 20 MHz). Pode se visualizar os efeitos da redução da taxa de símbolo na Figura 29. A relação sinal ruído (dBc) foi de aproximadamente 45 dB para a taxa de 12,5 MS/s, enquanto para taxa de 6,25 MS/s é de 46 dB. Este resultado foi considerado satisfatório para aplicações comerciais [44].

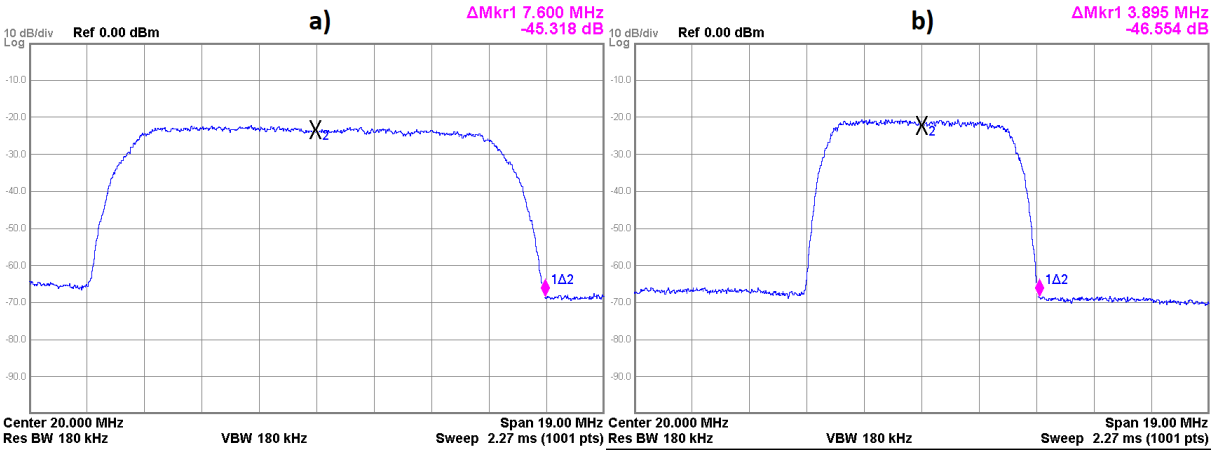

Figura 29 – Diferentes taxa de símbolo observadas no Analisador Vetorial de Sinais Keysight CXA N9000A. a)12,5 MS/s e b)6,25 MS/s.

3.6 COMPARAÇÃO DE DESEMPENHO COM UM KIT COMERCIAL

Como parâmetro de comparação da qualidade do gerador em FPGA, foi utilizado um kit didático comercial da empresa Dreamcatcher. O ME1100, que faz uso de dois Geradores de Funções Keysight 33500B, envia os dados via comandos SCPI para esses geradores, emulando assim os sinais I e Q de um modulador digital padrão, que gera os sinais em fase e quadratura [45].

A base utilizada para comparação é um sistema de transmissão utilizando um filtro de RO igual a 1. A magnitude de erro vetorial (EVM) será utilizada para validar a comparação. A EVM é um índice de desempenho que mede a precisão da constelação transmitida. Quanto maior for a porcentagem da EVM, maior será a distância que o símbolo estará em média da sua posição ideal. Esse valor é obtido através da Equação (2.12) [39].

A Figura 30 ilustra o desempenho do kit ME1100 medido através de um Analisador Vetorial de Sinais Keysight CXA N9000A. Os Geradores de Funções 33500B apresentaram uma EVM de 10% como desempenho. Vale ressaltar que são equipamentos de medidas de alto custo.

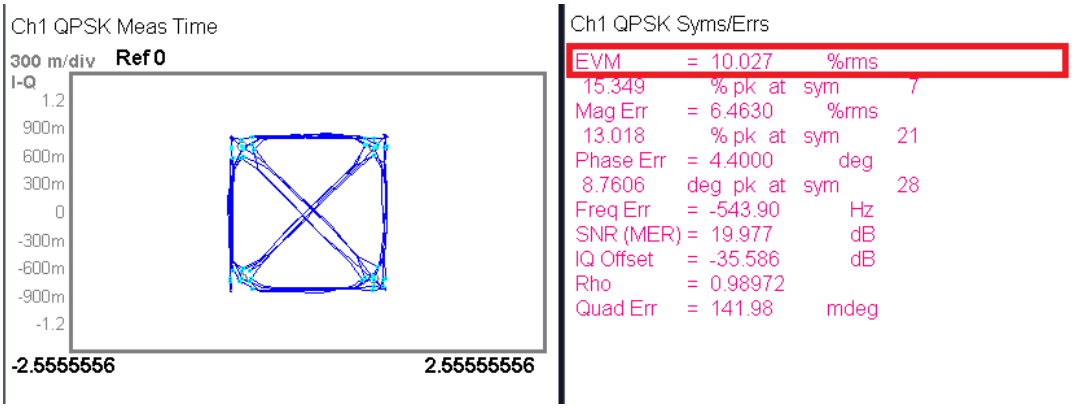

Figura 30 – Desempenho da modulação utilizando dois Geradores de Funções Keysight 33500B.

O Gerador de Funções foi limitado a uma amplitude de 3 Vpp para que a comparação ocorresse no mesmo patamar da FPGA. O resultado apresentado pelo sistema implementado em FPGA pode ser visto na Figura 31.

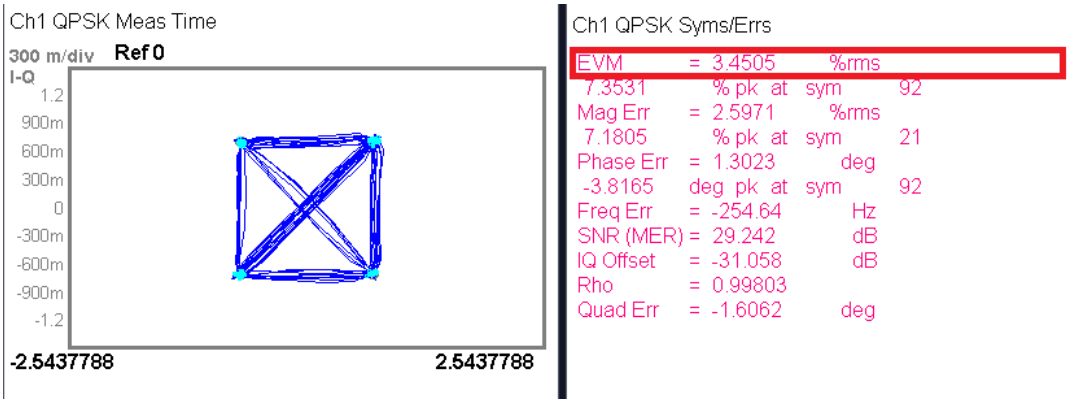

Figura 31 – Desempenho da modulação utilizando o kit contendo FPGA.

O EVM obtido foi de 3,45%, superando o desempenho do kit Dreamcatcher ME1100. Logo, conclue-se que esse sistema implementado em FPGA pode ser utilizado para a geração de modulações digitais utilizadas frequentemente em telecomunicações sem prejuízos na constelação e na EVM. Na Tabela 1, dentro da seção 5.2 pode-se ver a EVM para várias modulações disponíveis no FPGA.

3.7 RESUMO DO CAPÍTULO

Nesse capítulo, foi detalhado o funcionamento de um sistema para a geração de modulações digitais vetoriais. Foram abordados detalhes como gerador PN, constelação de bits, codificação de pulsos, filtro de saída e variação da taxa de símbolo. Ao final, foi apresentada uma comparação com um kit comercial, em que o modulador implementado mostrou um desempenho superior.

4 CONSTRUÇÃO DE UMA PLATAFORMA PARA IDENTIFICAÇÃO DE SISTEMAS

Neste capítulo serão detalhados os passos para a construção de uma plataforma para identificação de dispositivos ou canais. O estimador foi implementado em MATLAB®, e os coeficientes são enviados para o FPGA. Em seguida, foi realizado um teste com um dispositivo real, um filtro Butterworth de quinta ordem, e resultados satisfatórios foram obtidos.

4.1 IMPLEMENTAÇÃO DO ESTIMADOR EM MATLAB

Nesta seção será discutido a implementação de um estimador MQ em MATLAB® para a identificação de um sistema real.

4.1.1 MODELAGEM CAIXA PRETA

Modelagem Caixa Preta ou modelagem empírica é o tipo de modelagem onde o conhecimento das leis físicas que descrevem o processo não é necessário [5]. Ou seja, nesse tipo de modelagem, o modelo é construído, unicamente, através dos dados provenientes do processo a ser descrito [10]. O objetivo do algoritmo implementado em MATLAB® é construir um modelo do tipo Caixa Preta baseado em dados reais provenientes do FPGA.

Os dados devem ser analisados por um estimador, que é o algoritmo que retornará os coeficientes do modelo. Neste trabalho, faz-se o uso do estimador de Mínimos Quadrados (MQ), que é descrito na seção 4.1.2.

4.1.2 ESTIMADOR IDEAL

O método de Mínimos Quadrados (MQ) tem origem nos trabalhos de Gauss sobre astronomia. Nesses estudos, Gauss tinha como objetivo prever as órbitas dos corpos celestes a partir de um conjunto de dados provenientes de observações. Os dados não eram suficientes para descrever por completo as órbitas mas, utilizando o método proposto, Gauss poderia descrever os pontos desconhecidos através de uma aproximação linear [5].

O objetivo dessa seção é descrever o funcionamento do método de Mínimos quadrados.

Para começar, pode-se escrever que y é função de u e θ como na equação (4.1).

$$y = f(u, \theta) \tag{4.1}$$

A equação (4.1) pode ser reescrita como em (4.2).

$$y = u^T y \tag{4.2}$$

As equações apresentadas em (4.1) e (4.2) podem ser reescritas na forma matricial descrita em (4.3) [15]. Utilizando essa metodologia, pode-se ter u_n valores gerando y_n resultados através de estimadores θ_n [5] [46].

$$Y = U\theta \tag{4.3}$$

Pode-se utilizar a entrada e a saída do sistema para se determinar os estimadores θ_n, como descrito na equação (4.4).

$$\theta = U^{-1}Y \tag{4.4}$$

Na equação (4.4), pode ocorrer das matrizes U e Y não obedecerem as regras de multiplicação matricial. Uma solução elegante para esse problema é multiplicar ambos os lados da equação (4.3) por U^H, onde H denota o transposto Hermitiano, como descrito em (4.5) [5] [46] [47].

$$U^H Y = U^H U\theta \tag{4.5}$$

A partir da equação (4.5) é possível obter (4.6) em que a matriz $[U^H U]^{-1}$ é conhecida como pseudo-inversa.

$$\theta = [U^H U]^{-1} U^H Y \tag{4.6}$$

Sabe-se que (4.6) é uma solução do sistema proposto, mas não é possível ter certeza que essa é a melhor solução possível. Para se obter uma resposta satisfatória pode-se escrever a equação (4.7), visando encontrar o erro quadrático do valor estimado $\hat{y}$ para o valor real y [48].

$$J(\theta) = \sum_{n=1}^{N} [y - \hat{y}[n]]^2 \tag{4.7}$$

As equações (4.8), (4.9), (4.10) e (4.11) são desenvolvimentos da equação (4.7).

$$J(\theta) = (Y - \hat{Y})^2 \tag{4.8}$$

$$J(\theta) = (Y - U\theta)^H (Y - U\theta) \tag{4.9}$$

$$J(\theta) = Y^H Y - Y^H U\theta - \theta^H U^H Y + \theta^H U^H U\theta \tag{4.10}$$

$$J(\theta) = Y^H Y - 2Y^H U\theta + \theta^H U^H U\theta \tag{4.11}$$

Para achar os parâmetros θ que minimizam o erro, pode-se derivar a função de erro $J(\theta)$ e igualar o resultado a zero [49] [5]. As equações (4.12) e (4.13) descrevem esse processo.

$$\frac{\partial J(\theta))}{\partial \theta} = -2Y^H U + 2U^H U\theta = 0 \tag{4.12}$$

$$\theta = [U^H U]^{-1} U^H Y \tag{4.13}$$

A equação (4.13) prova que o estimador, descrito em (4.6), é o que minimiza o erro quadrático, pois ambas obtiveram o mesmo valor de θ. Esse modo de se obter θ é conhecido como estimador de Mínimos Quadrados. A implementação desse estimador no MATLAB® é feita pelo comando: $theta = U \backslash Y$. Onde a variável theta é o θ da equação (4.13).

4.1.2.1 MÍNIMOS QUADRADOS SEM REALIMENTAÇÃO

Se um processo puder ser descrito apenas pela combinação do sinal de entrada com as entradas passadas multiplicados por coeficiente b_m, ele pode ser estimado por um MQ FIR. A ordem desse estimador é definida por m e o número de dados por N. A matriz U, nesse caso, é definida por (4.14) [46], em que x são os dados antes da interferência do dispositivo a ser identificado e U a matriz de regressão.

$$U = \begin{bmatrix} x(m+1) & ... & x(1) \\ x(m+2) & ... & x(2) \\ \vdots & \vdots & \vdots \\ x(N) & ... & x(N-m) \end{bmatrix} \tag{4.14}$$

A saída do sistema é definida pela matriz (4.15), em que y são os dados capturados após o dispositivo a ser identificado. O número de dados N da entrada tem que ser igual o número de dados da saída.

$$Y = \begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(N) \end{bmatrix} \tag{4.15}$$

Os estimadores θ são definidos pela matriz (4.16), e são calculados pela equação (4.13), em que b representa os coeficientes do estimador FIR.

$$\theta = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \tag{4.16}$$

O erro entre a saída do sistema e a saída dos dados no estimador está ilustrado na Figura 32. O modelo descrito pelo estimador é igual a um modelo do filtro FIR.

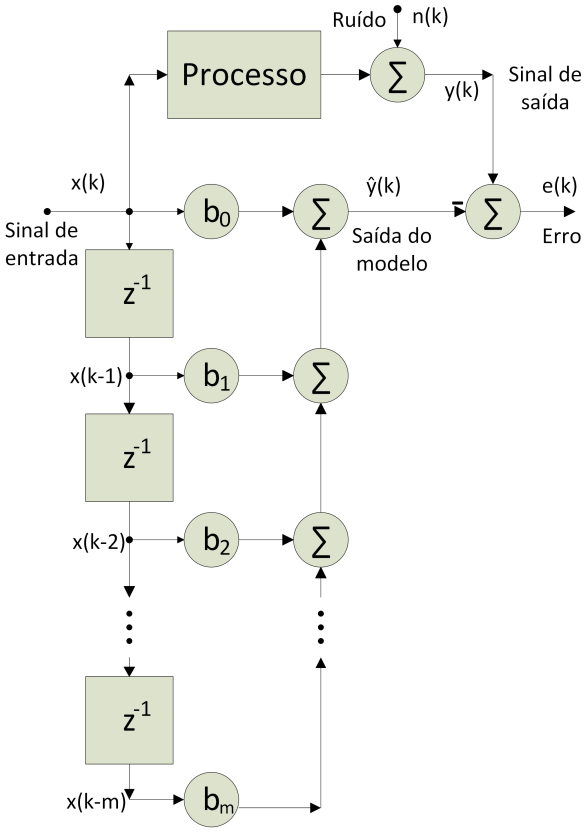

Figura 32 – Erro comparado com um MQ não-recursivo.

4.1.2.2 MÍNIMOS QUADRADOS COM REALIMENTAÇÃO

Alguns sistemas não podem ser definidos por um estimador sem realimentação. Para esses casos mais complexos, a matriz U é definida por (4.17), onde x são os dados antes da interferência do dispositivo a ser identificado, e y são os dados que foram submetidos à interferência.

$$U = \begin{bmatrix} x(m+1) & ... & x(1) & -y(n+1) & ... & -y(2) \\ x(m+2) & ... & x(2) & -y(n+2) & ... & -y(3) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \\ x(N) & ... & x(N-m) & -y(N) & ... & -y(N-n+1) \end{bmatrix} \tag{4.17}$$

As matrizes θ e Y são definidas por (4.16) e (4.15), como no MQ sem realimentação. O erro entre a saída do sistema e a resposta do estimador está ilustrado na Figura 33. O modelo descrito pelo estimador é igual a um modelo do filtro IIR.

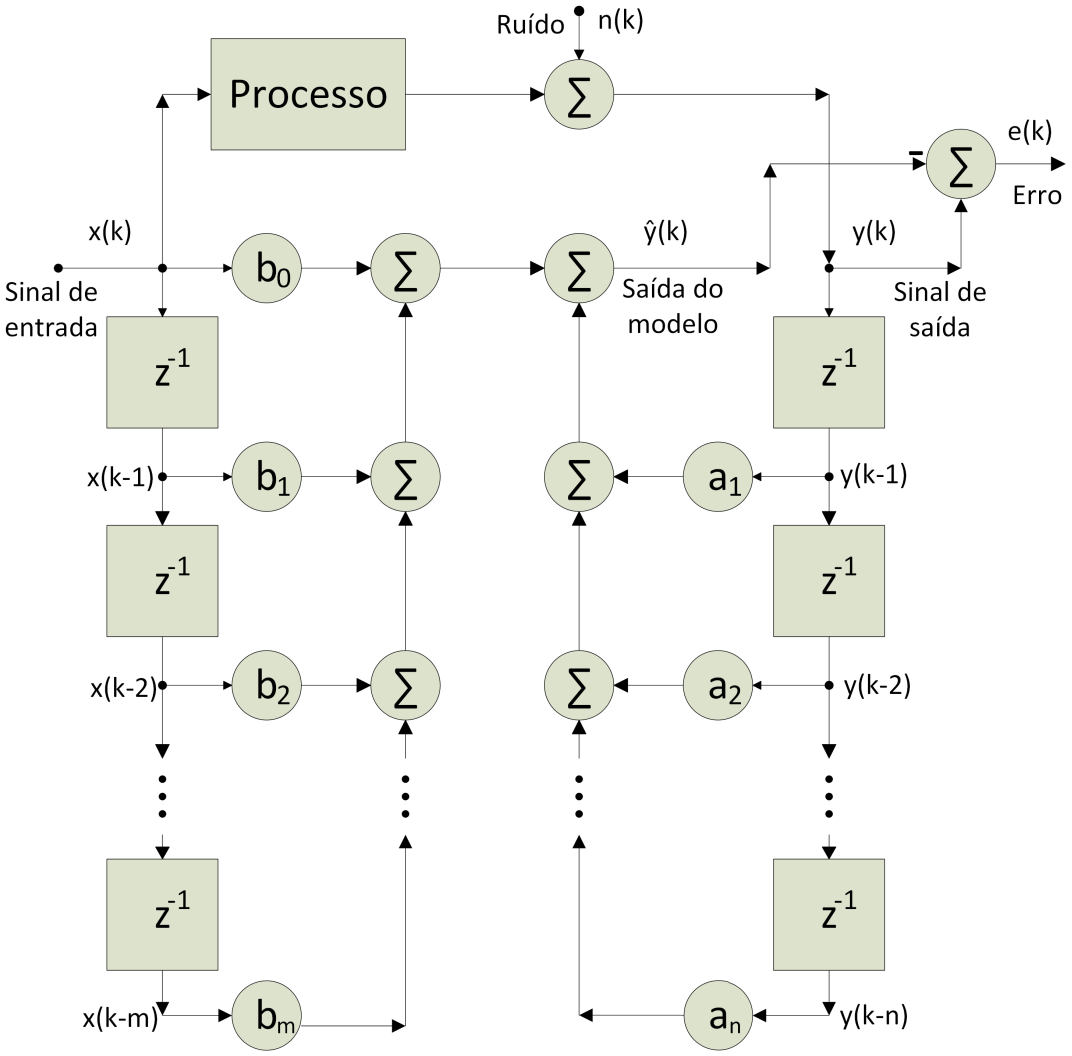

Figura 33 – Erro comparado com um MQ recursivo.

4.1.3 ALGORITMO DE ESTIMAÇÃO EFICIENTE

Com base na seção 4.1.2, pode-se imaginar que o problema de estimação utilizando o método de Mínimos Quadrados se resume a uma simples operação matemática, mas o problema da modelagem Caixa Preta é mais complexo. Nas próximas seções, são discutidos mais alguns detalhes que foram considerados para a construção do estimador.

4.1.3.1 DIVISÃO DOS DADOS DENTRO DO ESTIMADOR

Um modelo construído a partir de um único conjunto de dados pode não ter a generalidade necessária para descrever um sistema, podendo ter uma resposta satisfatória apenas para o conjunto de dados para o qual ele foi treinado. Por isso, dentro de um conjunto de dados utilizado para a estimação, os dados são divididos em: treinamento, validação e teste [46].

Essa divisão serve para que o modelo descreva o sistema sem ser enviesado somente

para um conjunto de dados. A Figura 34 descreve o funcionamento de um processo de identificação.

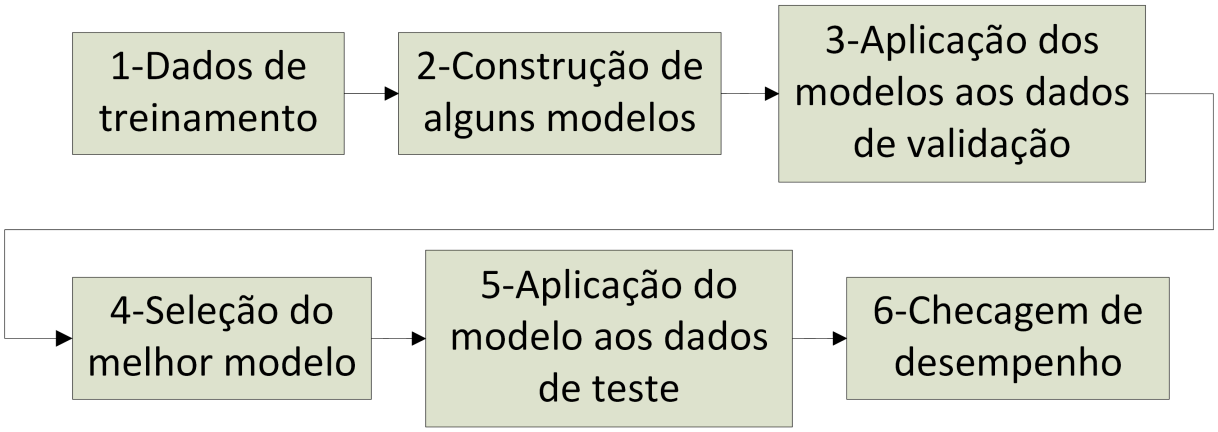

Figura 34 – Fluxograma da criação de um modelo.

1. Dados de treinamento são aqueles dados que serão utilizados para a construção do θ no estimador MQ descrito na seção 4.1.2.
2. São construídos diferentes modelos a partir de alguns conjuntos de treinamento.
3. O modelo criado é aplicado aos dados de validação e é realizado o cálculo da NMSE para garantir que o modelo selecionado não seja viciado nos dados de treinamento.
4. Para a definição do melhor modelo é utilizado o NMSE dos resultados nos dados de validação.
5. O modelo selecionado anteriormente é aplicado aos dados de teste.
6. É realizado o cálculo do NMSE para uma segunda checagem da qualidade do modelo e é esse o valor considerado para a medição da qualidade dos coeficientes estimados.

Cada conjunto de dados dentro do projeto implementado possui 32000 amostras.

4.1.3.2 DEFINIÇÃO DOS MODELOS

Na subseção 4.1.3.1 foi descrito que diferentes modelos são criados a partir dos dados de treinamento. Esses são construídos utilizando diferentes ordens, tipos de realimentação e atrasos. Por exemplo, pode-se supor que um conjunto de dados provenientes de um filtro Butterworth de quinta ordem será bem descrito por um estimador recursivo de ordem cinco. Porém, para uma estimação correta, os dados de entrada e saída precisam estar alinhados como descrito nas equações (4.14), (4.15) e (4.16). Por exemplo, para o sistema $y[k] = 0.25x[k] + 0.5x[k-1] + 0.25x[k-2]$ os dados devem estar alinhados, como na Figura 35.

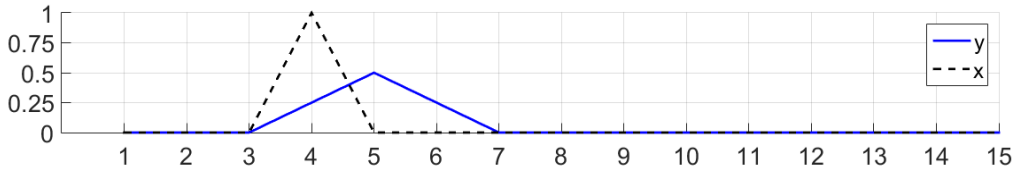

Figura 35 – Sinal com alinhamento correto.

Na captura de um sinal real, o atraso é difícil de se prever, pois dentro de um sistema existe o atraso provocado pela captura dos dados, pela transmissão e pelo sistema [50]. O sistema descrito anteriormente pode ter um atraso na recepção dos dados como descrito na Figura 36.

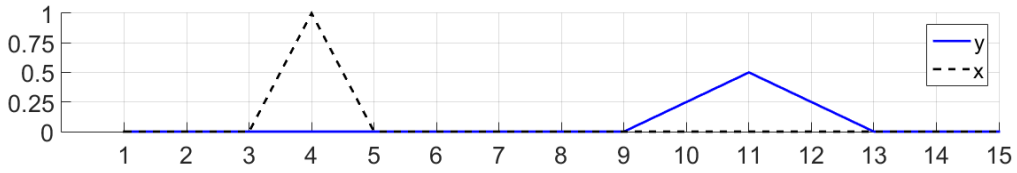

Figura 36 – Sinal y possui uma resposta correta, mas o atraso de 6 amostras impossibilitaria a identificação por um MQ de ordem 3.

Uma maneira de se compensar esse efeito é utilizando a função de correlação cruzada definida por (4.18) [47], onde $E[\cdot]$ representa o operador esperança matemática e $\phi_{xy}(\tau)$ a correlação. Esta é utilizada para se determinar onde os sinais são mais parecidos e pode ser facilmente aplicada em MATLAB® pelo comando xcorr [51].

$$\phi_{xy}(\tau) = E[x(t-\tau)y(t)] \tag{4.18}$$

A Figura 37 ilustra como os sinais podem ser alinhados no ponto de maior correlação.

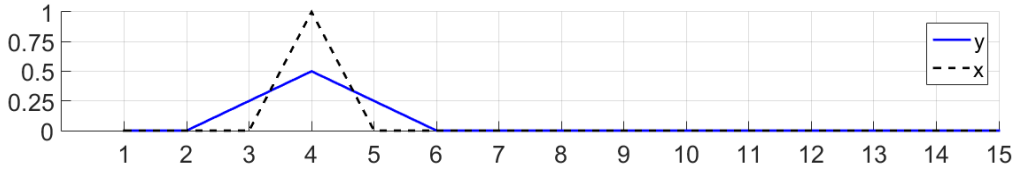

Figura 37 – O ponto de maior correlação só pode ter uma resposta não-causal para esse caso.

Como visto na Figura 37, os sinais ainda não possuem um alinhamento igual à Figura 35. Não necessariamente quando os dados estão alinhados no ponto de melhor correlação, que é o caso mostrado, é quando o modelo será criado da melhor maneira. A solução encontrada no algoritmo desenvolvido é que, a partir do ponto de maior correlação,

são criados $n+1$ modelos com diferentes atrasos, onde n representa a ordem máxima a ser analisada e os atrasos são inseridos um a um como mostrado na Figura 38

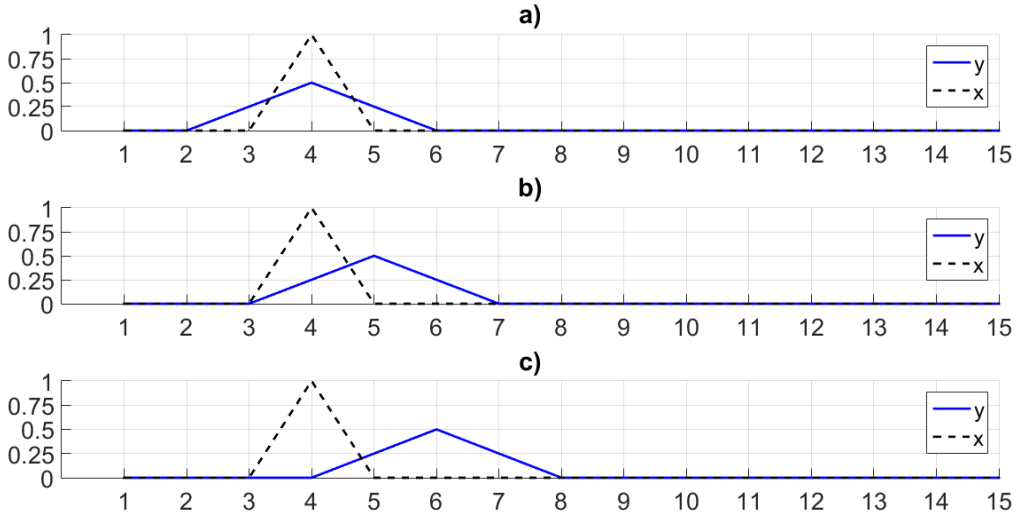

Figura 38 – a) O ponto de maior correlação só pode ter uma resposta não-causal para esse caso; b) Com 1 delay a resposta é correta e c) Com 2 delays a estimação ficará errada.

Portanto, o algoritmo desenvolvido recebe o conjunto de dados de treinamento e constrói vários modelos com diferentes ordens, atrasos e recursividades. Os dados de validação são aplicados a estes modelos. O modelo que apresentar o menor NMSE é selecionado. Após isso, o modelo é aplicado aos dados de teste, e o seu NMSE é calculado.

4.2 IDENTIFICAÇÃO DE UM DISPOSITIVO REAL

Utilizando o algoritmo de MQ, descrito na seção 4.1 como estimador, é possível a identificação de um sistema real. Para aproximar o processo de identificação de aplicações práticas, optou-se por se considerar dois sinais (em fase e quadratura). Ou seja, toda a parte de estimação do modelo é feita utilizando dados complexos. Um número complexo possui uma parte real e outra imaginária. Em MATLAB®, a declaração de um número complexo pode ser feita pela soma do sinal em fase com a multiplicação do sinal em quadratura por $1i$. Essa operação é descrita pela equação (4.19).

$$Sinal_{complexo} = Sinal_{emfase} + 1i * Sinal_{quadratura} \tag{4.19}$$

Com o algoritmo em FPGA e o estimador em MATLAB® construídos, é possível aplicar estas ferramentas para a identificação de um sistema real. Neste trabalho foi utilizado como sistema real um filtro Butterworth de quinta ordem, construído de acordo com o Anexo A. A Figura 39 mostra o filtro a ser identificado prototipado em uma placa com conectores SMA.

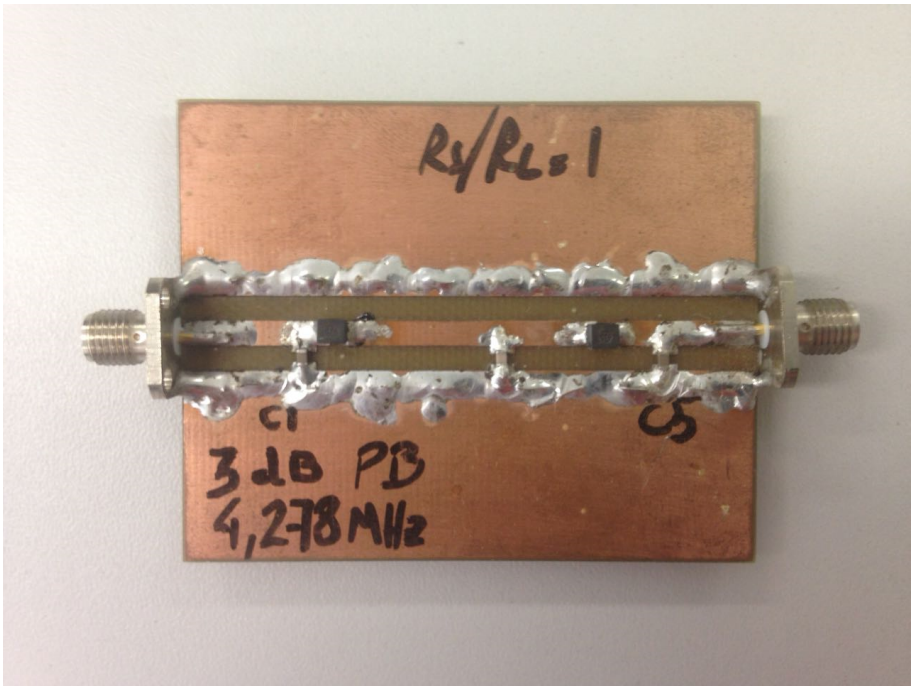

Figura 39 – Filtro de quinta ordem construído para identificação.

Na dissertação, foram utilizados dois filtros idênticos para a transmissão do sinal, um para o sinal em fase (I) e outro para o sinal em quadratura (Q). Devido à presença desses dois sinais, o estimador considera um sinal complexo, onde a parte real representa o sinal em fase e a imaginária o sinal em quadratura.

O periférico responsável pela conversão Digital-Analógica possui uma saída de sinal com uma impedância de 12 Ω, enquanto a sua entrada do conversor Analógico-Digital possui uma impedância de 50 Ω. É possível visualizar essas duas partes na Figura 40.

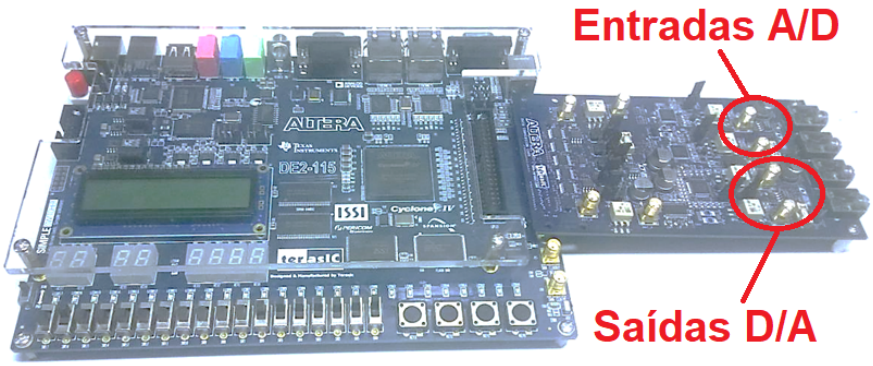

Figura 40 – FPGA com sua placa de conversão de dados.

Para um setup em que a saída é simplesmente uma linha, como o utilizado para se visualizar as constelações na Figura 41, não há problema, mas quando entre a fonte e a carga existem mais componentes com ligação para terra, esse descasamento pode ocasionar mau funcionamento [19].

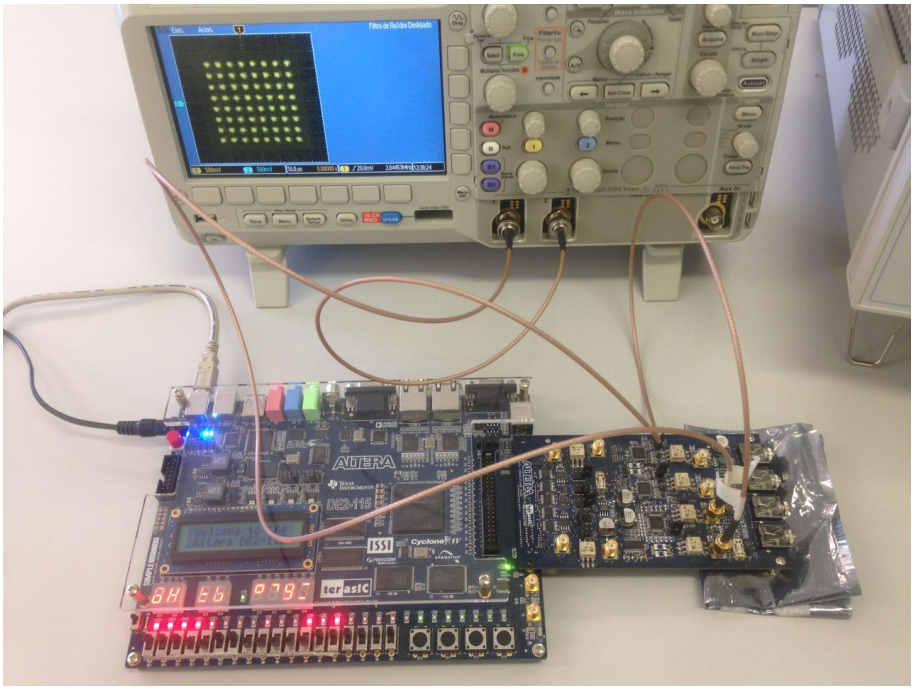

Figura 41 – Constelação 64-QAM sendo observada na saída do FPGA.

Para se contornar o problema de casamento de impedâncias, faz-se uso de um transformador 1:4 para se casar a impedância da fonte com a carga, pois o ADC, o CXA e o mixer utilizados possuem entradas de 50 Ω. Esse transformador é um Circuito Integrado (CI) wb4-6 da coilcraft prototipado em uma placa com conectores SMA. Essa placa é mostrada na Figura 42.

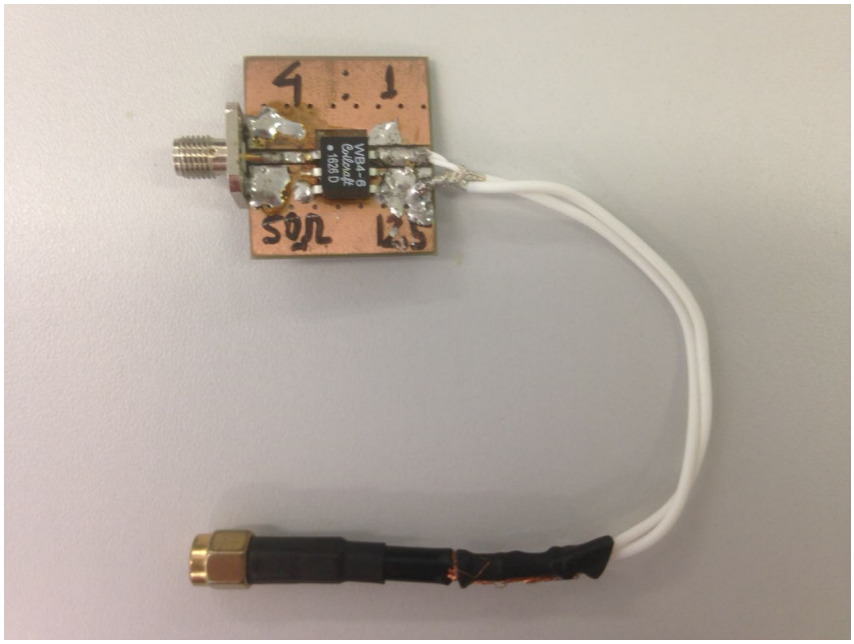

Figura 42 – Transformador 1:4 construído.

Com o transformador é possível montar o setup em que o sinal se origina no DAC,

é enviado ao transformador, ao filtro e é recebido no ADC como mostrado na Figura 43. Essa implementação foi necessária para realizar a identificação do dispositivo, pois esse foi construído para cargas desbalanceadas.

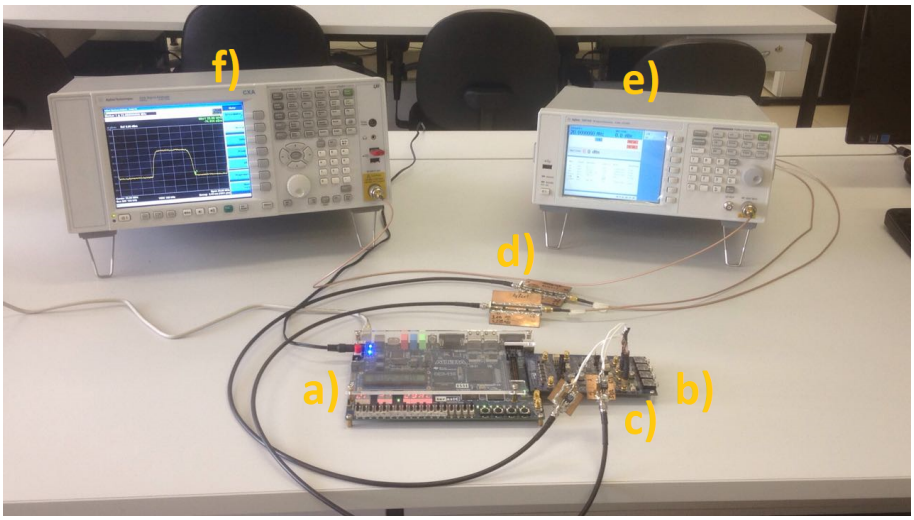

Figura 43 – Sistema para avaliação do canal. a) Altera FPGA DE2-115; b) Altera ADC/DAC; c) Transformadores d) Filtros e) Gerador de sinais de RF Keysight N9310A; f) Analisador Vetorial de Sinais Keysight CXA N9000A.

Após a identificação, espera-se que a resposta do modelo em frequência seja parecida com a resposta do Filtro, medida previamente utilizando um Analisador de Redes Keysight E5061B, conforme a Figura 44.

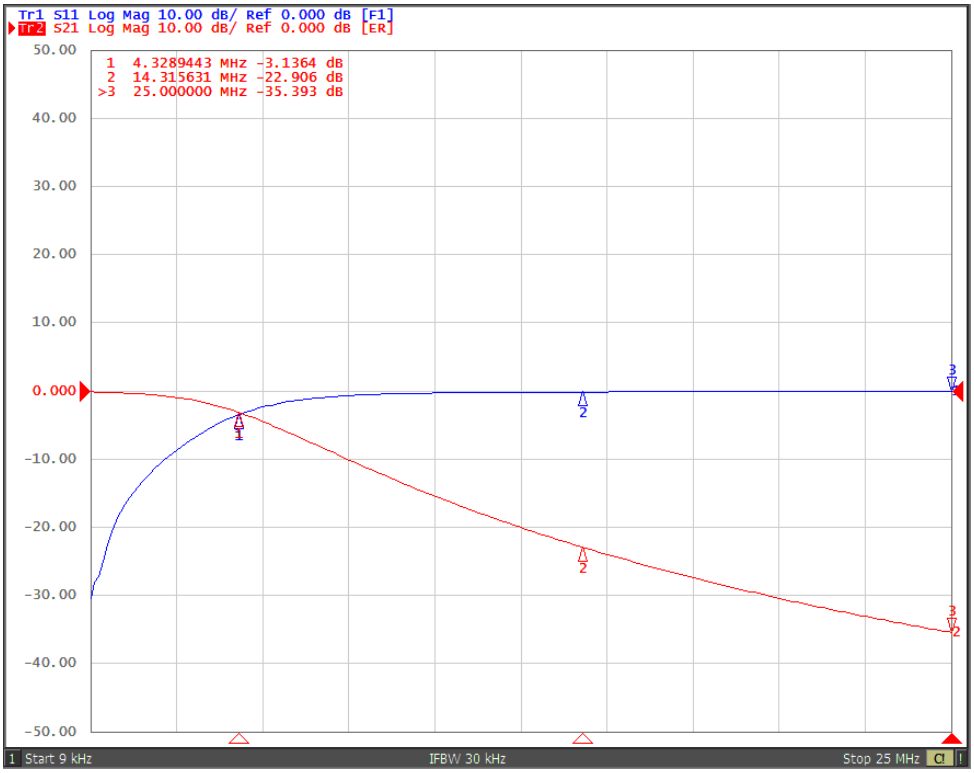

Figura 44 – Curva do filtro analisada pelo Analisador de Redes Keysight E5061B.

4.2.1 CAPTURA DOS DADOS NO FPGA

Como visto na seção 4.1, os dados provenientes do FPGA são processados no MATLAB® para que o estimador determine os coeficientes. Portanto, para a criação de um modelo, é necessário uma plataforma que faça a comunicação do FPGA com o MATLAB®.

Uma ferramenta disponível no Quartus é o SignalTap [11]. Essa plataforma possibilita a captura do sinal enviado e sua conversão para um arquivo de texto. A Figura 45 mostra a tela de captura do SignalTap.

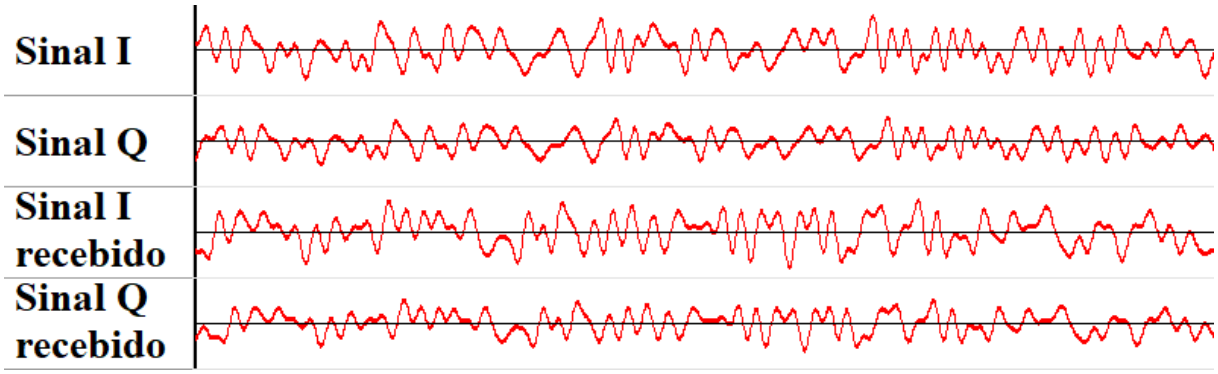

Figura 45 – SignalTap disponível no Quartus II.

A partir desta plataforma é possível capturar o sinal no domínio do tempo, como visualizado na Figura 46. No caso apresentado, o SignalTap limita a 32.000 pontos a

captura dos 61 bits (14 de cada um dos 4 sinais e 5 da variável de acompanhamento). A Figura 46 mostra o mesmo sinal capturado no MATLAB®.

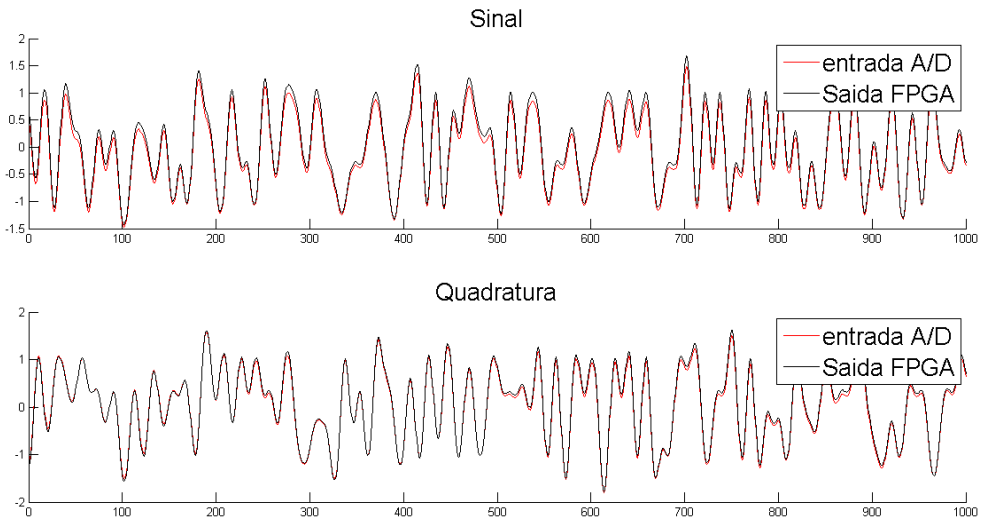

Figura 46 – Sinal IQ no MATLAB®.

4.2.2 SINAL UTILIZADO PARA IDENTIFICAÇÃO

O sinal ideal a ser utilizado para identificação de sistemas é um ruído branco gaussiano [52] [53]. Pelo fato deste sinal possuir componentes em todo espectro de frequências, ele se torna ideal para a identificação de um sistema, porém, é impossível de ser implementado.

Os sinais utilizados no projeto são, na maioria, transmitidos com algum pulso cosseno levantado com largura de banda restrita conforme descrito no capítulo 3. Um sinal de banda limitada não excitaria o sistema por completo. Por essa razão, nessa parte foi utilizado um Trem de Pulsos complexo (como uma componente em fase e outra em quadratura) com diferentes amplitudes para a excitação do sistema, pois, dentro das possibilidades de geração do FPGA, esse sinal é o que mais se aproxima de um ruído branco.

4.2.3 RESULTADOS DA IDENTIFICAÇÃO

Primeiramente, para captura dos dados de entrada denominados X, o algoritmo PN foi desligado. Ao invés de uma sequência aleatória, um conjunto de 16000 bits gerados aleatoriamente no MATLAB® é convertido para binário, e essa sequência é escrita no FPGA para ser transmitida. Esses dados passam somente pelo transformador, como descrito na Figura 47.

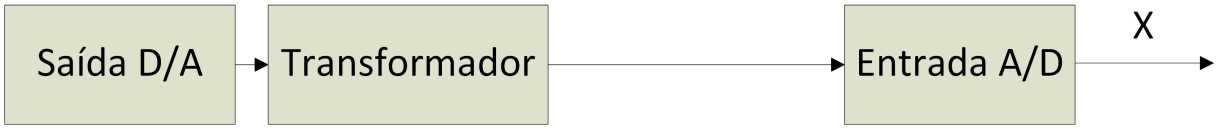

Figura 47 – Fluxograma da identificação.

Para a identificação do filtro, os mesmos 16000 dados utilizados para obtenção do X são transmitidos com o filtro, como demonstrado na Figura 48. Essa transmissão é utilizada para obtenção do Y.

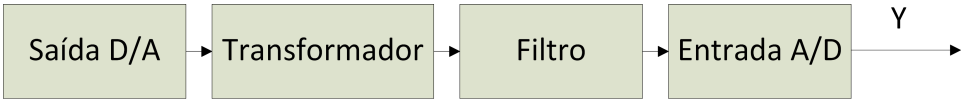

Figura 48 – Fluxograma da identificação.

Devido a capacidade do algoritmo implementado em MATLAB® de encontrar o ponto de maior correlação, pode-se utilizar os dados de saída descritos na Figura 47 como os dados de entrada X do identificador, e os dados descritos na Figura 48, como os dados de saída Y. A captura de dados realizada dessa maneira remove os efeitos provocados pelo transformador e se tem uma identificação somente do filtro proposto.

4.2.3.1 IDENTIFICAÇÃO SEM REALIMENTAÇÃO

Nesta seção, foi selecionada a opção de identificação sem realimentação no algoritmo de identificação. A ordem foi limitada a dez e os sinais utilizados foram os descritos na seção 4.2.2. A Figura 49 mostra a evolução do Erro Quadrático Médio Normalizado (NMSE) para as diferentes ordens.

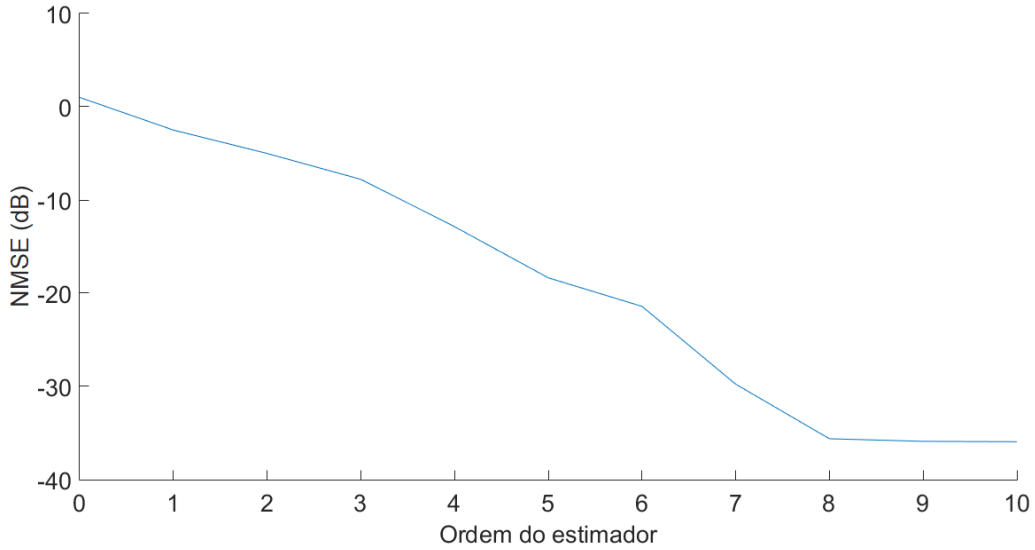

Figura 49 – NMSE da Identificação FIR em função da ordem do estimador.

Na Figura 49 observa-se que, a partir da ordem 8, o algoritmo alcança um mínimo de -35.9 dB. Esperava-se que a ordem necessária fosse maior do que 5, pois o filtro Butterworth possui realimentação, e para se representar um modelo IIR por um FIR seria necessária uma ordem mais elevada.

Os coeficientes b_k obtidos no estimador FIR foram: [0,051371 0,24276 0,31508 0,28575 0,19896 0,11483 0,052801 0,017799 0,0026624]. Como o filtro Butterworth possui uma fase aproximadamente linear, os coeficientes complexos apresentaram valores próximos a zero, por isso, considera-se apenas os coeficientes reais.

Agora, com os coeficientes obtidos pelo algoritmo pode-se observar a resposta do modelo em MATLAB® utilizando o comando fdatool. A Figura 50 mostra os resultados obtidos.

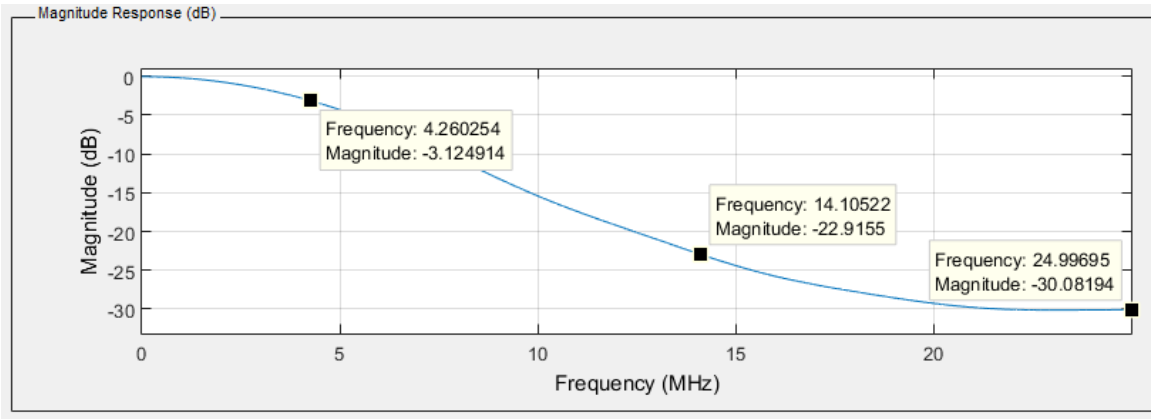

Figura 50 – Curva observada no fdatool para uma estimação FIR.

Na Figura 50, as atenuações do modelo, nos pontos de 4.3 e 14.3 MHz, são as

mesmas se comparado com a curva do filtro analisada pelo Analisador de Redes Keysight E5061B, mostrada na Figura 44. Em 25 MHz ocorre uma diferença de 5 dB para o modelo FIR, devido provavelmente ao nível do sinal ser extremamente pequeno.

4.2.3.2 IDENTIFICAÇÃO COM REALIMENTAÇÃO

Outra possibilidade de identificação é a opção com realimentação. Assim como, no sistema sem realimentação, a ordem foi limitada a dez e os sinais utilizados foram os descritos na seção 4.2.2. A Figura 51 mostra a evolução do Erro Quadrático Médio Normalizado (NMSE) para as diferentes ordens.

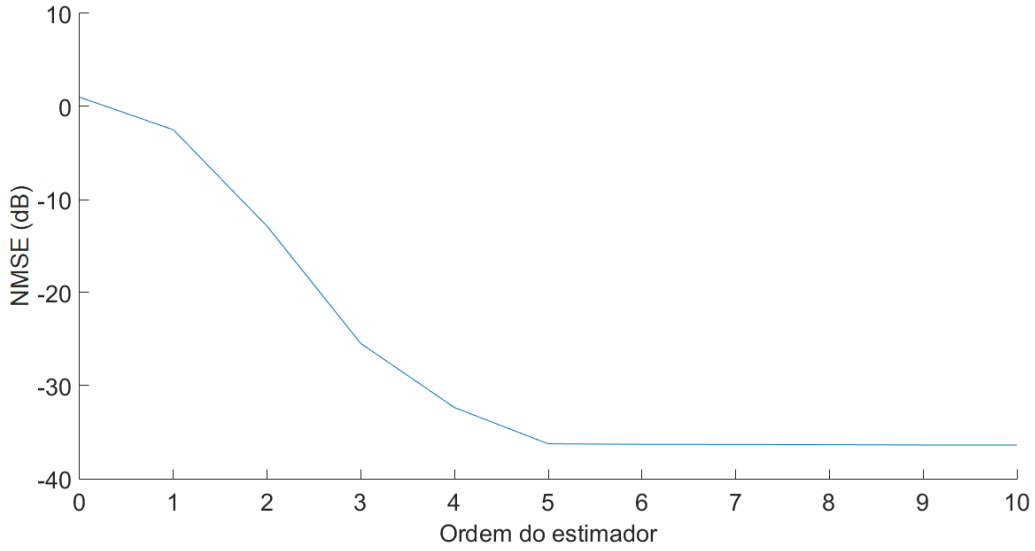

Figura 51 – NMSE da Identificação IIR em função da ordem do estimador.

Observa-se que, a partir da ordem cinco, o algoritmo alcança um mínimo de -36.4 dB. Era esperado que essa ordem fosse suficiente para descrever o filtro, pois na sua construção foi definido que ele é de ordem cinco, como observado no início dessa seção.

Os coeficientes b_k obtidos no estimador IIR foram: [0,00000 -0,0002 -0,0001 0,0403 0,1557 0,0811], enquanto os a_k foram [1 -0,8275 -0.1972 0,43089 -0,12771 -0,006127]. Da mesma forma que no modelo FIR, o filtro Butterworth possui uma fase aproximadamente linear, os coeficientes complexos apresentaram valores próximos a zero, por isso, considera-se apenas os coeficientes reais.

Com os coeficientes obtidos pelo algoritmo pode-se observar a resposta do modelo em MATLAB® utilizando o fdatool. A Figura 52 mostra os resultados obtidos.

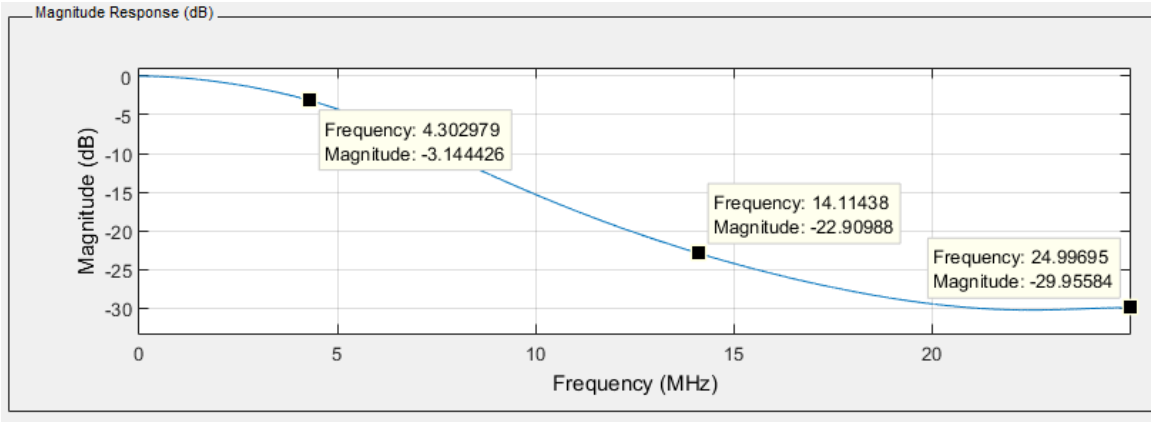

Figura 52 – Curva observada no fdatool para uma estimação IIR.

Observa-se, na Figura 52, que nas frequências de 4.3 e 14.3 MHz a resposta do modelo se comparado com a do filtro analisada pelo Analisador de Redes Keysight E5061B são muito próximos. Em 25 MHz ocorre uma diferença de 5 dB, provavelmente, por essa região já estar sujeita a identificação de ruídos e a descasamento de impedância provocada pela faixa de operação do transformador.

4.3 RESUMO DO CAPÍTULO

Viu-se nesse capítulo que é possível a construção de um modelo Caixa Preta somente a partir dos dados de entrada e saída. O estimador utilizado é o MQ, que pode ser recursivo ou não-recursivo. O algoritmo de estimação cria vários modelos para superar a dificuldade de se alinhar os dados de entrada e saída.

O algoritmo implementado em MATLAB® estimou com uma boa precisão um filtro inserido como dispositivo a ser identificado. Esse fato pode ser comprovado pela comparação com a curva gerada pelo Analisador de Redes Keysight E5061B e pela ordem do estimador IIR, que foi exatamente a ordem proposta pelo filtro. A qualidade do modelo também pode ser avaliada pelo NMSE apresentado, que em ambos os casos foi inferior a -35dB.

Um filtro IIR de ordem cinco possui 12 coeficientes, enquanto o FIR de ordem 8 possui 9. A depender da aplicação, esse número menor de coeficientes pode indicar que a escolha de um modelo FIR para identificação de um filtro Butterworth é melhor, já que a diferença de desempenho é de apenas 0,5 dB.

5 CONSTRUÇÃO DE UMA PLATAFORMA DE PRÉ-DISTORÇÃO PARA USO COM FPGA

O Filtro identificado no Capítulo 4 possui um corte em 4,3 MHz. Como visto no Capítulo 3, para taxa de 12,5 MS/s, o FPGA gera sinais cosseno-levantado com larguras de banda entre 6,25 e 12,5 MHz. Portanto, ao transmitir-se esses sinais através do filtro as frequências acima de 4,3 MHz serão atenuadas. O filtro é implementado para ser uma alteração induzida no canal.

A alteração induzida no canal por um dispositivo modifica o sinal transmitido. Essa modificação não é desejada, pois espera-se que o sinal recebido seja o mais próximo possível do transmitido. Figuras de Mérito são utilizadas para quantificar precisamente estas alterações no sinal.

Neste capítulo será detalhado um algoritmo em FPGA capaz de aplicar uma função FIR ou IIR de ordem igual ou inferior a 7 ao sinal antes de enviá-lo ao DAC, com o propósito de realizar uma pré-distorção e eliminar o efeito indesejado de um dispositivo no sinal.

5.1 SISTEMAS INVERSOS

Um sistema é tudo aquilo que altera um sinal. Por exemplo, na equação (5.1), o sinal $y[n]$ é gerado pela convolução da entrada $x[n]$ com o sistema $h_1[n]$ [31].

$$y[n] = x[n] * h_1[n] \tag{5.1}$$

O sistema descrito na equação (5.1) pode ser representado no espectro das frequências, como mostrado na equação (5.2).

$$Y(j\omega) = X(j\omega)H_1(j\omega) \tag{5.2}$$

Um sistema h_2 pode ser descrito como o inverso de h_1 quando a convolução dos dois resulta em um impulso unitário $\delta[n]$ [54]. Essa representação é demonstrada na equação (5.3).

$$h_1[n] * h_2[n] = \delta[n] \tag{5.3}$$

No espectro das frequências, um sistema inverso H_2 pode ser descrito por meio da equação (5.4).

$$H_1(j\omega)H_2(j\omega) = 1 \tag{5.4}$$

O sistema inverso obedece a equação (5.5).

$$H_2(j\omega) = \frac{1}{H_1(j\omega)} \tag{5.5}$$

Uma maneira de visualizar a complementariedade dos sistemas $H_1(j\omega)$ e $H_2(j\omega)$ é através da Figura 53, onde o módulo da função inversa é exatamente o complementar de $H_1(j\omega)$ dentro da largura de banda B. Essa definição se aplica a qualquer sistema inverso [4].

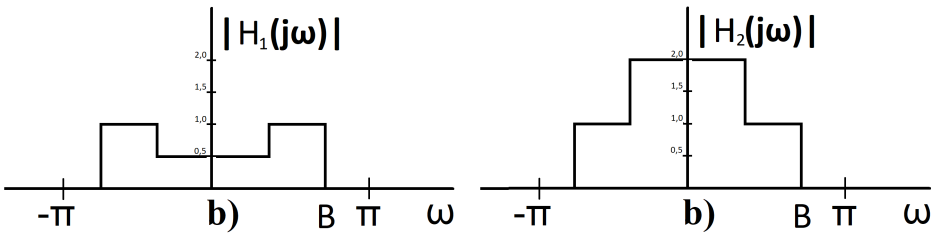

Figura 53 – Representação do sistema $H_1(j\omega)$ e do seu inverso $H_2(j\omega)$.

Como o $h_2[n]$ pode ser não causal (a representação no tempo discreto de $H_2(j\omega)$, obtido na equação (5.5)), essa solução pode ser impossível em um sistema implementado em um processador. Uma solução que satisfaça a equação (5.6) pode ser uma resposta possível, onde A representa o atraso.

$$h_1[n] * h_2[n] = \delta[n - A] \tag{5.6}$$

O atraso na equação (5.6) não interferirá no módulo de $H_1(j\omega)$ e $H_2(j\omega)$.

5.1.1 IMPLEMENTAÇÃO DA PRÉ-DISTORÇÃO EM FPGA

O FPGA utiliza linguagem de baixo nível se comparado com o MATLAB®. Devido a essa maior simplicidade das instruções, a implementação de uma lógica com números complexos é dificultada, pois, para se aplicar os coeficientes complexos no FPGA, são necessárias operações adicionais. A seguir será descrito como as operações com números complexos são realizadas no FPGA.

A equação (5.7) define como se encontra o sinal em fase y_r a partir das entradas em fase x_r e em quadratura x_q combinadas com a parte real do estimador a e a parte

complexa a^*. São somadas à equação as saídas passadas reais y_r e as saídas em quadratura y_q combinadas com a parte real do estimador b e a parte complexa b^*. A ordem n pode ser no máximo 7 para o algoritmo implementado em FPGA com e sem realimentação.

$$\begin{aligned} & y_r[k] = \\ & x_r[k]b_0 + x_r[k-1]_1 ... + x_r[k-m]b_m + \\ & + y_r[k-1]a_1 ... + y_r[k-n]a_n + \\ & + x_q[k]b_0^* + x_q[k-1]b_1^* ... + x_q[k-m]b_m^* + \\ & + y_q[k-1]a_1^* ... + y_q[k-n]a_n^* \end{aligned} \tag{5.7}$$

Como feito para o sinal em fase, na equação anterior, obtém-se a saída para o sinal em quadratura $y_q(k)$ na equação (5.8).

$$\begin{aligned} & y_q[k] = \\ & x_q[k]b_0 + x_q[k-1]_1 ... + x_q[k-m]b_m + \\ & + y_q[k-1]a_1 ... + y_q[k-n]a_n + \\ & + x_r[k]b_0^* + x_r[k-1]b_1^* ... + x_r[k-m]b_m^* + \\ & + y_r[k-1]a_1^* ... + y_r[k-n]a_n^* \end{aligned} \tag{5.8}$$

Uma maneira de visualizar as duas equações anteriores é através da Figura 54, que sintetiza como as operações são implementadas. Os ramos de coeficientes são aplicados todos em paralelo.

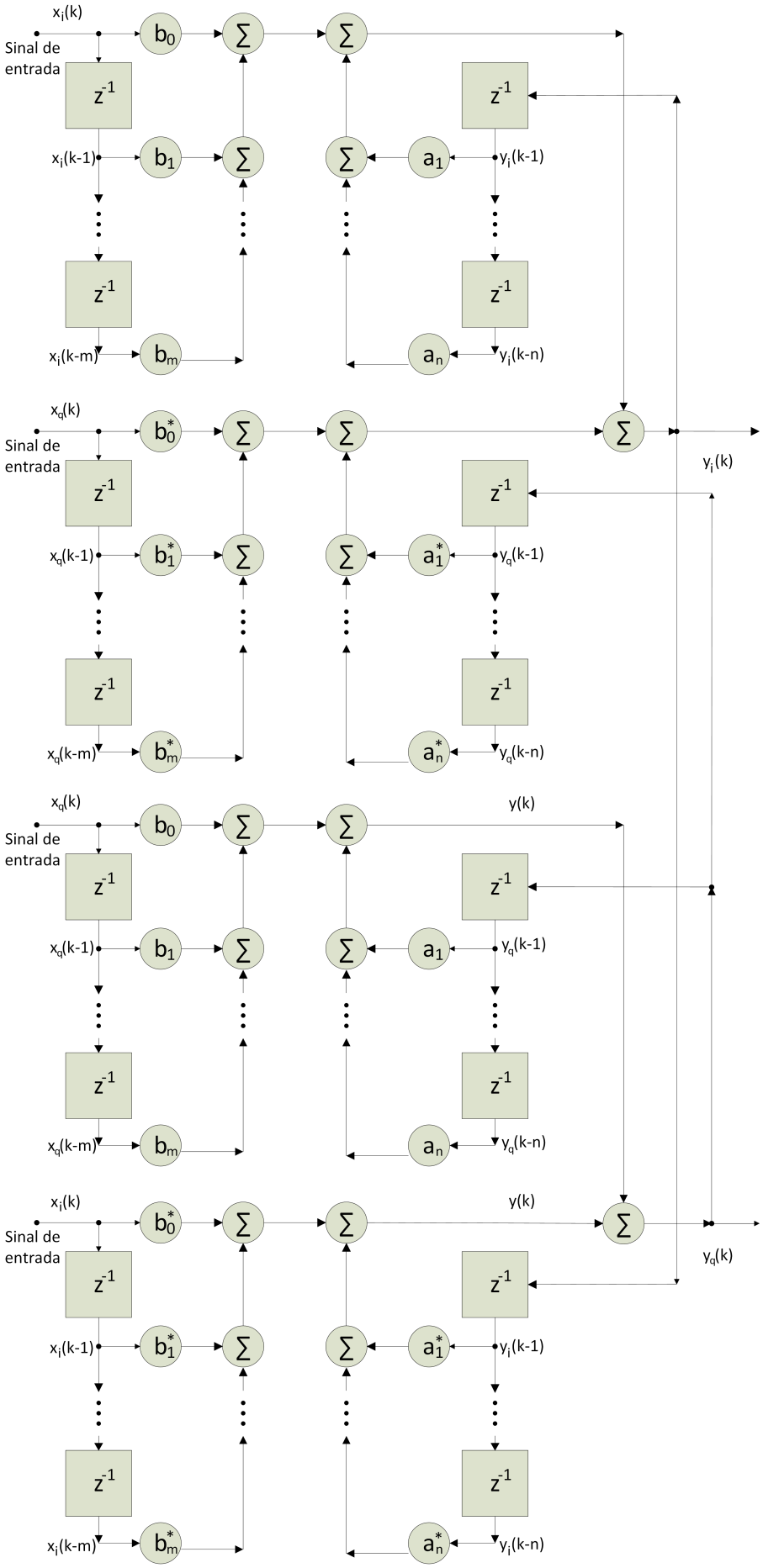

Figura 54 – Fluxograma da pré-distorção complexa implementado no FPGA.

5.1.2 SINAL UTILIZADO PARA CRIAÇÃO DE UM MODELO DE PRÉ- DISTORÇÃO

Um método de estimação indireta pode ser utilizado para a criação de um modelo inverso [20] [21]. Nesse método, a ordenação dos dados é invertida. A entrada agora é o sinal anteriormente capturado como saída do sistema Y e a saída é o X. A captura dos dados é feita no SignalTap, como descrito na seção 4.2.1, e a ordem da captura é demonstrada pela Figura 55.

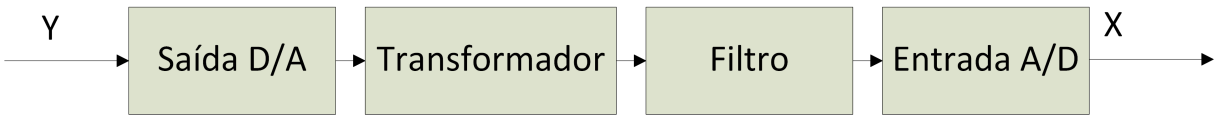

Figura 55 – Fluxograma da identificação do sistema completo.

A captura dos dados foi feita com o transformador para que os efeitos não-lineares ocasionados pelo mesmo fossem levados em conta. Quando o sistema inverso foi montado, utilizando apenas o modelo do filtro, o NMSE ficou limitado a -16 dB, não sendo considerado um desempenho suficiente.

Primeiramente, o sinal utilizado para encontrar os coeficientes da pré-distorção foi o trem de impulsos. Esse gerou os resultados mostrados nas Figuras 56 e 57.

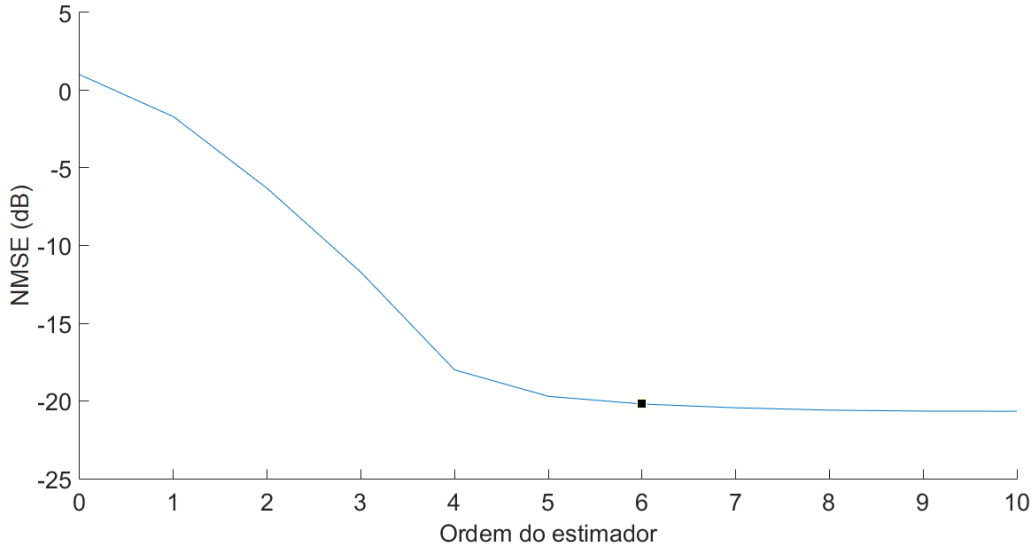

Figura 56 – NMSE da pré-distorção FIR de acordo com a ordem.

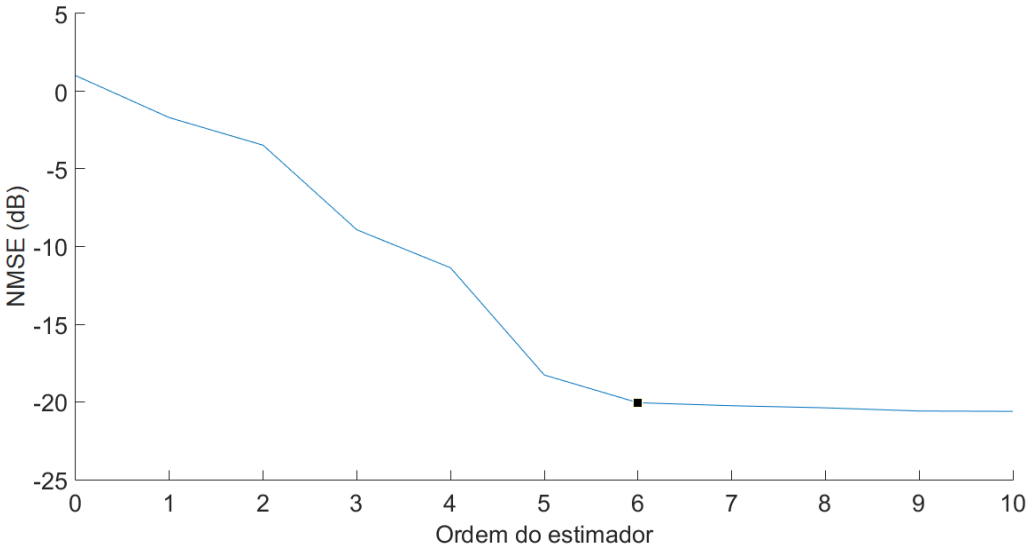

Figura 57 – NMSE da pré-distorção IIR de acordo com a ordem.

A aplicação destes coeficientes ao FPGA gerou resultados reais com NMSE acima dos -20dB. Além disso, a $\%EVM$ medida pelo CXA ficava acima dos 10%. Esses resultados representavam uma significativa melhora se comparados ao sinal sem pré-distorção (NMSE acima dos -13dB e $\%EVM$ acima dos 20%), mas poderiam ser melhorados.

Uma alternativa interessante é utilizar um sinal filtrado para estimação, pois os sinais transmitidos são de banda limitada. A pré-distorção não precisa compensar os efeitos do filtro em uma frequência onde o sinal não existe. Por isso foi utilizado o sinal 4-QAM com RO igual a 1 para a identificação. Esse RO foi escolhido por ser o que apresenta a maior banda entre os sinais cosseno levantado disponíveis. Montar uma pré-distorção com uma banda muito estreita prejudicaria o desempenho quando essa for aplicada a um sinal com uma banda superior à do sinal utilizado para a obtenção dos coeficientes de pré-distorção.

As figuras 58 e 59 demonstram os resultados utilizando esse método para diferentes ordens.

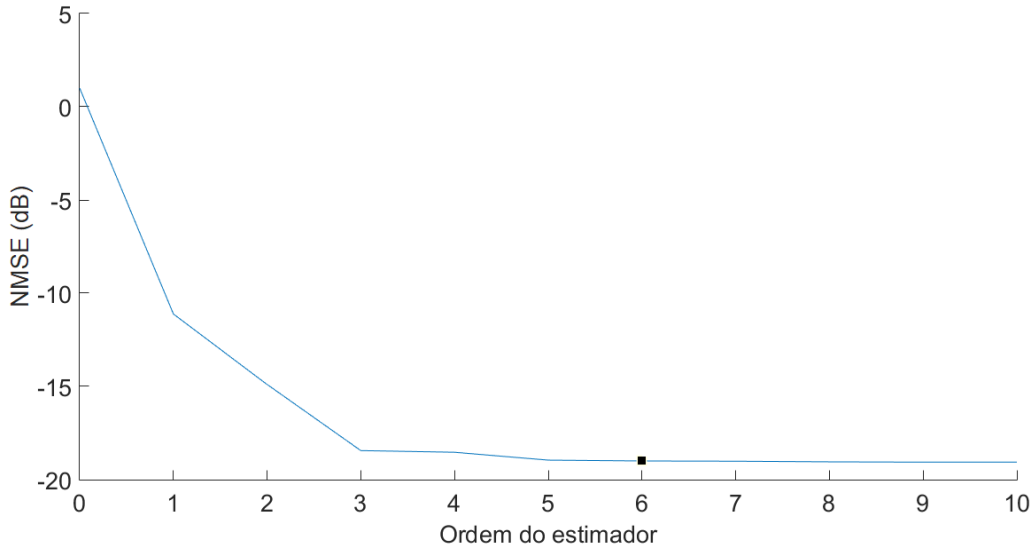

Figura 58 – NMSE da pré-distorção FIR de acordo com a ordem.

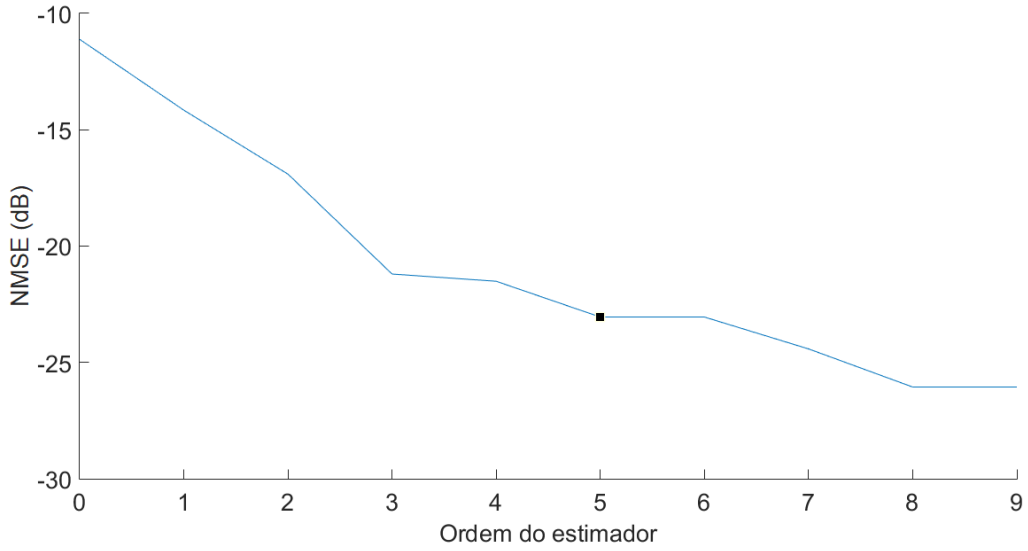

Figura 59 – NMSE da pré-distorção IIR de acordo com a ordem.

Os coeficientes b_k para o estimador FIR foram: [-0,56877 1,4852 2,6329 -4,5028 1,6175 0,41514 -0,26085]. Os coeficientes b_k para o estimador IIR foram:[-0,28815 1,9135 -2,5342 1,0566 0,021433 -0,16563], enquanto os a_k são definidos por: [-2,2247 2,8344 -2,5229 1,3263 -0,41186]. Assim como nos modelos de pré-distorção, a parte complexa dos coeficientes pode ser desconsiderada por possuir uma amplitude bem menor.

Comparando-se as Figuras 57 e 58 pode-se inferir que utilizar o trem de impulsos FIR é melhor. Os testes reais demonstraram que ao utilizar essa função em um pulso cosseno levantado o NMSE ficava limitado em -17dB, enquanto o segundo método mantém

aproximadamente os mesmos -19 dB. O erro deste estimador ficou concentrado em baixas frequências, ocasionando uma interferência maior no sinal de banda limitada.

Na Figura 60 pode-se ver as curvas de todas as funções de pré-distorção no fdatool.

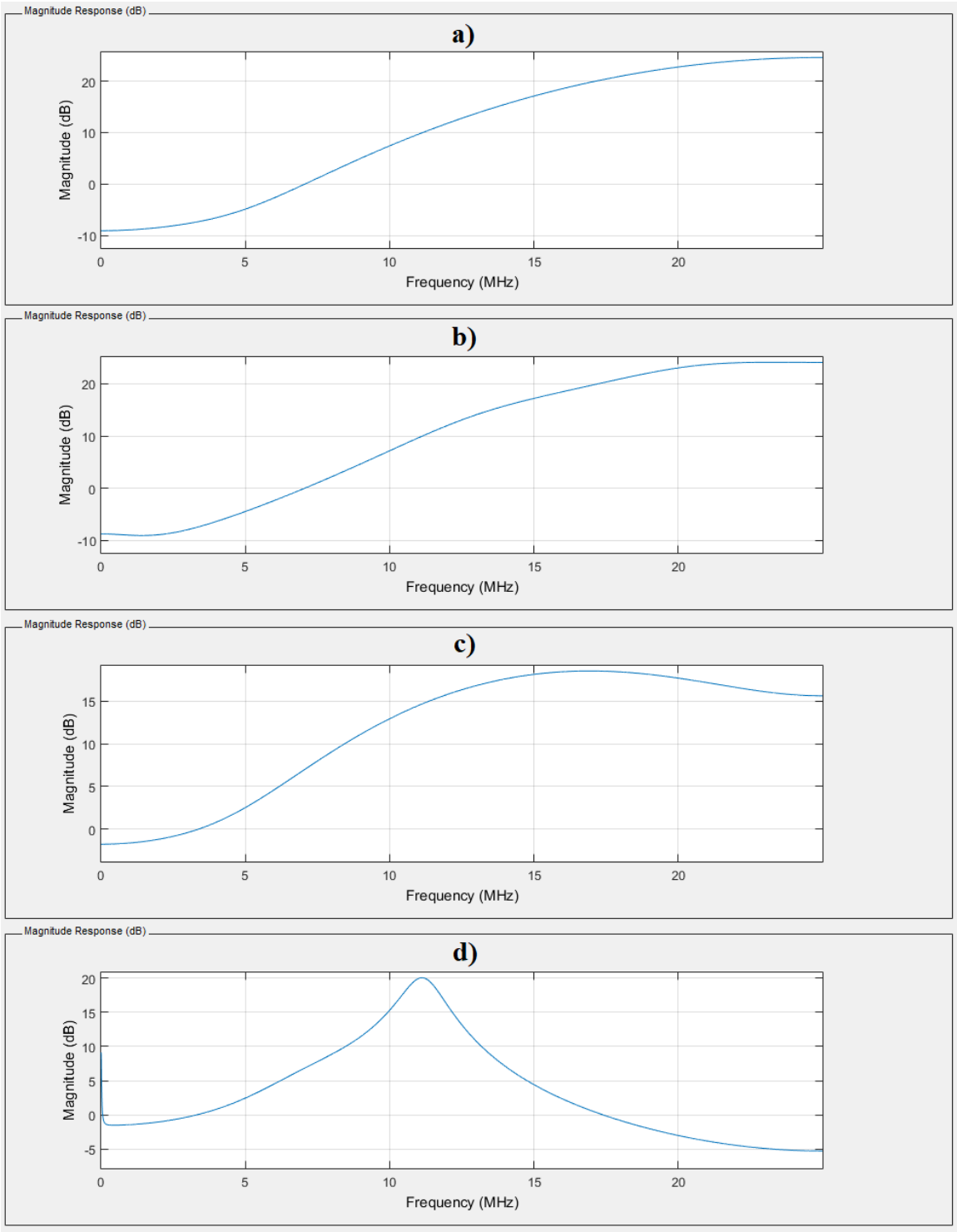

Figura 60 – Curvas de pré-distorção estimados com: a) Trem de impulsos e estimador FIR; b) Trem de impulsos e estimador IIR; c) Cosseno-levantado e estimador FIR; d) Cosseno-levantado e estimador IIR.

Como visto na Figura 60, o estimador IIR com sinal cosseno-levantado não amplificou o ruído para as frequências mais altas, além disso, compensou o efeito do transformador nas frequências mais baixas. Para a transmissão de sinais com banda limitada esse estimador apresentou um melhor desempenho. Nas próximas seções, são mostrados os resultados da aplicação desses coeficientes no FPGA.

5.2 RESULTADOS NA EVM OBTIDOS APLICANDO PRÉ-DISTORÇÃO

A EVM descreve a diferença entre o ponto onde o sinal deveria estar e o ponto onde ele está no momento da decisão. Para a medição do EVM, o sinal digital gerado pelo FPGA é convertido para um sinal contínuo pelo DAC, após isso, o sinal passa pelo transformador e pelo filtro utilizando cabos coaxiais. Saindo do filtro, o sinal não volta para o ADC como no processo utilizado para a identificação na seção 4.2, mas agora é enviado para o Gerador de RF Keysight N9310A, que translada esse sinal para frequência de 20 MHz. O sinal em banda de RF é analisado pelo CXA, assim como o processo utilizado para a visualização dos espectros no capítulo 3. Esse setup pode ser visualizado na Figura 61.

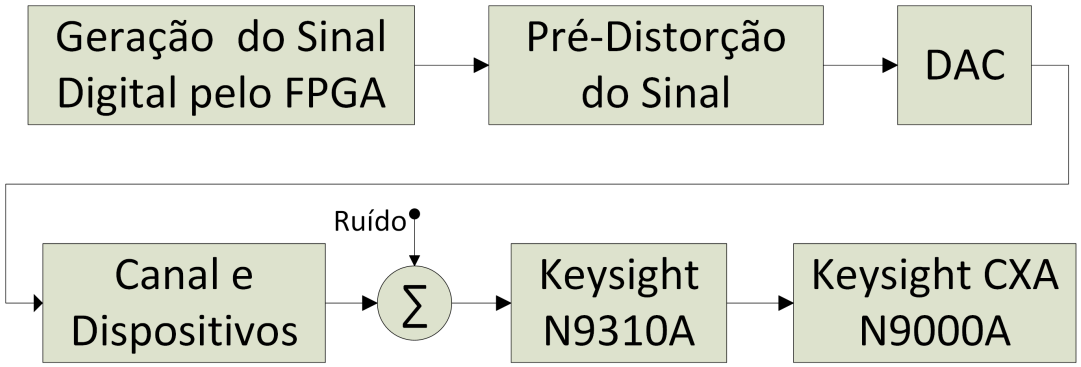

Figura 61 – Setup para medição de EVM pelo CXA 9000A.

Na Figura 62, observa-se o EVM medido pelo CXA 9000A de um sinal QPSK submetido a um filtro de saída cosseno levantado com RO=1.

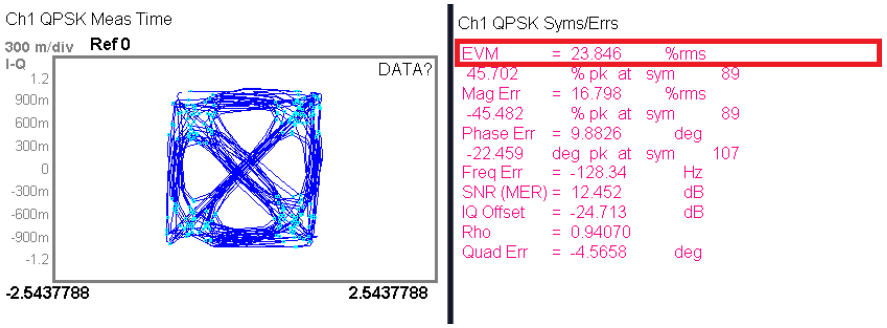

Figura 62 – EVM sem pré-distorção para um sinal QPSK submetido a um filtro de saída cosseno levantado com RO=1, de 23,5 %.

Aplicando-se a pré-distorção IIR, a EVM apresenta uma grande melhora, como descrito na Figura 63.

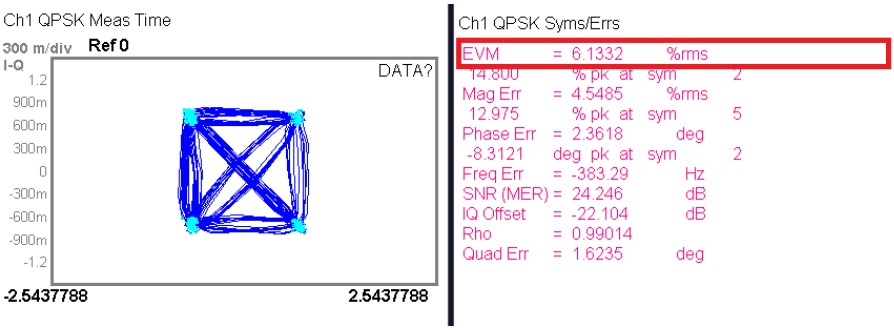

Figura 63 – EVM com pré-distorção IIR para um sinal QPSK submetido a um filtro de saída cosseno levantado com RO=1, de 6,13 %.

O caso anterior demonstra a aplicação da pré-distorção em um sinal com RO=1, que é o mesmo caso utilizado para se encontrar os coeficientes para pré-distorção. Ficaria inviável, no trabalho, apresentar a tela do equipamento para todos os sinais que o FPGA pode gerar. Por isso, optou-se por mostrar esses resultados em tabelas.

A Tabela 1 mostra o EVM para todos os casos que o equipamento consegue medir. Para cada possibilidade, a medida foi realizada três vezes. Não foi possível o cálculo do EVM para sinais com RO menor do que 0.05 e para as constelações 4-PSK e 16-PSK, pois não existem essas opções no CXA.

Tabela 1 – EVM antes da pré-distorção

RO	**0,0**			**0,2**			**0,4**		
Modulação \ Teste	1	2	3	1	2	3	1	2	3
BPSK	-	-	-	21,7	19,8	18,7	21,4	21,2	20,8
8-PSK	-	-	-	21,5	25,6	23,3	23,5	21,5	25,9
4-QAM	-	-	-	21,3	20,7	20,5	19,7	22,7	21,2
16-QAM	-	-	-	19,5	20,0	20,6	18,9	18,9	20,6
32-QAM	-	-	-	11,2	11,9	11,7	12,7	12,0	11,5
64-QAM	-	-	-	8,7	8,1	9,0	8,2	8,3	9,1
RO	**0,6**			**0,8**			**1,0**		
Modulação \ Teste	1	2	3	1	2	3	1	2	3
BPSK	21,7	22,4	23,7	22,3	24,8	23,3	22,7	23,2	23,3
8-PSK	27,0	26,8	22,9	27,3	28,0	28,9	28,8	26,0	28,4
4-QAM	23,0	22,5	23,1	23,4	24,4	24,6	23,8	24,8	24,8
16-QAM	20,0	19,7	20,1	18,8	18,8	20,4	20,1	19,8	20,3
32-QAM	12,6	11,9	11,6	12,6	11,6	12,3	11,5	13,7	12,1
64-QAM	8,2	9,3	8,8	8,9	8,4	8,2	8,4	8,3	8,8

De acordo com a Tabela 1, o EVM melhora conforme o número de bits transmitidos por símbolo aumenta. Não necessariamente este número demonstra que o sinal transmitido chegou íntegro para estes casos. Quando se tem mais possibilidades de símbolos, pode-se supor que a distância do sinal para algum símbolo será menor no ponto de decisão, melhorando, assim, o resultado para sinais que têm mais possibilidades. A Figura 64 demonstra que, em comparação com um 4-QAM, a distância mínima entre os símbolos de um 16-QAM é três vezes menor. Para o caso 32-QAM a distância mínima é cinco vezes menor e para o 64-QAM é sete vezes menor.

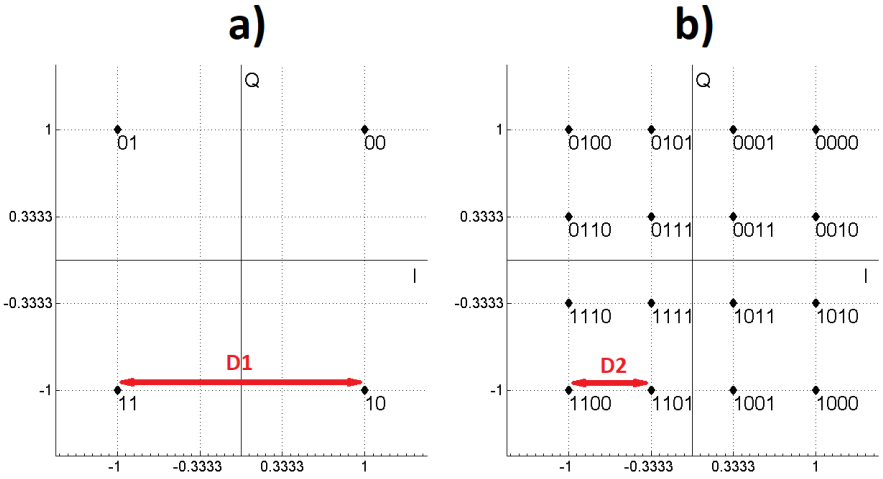

Figura 64 – Distância entre os símbolos de uma constelação: a) 4-QAM e b)16-QAM.

Essa particularidade faz com que a EVM seja boa para medir a melhora de um

sinal após a pré-distorção quando se considera a mesma constelação, mas não seja ideal para se comparar a qualidade de sinais com constelações diferentes.

A Tabela 2 mostra os resultados para uma pré-distorção FIR.

Tabela 2 – EVM com pré-distorção FIR

RO	**0,0**			**0,2**			**0,4**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	-	-	-	9,0	9,3	9,3	8,4	9,6	8,5
8-PSK	-	-	-	8,9	10,3	9,4	9,4	9,1	8,2
4-QAM	-	-	-	9,7	9,5	10,3	9,3	8,5	8,4
16-QAM	-	-	-	9,0	7,5	7,7	6,5	8,3	6,8
32-QAM	-	-	-	6,6	6,6	6,8	5,9	5,6	6,4
64-QAM	-	-	-	8,1	6,2	6,2	8,3	8,1	7,6
RO	**0,6**			**0,8**			**1,0**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	9,6	9,1	7,5	7,9	7,4	7,5	8,7	8,4	8,3
8-PSK	8,8	7,8	8,4	6,8	7,8	7,9	9,3	8,6	8,2
4-QAM	9,1	7,5	9,1	5,6	8,2	8,8	7,4	6,8	8,1
16-QAM	6,5	6,6	7,7	4,6	5,0	6,3	5,1	6,1	5,1
32-QAM	6,5	5,5	5,6	5,1	4,7	4,6	5,6	4,3	5,5
64-QAM	6,2	7,4	7,4	7,0	5,9	5,4	4,4	5,3	5,8

A Tabela 3 mostra o EVM após a pré-distorção IIR.

Tabela 3 – EVM com pré-distorção IIR

RO	**0,0**			**0,2**			**0,4**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	-	-	-	7,6	7,8	8,0	7,7	7,0	7,1
8-PSK	-	-	-	8,4	8,4	7,5	7,9	7,3	7,3
4-QAM	-	-	-	8,3	7,8	8,0	7,7	7,4	7,4
16-QAM	-	-	-	6,0	5,8	6,0	5,7	5,6	5,8
32-QAM	-	-	-	5,3	5,7	5,7	4,9	4,8	5,1
64-QAM	-	-	-	5,4	5,3	6,0	4,8	5,3	4,9
RO	**0,6**			**0,8**			**1,0**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	7,1	6,9	7,3	6,6	6,5	6,1	6,6	6,6	7,1
8-PSK	7,1	6,4	7,2	6,8	6,9	6,3	6,4	6,2	6,5
4-QAM	6,6	6,8	6,9	5,6	5,8	5,9	6,1	6,2	6,7
16-QAM	5,3	4,9	5,4	4,6	5,0	5,0	4,8	4,4	4,8
32-QAM	4,6	4,6	4,5	4,3	4,2	4,1	3,8	4,2	4,0
64-QAM	4,9	4,4	4,6	4,4	4,2	4,0	4,2	4,5	4,3

A pré distorção melhorou o EVM para todos os casos. O EVM para os sinais com mais bits por símbolo continuam melhores pelo mesmo fato do caso sem pré-distorção.

Como esperado, os resultados para uma pré-distorção FIR foram piores, se comparados com a IIR, mas se comparado com o caso sem pré-distorção, houve uma melhora significativa. A Figura 65 mostra a média das EVMs com e sem pré-distorção para cada constelação.

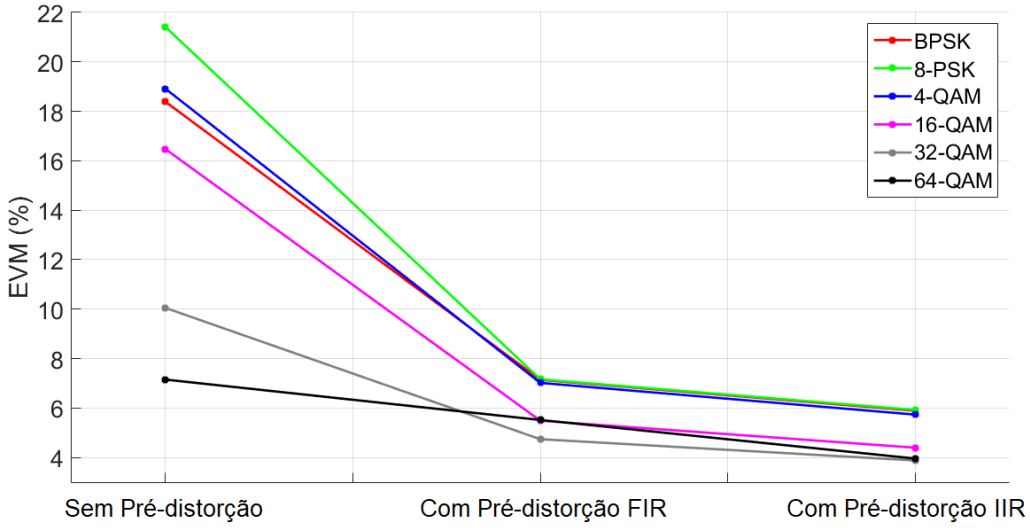

Figura 65 – EVM com e sem pré-distorção

5.3 RESULTADOS CALCULADOS NO MATLAB

Assim como na seção 4.2, para o cálculo do NMSE, MAE e BER foi utilizado o setup descrito na Figura 66.

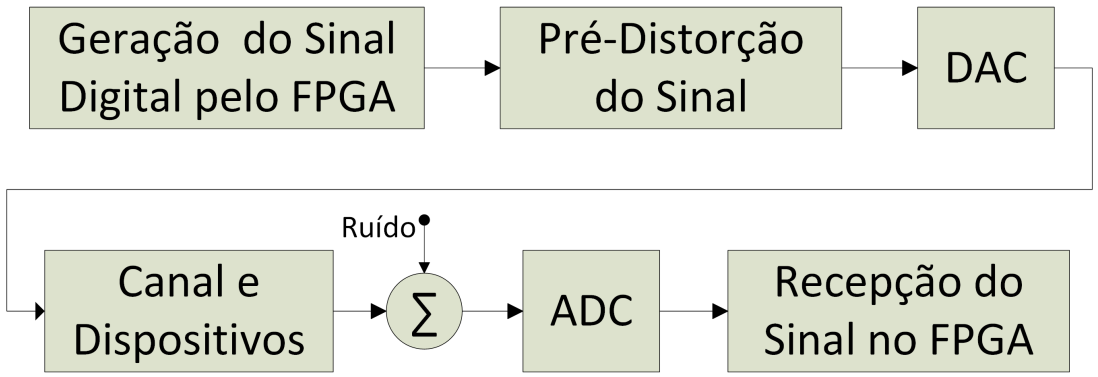

Figura 66 – Setup utilizado para medida do NMSE, MAE e BER.

Os dados são capturados no SignalTap descrito na Figura 67. O cálculo do NMSE e do MAE é feito pela comparação dos sinais a) e b) com os sinais e) e f). Para o cálculo da BER é construído um algoritomo em MATLAB® que a partir dos sinais e) e f) descreve qual símbolo foi recebido, e através da comparação com os bits transmitidos g) realiza o cálculo da BER.

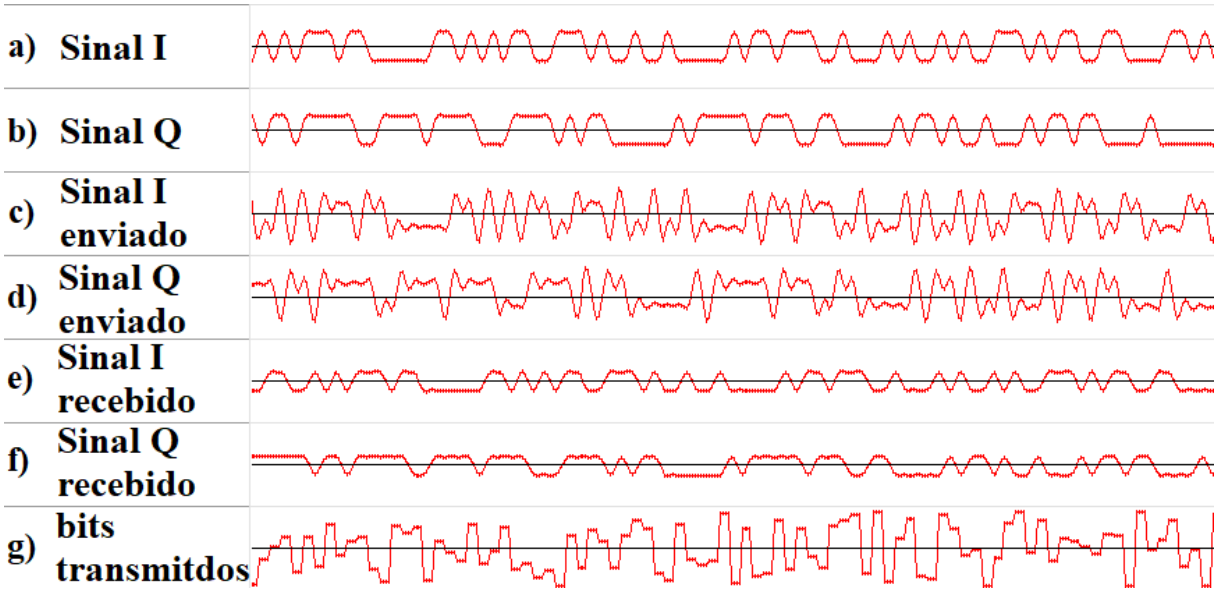

Figura 67 – Transmissão com pré-distorção IIR: a)Sinal I; b)Sinal Q; c) Sinal I com pré-distorção; d) Sinal Q com pré-distorção; e) Sinal I recebido; f) Sinal Q recebido; g) bits transmitidos

Para todos os casos são capturados 32000 pontos de cada sinal.

5.3.1 RESULTADOS POR ERRO QUADRÁTICO MÉDIO NORMALIZADO

O cálculo do NMSE foi realizado utilizando os dados provenientes do SignalTap analisados em MATLAB®. Esses resultados, para um sistema sem pré-distorção, podem ser vistos na Tabela 4.

Tabela 4 – NMSE sem pré-distorção

RO	**0,0**			**0,2**			**0,4**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	-11,9	-12,0	-12,1	-12,3	-12,3	-12,3	-12,5	-12,5	-12,4
4-PSK	-12,1	-12,1	-12,1	-12,5	-12,4	-12,5	-12,6	-12,6	-12,5
8-PSK	-12,1	-12,1	-12,1	-12,5	-12,5	-12,4	-12,5	-12,6	-12,5
16-PSK	-12,2	-12,2	-12,2	-12,4	-12,5	-12,5	-12,6	-12,5	-12,6
4-QAM	-12,1	-12,1	-12,1	-12,4	-12,4	-12,5	-12,5	-12,5	-12,6
16-QAM	-12,1	-12,1	-12,1	-12,4	-12,5	-12,5	-12,5	-12,6	-12,5
32-QAM	-12,2	-12,1	-12,2	-12,5	-12,4	-12,5	-12,6	-12,5	-12,5
64-QAM	-12,1	-12,1	-12,1	-12,4	-12,4	-12,4	-12,5	-12,5	-12,5
RO	**0,6**			**0,8**			**1,0**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	-12,2	-12,1	-12,1	-11,6	-11,6	-11,8	-11,1	-11,0	-11,0
4-PSK	-12,2	-12,3	-12,2	-11,8	-11,8	-11,8	-11,0	-11,1	-11,0
8-PSK	-12,3	-12,3	-12,2	-11,8	-11,7	-11,7	-11,0	-11,1	-11,1
16-PSK	-12,2	-12,2	-12,3	-11,7	-11,7	-11,7	-11,1	-11,1	-11,1
4-QAM	-12,3	-12,3	-12,2	-11,8	-11,8	-11,8	-11,0	-11,2	-11,1
16-QAM	-12,3	-12,3	-12,3	-11,7	-11,7	-11,7	-11,1	-11,0	-11,1
32-QAM	-12,3	-12,3	-12,2	-11,8	-11,7	-11,7	-11,1	-11,1	-11,1
64-QAM	-12,3	-12,2	-12,3	-11,8	-11,7	-11,7	-11,1	-11,1	-11,1

Como era de se esperar, para os sinais com RO maior, o NMSE foi pior. Isso é causado por uma maior influência do filtro para esses casos, pois quanto maior o RO, maior é a largura de banda do sinal e maior será a banda após o corte de 4.3 MHz do filtro.

A influência da pré-distorção FIR no NMSE pode ser vista na Tabela 5.

Tabela 5 – NMSE com pré-distorção FIR

RO	**0,0**			**0,2**			**0,4**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	-20,1	-20,5	-20,8	-20,4	-19,8	-20,5	-20,3	-19,8	-19,8
4-PSK	-20,0	-20,0	-20,0	-19,7	-19,6	-20,1	-19,6	-19,6	-19,5
8-PSK	-20,1	-19,9	-20,1	-19,8	-19,6	-20,2	-19,8	-19,8	-19,5
16-PSK	-20,2	-20,5	-20,3	-20,2	-20,2	-20,1	-19,7	-20,0	-20,0
4-QAM	-20,0	-20,3	-20,0	-20,1	-19,8	-19,7	-19,4	-19,9	-19,4
16-QAM	-20,1	-20,1	-20,1	-19,9	-19,7	-20,2	-19,9	-19,7	-19,9
32-QAM	-20,3	-20,5	-20,4	-20,0	-20,3	-20,3	-19,9	-20,1	-20,1
64-QAM	-19,8	-20,1	-20,1	-19,6	-20,0	-19,6	-19,7	-19,8	-19,7
RO	**0,6**			**0,8**			**1,0**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	-20,0	-19,5	-19,2	-19,4	-19,3	-19,3	-18,7	-19,1	-19,1
4-PSK	-20,3	-20,6	-20,3	-18,9	-19,2	-18,9	-18,8	-18,6	-18,7
8-PSK	-19,5	-19,4	-19,2	-19,5	-19,2	-19,2	-18,9	-18,7	-19,2
16-PSK	-19,6	-19,7	-19,5	-19,5	-19,5	-19,4	-19,2	-18,7	-18,9
4-QAM	-19,5	-19,7	-19,6	-19,3	-19,4	-19,4	-18,7	-19,0	-18,8
16-QAM	-19,6	-19,7	-19,5	-19,2	-19,3	-19,4	-18,9	-19,1	-19,0
32-QAM	-19,4	-19,7	-19,7	-19,6	-19,3	-19,5	-19,2	-19,3	-18,9
64-QAM	-19,6	-19,4	-19,4	-19,5	-19,1	-19,1	-18,5	-18,8	-19,0

Da mesma forma, influência da pré-distorção IIR no NMSE pode ser vista na Tabela 6.

Tabela 6 – NMSE com pré-distorção IIR

RO	**0,0**			**0,2**			**0,4**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	-23,9	-23,6	-23,6	-23,4	-23,4	-23,3	-23,2	-23,9	-23,2
4-PSK	-23,6	-23,7	-23,7	-23,1	-23,1	-23,8	-23,4	-23,4	-23,5
8-PSK	-23,6	-23,6	-23,9	-24,0	-23,7	-22,9	-22,5	-22,7	-23,1
16-PSK	-24,0	-24,2	-23,4	-23,8	-23,4	-23,1	-23,4	-22,9	-23,8
4-QAM	-23,7	-23,3	-23,6	-23,5	-23,5	-23,0	-23,2	-23,1	-23,5
16-QAM	-23,3	-24,0	-23,2	-23,0	-23,6	-23,2	-23,5	-23,4	-22,9
32-QAM	-23,2	-24,0	-23,9	-23,3	-23,7	-23,3	-23,5	-23,1	-22,8
64-QAM	-23,1	-23,5	-23,8	-23,1	-22,9	-23,6	-23,5	-23,2	-22,8
RO	**0,6**			**0,8**			**1,0**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	-23,6	-23,7	-23,0	-23,3	-22,8	-22,8	-22,4	-22,3	-22,3
4-PSK	-23,1	-22,7	-23,0	-22,5	-22,8	-22,9	-21,9	-21,9	-22,4
8-PSK	-23,0	-23,0	-23,0	-22,4	-22,2	-23,1	-22,6	-21,8	-22,6
16-PSK	-23,1	-22,7	-22,7	-22,6	-23,3	-23,1	-22,3	-22,5	-22,3
4-QAM	-23,0	-23,3	-22,9	-22,5	-22,4	-22,5	-22,1	-22,2	-22,0
16-QAM	-22,6	-23,4	-23,2	-22,5	-22,9	-23,1	-22,6	-22,4	-22,8
32-QAM	-23,3	-22,6	-23,3	-23,0	-23,0	-23,0	-22,6	-22,3	-22,6
64-QAM	-23,1	-22,5	-22,4	-22,5	-22,8	-22,2	-22,4	-22,4	-22,6

A pré-distorção FIR melhorou os resultados de NMSE, mas não superou o desempenho mostrado pela pré-distorção IIR. A Figura 68 mostra a média das NMSEs com e sem pré-distorção para cada constelação.

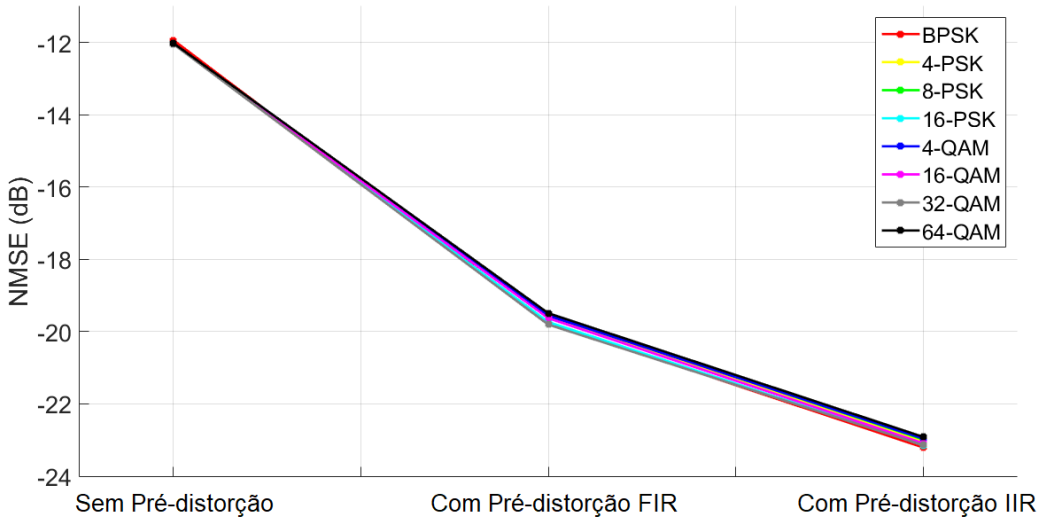

Figura 68 – NMSE com e sem pré-distorção

5.3.2 RESULTADOS POR ERRO MÉDIO ABSOLUTO

Assim como na seção 4.2, para o cálculo do MAE foi utilizado o setup descrito na Figura 43. Como se trata de uma medida absoluta, a normalização utilizada nas três tabelas apresentadas tem que ser a mesma. Essa técnica normaliza o bit de maior amplitude com energia unitária. O cálculo do MAE foi realizado utilizando os dados provenientes do SignalTap analisados em MATLAB®. Esses resultados, para um sistema sem pré-distorção, podem ser vistos na Tabela 7.

Tabela 7 – MAE sem pré-distorção

RO	**0,0**			**0,2**			**0,4**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	0,21	0,21	0,20	0,18	0,18	0,18	0,17	0,17	0,17
4-PSK	0,22	0,22	0,22	0,20	0,20	0,20	0,19	0,19	0,19
8-PSK	0,22	0,21	0,22	0,20	0,20	0,20	0,20	0,20	0,19
16-PSK	0,22	0,22	0,22	0,21	0,21	0,21	0,20	0,20	0,20
4-QAM	0,21	0,27	0,32	0,28	0,29	0,29	0,26	0,26	0,27
16-QAM	0,23	0,23	0,23	0,22	0,21	0,21	0,21	0,21	0,21
32-QAM	0,19	0,19	0,19	0,18	0,18	0,18	0,17	0,17	0,18
64-QAM	0,20	0,20	0,20	0,19	0,19	0,19	0,18	0,18	0,18
RO	**0,6**			**0,8**			**1,0**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	0,19	0,17	0,17	0,18	0,18	0,18	0,20	0,20	0,20
4-PSK	0,19	0,19	0,19	0,19	0,19	0,19	0,20	0,20	0,21
8-PSK	0,20	0,20	0,20	0,20	0,20	0,20	0,21	0,21	0,21
16-PSK	0,20	0,20	0,20	0,20	0,20	0,20	0,23	0,22	0,22
4-QAM	0,27	0,29	0,28	0,28	0,28	0,28	0,31	0,31	0,31
16-QAM	0,21	0,21	0,21	0,21	0,21	0,21	0,22	0,23	0,22
32-QAM	0,17	0,17	0,17	0,18	0,18	0,18	0,21	0,21	0,23
64-QAM	0,18	0,18	0,18	0,19	0,19	0,18	0,19	0,20	0,20

Da mesma maneira, a influência da pré-distorção FIR no MAE pode ser vista na Tabela 8.

Tabela 8 – MAE com pré-distorção FIR

RO	**0,0**			**0,2**			**0,4**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	0,08	0,07	0,07	0,07	0,07	0,07	0,06	0,07	0,07
4-PSK	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08
8-PSK	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08
16-PSK	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08
4-QAM	0,08	0,09	0,11	0,11	0,11	0,11	0,11	0,11	0,11
16-QAM	0,09	0,08	0,09	0,09	0,09	0,08	0,08	0,09	0,08
32-QAM	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07
64-QAM	0,08	0,07	0,07	0,08	0,07	0,08	0,08	0,07	0,08
RO	**0,6**			**0,8**			**1,0**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	0,06	0,07	0,07	0,07	0,07	0,07	0,08	0,07	0,07
4-PSK	0,05	0,06	0,05	0,08	0,08	0,08	0,08	0,08	0,08
8-PSK	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08
16-PSK	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,09	0,08
4-QAM	0,08	0,08	0,08	0,11	0,11	0,11	0,12	0,11	0,12
16-QAM	0,09	0,08	0,09	0,09	0,09	0,08	0,09	0,08	0,08
32-QAM	0,07	0,07	0,07	0,07	0,07	0,07	0,09	0,09	0,09
64-QAM	0,07	0,08	0,08	0,07	0,08	0,08	0,08	0,08	0,07

A influência da pré-distorção IIR no MAE pode ser vista na Tabela 9.

Tabela 9 – MAE com pré-distorção IIR

RO	**0,0**			**0,2**			**0,4**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	0,04	0,05	0,04	0,04	0,04	0,04	0,04	0,04	0,04
4-PSK	0,05	0,05	0,05	0,05	0,05	0,05	0,05	0,05	0,05
8-PSK	0,05	0,05	0,05	0,05	0,05	0,05	0,05	0,06	0,05
16-PSK	0,05	0,05	0,05	0,05	0,05	0,05	0,05	0,05	0,05
4-QAM	0,06	0,07	0,08	0,07	0,07	0,07	0,07	0,07	0,07
16-QAM	0,06	0,05	0,06	0,06	0,06	0,06	0,05	0,05	0,06
32-QAM	0,05	0,04	0,05	0,05	0,04	0,05	0,04	0,05	0,05
64-QAM	0,05	0,05	0,05	0,05	0,05	0,05	0,05	0,05	0,05
RO	**0,6**			**0,8**			**1,0**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	0,04	0,04	0,04	0,04	0,05	0,04	0,05	0,05	0,05
4-PSK	0,05	0,06	0,05	0,05	0,05	0,05	0,06	0,06	0,05
8-PSK	0,05	0,05	0,05	0,06	0,06	0,06	0,05	0,06	0,06
16-PSK	0,05	0,05	0,05	0,05	0,05	0,05	0,05	0,06	0,05
4-QAM	0,08	0,07	0,08	0,07	0,07	0,07	0,08	0,08	0,08
16-QAM	0,06	0,05	0,06	0,06	0,05	0,05	0,06	0,06	0,05
32-QAM	0,05	0,05	0,04	0,05	0,05	0,05	0,05	0,05	0,05
64-QAM	0,05	0,05	0,05	0,05	0,05	0,05	0,05	0,05	0,05

A pré-distorção FIR melhorou os resultados de MAE, mas, assim como no NMSE, não superou o desempenho mostrado pela pré-distorção IIR. A Figura 69 mostra a média das MAEs com e sem pré-distorção para cada constelação.

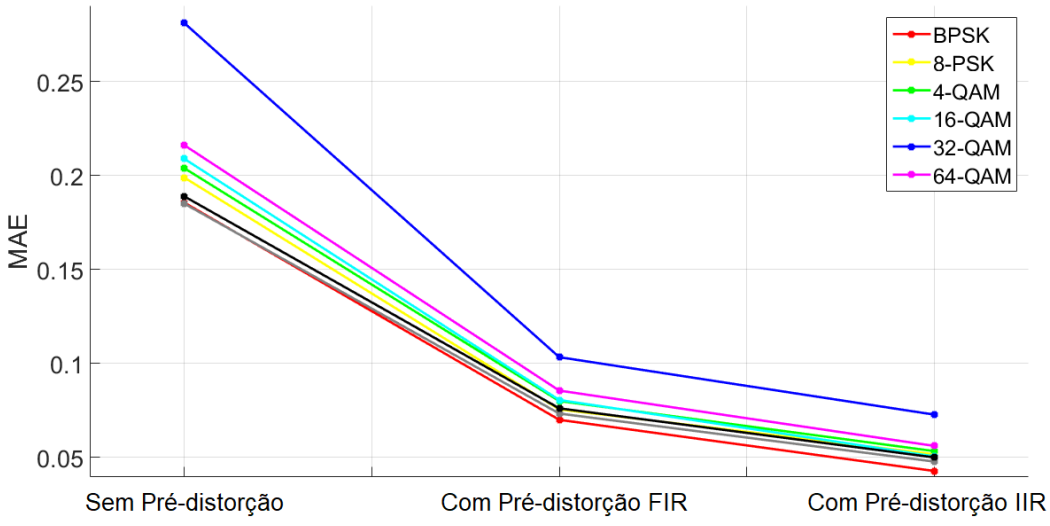

Figura 69 – MAE com e sem pré-distorção

5.3.3 RESULTADOS POR TAXA DE ERRO DE BIT

O cálculo da Taxa de Erro de Bit (BER) é feito utilizando o MATLAB®. Como são capturados 32000 pontos, 8000 símbolos são recebidos. O número de bits analisados varia de acordo com a constelação: BPSK (8000 bits), 4-PSK (16000 bits), 8-PSK (24000 bits), 16-PSK (32000 bits), 4-QAM (16000 bits), 16-QAM (32000 bits), 32-QAM (40000 bits) e 64-QAM (48000 bits).

Sem pré-distorção, os resultados encontrados podem ser vistos na Tabela 10.

Tabela 10 – BER sem pré-distorção

RO	**0,0**			**0,2**			**0,4**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	0	0	0	0	0	0	0	0	0
4-PSK	0	0	0	0	0	0	0	0	0
8-PSK	0,02	0,02	0,02	0,02	0,02	0,02	0,02	0,02	0,01
16-PSK	0,13	0,13	0,13	0,13	0,13	0,13	0,13	0,13	0,13
4-QAM	0	0	0	0	0	0	0	0	0
16-QAM	0,04	0,04	0,04	0,04	0,04	0,04	0,04	0,04	0,04
32-QAM	0,10	0,11	0,10	0,11	0,10	0,10	0,10	0,10	0,10
64-QAM	0,14	0,14	0,14	0,14	0,14	0,14	0,14	0,14	0,14
RO	**0,6**			**0,8**			**1,0**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	0	0	0	0	0	0	0	0	0
4-PSK	0	0	0	0	0	0	0	0	0
8-PSK	0,02	0,02	0,02	0,02	0,02	0,02	0,02	0,02	0,02
16-PSK	0,14	0,14	0,14	0,14	0,14	0,14	0,14	0,14	0,14
4-QAM	0	0	0	0	0	0	0	0	0
16-QAM	0,05	0,05	0,05	0,05	0,05	0,05	0,06	0,06	0,06
32-QAM	0,11	0,11	0,11	0,12	0,12	0,12	0,12	0,12	0,12
64-QAM	0,14	0,14	0,14	0,15	0,15	0,15	0,15	0,15	0,15

Para transmissão de 2 bits por símbolo (4-QAM e 4-PSK) ou para transmissão de 1 bit por símbolo (BPSK), não houveram erros de bit. Isso se deve à distância que um símbolo apresenta para o outro nesses casos. A alteração provocada pelo filtro não foi suficiente para que a informação binária transmitida se alterasse nesses casos. Para os outros casos, a alteração induzida provocou erros proporcionais à distância entre os símbolos, por exemplo, o 16-QAM, apesar de ter o mesmo número de bits por símbolo do 16-PSK, apresentou um melhor resultado devido a maior distância entre os seus símbolos.

Na Tabela 11, é possível ver os resultados para uma pré-distorção FIR.

Tabela 11 – BER após a pré-distorção FIR

RO	**0,0**			**0,2**			**0,4**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	0	0	0	0	0	0	0	0	0
4-PSK	0	0	0	0	0	0	0	0	0
8-PSK	0	0	0	0	0	0	0	0	0
16-PSK	3,2e-3	1,7e-3	3,2e-3	2,8e-3	2,8e-3	3,0e-3	3,4e-3	2,4e-3	2,8e-3
4-QAM	0	0	0	0	0	0	0	0	0
16-QAM	0	0	0	0	0	0	0	0	0
32-QAM	1,5e-4	1,2e-4	1,5e-4	1,2e-4	1,2e-4	2,5e-4	1,2e-4	1,7e-4	1,5e-4
64-QAM	1,0e-2	8,7e-3	8,2e-3	1,0e-2	8,9e-3	1,0e-2	7,5e-3	8,8e-3	7,9e-3
RO	**0,6**			**0,8**			**1,0**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	0	0	0	0	0	0	0	0	0
4-PSK	0	0	0	0	0	0	0	0	0
8-PSK	0	0	0	0	0	0	0	0	0
16-PSK	3,0e-3	2,2e-3	3,0e-3	2,1e-3	1,8e-3	2,9e-3	2,6e-3	3,4e-3	3,5e-3
4-QAM	0	0	0	0	0	0	0	0	0
16-QAM	0	0	0	0	0	0	0	0	0
32-QAM	3,0e-4	2,7e-4	2,0e-4	1,7e-4	1,0e-4	5,0e-5	3,4e-2	3,4e-2	3,5e-2
64-QAM	6,9e-3	8,4e-3	7,3e-3	6,5e-3	9,2e-3	9,1e-3	1,0e-2	8,3e-3	7,4e-3

A Tabela 12 apresenta os resultados após uma pré-distorção IIR.

Tabela 12 – BER após a pré-distorção IIR

RO	**0,0**			**0,2**			**0,4**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	0	0	0	0	0	0	0	0	0
4-PSK	0	0	0	0	0	0	0	0	0
8-PSK	0	0	0	0	0	0	0	0	0
16-PSK	0	6,2e-5	3,1e-5	0	3,1e-5	3,1e-5	3,1e-5	0	3,1e-5
4-QAM	0	0	0	0	0	0	0	0	0
16-QAM	0	0	0	0	0	0	0	0	0
32-QAM	0	0	0	0	0	0	0	0	0
64-QAM	1,6e-4	1,2e-4	6,2e-5	8,3e-5	1,2e-4	1,0e-4	1,0e-4	4,1e-5	1,0e-4
RO	**0,6**			**0,8**			**1,0**		
Teste / Modulação	1	2	3	1	2	3	1	2	3
BPSK	0	0	0	0	0	0	0	0	0
4-PSK	0	0	0	0	0	0	0	0	0
8-PSK	0	0	0	0	0	0	0	0	0
16-PSK	3,1e-5	3,1e-5	3,1e-5	0	3,1e-5	0	0	0	0
4-QAM	0	0	0	0	0	0	0	0	0
16-QAM	0	0	0	0	0	0	0	0	0
32-QAM	0	0	0	0	0	0	0	0	0
64-QAM	6,2e-5	8,3e-5	1,0e-4	6,2e-5	4,1e-5	1,0e-4	1,8e-4	1,8e-4	2,0e-4

A pré-distorção IIR diminuiu o erro de bit para a constelação 64-QAM e 16-PSK, e também reduziu a zero o erro de bit para os outros casos. Para o caso FIR houve erros para o 32-QAM, o 64-QAM e o 16-PSK. A Figura 70 mostra a média das BERs com e sem pré-distorção para cada constelação.

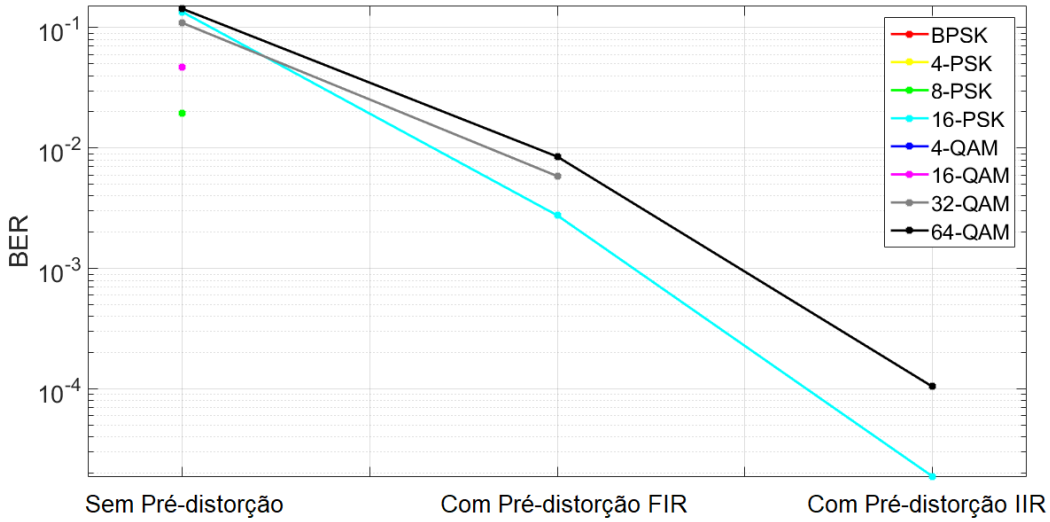

Figura 70 – BER com e sem pré-distorção

5.4 RESUMO DO CAPÍTULO

Os valores mostrados nas tabelas mostram que houve uma significativa melhora se comparado aos dados sem pré-distorção, sendo que a estimação IIR obteve valores de NMSE superiores em 3 dB se comparado com a FIR. É importante ressaltar que os dados foram obtidos através de medições reais que foram analisadas em MATLAB®. O modelo de pré-distorção foi feito considerando o ADC/DAC, os cabos, o transformador e o filtro.

6 CONCLUSÕES

Na dissertação, foi criada uma plataforma em FPGA que gera diferentes modulações digitais a partir de uma sequência PN. Utilizando esse sinal, implementou-se um algoritmo em MATLAB® que identifica um dispositivo pelo qual o sinal é transmitido. Uma terceira parte utiliza-se o mesmo algoritmo em MATLAB® para a criação de um modelo inverso. Os coeficientes deste modelo inverso são programados no FPGA para a implementação de uma pré-distorção a fim de compensar distorções inseridas no sinal por um dispositivo. Para essa dissertação, o dispositivo foi um filtro Butterworth de quinta ordem.

O filtro foi introduzido como uma alteração induzida no canal. Para a sua correta identificação, um sinal foi transmitido através do filtro. A partir da comparação desse sinal com o mesmo transmitido sem o filtro, foi construído um modelo em MATLAB®. A curva estimada pelos coeficientes desse modelo foi comparada com a curva real do filtro medida por um equipamento comercial. Concluiu-se que o algoritmo é apropriado para a identificação de um sistema real. A ordem estimada do filtro calculada pelo algoritmo foi exatamente a projetada. Essa ordem foi obtida por um modelo IIR, que é a característica esperada em um filtro Butterworth.

Como uma terceira aplicação, construiu-se um sistema de pré-distorção implementado no FPGA. Primeiramente, é necessária a obtenção dos coeficientes do modelo Caixa Preta no MATLAB®. Para isso, foi utilizado um método de estimação indireta, onde os dados de entrada e saída são invertidos para a criação de um modelo inverso pelo algoritmo de estimação. Após a obtenção dos coeficientes dos modelos FIR e IIR, estes podem ser implementados no FPGA e os seus respectivos desempenhos podem ser analisados.

O modelo de pré-distorção envolveu o DAC, o transformador, o filtro e o ADC. Considerou-se todos os componentes devido a não-linearidade do transformador, porque resultados insatisfatórios foram obtidos quando a estimação considerava apenas o filtro. Para garantir a qualidade do trabalho, foram utilizadas quatro figuras de mérito: EVM, NMSE, MAE e BER; sendo que a primeira é feita através de um equipamento e as outras três pelos dados capturados no MATLAB®. Em todas as medidas, observou-se uma melhora da qualidade da transmissão ao se aplicar a pré-distorção. Essa melhora foi mais significativa ao se aplicar os coeficientes do modelo IIR, como visto no trabalho.

O trabalho demonstrou que é possível utilizar um algoritmo de identificação para a criação de modelos que descrevem um sistema real e também criar modelos que podem ser utilizados na pré-distorção. Os resultados da identificação foram submetidos à figuras de mérito, além de serem comparados com uma curva real. Na pré-distorção, os resultados foram submetidos a quatro medidas de desempenho, comparando a qualidade antes e após a aplicação dos coeficientes no sinal.

Como trabalhos futuros, espera-se a identificação de dispositivos mais complexos

como amplificadores de potência. Também poderá ser implementado em MATLAB® um algoritmo que estime polinômios não-lineares para identificação. Esses polinômios implementados no FPGA serão comparados com os modelos FIR e IIR implementados nesse trabalho.

REFERÊNCIAS

[1] Altera, *DE2-115 User Manual*, 2010.

[2] L. O. M. Stefano, F. Santana, "O retrato dos novos consumidores brasileiros," *Revista Exame*, 2008.

[3] D. A. Guimaraes, *Digital Transmission – A Simulation Aided Introduction with Vissim/Comm*, 2009, no. 1.

[4] A. P. Lathi, *Digital Signal Processing and Linear Systems*, 2000, vol. 1, no. 1.

[5] L. Aguirre, *Introdução à Identificação de Sistemas*, 2004, vol. 1, no. 2.

[6] J. Wood and D. E. Root, *Fundamentals of Nonlinear Behavioral Modeling of RF and Microwave Design*. Artech House, 2005.

[7] Altera, *Data Conversion HSMC Reference Manual*, 2008.

[8] A. C. Luiz, A. A. M. Medeiros, I. A. Oliveira, A. B. dos Santos, T. V. N. Coelho, and D. D. Silveira, "Modulador digital educacional em FPGA para uso em aulas de laboratório de telecomunicações," in *COBENGE 2017*, 2017.

[9] A. C. Luiz, A. J. S. Marco, A. A. M. Medeiros, A. B. dos Santos, T. V. N. Coelho, and D. D. Silveira, "16-QAM FPGA digital modulator and a tool to evaluate bit error rate," in *2017 SBMO/IEEE MTT-S International Microwave and Optoelectronics Conference (IMOC)*, Aug 2017, pp. 1–5.

[10] L. Ljung, *System Identification*. Prentice Halls, 1999, vol. 2.

[11] Altera, *SignalTap II with Verilog Designs*, 2012.

[12] C. Erdogan, I. Myderrizi, and S. Minaei, "FPGA implementation of BASK-BFSK-BPSK digital modulators [testing ourselves]," *IEEE Antennas and Propagation Magazine*, vol. 54, no. 2, pp. 262–269, April 2012.

[13] T. Kazaz, M. Kulin, and M. Hadzialic, "Design and implementation of SDR based QPSK modulator on FPGA," in *2013 36th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, May 2013, pp. 513–518.

[14] Y. Linn, "An ultra low cost wireless communications laboratory for education and research," *IEEE Transactions on Education*, vol. 55, no. 2, pp. 169–179, May 2012.

[15] Mathworks, "mldivide, " Available in: https://www.mathworks.com/help/matlab/ref/mldivide.htmll, 2017.

[16] A. P. and S. S. Pillai, "Mmse channel estimation with lms equalization for sc fde systems," in *2015 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, Dec 2015, pp. 134–138.

[17] R. W. Stewart, J. J. Soraghan, W. S. Gan, and T. S. Durrani, "Channel equalisation using the non-canonical lms," in *[1991] Conference Record of the Twenty-Fifth Asilomar Conference on Signals, Systems Computers*, Nov 1991, pp. 1160–1164 vol.2.

[18] A. H. I. Makki, A. K. Dey, and M. A. Khan, "Comparative study on lms and cma channel equalization," in *2010 International Conference on Information Society*, June 2010, pp. 487–489.

[19] c. Bowick, *RF Circuit Desing*, 2008.

[20] S. Amin, E. Zenteno, P. N. Landin, D. Rönnow, M. Isaksson, and P. Händel, "Noise impact on the identification of digital predistorter parameters in the indirect learning architecture," in *2012 Swedish Communication Technologies Workshop (Swe-CTW)*, Oct 2012, pp. 36–39.

[21] M. E. Gadringer, D. Silveira, and G. Magerl, "Efficient power amplifier identification using modified parallel cascade hammerstein models," in *2007 IEEE Radio and Wireless Symposium*, Jan 2007, pp. 305–308.

[22] Z. Weiliang, P. Changyong, G. Xingbo, and Y. Zhixing, "Design and FPGA implementation of high-speed square-root-raised-cosine FIR filters," in *Proceedings of 2002 IEEE 10th Digital Signal Processing Workshop, 2002 and the 2nd Signal Processing Education Workshop.*, Oct 2002, pp. 232–235.

[23] P. K. Meher, S. Chandrasekaran, and A. Amira, "FPGA realization of FIR filters by efficient and flexible systolization using distributed arithmetic," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3009–3017, July 2008.

[24] S. M. R. Islam, R. Sarker, S. Saha, and A. F. M. N. Uddin, "Design of a programmable digital IIR filter based on FPGA," in *2012 International Conference on Informatics, Electronics Vision (ICIEV)*, May 2012, pp. 716–721.

[25] S. S. Pujari, P. P. Muduli, A. Panda, R. Badhai, S. Nayak, and Y. Sahoo, "Design amp; implementation of fir filters using on-board adc-dac amp; fpga," in *International Conference on Information Communication and Embedded Systems (ICICES2014)*, Feb 2014, pp. 1–6.

[26] A. Hagenblad and L. Ljung, "Maximum likelihood identification of Wiener models with a linear regression initialization," in *Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No.98CH36171)*, vol. 1, 1998, pp. 712–713 vol.1.

[27] M. Isaksson, D. Wisell, and D. Ronnow, "A comparative analysis of behavioral models for RF power amplifiers," *IEEE Transactions on Microwave Theory and Techniques*, vol. 54, no. 1, pp. 348–359, Jan 2006.

[28] J. H. Lin, T. M. Sellke, and E. J. Coyle, "Adaptive stack filtering under the mean absolute error criterion," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 6, pp. 938–954, Jun 1990.

[29] N. Sasaki, H. Shimada, S. Shimada, and H. Kobayashi, "Evaluation of transmission quality of visible light communication using bit error rate measurement," in *2016 16th International Conference on Control, Automation and Systems (ICCAS)*, Oct 2016, pp. 1362–1365.

[30] *Keysight N9000A CXA Signal Analyzer*, 2014.

[31] A. V. Oppenheim and A. S. Willsky, *Sinais e Sistemas*, 2005, vol. 1, no. 2.

[32] B. Kovačević, Z. Banjac, and M. Milosavljević, *Adaptive Digital Filters*. Springer, 2013.

[33] L. Philipose, *Using the ADS8380 With the TMS320C6713 DSP*, 2005.

[34] T. Instruments, *Tiva™ TM4C123GH6PM Microcontroller*, 2013.

[35] Intel, *FPGAs For Dummies*, 2017.

[36] S. T. Karris, *Digital Circuit Analysis and Design with Simulink Modeling and Introduction to CPLDs and FPGAs*, 2007, vol. 2.

[37] Altera, "Quartus prime lite edition," Available in: http://dl.altera.com/?edition=lite, 2017.

[38] R. Tocci, N. S. Widmer, and G. L. Mo., *Sistemas Digitais: Princípios e Aplicações*, 2011.

[39] M. D. McKinley, "EVM calculation for broadband modulated signals," *64th ARFTG*, pp. 45–52, 2004.

[40] S. Malviya and P. Kumari, "Implementation of pseudo-noise sequence generator on FPGA using Verilog," *International Journal of Electronic and Electrical Engineering.*, vol. 7, 2014.

[41] R. Tocci and N. Widmer, *SISTEMAS DIGITAIS Princípios e Amplicações*, 2000, vol. 7, no. 1.

[42] A. A. Hasan and I. D. Marsland, "Low complexity LLR metrics for polar coded QAM," in *2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)*, April 2017, pp. 1–4.

[43] Altera, *Megafunction Overview User Guide*, 2009.

[44] Associação Brasileira de Normas Técnicas, "Televisão digital terrestre - sistema de transmissão," https://www.abntcolecao.com.br/normavw.aspx?ID=1606g, 2007.

[45] Dreamcatcher, *SME1100 Digital RF Communications Courseware*, 2017.

[46] O. Nells, *Nonlinear System Identification*, 2001, vol. 1, no. 1.

[47] D. T. Westwick and R. E. Kearney, *Identification of Nonlinear Physiological Systems*. IEEE PRESS, 2003.

[48] R. Yates and D. Goodman, *Probability and Stochastic Processes*, 2005, vol. 1, no. 2.

[49] L. Hamilton, *Um Curso de Cálculo*, 2001, vol. 1, no. 5.

[50] J. G. Proakis, *Digital Communications*. McGraw Hill, 2002.

[51] Mathworks, "Cross-correlation - matlab xcorr," Available in: https://www.mathworks.com/help/signal/ref/xcorr.htmll, 2017.

[52] M. Schetzen, *The Volterra and Wiener Theories of Nonlinear Systems*. KriegerPublishing Company, 1980.

[53] V. Z. Marmarelis, *Nonlinear Dynamic Modeling of Physiological Systems*. J. Wiley Sons, 2004.

[54] S. K. Mitra, *Digital Signal Processing – A Computer-Based Approach*, 2005, vol. 3, no. 1.

APÊNDICE A – CASAMENTO DE IMPEDÂNCIA

Um circuito com impedância descasada é aquele onde a impedância da fonte difere da impedância de carga [19]. A Figura 71 ilustra como essas impedâncias são representadas.

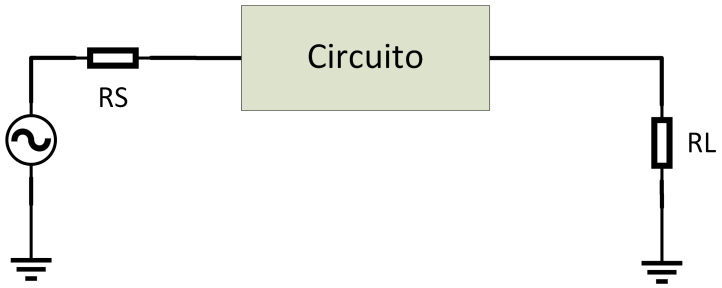

Figura 71 – Circuito com impedância de carga (RL) e impedância de fonte (RS).

Caso o circuito não possua retorno para terra, a medição não será afetada pelo descasamento. Mas quando se adiciona um componente como um filtro, essa diferença entre RS e RL faz com que o circuito não funcione corretamente. Para resolver esse problema, adiciona-se um transformador com uma relação de transformação que compense essa diferença. A Figura 72 ilustra onde o transformador pode ser colocado no circuito.

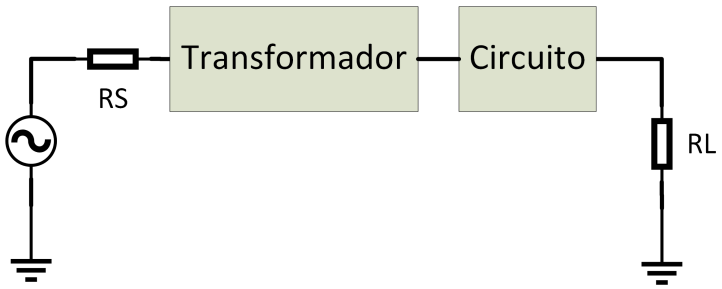

Figura 72 – Circuito com transformador.

Em um circuito onde a impedância da fonte é 4 vezes menor do que a da carga é utilizado um transformador 1:4 para se corrigir essa diferença.

APÊNDICE B – FILTRO PROTÓTIPO

Um filtro Butterworth passa-baixas pode ser projetado a partir de um filtro protótipo [19]. Esse filtro possui a estrutura mostrada na figura 73.

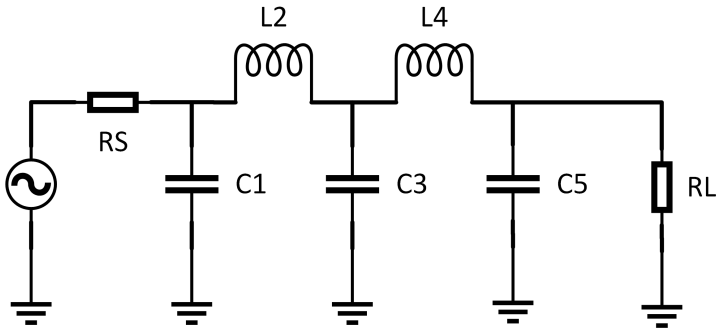

Figura 73 – Filtro Butterworth de quinta ordem.

Para um filtro com corte em 1 Hz os valores dos componentes são mostrados na Tabela 13.

Tabela 13 – Valores para um corte em 1 Hz

n	RS/RL	C1	L2	C3	L4	C5
5	0.900	0.442	1.027	1.910	1.756	1.389
	0.800	0.470	0.866	2.061	1.544	1.738
	0.700	0.517	0.731	2.285	1.333	2.108
	0.600	0.588	0.609	2.600	1.126	2.552
	0.500	0.686	0.496	3.051	0.924	3.133
	0.400	0.838	0.388	3.736	0.727	3.965
	0.300	1.094	0.285	4.884	0.537	5.307
	0.200	1.608	0.186	7.185	0.352	7.935
	0.100	3.512	0.091	14.095	0.173	15.710

A partir dos valores da Tabela 13, pode-se construir um filtro Butterworth de quinta ordem, utilizando as equações (B.1) e (B.2) para diferentes frequências de corte f_c.

$$C = \frac{C_n}{2\pi f_c R_L} \tag{B.1}$$

$$L = \frac{R_L L_n}{2\pi f_c} \tag{B.2}$$