Serviço Público Federal
Ministério da Educação
Fundação Universidade Federal de Mato Grosso do Sul

Curso de ENGENHARIA FÍSICA

Trabalho de Conclusão de Curso

Geração e Controle de Pulsos de Radiofrequência para um Protótipo de Imagem por Ressonância Magnética

Jhoenne Helena Vasconcelos Pulcherio

Orientador: Prof. Dr. Renan Albuquerque Marks

Coorientador: Pedro Henrique Andrade Trindade

Trabalho de Conclusão de Curso apresentado ao curso de Engenharia Física do Instituto de Física (INFI), da Universidade Federal de Mato Grosso do Sul (UFMS).

Campo Grande – MS

Dezembro/2025

AGRADECIMENTOS

Agradeço à minha família, por todo amor, apoio e incentivo nesta etapa da minha vida. Em especial à minha mãe, Juliene Cristina de Vasconcelos, que sempre me acolheu, me motivou nos momentos difíceis e nunca mediu esforços para que eu alcançasse meus objetivos. Aos meus avós, Aparecida Helena Pavesi Vasconcelos e Pedro Saturnino de Vasconcelos, que foram um grande suporte e me ajudaram a chegar até aqui. E ao meu irmão, João Pedro Vasconcelos Pulcherio, pela presença e companheirismo em todo o processo.

Ao meu orientador, Prof. Dr. Renan Albuquerque Marks, pela disponibilidade e orientação ao longo deste trabalho. Ao meu coorientador, Pedro Henrique Andrade Trindade, pelos ensinamentos, paciência e conselhos. Agradeço a todos os meus colegas do Centro Nacional de Pesquisa em Energia e Materiais (CNPEM), em especial aos grupos de Eletrônica e Sistemas Embarcados e de Arquitetura e Modelagem de Sistemas, pelos aprendizados, apoio e pela convivência no dia a dia. Às minhas antigas supervisoras, Marisa Fonseca e Maria Eduarda Tempesta, pelo acolhimento e incentivo no primeiro ano de estágio. Às minhas amigas Aline, Letícia e Paula, pelas conversas, risadas e apoio em todos os momentos.

Aos meus amigos da Universidade Federal de Mato Grosso do Sul (UFMS), em especial ao Guilherme Costa, pelo apoio, incentivo e amizade que ajudaram a tornar essa jornada mais leve. Também aos amigos Felipe, Pedro e Rayane, pelo companheirismo ao longo de toda a graduação. Sem vocês, esse processo teria sido muito mais difícil. Ao Grupo de Óptica e Fotônica (GOF), onde fui bolsista durante grande parte da graduação, agradeço pelos aprendizados, pelas conversas e pelos cafés da tarde.

À República Danoninho, que se tornou um lar nos últimos dois anos, em especial à Emanuele Dória e ao Breno Porto, pela compreensão, apoio e por todos os momentos de conversa e descontração.

Ao CNPEM, pela oportunidade de estágio e pela bolsa concedida. Agradeço também ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e ao Programa de Auxílio Permanência da UFMS, pelo apoio financeiro que possibilitou cursar a graduação em tempo integral.

Por fim, agradeço a todos que, de alguma forma, contribuíram para a realização deste trabalho e para a minha formação pessoal e profissional. Sem vocês, eu não teria chegado até aqui.

RESUMO

A Ressonância Magnética (RM) é uma técnica de diagnóstico por imagem não invasiva e de alta resolução, amplamente utilizada na medicina por permitir a diferenciação de tecidos sem o uso de radiação ionizante. No entanto, há uma desigualdade relevante entre o acesso a esses equipamentos em função da região no Brasil, que apresenta áreas extensas com oferta baixa ou inexistente desses exames. O desenvolvimento de equipamentos nacionais de RM representa uma alternativa para reduzir custos e ampliar o acesso, promovendo autonomia tecnológica. Considerando esse contexto, o CNPEM está desenvolvendo competências nesse campo e, atualmente conta com um protótipo de RM de 0,4 T desenvolvido parcialmente no centro, com exceção de amplificadores de gradiente, amplificadores de potência e console. O presente trabalho marca o início do desenvolvimento de um console, com foco no sistema de transmissão de sinais de Radiofrequência (RF). O objetivo foi desenvolver e validar um gerador digital de pulsos de RF, implementado em Field-Programmable Gate Array (FPGA), utilizando a placa de desenvolvimento Red Pitaya e a linguagem de descrição de hardware VHSIC Hardware Description Language (VHDL). A metodologia empregou a técnica de Síntese Digital Direta (Direct Digital Synthesis (DDS)) para a geração de portadoras senoidais e cossenoidais por meio de um Oscilador Controlado Numericamente (Numerically Controlled Oscillator (NCO)), em conjunto com a modulação em quadratura (IQ). O projeto digital foi desenvolvido em VHDL e validado através de *testbenches* em ambiente de simulação no software Vivado. Os resultados das simulações mostraram o correto funcionamento da integração dos módulos e a geração do pulso com a frequência de acordo com o parâmetro de controle. Esse comportamento validou a geração de sinais com controle de frequência a partir da técnica de DDS e a modulação do pulso. Os testes em bancada mostraram um pulso modulado no osciloscópio compatível com as simulações realizadas, confirmando a correta implementação em FPGA para a geração dos pulsos de RF, com valores experimentais próximos aos simulados. O trabalho contribuiu com a capacitação da equipe, servindo como uma etapa inicial no processo de desenvolvimento do console conceitual. Os resultados obtidos nesse processo podem ser otimizados e aprimorados de forma que, no futuro, contribuam para o objetivo do CNPEM de desenvolver um equipamento de RM integralmente nacional.

Palavras-Chave: Ressonância Magnética, Radiofrequência, FPGA, DDS, VHDL, Red Pitaya.

ABSTRACT

Magnetic Resonance Imaging (MRI) is a non-invasive, high-resolution imaging technique widely used in medicine for its ability to differentiate soft tissues without the use of ionizing radiation. However, there is a significant inequality in access to such equipment across Brazil, with large regions showing limited or no availability of MRI exams. The development of national MRI equipment represents an alternative to reduce costs and expand access, promoting technological autonomy. In this context, the CNPEM has been developing expertise in this field and currently has a 0.4 T MRI prototype partially developed at the center, except for the gradient amplifiers, power amplifiers, and console. This work marks the beginning of the console development, focusing on the RF transmission system. The objective was to design and validate a digital RF pulse generator implemented on an FPGA, using the Red Pitaya development board and the VHDL hardware description language. The methodology employed the DDS technique for generating sine and cosine carriers through a NCO, combined with quadrature (IQ) modulation. The digital design was developed in VHDL and validated through *testbenches* in a simulation environment using Vivado software. Simulation results demonstrated the correct integration of the modules and the generation of the pulse with frequency matching the control parameter. This behavior validated the frequency-controlled signal generation using the DDS technique and pulse modulation. Bench tests showed a modulated pulse on the oscilloscope consistent with the simulations, confirming the correct FPGA implementation for RF pulse generation, with experimental values close to those obtained in simulation. This work contributed to the team's technical training and served as an initial step in the development of the conceptual console. The results obtained in this process can be further optimized and improved to support CNPEM's goal of developing a fully national MRI system in the future.

Keywords: Magnetic Resonance Imaging, Radiofrequency, FPGA, DDS, VHDL, Red Pitaya.

Lista de Figuras

Lista de Tabelas

Siglas

ADC Analog-to-Digital Converter.

CNPEM Centro Nacional de Pesquisa em Energia e Materiais.

DAC Digital-to-Analog Converter.

DDS Direct Digital Synthesis.

DFT Discrete Fourier Transform.

DUT Device Under Test.

FFT Fast Fourier Transform.

FID Free Induction Decay.

FPGA Field-Programmable Gate Array.

GPA Gradient Power Amplifier.

IRM Imagem por Ressonância Magnética.

MRI Magnetic Resonance Imaging.

NCO Numerically Controlled Oscillator.

RF Radiofrequência.

RM Ressonância Magnética.

RMN Ressonância Magnética Nuclear.

SLR Shinnar-Le Roux.

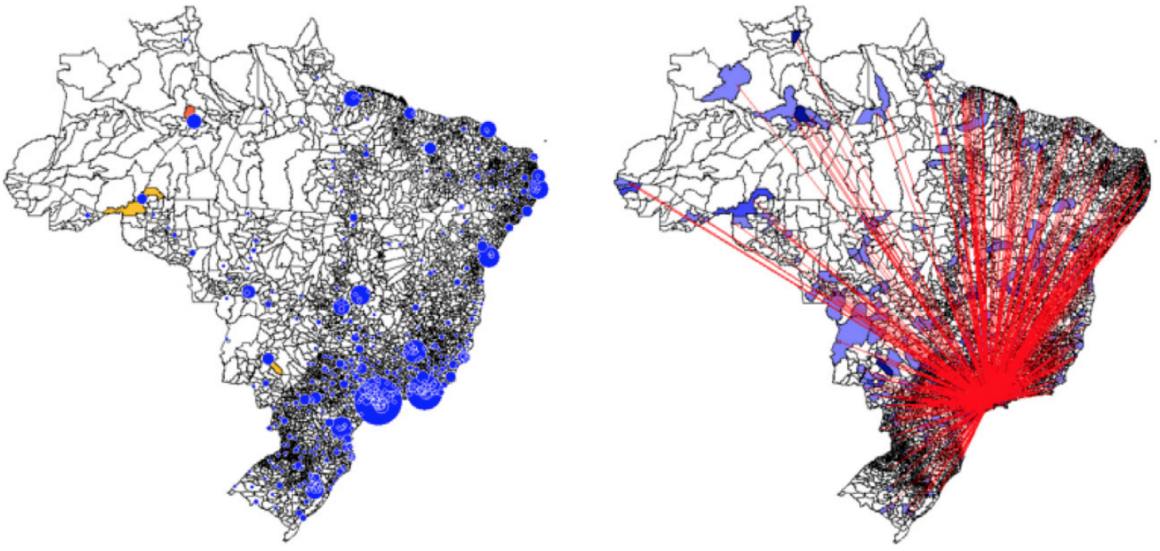
VHDL VHSIC Hardware Description Language.

Serviço Público Federal
Ministério da Educação
Fundação Universidade Federal de Mato Grosso do Sul

Sumário

1 INTRODUÇÃO

A RM é uma tecnologia capaz de fornecer diagnósticos por imagem com alta resolução espacial e capacidade de diferenciação de tecidos, podendo ser aplicada por toda a extensão do corpo humano [1]. Utilizando um campo magnético intenso, pulsos de RF e um sistema computacional, a Imagem por Ressonância Magnética (IRM) (em inglês, Magnetic Resonance Imaging, MRI) produz imagens detalhadas de órgãos internos, possibilitando diagnósticos precisos para uma ampla variedade de condições médicas [2]. Além disso, a técnica se destaca por não utilizar radiação ionizante, sendo considerada não invasiva e permitindo exames repetidos de forma segura [3]. No entanto, a disseminação e a disponibilidade equitativa dessa tecnologia representam um desafio significativo, especialmente em países em desenvolvimento como o Brasil. O cenário nacional revela uma distribuição de equipamentos de RM desigual. Dados de 2017 mostram que apenas 462 dos 5569 municípios brasileiros possuíam esse tipo de equipamento, deixando 5107 municípios sem acesso ao serviço [2]. Essa carência generalizada e a má distribuição geográfica impactam diretamente na saúde da população, forçando milhares de pacientes a percorrerem longas distâncias para ter acesso ao exame. Em 2017, mais de 39% dos exames foram realizados fora do município de residência do paciente, conforme ilustrado na Figura 1 (a); já a Figura 1 (b) ilustra a gravidade dessa desigualdade ao destacar o fluxo de pacientes em direção ao estado de São Paulo, que concentrou a realização de exames para diferentes regiões do país [2].



(a) Distribution of MRIs in Brazil. (b) Flow for MRI scan in São Paulo.

Figura 1: Distribuição de equipamentos de MRI no Brasil, destacando: (a) desigualdade de acessibilidade dos exames no país e (b) fluxo para exames de MRI em São Paulo [2].

Essa disparidade evidencia uma situação crítica de acessibilidade e iniquidade no

Sistema Único de Saúde (SUS), contribui para o baixo aproveitamento da capacidade instalada nacionalmente, que em 2017 correspondia a apenas 21,45% do total disponível; além disso, o setor privado apresenta maior aquisição de equipamentos e crescimento no número de exames, superando as metas recomendadas pelo Ministério da Saúde, enquanto o sistema público não as atinge, aprofundando a desigualdade no acesso[4].

Nesse contexto, o desenvolvimento de tecnologia nacional na área de RM surge como uma alternativa para reduzir custos e ampliar o acesso a esse tipo de exame. Em 2024, o CNPEM desenvolveu parcialmente um protótipo de MRI de 0,4 T, demonstrando capacidade em produzir equipamentos nesse campo. Entretanto, devido ao curto período do projeto, de apenas um ano, parte do equipamento foi desenvolvido no Centro, com exceção dos sistemas de eletrônica e controle, que foram adquiridos de fornecedores externos. Com o objetivo de desenvolver um equipamento integralmente nacional, torna-se necessário o desenvolvimento do console, responsável pelo controle do sistema de RM. Dentro desse contexto, o presente trabalho tem como foco uma etapa do processo de transmissão do console, a geração de pulsos de RF. Além de ser uma etapa inicial de prova de conceito para o desenvolvimento do equipamento, a experiência adquirida nesse processo é importante para a capacitação da equipe e o aprofundamento do conhecimento técnico. Este projeto serve como uma base de aprendizado, de modo que, no futuro, os desenvolvimentos subsequentes possam ser aprimorados e otimizados, contribuindo para a autonomia tecnológica do país na área de saúde.

2 OBJETIVOS

Este capítulo abordará na Seção 2.1, o objetivo geral e, na Seção 2.2 os objetivos específicos do presente trabalho.

2.1 Objetivo Geral

Desenvolver e validar um gerador digital de pulsos de RF implementado em FPGA, utilizando a placa de desenvolvimento Red Pitaya e a linguagem de descrição de hardware VHDL para um protótipo de RM, contribuindo para o desenvolvimento de um equipamento com tecnologia nacional.

2.2 Objetivos Específicos

- Estudar os princípios físicos e eletrônicos envolvidos na geração de pulsos de RF utilizados em sistemas de RM.
- Analisar os principais tipos de pulsos de RF para equipamentos de RM e técnicas de janelamento (*windowing*).
- Implementar a técnica de Síntese Digital Direta (em inglês, Direct Digital Synthesis DDS) em FPGA, desenvolvendo módulos dedicados à geração de pulsos, à geração de portadoras e à modulação em quadratura, visando a construção de um gerador digital de sinais de RF.

- Validar o funcionamento dos módulos digitais por meio de simulações e *testbenches* desenvolvidos na ferramenta Vivado, comparando os resultados com referências obtidas em ambiente Python.
- Integrar os módulos desenvolvidos e realizar a verificação experimental do sinal transmitido em um osciloscópio, avaliando a coerência entre os resultados simulados e os obtidos na implementação física.

3 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, serão apresentados alguns conceitos teóricos necessários para entender o trabalho desenvolvido. A Seção 3.1 aborda os princípios físicos da técnica de RM.

A Seção 3.2 apresenta o console, equipamento que controla o sistema de RM, discutindo suas funções de sincronização, transmissão e recepção de sinais. Por fim, a Seção 3.3 trata do processamento digital de sinais de RF, abordando a modulação IQ (Seção 3.3.1) e a geração de pulsos de RF utilizando a técnica de DDS (Seção 3.3.2).

3.1 Ressonância Magnética Nuclear (RMN)

Nesta seção, são apresentados os conceitos fundamentais de RMN, detalhando o fenômeno de na Seção 3.1.1 e os componentes de um equipamento de RM na Seção 3.1.2. Em seguida, são descritos os principais tipos de pulsos de RF utilizados em RM (Seção 3.1.3).

3.1.1 Fenômeno de RMN

O fenômeno de RMN consiste na interação de um campo magnético estático intenso com os núcleos atômicos da amostra que possuem momento angular e momento magnético não-nulos. Sob uma perspectiva clássica, núcleos como o do hidrogênio, ao serem submetidos a esse campo, exibem um movimento de precessão, análogo a um pião sob a ação do campo gravitacional. Esse fenômeno ocorre graças a uma propriedade intrínseca de partículas subatômicas conhecida como *spin*. Para que um núcleo possa ser utilizado em RMN, ele deve possuir um *spin* nuclear resultante não-nulo, característica comum em núcleos com número de massa ímpar, como o hidrogênio, principal alvo em IRM devido à sua abundância no corpo e alta sensibilidade à técnica [1, 3].

Quando submetidos a um campo magnético externo, o momento magnético associado ao spin de cada próton se orienta em relação ao campo. Sob o tratamento da mecânica quântica, o próton pode se orientar em duas orientações em relação ao campo externo, paralelo ao B_0 (*spin-up*), que corresponde ao estado de menor energia, ou antiparalelo (*spin-down*), que representa o estado de maior energia. A equação que relaciona a frequência de oscilação (denominada de frequência de Larmor) e a instensidade do campo magnético é descrita na Equação 1 [1, 5, 5].

$$\omega = \gamma B_0 \tag{1}$$

Onde γ é a razão giromagnética que, no caso do hidrogênio é de 42,58 MHz/T, e B_0 é o campo externo ao qual os spins estão submetidos. A soma vetorial dos momentos magnéticos dos spins submetidos a B_0 forma um vetor de magnetização líquida (M), que, no estado de equilíbrio, possui apenas a componente longitudinal, orientada na mesma direção do campo. Para realizar um experimento de RMN, é necessário perturbar o estado de equilíbrio da magnetização, de forma que o vetor M esteja no plano transversal. Essa reorientação do vetor magnetização acontece com a aplicação de um pulso de RF (B_1), perpendicular a B_0 e emitido na frequência de Larmor, fenômeno conhecido como ressonância. Quando esse pulso é aplicado, os spins absorvem energia e o vetor magnetização desloca, formando um ângulo em relação a B_0 [1, 6].

Para que uma corrente elétrica seja induzida na bobina de recepção, posicionada perpendicularmente ao plano transversal, é necessário que o vetor de magnetização, total ou parcialmente, esteja nesse plano e que os spins apresentem coerência de fase. Quando todos os momentos magnéticos individuais são desviados em $90^{\underline{o}}$ para o plano transversal e precessionam na mesma fase, obtém-se o sinal máximo induzido na bobina [1].

Quando a aplicação do pulso de RF é cessada, o sistema retorna ao seu estado de equilíbrio através dos processos de relaxação da magnetização, gerando um sinal com formato de uma onda senoidal amortecida conhecido como FID [1]. O processo está ilustrado na Figura 2.

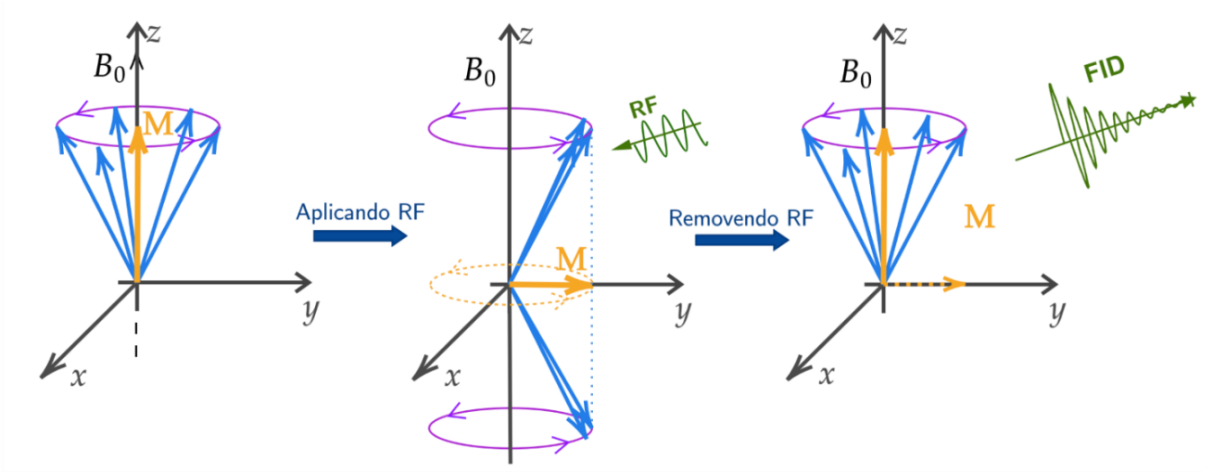


Figura 2: Representação do fenômeno de ressonância magnética envolvendo pulso de RF para gerar a magnetização transveral e a geração do FID durante a relaxação [6].

O sinal de RM (FID) detectado pela bobina de recepção do equipamento é digitalizado e processado para formar a imagem, permitindo a visualização detalhada das estruturas anatômicas e funcionais da área em estudo [6]. A codificação espacial da informação é realizada pela aplicação de gradientes de campo magnético, que alteram sutilmente o campo magnético em diferentes posições. Isso faz com que a frequência de Larmor varie espacialmente, permitindo localizar a origem do sinal. Os dados adquiridos

são organizados em uma matriz de frequências espaciais denominada espaço-k (*k-space*). [6, 7]. Após o preenchimento completo do *k-space*, aplica-se a Transformada de Fourier, que converte os dados do domínio da frequência para o domínio espacial, resultando na imagem final [7].

A relaxação dos spins, responsável pelo FID, ocorre devido às trocas de energia entre os próprios spins e entre os spins e o ambiente ao seu redor (rede). Duas constantes de tempo são utilizadas para caracterizar esses processos: T_1 e T_2. A constante T_1, ou tempo de relaxação longitudinal, representa o tempo de retorno da magnetização longitudinal para seu estado de equilíbrio, enquanto a constante T_2, ou tempo de relaxação transversal, descreve o decaimento da magnetização criada no plano transversal. As diferenças intrínsecas nos tempos de relaxação entre os tecidos biológicos são a principal fonte de contraste nas imagens de RM [1]. Esse contraste pode ser acentuado e manipulado pelo uso de sequências de pulsos, que são combinações precisamente temporizadas de pulsos de RF e gradientes de campo magnético, permitindo gerar imagens ponderadas para realçar características teciduais específicas [5].

3.1.2 Componentes de um equipamento de MRI

Um equipamento de MRI consiste em um sistema tecnológico complexo, constituído por quatro subsistemas principais que funcionam de forma integrada e sincronizada, sendo eles [7]:

- **Magneto:** É o principal componente do sistema de RM, responsável por produzir o campo magnético estático, intenso e homogêneo, que alinha os momentos magnéticos dos prótons de hidrogênio nos tecidos do corpo. Os tipos de magnetos mais comuns em sistemas de RM são os supercondutores, permanentes e resistivos [7].
- **Sistema de RF:** Este sistema é composto por bobinas de RF responsáveis pela transmissão dos pulsos de excitação e pela recepção dos sinais de RM emitidos pelo corpo. Além disso, o sistema inclui amplificadores de RF, que têm a função de amplificar os pulsos gerados no console e os sinais emitidos pela amostra [6].
- **Sistema de Gradiente:** É responsável pela codificação espacial, que permite localizar os sinais de RM para a formação da imagem [7]. Esse sistema é composto por bobinas de gradiente que geram uma variação linear na intensidade do campo magnético estático ao longo dos três eixos espaciais (x, y e z), e amplificadores de gradiente, responsáveis pela amplificação dos pulsos de gradiente gerados no espectrômetro [6, 8].
- **Console:** Atua como o sistema de controle e processamento do equipamento. É responsável por processar as sequências de pulso, gerando as formas de onda para os sistemas de RF e de gradiente de maneira síncrona. Além disso, realiza a aquisição e a digitalização dos sinais de RM e, posteriormente, a reconstrução da imagem a partir dos dados brutos. O console também serve como uma interface de operação entre o usuário e o equipamento [9]. Esse componente e seus detalhes serão aprofundados na Seção 3.2

Os subsistemas listados estão representados na Figura 3:

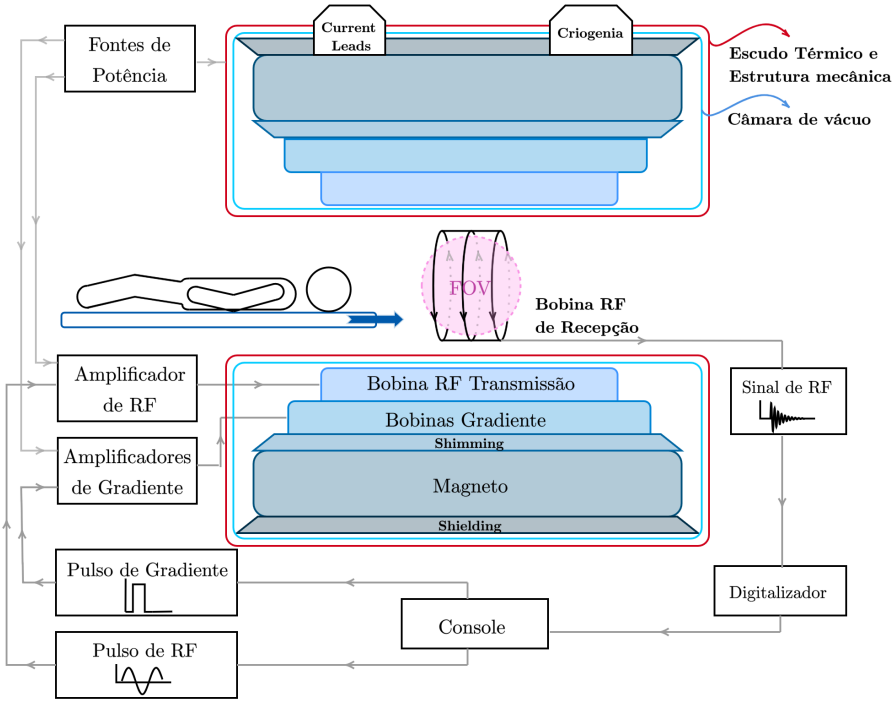


Figura 3: Subsistemas de um equipamento de MRI tradicional com magneto do tipo supercondutor, que requer escudo térmico e câmara de vácuo para operação[6].

3.1.3 Pulsos de Radiofrequência em RM

Como descrito na Seção 3.1.1, em um sistema de RM, os pulsos de RF são fundamentais para excitar os núcleos de hidrogênio, que ao relaxarem, emitem sinais eletromagnéticos que são captados pelas bobinas e utilizados para a formação das imagens. O transmissor gera pulsos de RF com parâmetros controlados, como frequência central, largura de banda, amplitude e fase, que determinam a excitação dos núcleos dentro da região desejada [7, 10].

A geração de um pulso de RF envolve a modulação, isto é, a multiplicação de uma onda portadora senoidal por uma envoltória de RF. A frequência da portadora é ajustada para a *frequência de Larmor*, podendo incluir um pequeno deslocamento (Δf) necessário para a localização da fatia. A envoltória, por outro lado, é uma função que varia mais lentamente no tempo e define a forma do pulso. [8]

A duração do pulso (T) geralmente é medida em milissegundos ou segundos. Outro parâmetro importante é a largura de banda, especificada em hertz ou quilohertz, que está relacionada ao conteúdo em frequência do pulso. Além disso, os pulsos de RF podem ser descritos por seu *flip angle* (θ), que representa o ângulo de desvio produzido pela aplicação do pulso. Por exemplo, um pulso de excitação que transfere completamente a

magnetização longitudinal para o plano transversal tem um *flip angle* de 90° ($\pi/2$ rad). Esse ângulo pode ser calculado por meio da área sob a curva da envoltória do pulso [8].

Estes pulsos são gerados digitalmente, enviados a um DAC, onde são convertidos em sinais analógicos, e posteriormente amplificados para atingir a intensidade necessária para excitar os spins [7], o processo de geração dos sinais digitais será mais aprofundado na Seção 3.3.

Existem várias formas de pulsos de RF, alguns dos formatos mais comumente usados são descritos a seguir:

- **Pulsos Retângulares ("*Hard pulses*"):** possuem a forma de uma função *RECT* no domínio do tempo (Figura 4). Eles são caracterizados por possuirem uma duração temporal curta e amplitude constante. Os *hard pulses* são utilizados quando não há necessidade de seleção espectral ou espacial, uma vez que sua largura de banda é suficientemente ampla para afetar spins com uma grande variedade de frequências de ressonância. Os pulsos retangulares são aplicados sem a presença de um gradiente simultâneo [8].

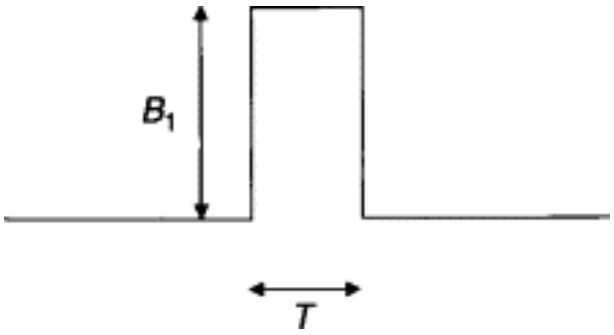


Figura 4: Forma de onda de um pulso retangular (função *RECT*) no domínio do tempo [8].

A banda de frequências de um pulso *hard* é ampla, e sua forma no domínio da frequência é uma função *sinc* (sin(x)/x), que é a Transformada de Fourier de uma função retangular. O *flip angle* de um *hard pulse* é determinado por [8]:

$$\theta = \gamma B_1 T \tag{2}$$

A implementação de pulsos retangulares pode ser desafiadora devido às descontinuidades na sua forma de onda, o que pode comprometer a reprodução com fidelidade por alguns amplificadores de RF. Nesses casos, uma opção é utilizar um pulso retangular com janelamento, que geralmente funciona melhor. Uma outra alternativa é a utilização de um pulso com forma de onda trapezoidal. Para que um pulso *hard* seja eficaz, ele deve ter uma duração suficientemente curta para garantir que sua largura de banda cubra todas as frequências de interesse no volume de imagem. Por exemplo, durações entre 100 e 500 µs resultam em larguras de banda de 2 a 10 kHz, que são ideais para a maioria das aplicações na ausência de gradientes [8].

- **Pulsos *sinc*:** amplamente utilizados para excitação seletiva, saturação e refocalização em RM, sendo uma das formas mais comuns de pulsos de RF por produzir perfis de fatia próximos do ideal [8]. A forma de onda de um pulso *sinc* consiste em vários lóbulos adjacentes de polaridade alternada. O lóbulo central apresenta a maior amplitude e é duas vezes mais largo que os lóbulos laterais, cujas amplitudes decaem progressivamente à medida que se afastam do centro [8]. A Figura 5 ilustra o formato de um pulso *sinc*.

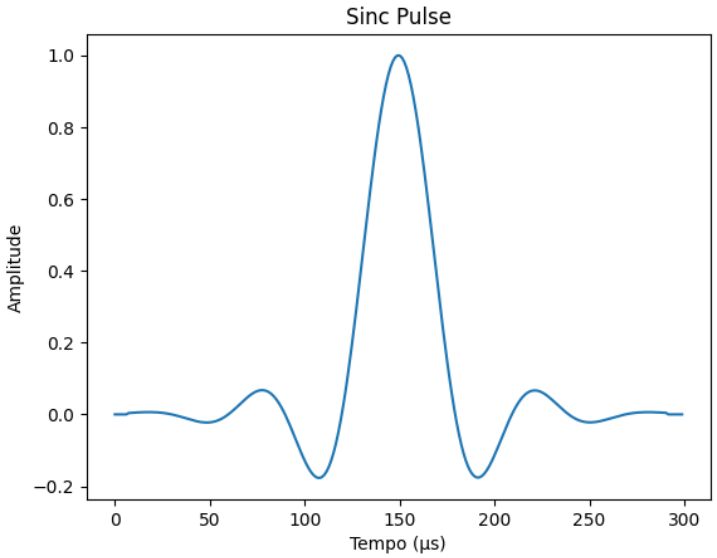


Figura 5: Forma de onda de um pulso *sinc* no domínio do tempo.

A Transformada de Fourier de um pulso *sinc* (função $sinc(x) = sin(x)/x$) infinito no domínio do tempo é uma função *RECT* que, quando aplicada na presença de um gradiente de seleção de fatia, resulta em um perfil de fatia uniforme. No entanto, para que o pulso possa ser gerado, sua duração deve ser finita, sendo obtida pela truncagem dos lóbulos laterais, mantendo apenas o lóbulo central e alguns de seus vizinhos. Essa truncagem introduz ondulações (ripples) no perfil de excitação, principalmente nas bordas da fatia[8, 7]. Quanto maior o número de lóbulos incluídos, melhor a aproximação ao perfil ideal de frequência, mas isso aumenta a duração do pulso, o que pode prolongar os tempos mínimos de eco (TE) e de repetição (TR) da sequência de pulsos [8].

Para atenuar os efeitos causados pelo truncamento e suavizar o perfil da fatia, geralmente é aplicada uma janela de apodização ao pulso *sinc*, isso consiste em multiplicar o pulso por uma função suavemente variável, como Hanning ou uma Gaussiana, tendo como resultado um perfil de excitação com muito menos ondulações. [8, 7].

- **Pulsos Shinnar-Le Roux (SLR):** desenvolvidos para resolver de forma direta e eficiente o problema inverso do design de pulsos seletivos. Esse problema consiste em determinar qual pulso de RF deve ser aplicado para se obter um perfil de fatia

desejado, a partir das condições iniciais da magnetização. Para pulsos com *flip angles* pequenos, a forma de um pulso de excitação pode ser aproximada pela Transformada Inversa de Fourier do perfil da fatia. No entanto, essa aproximação falha para *flip angles* maiores (entre 30° e 90°), devido à não linearidade das equações de Bloch, exigindo métodos mais avançados para o design preciso desses pulsos [8].

O algoritmo SLR permite calcular o pulso de RF de forma direta, sem a necessidade de processos iterativos. Ele utiliza parâmetros como a largura de banda de RF, duração do pulso, *flip angle*, percentual de *ripple* na banda passante e na banda de rejeição, e o algoritmo retorna o pulso de RF ideal por meio de um processo computacional direto. Essa abordagem também permite ao projetista realizar ajustes entre os parâmetros antes mesmo de gerar o pulso [8].

3.2 Console de Controle

Um console de controle, também conhecido como espectrômetro, é um componente eletrônico fundamental dos sistemas de RM, sendo responsável pela transmissão dos pulsos de RF, controle das sequências de pulsos de RF e gradiente de forma sincronizada e pela aquisição e digitalização dos sinais de RM para que possam ser processados para a reconstrução da imagem. O console atua como o "cérebro" do sistema, controlando a aquisição, gerenciando o tempo de eventos e enviando comandos para os componentes de hardware. Além disso, como já mencionado, ele pode atuar como a interface entre o operador e o equipamento. No entanto, o conhecimento detalhado sobre esse componente em grande parte permanece inacessível por conta da natureza proprietária dos sistemas comerciais. Nesse contexto, uma série de trabalhos de código aberto (*open-source*) surgiram como uma alternativa a esses sistemas fechados, buscando fornecer plataformas mais flexíveis, com componentes de baixo custo e facilmente programáveis. Esses projetos, como o Open-source Console for Real-time Acquistion (OCRA) e o "MAgnetic Resonance COntrol System (MaRCoS)", buscam romper com as limitações de hardware e software, facilitando o desenvolvimento e a pesquisa na área de RM [10, 9, 11].

O funcionamento do console pode ser dividido em duas funções principais: transmissão de pulsos de RF e a aquisição e processamento de sinais de RM. A maioria dos consoles utiliza a tecnologia de Síntese Digital Direta (DDS) para a geração das formas de onda de RF digitais, um tópico que será abordado com mais detalhes na Seção 3.3. Para a realização das tarefas do console, é essencial uma alta coerência de fase entre os diversos pulsos, o que impõe fortes restrições ao controle de tempo. Devido a essa necessidade de precisão e sincronização, placas com FPGA embarcada são frequentemente utilizados como o núcleo dos consoles de RM, sendo adequados para tarefas rápidas e sincronizadas no tempo [10, 9].

Após a geração dos sinais de banda base e de portadora utilizando a tecnologia DDS, esses sinais são modulados por um modulador de forma de onda. Esse processo alinha a frequência da portadora com a frequência de ressonância dos núcleos de hidrogênio, convertendo o sinal de banda base em um sinal de RF. O sinal modulado passa por um DAC e é transformado em um pulso de RF analógico desejado. Os pulsos são intensificados

por amplificadores de potência antes de serem enviados para a sonda [10, 6].

Simultaneamente, o console também gerencia os pulsos de gradiente. As formas de onda de gradiente, geradas digitalmente, são enviadas para um Gradient Power Amplifier (GPA), que as amplifica. Essas formas de onda amplificadas e filtradas acionam as bobinas de gradiente, que são essenciais para codificar espacialmente a imagem. Após a excitação dos núcleos de hidrogênio, o console gerencia a recepção e o processamento dos sinais emitidos pela amostra. O FID é captado pela bobina receptora e passa por um pré-amplificador antes da digitalização pelo Analog-to-Digital Converter (ADC) [12, 10].

Após a digitalização do sinal de RM pelo ADC, inicia-se o processo de recepção digital, no qual o sinal FID precisa passar por um fluxo de processamento que inclui *downconversion* digital e decimação. A etapa de *downconversion* tem como objetivo demodular o sinal de RM. O processo é realizado fazendo a multiplicação do sinal do ADC pelas saídas I e Q de um oscilador em quadratura que opera na frequência de Larmor. Esse processo permite extrair o sinal em banda base. Em seguida, aplicam-se filtros para os processos de decimação e filtragem, utilizados para reduzir a alta taxa de dados dos sinais, com o intuito de melhorar o processamento subsequente [10].

Os dados de RM são temporariamente armazenados e depois transmitidos via Ethernet para um PC. No PC, os dados do espaço-k são submetidos a uma Transformada inversa de Fourier para reconstruir a imagem final que é exibida para o operador [10, 3]. A Figura 6 apresenta o diagrama de blocos de um console de controle em RM, destacando o fluxo de dados no processo de transmissão dos pulsos de RF e gradiente, além do fluxo de recepção e digitalização dos sinais até a reconstrução e visualização da imagem.

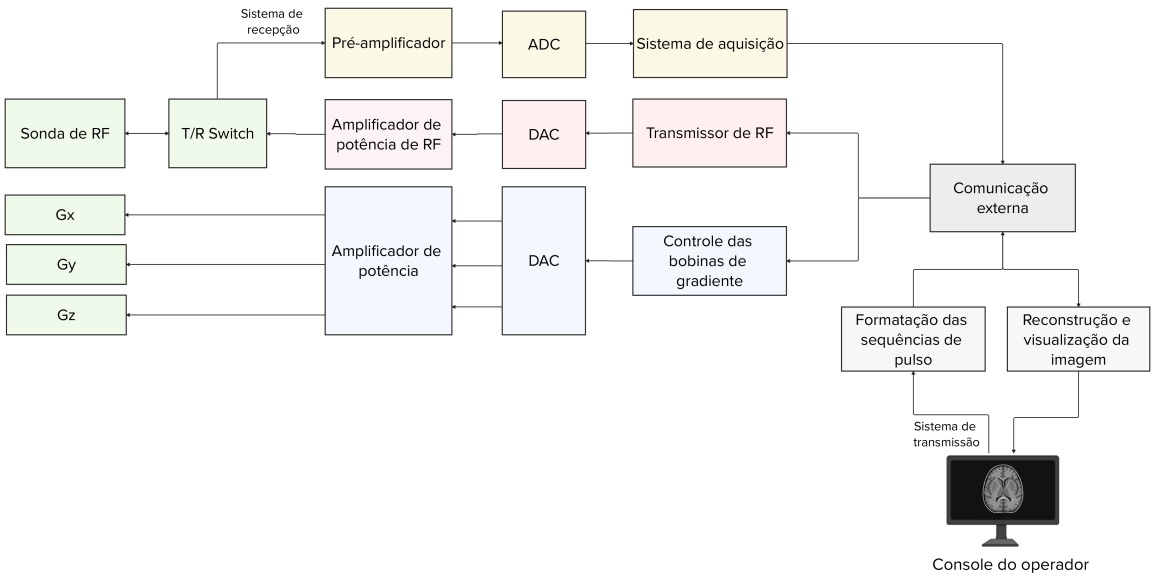


Figura 6: Diagrama de blocos de um console de controle em ressonância magnética, ilustrando os fluxos de transmissão e recepção dos sinais.

3.3 Processamento Digital de Sinais de RF

Nesta seção, são apresentados os princípios de processamento digital envolvidos na geração de sinais de RF utilizados em sistemas de RM. A Seção 3.3.1 introduz o conceito de modulação em quadratura (IQ). Em seguida, são descritos os métodos de geração digital de sinais, com ênfase na técnica de Síntese Digital Direta (DDS) e na arquitetura do Oscilador Controlado Numericamente (NCO), discutidos na Seção 3.3.2.

3.3.1 Modulação *IQ*

A modulação em quadratura *(IQ – In-phase/Quadrature)* é um conceito utilizado em sistemas de processamento digital de sinais de RF, baseando-se na decomposição de sinais senoidais em duas componentes ortogonais, denominadas em fase (I) e em quadratura (Q). A terminologia "IQ" tem como origem a representação de um sinal de RF que pode ter sua representação por coordenadas polares (representação de amplitude/fase) ou em coordenadas cartesianas. Qualquer sinal senoidal de RF pode ser modelado como um fasor, um vetor girante com amplitude A, frequência ω e fase inicial e ϕ_0 [13]:

$$y(t) = A \cdot \sin(\omega t + \phi_0) \tag{3}$$

Usando funções trigonométricas, o sinal pode ser decomposto em suas componentes seno e cosseno:

$$y(t) = I \cdot \cos(\omega t) + Q \cdot \sin(\omega t) \tag{4}$$

Onde a amplitude da componente cosseno é definida como a componente em fase (I), e a amplitude da componente seno é chamada de componente em quadratura (Q) [13].

3.3.2 Geração de Pulsos de RF

Atualmente, a maioria dos sistemas de RM emprega tecnologia de Síntese Digital Direta (DDS) para geração de pulsos de RF. Em comparação com métodos analógicos tradicionais de geração de frequência, o DDS oferece vantagens significativas, como a alta resoluação de frequência, precisão de fase e rápida comutação de frequência. A arquitetura do DDS é comumente implementada em FPGAs devido à sua aplicação em circuitos de processamento de sinais de alta velocidade, que incluem tabelas de busca e registradores [10].

A técnica de DDS é usada para a geração de formas de ondas analógicas de alta frequência. A arquitetura se baseia em um Oscilador Controlado Numericamente (NCO) e um DAC conforme ilustrado na Figura 7 [13].

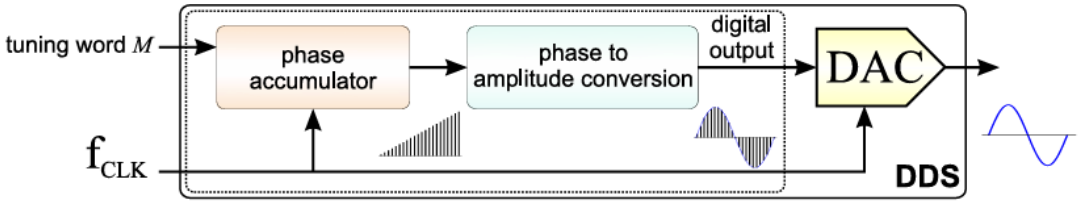


Figura 7: Diagrama de um DDS contendo um NCO (linha tracejada) e DAC [13].

Na revisão da literatura foi observado uma variação no uso dos conceitos de DDS e NCO. Alguns trabalhos tratam o NCO como o bloco responsável pela geração digital da potadora, enquanto a técnica de DDS se baseia nos mesmos blocos utilizados no NCO, mas é expandida por meio de um DAC para gerar as formas de onda analógicas [13]. Em outros trabalhos, o processo completo de geração digital de sinais é tratado inteiramente como DDS, sem destacar o NCO como um componente [10]. A utilização dos termos pode variar de acordo com o contexto e o trabalho, embora ambas as abordagens descrevam praticamente a mesma teoria de geração digital de sinais. Nesse trabalho, vamos adotar a ideia de que o NCO se refere especificamente à geração digital da portadora, enquanto o DDS será considerado o sistema completo, englobando o NCO e o DAC para a geração dos sinais analógicos.

O NCO é implementado como um oscilador digital em quadratura, gerando uma sequência de amostras de seno e cosseno. A funcionalidade básica de um NCO consiste em um acumulador de fase (*Phase accumulator*) e um módulo de conversão de fase para amplitude (*Phase to amplitude conversion*) conforme ilustrado na Figura 8 [13].

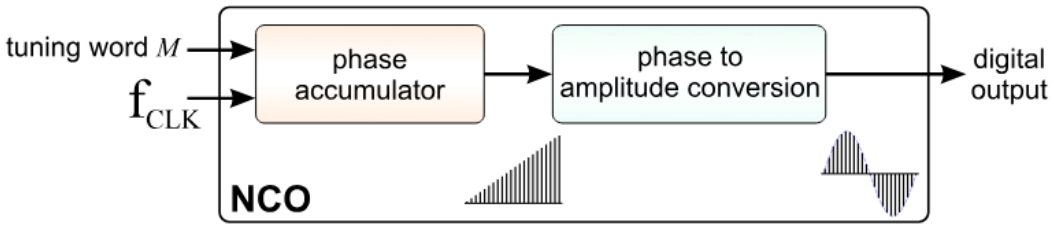


Figura 8: Diagrama de um NCO com os módulos *phase accumulator* e *phase to amplitude conversion* [13].

O processo inicia com um incremento de fase programável que é fornecido como entrada. A cada ciclo de clock, o incremento de fase (*tuning_word M*) é adicionada ao *phase accumulator*, que gera um endereço de fase correspondente. A fase total resultante é convertida para o valor de amplitude correspondente, usando uma tabela de consulta (*lookup table*, LUT) onde uma onda senoidal está armazenada. A produção sequencial desses valores constrói a forma de onda digital, que depois é convertida em uma forma de

onda analógica por um DAC. A frequência de saída da forma de onda (f_out) é determinada pela frequência de clock (f_clk), pelo *tuning word* M e pela largura de bits do *phase accumulator* (N), a relação é dada pela Equação 5 [13, 10].

$$f_{out} = \frac{M \cdot f_{CLK}}{2^N} \tag{5}$$

O *tuning word* é programável, o que permite a seleção de praticamente qualquer frequência de saída até metade da frequência de clock (frequência de *Nyquist*) e a alteração da frequência ocorre imediatamente após o carregamento de um novo incremento de fase no registrador de sintonia M. A *lookup table* com as amostras do seno pode conter uma onda senoidal completa ou apenas um quarto de período. Esta otimização é suficiente para a reconstrução da onda senoidal devido a propriedade de simetria da função e reduz a quantidade de espaço de memória necessário (entretanto, os cálculos adicionais necessários para obter a amplitude correta devem ser ponderados em relação à economia de espaço de memória)[13, 14].

Na FPGA, a forma de onda em banda base, como por exemplo, um pulso do tipo *sinc*, pode ser combinada com uma portadora, possibilitando modulação em amplitude e fase, resultando na geração do pulso de RF, esse processo pode ser descrito como uma etapa de *up-conversion*. O termo *up-conversion* refere-se ao processo de conversão de um sinal digital em banda base para um sinal em banda passante, isto é, a transposição da envoltória do pulso para a frequência de operação necessária para excitar a amostra. Essa conversão é realizada pela combinação da portadora, na frequência de interesse, com a envoltória modulante, que controla a amplitude da portadora. O resultado é um sinal de RF centrado na frequência desejada, mas preservando o formato temporal do pulso original. A representação da envoltória em banda base é feita através das componentes em fase (I) e em quadratura (Q), que permitem a modulação IQ, técnica que possibilita o controle independente de amplitude e fase do sinal resultante [10, 13, 11].

4 METODOLOGIA

Este capítulo descreve a metodologia adotada para o desenvolvimento do gerador digital de pulsos de RF proposto neste trabalho.

A Seção 4.1 apresenta os requisitos e especificações do sistema. As Seções 4.2 e 4.3 apresentam a geração das amostras em Python utilizadas nos módulos em VHDL. A Seção 4.4 detalha a simulação do comportamento do NCO em Python. Em seguida, a Seção 4.5 apresenta a implementação em VHDL, destacando os módulos de integração. A Seção 4.6 descreve as funcionalidades dos módulos básicos e reutilizáveis. A Seção 4.7 discute o processo de validação dos módulos por meio de *testbenches* e simulações no Vivado. Por fim, a Seção 4.8 mostra os testes em bancada realizados com a FPGA.

4.1 Requisitos e Especificações do Sistema

O protótipo de equipamento de MRI desenvolvido pelo CNPEM se baseia em um magneto central, responsável pela geração do campo magnético principal (B_0), construído com dois conjuntos de ímãs permanentes. Esses ímãs fornecem um campo de 0,4 T em sua região central. De acordo com a Equação 1 a frequência de Larmor para excitar os spins nesse campo deve ser aproximadamente 17 MHz. Para a operação, o projeto utiliza uma bobina solenoide que atua tanto na transmissão quanto na recepção dos sinais de RF.

Para atender aos requisitos de geração e aquisição de sinais nessa frequência, optou-se pela utilização da placa comercial Red Pitaya SDRLab 122–16 (Figura 9), empregada em projetos open-source como o OCRA e o MaRCoS [11, 9]. A Red Pitaya é baseada em um *System-on-Chip* (SoC) que integra um processador Dual-Core ARM Cortex-A9 e uma FPGA Xilinx Zynq 7020, com 512 MB de memória RAM DDR3, permitindo o processamento de sinais em tempo real.

Seus recursos de Entrada/Saída (*Input/Output*) incluem dois canais de entrada analógica com ADCs de 16 bits e dois canais de saída com DACs de 14 bits, ambos operando a uma taxa de amostragem de 122,88 MS/s. As entradas e saídas de RF possuem impedância de 50 Ω e largura de banda de operação que varia de 300 kHz a 550 MHz para as entradas e até 60 MHz para saídas. Essas características tornam a placa adequada para equipamentos de MRI com campos de até 1,17 T [9], cobrindo com margem a frequência de interesse deste trabalho de 17 MHz.

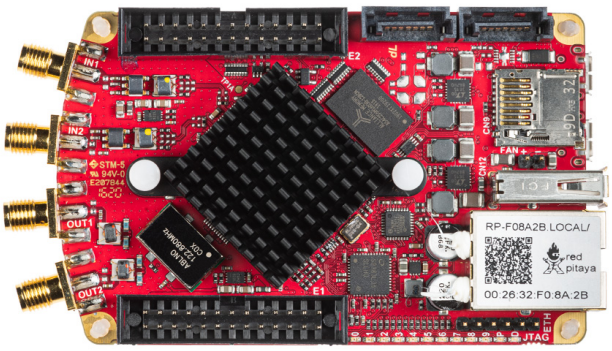


Figura 9: Placa comercial Red Pitaya 122-16.

4.2 Geração de Pulsos de RF em Python e Conversão para Ponto Fixo

Nesta etapa, foram gerados diferentes formatos de pulsos de RF em ambiente *Python*, com o objetivo de gerar a envoltória com as amostras dos formatos de pulsos para a implementação em VHDL. Foram explorados diferentes formatos de pulsos e janelas de apodização, variando parâmetros e realizando a análise espectral para observar a resposta

em frequência do pulso. Essa resposta determina, mediante a aplicação simultânea de um gradiente de seleção de fatia, a espessura e a forma da fatia a ser imageada.

A geração dos pulsos foi realizada por meio de uma classe desenvolvida em Python para este trabalho, que permite configurar parâmetros como o tipo de pulso, amplitude, taxa de amostragem, duração, e tipo de janela de apodização. Foram implementadas janelas do tipo Hamming, Hanning e Gaussiana, de modo a permitir o estudo comparativo de diferentes perfis de suavização. Após a geração, cada forma de onda foi analisada tanto no domínio do tempo quanto no domínio da frequência. Neste contexto, utilizamos a Transformada Discreta de Fourier (DFT) (implementada via FFT) para obter o espectro de magnitude dos sinais, expresso em dB relativos, o que é suficiente para a determinação da largura de banda dos pulsos.

Para a validação do fluxo de processamento, optou-se pela utilização de um caso representativo, um pulso *sinc* janelado com Hamming, ilustrado na Figura 10. O pulso *sinc* foi gerado com uma duração de 300 μs e 5 cruzamentos por zero (*zero-crossings*), utilizando uma taxa de amostragem (*sample rate*) de aproximadamente 833,33 kHz (definida pela relação `50*(N_CROSSINGS/PULSE_DURATION)`), garantindo que o pulso fique bem amostrado. A janela Hamming foi aplicada com comprimento equivalente a 95% da duração total do pulso, totalizando 250 amostras e suavizando as transições nas extremidades do pulso. Depois, as amostras calculadas (pontos) da forma de onda foram convertidas para representação em aritmética de ponto fixo de 14 bits: os 13 bits menos significativos codificam a parte fracionária normalizada ($0 < valor <= 1$) do valor e o 1 bit mais significativo restante é usado para codificar o sinal (0=positivo e 1=negativo). As amostras foram passadas para binário e ao final, gerou-se um arquivo txt contendo uma amostra por linha para ser usado na implementação em VHDL.

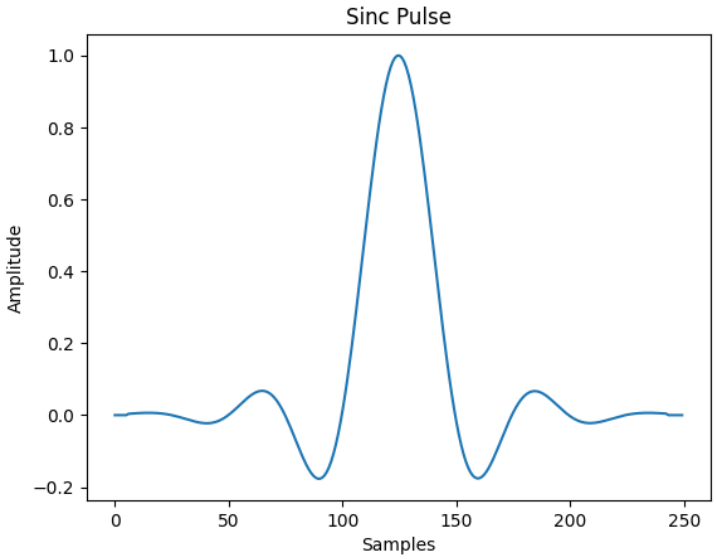


Figura 10: Pulso *sinc* janelado com Hamming (250 amostras) gerado em Python.

Os parâmetros do pulso foram selecionados de forma a gerar um conjunto de amos-

tras do pulso *sinc* com o objetivo de validar a implementação em VHDL, sem a pretensão de otimizar para uma aplicação específica em MRI. A arquitetura desenvolvida é parametrizável, permitindo que diferentes formatos de pulso possam ser gerados modificando os parâmetros no script Python e gerando um novo arquivo de dados.

Para estabelecer uma referência para a validação temporal e espectral, o pulso de 250 amostras foi interpolado em Python simulando o comportamento do bloco `Pulse_Gen` (Seção 4.5.2), com o fator de divisão (`division_factor`) igual a 400 e frequência de clock de 100 MHz. Realizou-se a DFT via FFT para determinar a largura de banda da envoltória em banda base, usando como critério -10 dB em relação ao pico central do espectro. Essa análise estabelece o valor esperado para o sistema, servindo como referência para a validação dos resultados da implementação em VHDL e das medições experimentais. Este pulso interpolado também serviu para uso como vetor de teste para validação do módulo `Pulse_Gen` posteriormente.

O pulso utilizado neste trabalho serve como estudo de caso para validar o fluxo completo de desenvolvimento, desde a geração do sinal em ambiente de alto nível até sua implementação em hardware. Para aplicações futuras, pulsos com características específicas podem ser integrados ao sistema.

4.3 Geração de Um Quarto da Onda Senoidal

Para a geração das amostras armazenadas na ROM, foi desenvolvido um script em Python responsável por criar um quarto do período (1/4) de uma onda senoidal, que é posteriormente utilizado pelo módulo `Phase_To_Address` em VHDL para reconstruir o período completo da senoide e gerar o cosseno defasado em 90°. Definiu-se o número total de amostras por período com uma resolução de 12 bits (`SAMPLES_SIZE = 4096`). A geração do sinal senoidal é feita através da função `numpy.sin()`, que calcula os valores de $\sin(\theta)$ para um vetor de pontos igualmente espaçados entre 0 e 2π. A Figura 11 ilustra a onda senoidal gerada com um período com 4096 amostras:

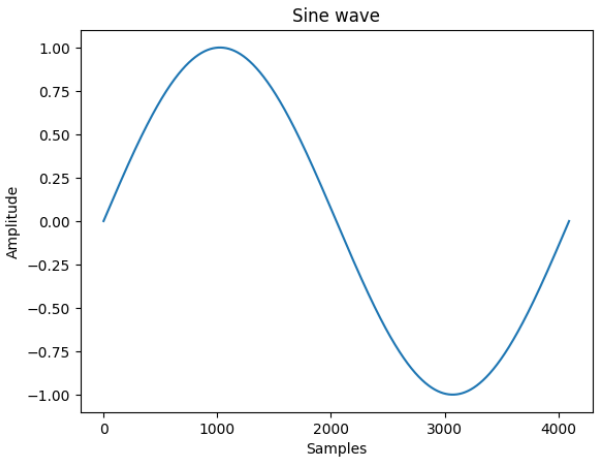


Figura 11: Um período da onda senoidal gerada com 4096 amostras.

Depois, seleciona-se apenas um quarto de período (Figura 12), fazendo `QUARTER_CYCLE = SAMPLES_SIZE/4`, resultando em 1024 amostras que serão armazenadas na `ROM`.

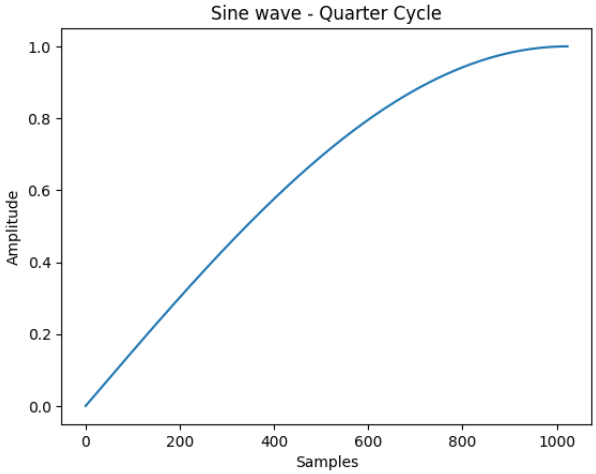


Figura 12: Um quarto do período da onda senoidal gerada com 1024 amostras.

Por fim, as amostras do um quarto de onda foram convertidas para ponto fixo com 14 bits (13 bits fracionários e 1 bit de sinal) e salvas em um arquivo de texto (.txt), seguindo o mesmo procedimento descrito na Seção 4.2 para o pulso *sinc*.

4.4 Simulação do comportamento do `NCO` em Python

Para validar o funcionamento do `NCO` antes de fazer a implementação em VHDL, foi desenvolvida uma simulação em Python que reproduz o seu comportamento. O objetivo foi validar se a frequência das portadoras muda de acordo com o valor de `tuning_word`, e se o seu espectro de magnitude esta centrada na frequência correta. Para fazer esse processo, foi criada uma classe para gerar as amostras das funções seno e cosseno. O número de amostras é determinado pela resolução de bits definida para o acumulador de fase (`bit_resolution`), correspondendo a 2^N pontos igualmente espaçados entre 0 e 2π. Essas amostras são usadas na classe `NCO` que faz o mecanismo de incremento e endereçamento de fase. A partir da Equação 5, o incremento de fase (`tuning_word` M) pode ser calculado conforme a Equação 6, em função da frequência desejada (f_{out}), da frequência de amostragem (f_s) e da resolução do acumulador (N):

$$M = \frac{f_{out} \cdot 2^N}{f_{CLK}} \tag{6}$$

A cada passo da simulação, o endereço é atualizado somando o valor do `tuning_word`, e as amostras correspondentes das ondas seno e cosseno são obtidas. Esse processo reproduz o comportamento do acumulador de fase e da geração periódica de portadoras senoidais e cossenoidais.

Para a análise, foram simuladas diferentes frequências de saída (1, 5, 10, 17, 20 e 25 MHz) considerando uma frequência de amostragem de 100MHz e resolução de 12 bits. Em seguida, os sinais foram quantizados em ponto fixo com 14 bits totais e 13 fracionários, reproduzindo o mesmo formato usado na implementação em VHDL.

O comportamento do `NCO` foi analisado tanto no domínio do tempo, observando as formas de onda senoidal e cossenoidal geradas, quanto no domínio da frequência, por meio do cálculo da DFT para validar se conforme a variação do `tuning_word` gerava a frequência de saída correspondente.

Também foi possível determinar a resolução de frequência do `NCO`, ou seja, o menor incremento de frequência obtido a cada variação unitária do incremento de fase. Essa resolução é dada por:

$$f_{res} = \frac{f_s}{2^N} \tag{7}$$

Onde f_s é a frequência de amostragem e N corresponde ao número de bits do acumulador de fase. Para os parâmetros utilizados na simulação ($f_s = 100MHz$ e $N = 12$), a resolução de frequência obtida foi de aproximadamente 24,4 kHz. Esse valor representa o "pulo" mínimo de frequência entre duas saídas possíveis do oscilador digital, e está diretamente relacionado ao tamanho do acumulador, ou seja, quanto maior o número de bits, menor será o passo de variação de frequência.

4.5 Implementação em VHDL

Esta seção apresenta a implementação em VHDL dos módulos que compõem o gerador digital de pulsos de RF. A Seção 4.5.1 descreve o módulo `IQ_Modulation`, responsável pela modulação do sinal. A Seção 4.5.2 apresenta o módulo `Pulse_Gen`, que realiza a geração da envoltória do pulso. Por fim, a Seção 4.5.3 apresenta o módulo `NCO` responsável por gerar as portadoras utilizadas no sistema.

4.5.1 Modulação IQ (`IQ_Modulation`)

O módulo `IQ_Modulation` integra os blocos `Pulse_Gen` e `NCO` para realizar a modulação do pulso de RF. O funcionamento do módulo inicia-se com a geração simultânea da envoltória do pulso *sinc* pelo bloco `Pulse_Gen` e das portadoras senoidal e cossenoidal pelo `NCO`. Nesta implementação, optou-se por gerar o pulso *sinc* somente com a componente real. Dessa forma, a mesma envoltória real gerada pelo `Pulse_Gen` é utilizada como entrada para ambas as componentes I e Q do modulador. Esta abordagem foi adotada propositalmente com o objetivo de simplificar o *hardware* e focar na etapa de controle de frequência via DDS/NCO. Extensões futuras podem explorar perfis complexos com fase arbitrária, como nos pulsos do tipo SLR.

O `Pulse_Gen` é configurado com parâmetros para gerar a envoltória do pulso *sinc* com duração controlada, enquanto o `NCO` é parametrizado para produzir as portadoras senoidal e cossenoidal na frequência de interesse. O bloco `IQ_Modulation` executa as

multiplicações entre a envoltória do pulso e cada uma das portadoras, esse processo é sincronizado pelo sinal de validação `valid_in_nco`, proveniente do `NCO`, este sinal atua como habilitador para o processo da modulação. Os sinais de saída do `IQ_Modulation` são então o `real_pulse` (multiplicação do pulso por cosseno) e `imaginary_pulse` (multiplicação do pulso por seno). O diagrama do módulo está ilustrado na Figura 13

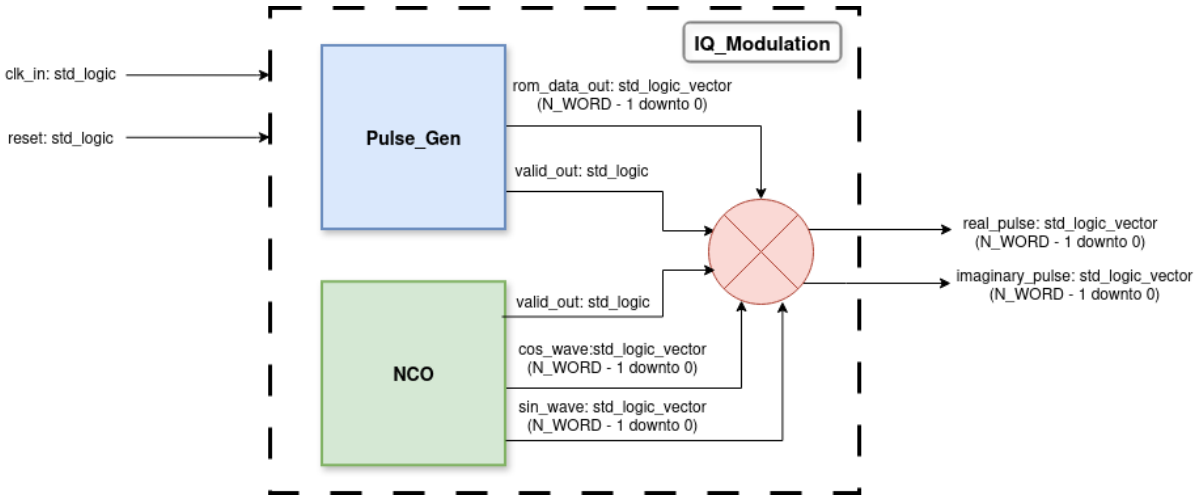


Figura 13: Diagrama do módulo `IQ_Modulation`. O símbolo circular em vermelho representa a operação de multiplicação entre os sinais da envoltória gerada pelo `Pulse_Gen` e as portadoras senoidal e cossenoidal do `NCO`.

4.5.2 Gerador de Pulso (`Pulse_Gen`)

O módulo *Pulse_Gen* tem como função integrar os módulos `Clk_Decimator` (Seção 4.6.1), `Phase_Accumulator` (Seção 4.6.2) e `ROM` (Seção 4.6.4) para a geração do pulso *sinc*. Sua principal função é conectar as portas de entrada e saída desses blocos, além de gerenciar os parâmetros genéricos do sistema, como o número de amostras (`SAMPLES_SIZE`), a largura de bits dos dados (`N_WORD`) e o caminho para o arquivo de amostras (`FILE_NAME`).

O fluxo de dados e controle dentro do *Pulse_Gen* inicia com o `Clk_Decimator`. Este módulo recebe o clock principal (`clk_in`) e um `division_factor` para gerar um clock de pulso mais lento (menor frequência), o `dec_clk_out`. Este sinal determina quando o `Phase_Accumulator` pode fazer o incremento de endereço.

O `Phase_Accumulator` atua como um contador, avançando em cada pulso de `dec_clk_in`. Ele gera endereços sequenciais (`rom_address`) que são usados para acessar as amostras na memória `ROM`. Para a geração do pulso *sinc*, o `tuning_word` foi definido como 1, garantindo que o acumulador avance em um passo por vez e leia todas as amostras em sequência. O `division_factor` foi mantido igual a 400 para gerar um pulso com duração de 1 ms (As definições desses valores são explicadas nas Seções 4.6.1 e 4.6.2). A cada avanço, o `Phase_accumulator` também gera um sinal de validação (`valid_acc_out`), indicando que um novo endereço está disponível.

O sinal de saída `valid_acc_out` conecta no `valid_in` da `ROM`, servindo como um sinal de habilitação. A `ROM` libera a amostra correspondente (`rom_data_1`) e um sinal de validação (`valid_rom_out`) que confirma a disponibilidade do dado. As saídas finais do módulo `Pulse_Gen` são o pulso *sinc* (`rom_data_out`) e seu sinal de controle (`valid_out`). O diagrama do módulo `Pulse_Gen` está ilustrado na Figura 14.

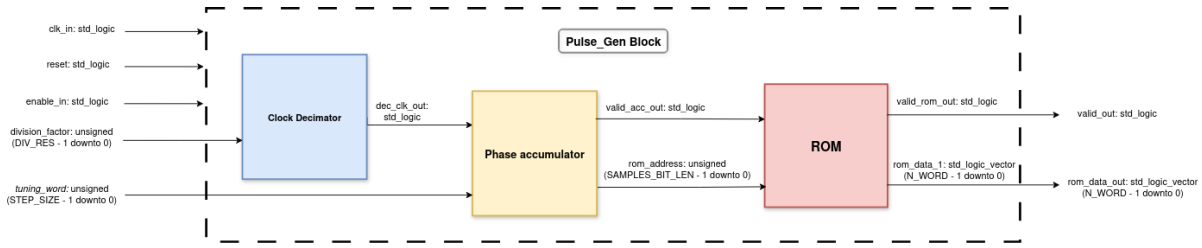


Figura 14: Diagrama do módulo `Pulse_Gen` com os sinais de entrada e saída.

4.5.3 Gerador das Portadoras (`NCO`)

O módulo `NCO` é responsável pela geração das portadoras senoidal e cossenoidal utilizadas na modulação do pulso de RF. Ele integra quatro módulos, o `Clk_Decimator` (Seção 4.6.1), o `Phase_Accumulator` (Seção 4.6.2), o `Phase_to_Address` (Seção 4.6.3) e a `ROM` (Seção 4.6.4). Sua função é conectar e gerenciar os sinais entre esses blocos, além de definir os parâmetros genéricos do sistema, como a quantidade de amostras e o arquivo .txt contendo um quarto de onda do seno.

O fluxo de operação inicia-se no bloco `Clk_Decimator`, que recebe o sinal de clock principal `clk_in` e o `division_factor`, gerando um clock de saída `dec_clk_out`. No contexto desta implementação, o valor de `division_factor` foi mantido igual a 1, de forma a utilizar a frequência máxima disponível do clock da placa. Esse sinal é então utilizado para sincronizar o incremento de fase no `Phase_Accumulator`.

O `Phase_Accumulator` atua como um contador controlado por clock, responsável por gerar os endereços de fase (`Phase_Accumulator`) a partir do valor de incremento (`tuning_word`). Esse parâmetro define a frequência das portadoras geradas. A cada incremento, o módulo também emite o sinal de validação `valid_acc_out`, indicando a disponibilidade de um novo endereço válido.

Em seguida, o módulo `Phase_to_Address` recebe o endereço `rom_address` e o converte em endereços específicos para leitura das amostras `sin_index` e `cos_index`. Esse bloco implementa a lógica necessária para reconstruir o sinal completo das portadoras utilizando apenas um quarto de onda armazenado na `ROM`. Ele também é responsável por gerar os sinais de controle `sin_negate` e `cos_negate`, que determinam quando a saída deve ser invertida para reproduzir corretamente os quatro quadrantes do período senoidal. O sinal `valid_phase_out` é emitido por este bloco para indicar que os endereços e sinais de negação estão prontos e válidos para leitura na `ROM`.

A ROM armazena as amostras do quarto de onda da senoide em formato de ponto fixo de 14 bits (1 bit de sinal e 13 fracionários). O arquivo especificado em FILE_NAME contém essas amostras, previamente geradas em Python. Durante a operação, a ROM recebe os endereços sin_index e cos_index e libera as amostras correspondentes rom_data_1 e rom_data_2, além de gerar o sinal de controle valid_out.

Por fim, as amostras lidas da ROM passam pela lógica de negação, implementada por dois processos síncronos. Essa etapa aplica o sinal de inversão (sin_negate ou cos_negate) às respectivas amostras, completando a reconstrução do sinal. O resultado são as saídas sin_wave e cos_wave, correspondentes às portadoras senoidal e cossenoidal defasadas em 90°.

O diagrama do módulo NCO, apresentado na Figura 15, ilustra a integração entre os blocos e o fluxo de dados e controle descritos.

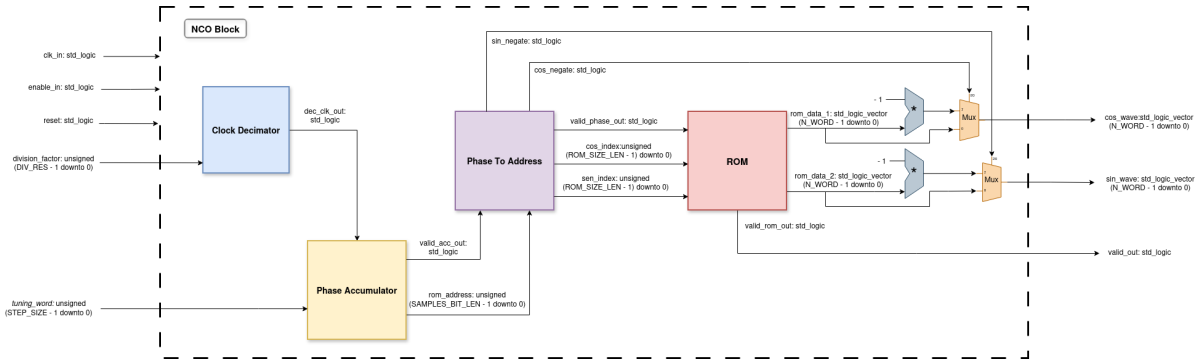


Figura 15: Diagrama do módulo NCO com os sinais de entrada e saída.

4.6 Módulos Básicos e Reutilizáveis

Os módulos foram desenvolvidos de forma que fossem reutilizáveis tanto para a geração do pulso de RF quanto para a geração das portadoras no NCO.

4.6.1 Clock Decimator (Clk_Decimator)

O módulo Clk_Decimator tem como objetivo controlar a duração total do pulso de RF, atuando como um divisor de frequência que gera um sinal de habilitação periódico para o módulo Phase_Accumulator. Ele funciona como um contador que recebe um sinal de entrada chamado division_factor. A cada ciclo do clock da FPGA, o contador é incrementado até atingir o valor definido por division_factor. Quando isso ocorre, o sinal de saída (dec_clk_out) é ativado por um ciclo de clock, e o contador é reiniciado. O diagrama do módulo com os sinais está represnetado na Figura 16.

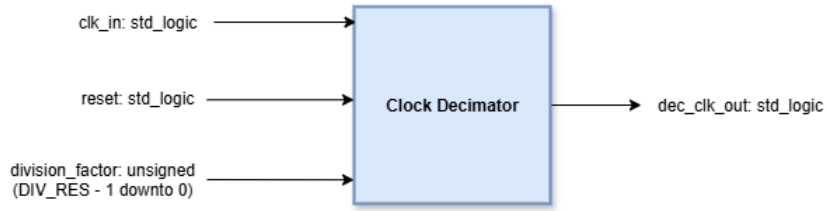


Figura 16: Diagrama do módulo `Clk_Decimator` com os sinais de entrada e saída.

O parâmetro genérico `DIV_RES` define a largura do contador, ou seja, o número de bits utilizados para representar o valor de `division_factor`. Esse parâmetro estabelece o intervalo máximo de contagem, permitindo ajustar a resolução temporal do sinal de saída.

Dessa forma, o sinal `dec_clk_out` atua como um pulso de habilitação (*enable*) que determina o instante em que uma nova amostra do pulso de RF deve ser liberada pelo `Phase_Accumulator`. Assim, o tempo total do pulso pode ser controlado a partir do número total de amostras e da frequência de clock da FPGA. Por exemplo, supondo uma duração desejada de 1 ms, um arquivo contendo 250 amostras e uma frequência de clock de 100 MHz, o fator de divisão (`division_factor`) é calculado como:

$$\text{division\_factor} = \frac{f_{\text{clk}} \times T_{\text{pulse}}}{N_{\text{samples}}} = \frac{100 \times 10^{6} \times 1 \times 10^{-3}}{250} = 400 \tag{8}$$

Nesse caso, o módulo libera uma nova amostra a cada 400 ciclos de clock, resultando em um pulso total de 1 ms. Para a geração das portadoras senoidais, o valor de `division_factor` foi mantido igual a 1, de modo a aproveitar a frequência máxima de clock da FPGA, já que essas ondas devem ser geradas em alta frequência. Dessa forma, o mesmo módulo pode ser reutilizado em diferentes partes do sistema, mas com parametrizações distintas conforme a função desempenhada.

4.6.2 Phase Accumulator (`Phase_Accumulator`)

O módulo `Phase_Accumulator` é responsável pelo controle do incremento de fase, atuando como um contador que gera os endereços da memória `ROM`, onde estão armazenadas as amostras das ondas senoidais e cossenoidais utilizadas como portadoras. Esse módulo recebe como entrada o sinal `tuning_word`, que define o passo de incremento de fase a cada ciclo de clock. O valor de `tuning_word` é obtido a partir da relação expressa na Equação 5, que relaciona a frequência de saída com o valor do incremento aplicado. A Figura 17 apresenta o diagrama esquemático do módulo, com suas principais entradas e saídas.

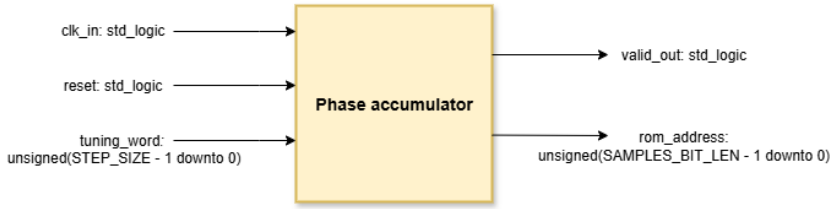


Figura 17: Diagrama do módulo `Phase_Accumulator` com os sinais de entrada e saída.

O módulo utiliza parâmetros genéricos que definem os tamanhos dos barramentos internos de forma automática, garantindo flexibilidade de uso:

- `SAMPLES_SIZE`: número total de amostras armazenadas na `ROM`. No caso do pulso de RF, esse valor é definido como 250.
- `SAMPLES_BIT_LEN`: quantidade de bits necessária para endereçar todas as amostras, calculada como $\lceil \log_2(\texttt{SAMPLES\_SIZE}) \rceil$. Para 250 amostras, são necessários 8 bits.
- `STEP_SIZE`: define a largura do sinal `tuning_word`, responsável pelo passo de incremento do acumulador. É calculado como $\lceil \log_2(\texttt{SAMPLES\_SIZE}/2) \rceil$, o `SAMPLES_SIZE` é dividido por 2 para limitar o tamanho do `tuning_word` de forma que o step pegue no mínimo 2 amostras a cada leitura da tabela (respeitando o Teorema de *Nyquist*).

Essas parametrizações tornam o módulo escalável, permitindo ajustar automaticamente o tamanho dos vetores de acordo com o número de amostras utilizado, sem necessidade de modificar o código-fonte.

A cada borda de subida do sinal `clk_in`, o módulo soma o valor atual do acumulador com o `tuning_word`. Essa soma gera o novo endereço da `ROM`, responsável por determinar o ponto da forma de onda que será lido. Quando o acumulador atinge o valor máximo (`SAMPLES_SIZE`), ele retorna a zero, garantindo a periodicidade da forma de onda. O incremento só é realizado quando o sinal `dec_clk_in` (proveniente do módulo `Clk_Decimator`) é igual a '1'. Dessa forma, o `Phase_Accumulator` atualiza o endereço da `ROM` apenas quando há um pulso de habilitação válido. O sinal `valid_acc_out` é ativado sempre que um novo endereço é gerado, funcionando como um indicativo de dado válido para o módulo subsequente.

De maneira análoga, o módulo `Phase_Accumulator` foi projetado para ser reutilizável tanto na geração das portadoras quanto no endereçamento do pulso de RF. No caso das portadoras, o parâmetro `tuning_word` controla a frequência de saída das ondas senoidal e cossenoidal. Para o pulso de RF, esse parâmetro foi definido como 1, de modo que as amostras sejam percorridas sequencialmente na `ROM`.

4.6.3 Phase to Address (`Phase_to_Address`)

O módulo `Phase_to_Address` tem como principal função realizar o mapeamento entre o endereço recebido do módulo `Phase_Accumulator`, e os endereços de leitura da `ROM` responsável por armazenar as amostras de um quarto de onda senoidal. Esse mapeamento

permite reconstruir o período completo (360°) a partir das amostras de um quarto de onda geradas em Python, isso pode ser feito devido as propriedades de simetria das funções seno e cosseno. Essa estratégia reduz o uso de memória sem comprometer a resolução da forma de onda, já que a quantidade de amostras na `ROM` pode ser ajustada para garantir alta precisão.

O parâmetro `ROM_SIZE` é derivado de `SAMPLES_SIZE` como `ROM_SIZE` = `SAMPLES_SIZE`/4, correspondendo ao número de amostras armazenadas para um quarto do período. O tamanho do índice da `ROM` (`ROM_SIZE_LEN`) é calculado automaticamente como $\lceil \log_2(\texttt{ROM\_SIZE}) \rceil$. A Figura 18 representa o diagrama do módulo, destacando as entradas e os sinais de saída.

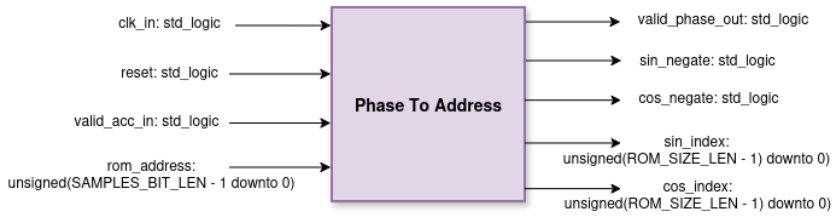


Figura 18: Diagrama do módulo `Phase_to_Address` com os sinais de entrada e saída.

A lógica de reconstrução do sinal consiste na divisão do ciclo completo da senoide em quatro quadrantes, cada um correspondente a um trecho de 90° do período. O endereço recebido (`rom_address`) é comparado com os limites desses quadrantes para determinar como deve ser feita a leitura das amostras da `ROM` e se a amostra deve ter inversão de sinal.

- **Primeiro quadrante (0° a 90°):** as amostras são lidas na ordem direta, sem inversão do sinal (VHDL: `sin_negate <= '0'`);
- **Segundo quadrante (90° a 180°):** as amostras são lidas de forma espelhada (do final para o início), sem inversão de sinal (VHDL: `sin_negate <= '0'`);
- **Terceiro quadrante (180° a 270°):** as amostras voltam a ser lidas em ordem direta, mas o sinal é invertido (VHDL: `sin_negate <= '1'`);
- **Quarto quadrante (270° a 360°):** as amostras são novamente lidas de forma espelhada, com inversão de sinal (VHDL: `sin_negate <= '1'`).

Essa lógica permite que o valor do endereço dentro da `ROM` varie sempre entre `0` e `ROM_SIZE - 1`, independentemente do quadrante atual, enquanto o sinal de negação (`sin_negate`) indica se o valor lido deve ser multiplicado por -1. Essa multiplicação é aplicada no módulo `NCO`, responsável pela geração das portadoras.

Para a geração da componente cossenoidal, é aplicada a mesma lógica, mas com um deslocamento de fase de 90°, implementado por meio da soma de `ROM_SIZE = 256` ao endereço recebido (`rom_address`). Assim, o módulo gera os endereços (`sin_index` e `cos_index`) e os sinais de inversão (`sin_negate` e `cos_negate`) correspondentes às formas de onda senoidal e cossenoidal.

4.6.4 ROM (Read-Only Memory)

O módulo `ROM` foi desenvolvido para armazenar as amostras utilizadas na geração das formas de onda senoidais e da forma de onda do pulso de RF. O conteúdo é carregado a partir dos arquivos (.txt) gerados em Python, contendo as amostras em formato binário com um tamanho de `N_WORD` bits. A arquitetura do módulo é genérica, permitindo sua reutilização em diferentes contextos. Os parâmetros configuráveis são o número de amostras (`SAMPLES_SIZE`), o tamanho de cada amostra (`N_WORD`) e o caminho do arquivo de entrada (`FILE_NAME`). O tamanho do endereço (`SAMPLES_BIT_LEN`) é calculado automaticamente como $\lceil \log_2(\texttt{SAMPLES\_SIZE}) \rceil$, garantindo que alterações no número de amostras ajustem o tamanho do endereço de forma automática.

A lógica interna do módulo consiste em ler linha a linha o arquivo de entrada durante a inicialização do sistema e armazenar e disponibilizar na saída a amostra correspondente ao endereço fornecido. Este módulo é utilizado de forma diferente na geração do pulso e das portadoras. Na geração das portadoras (`NCO`) a `ROM` armazena um quarto do período da onda senoidal. A partir desse arquivo, todo o período é reconstruído a partir da lógica de quadrantes implementada no módulo `Phase_To_Address`. Por esse motivo, a `ROM` recebe duas entradas de endereço, `sin_index` e `cos_index`, que permitem acessar as amostras para gerar as ondas senoidais e cossenoidais. O diagrama da `ROM` para uso no `NCO` está ilustrado na Figura 19:

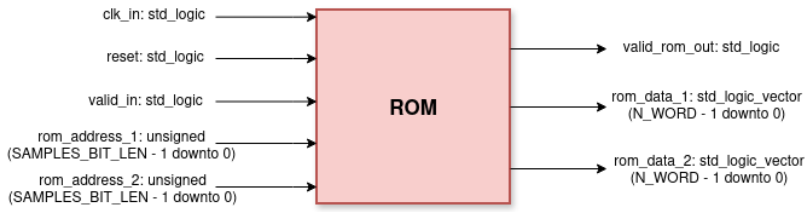


Figura 19: Diagrama do módulo `ROM` com os sinais de entrada e saída para o uso no `NCO`.

No caso da geração do pulso *sinc*, é utilizada apenas uma entrada de endereço (`rom_address`), já que na integração não é utilizado o módulo `Phase_to_address`. Assim, o módulo possui somente uma saída de amostra (`rom_data_out`), conforme ilustrado na Figura 20:

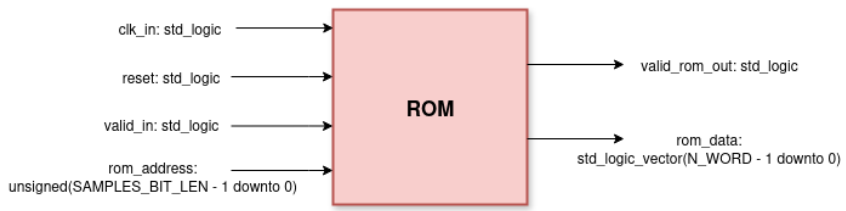


Figura 20: Diagrama do módulo `ROM` com os sinais de entrada e saída para uso no `Pulse_Gen`.

4.7 Metodologia de Validação dos Módulos em VHDL

A validação dos módulos desenvolvidos em VHDL foi realizada através de simulações no ambiente Vivado, com o objetivo de verificar o funcionamento lógico de cada bloco antes da etapa de síntese e integração no projeto completo. Essa abordagem permite identificar e corrigir erros de implementação de forma antecipada, garantindo que cada módulo atenda às suas especificações funcionais.

Para cada módulo desenvolvido, foi feito um código de *testbench* em VHDL. Esses *testbenches* foram feitos de forma que gerassem estímulos de entrada para observar os sinais de saída produzidos pelos módulos em teste. Cada *testbench* segue uma estrutura básica composta pela instanciação do módulo sob teste (Device Under Test (DUT)), geração dos sinais de *clock* e *reset*, e aplicação de estímulos controlados conforme os requisitos funcionais do componente. Durante as simulações, as formas de onda dos sinais foram observadas no simulador do Vivado, permitindo avaliar o comportamento temporal e lógico das saídas em resposta às entradas aplicadas.

No caso do módulo `NCO` (Gerador das portadoras), além da verificação das formas de onda senoidal e cossenoidal geradas no ambiente de simulação do Vivado, os dados de saída foram exportados em um arquivo no formato txt e depois reconstruídos em Python, onde calculou-se a FFT para verificar se o sinal possuía um pico na frequência esperada conforme o valor do parâmetro `tuning_word` era alterado. Essa etapa permitiu analisar se a frequência de saída variava de maneira linear em relação ao incremento de fase. Esse processo também foi realizado para o módulo `IQ_Modulation`, onde a saída (`real_pulse`) foi reconstruída em Python. O espectro de magnitude foi analisado para verificar se o pico espectral estava na frequência de portadora esperada e se a largura de banda correspondia ao valor de referência previamente calculado.

Além disso, foram gerados vetores de teste em Python para comparação com as saídas dos módulos que integravam os outros blocos, como o `Pulse_Gen`, responsável pela geração do pulso de RF e o `NCO`, que gera as portadoras senoidal e cossenoidal. O vetor de teste para o `Pulse_Gen` foi gerado como descrito na Seção 4.2, fazendo uma interpolação do pulso *sinc*, simulando o comportamento do `Clk_Decimator` com `division_factor` igual a 400. Para o `NCO`, o vetor de teste foi gerado usando a classe `NCO`. Essa estratégia permitiu verificar se as amostras obtidas nas simulações em VHDL correspondiam aos resultados previstos, validando o comportamento da implementação em alto nível e em hardware. A partir dessas validações individuais, foi possível avaliar o comportamento de cada módulo e do TopLevel `IQ_Modulation`. Após essa etapa, gerou-se o *bitstream* para a realização dos testes em bancada, com o objetivo de validar o pulso de RF modulado no osciloscópio com o uso da Red Pitaya.

4.8 Validação Experimental em Bancada com Osciloscópio

Para a validação experimental do sistema de geração dos pulsos de RF foram realizados testes em bancada utilizando a placa Red Pitaya (mencionada na Seção 4.1) e um osciloscópio. O objetivo foi verificar a geração de um pulso *sinc* modulado e na

frequência definida pelo parâmetro `tuning_word`.

Inicialmente, foi realizado o mapeamento dos pinos da FPGA para interfaceamento com os periféricos da placa. Esse processo utilizou o arquivo de constraints (.xdc), um formato aceito pelo software Vivado, disponibilizado pelo fabricante da Red Pitaya, que define a ligação entre os sinais lógicos do design VHDL e os pinos da FPGA, incluindo os conectores de entrada/saída, clocks e interfaces de comunicação.

Para os testes, foram sintetizadas no software Vivado diferentes variações do circuito apresentado nas seções anteriores, escolhendo-se distintos valores para o parâmetro `tuning_word`. Os valores selecionados para este teste foram: 41, 205, 410, 696 e 1024. Estes valores foram calculados previamente com base na Equação 5, considerando uma frequência de clock de 100 MHz e uma resolução de 12 bits para o acumulador de fase, o que resulta nas frequências esperadas de 1 MHz, 5 MHz, 10 MHz, 17 MHz e 25 MHz, respectivamente.

Para a montagem experimental, primeiramente a placa foi conectada a uma fonte de alimentação. A comunicação e a transferência de arquivos foram realizadas por meio da conexão USB entre a placa e o notebook, e da conexão Ethernet da placa à rede. O sinal de saída foi obtido do conector OUT1 (DAC) da Red Pitaya, utilizando um cabo SMA–BNC conectado a um dos canais de entrada do osciloscópio para visualização.

Com o *setup* montado, o *bitstream* (circuito sintetizado produzido a partir da modelagem em VHDL), gerado pelo Vivado (arquivo .bit) foi transferido para a Red Pitaya. Essa transferência foi realizada abrindo-se um terminal no computador e iniciando uma sessão com a placa via `sudo minicom`. Os arquivos *bitstream* foram copiados para o diretório da placa e posteriormente executados.

As imagens da tela do osciloscópio e os dados dos sinais adquiridos foram armazenados em um pendrive para posterior análise. O pulso de 17 MHz, de interesse para o protótipo, foi posteriormente reconstruído em um script em Python, no qual foi obtida a DFT para caracterizar o espectro do sinal e verificar sua largura de banda.

5 Resultados e Discussão

Neste capítulo são apresentados os resultados obtidos a partir das simulações e testes realizados com os módulos desenvolvidos, com o objetivo de validar o correto funcionamento do gerador de pulso de RF implementado em FPGA.

5.1 Largura de Banda do Pulso *Sinc*

O pulso *sinc* janelado com Hamming, gerado em Python e utilizado como vetor de teste para o bloco `Pulse_Gen`, foi submetido à análise espectral (via FFT) a fim de determinar de sua largura de banda no domínio da frequência. A Figura 21 apresenta o pulso *sinc* após o processo de interpolação, realizado para reproduzir o comportamento do módulo `Pulse_Gen` considerando um fator de divisão (`division_factor`) igual a 400.

Nesse caso, o sinal possui 100 mil amostras e duração total de 1 ms, assumindo um clock de 100 MHz.

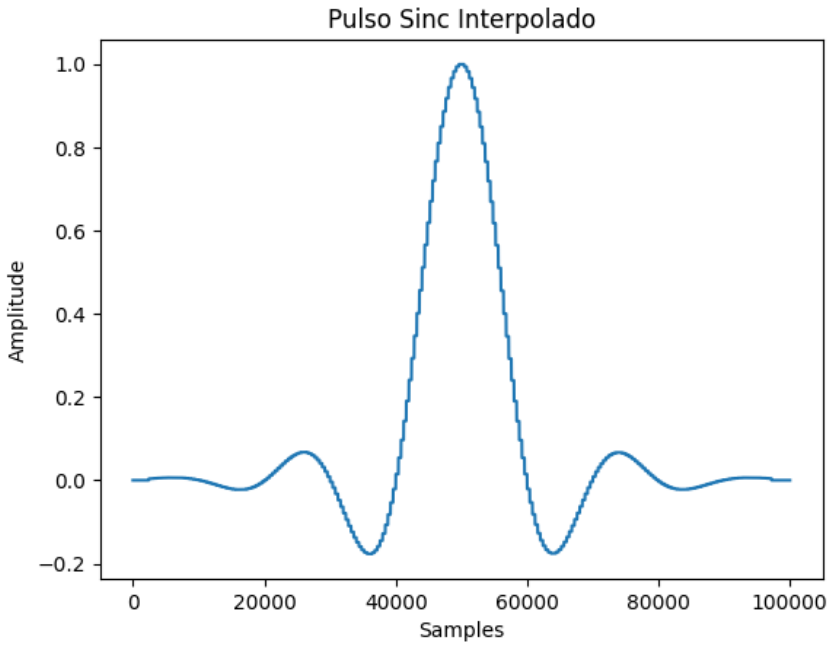


Figura 21: Pulso *sinc* com 100.000 amostras.

A Figura 22 mostra o espectro de magnitude do pulso *sinc* interpolado, onde é possível observar o espectro centrado em 0 Hz, característico da envoltória do sinal.

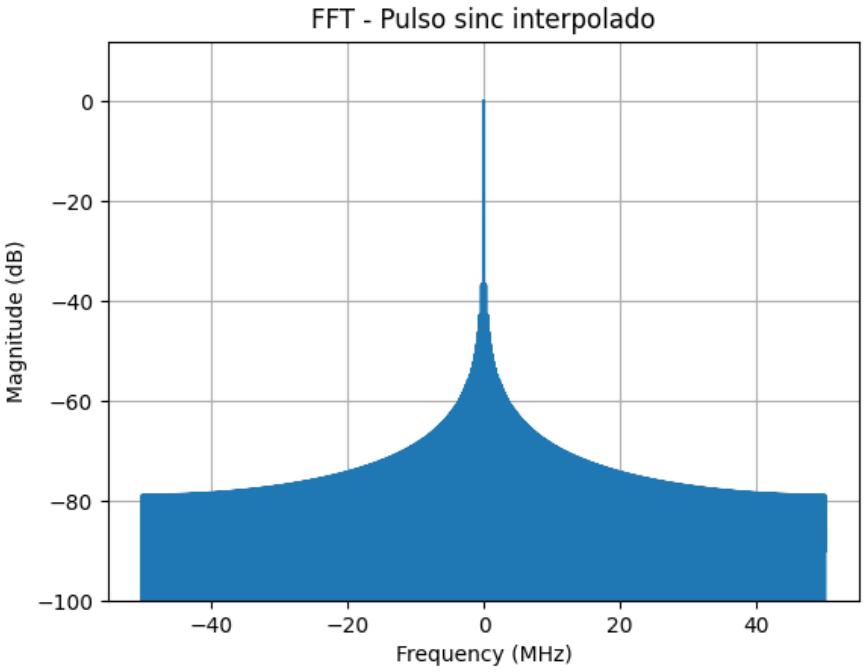


Figura 22: Espectro de magnitude (DFT) do pulso *sinc* interpolado.

A partir da análise do espectro e plotando um gráfico interativo, foi possível determinar a largura de banda da envoltória em banda base utilizando o critério de -10 dB em relação ao pico central. O valor obtido foi de aproximadamente 10,55 kHz, o que foi adotado como referência para as validações dos módulos `Pulse_Gen` e `IQ_Modulation`. Esse valor também foi utilizado como parâmetro comparativo para a largura de banda do pulso reconstruído em Python a partir dos dados adquiridos experimentalmente no osciloscópio.

5.2 Simulação do NCO em Python

Antes da implementação em VHDL, o comportamento do NCO foi validado por meio de uma simulação em Python, permitindo verificar a relação entre o incremento de fase, a frequência de saída e a precisão da síntese digital. O código desenvolvido utiliza a classe `NCO`, responsável por gerar as amostras das ondas senoidal e cossenoidal a partir do `tuning_word`. Foram testadas frequências de saída de 1, 5, 10, 17, 20 e 25 MHz, correspondendo aos valores de incremento de fase de 40, 204, 409, 696, 819 e 1024, respectivamente.

A Figura 23 apresenta as formas de onda senoidal e cossenoidal geradas para cada frequência. Observa-se que conforme o valor de `tuning_word` aumenta, a taxa de variação das amostras cresce proporcionalmente, resultando no aumento da frequência de saída.

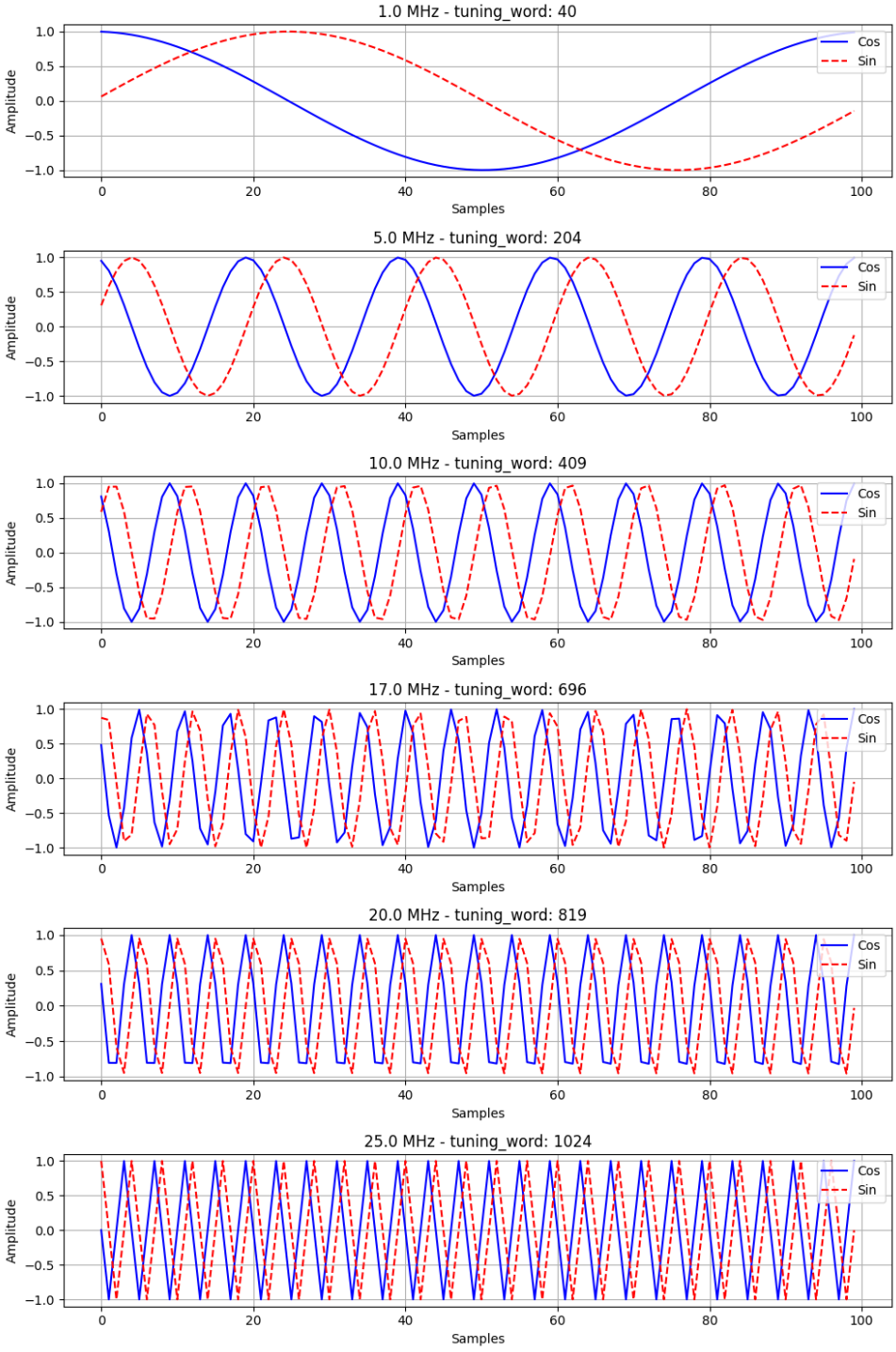


Figura 23: Formas de onda senoidal e cossenoidal geradas pelo `NCO` em diferentes frequências de saída.

A análise no domínio da frequência foi realizada por meio da DFT, permitindo verificar se as ondas estavam centradas na frequência esperada. A Figura 24 mostra os espectros de magnitude em dB obtidos para as ondas cossenoidais. Observa-se que cada

curva apresenta picos nas frequências correspondentes aos valores programados (1, 5, 10, 17, 20 e 25 MHz), confirmando a coerência entre a frequência de saída e o valor de controle aplicado.

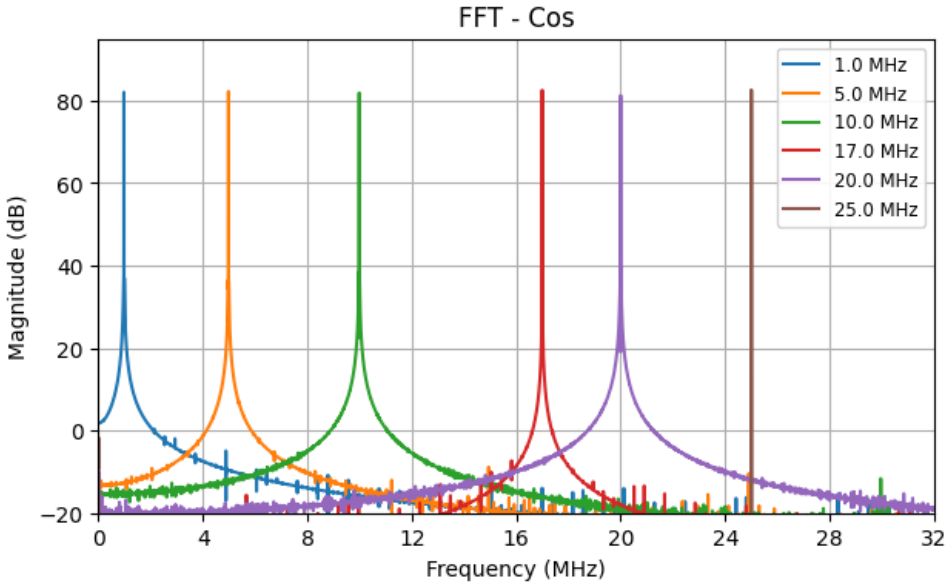


Figura 24: Espectros de magnitude das ondas cossenoidais geradas pelo `NCO` para diferentes valores de `tuning_word`.

Os resultados da simulação em Python validaram o funcionamento do NCO, evidenciando que o incremento de fase aplicado resulta em variações proporcionais na frequência de saída. Dessa forma, após a validação, foi possível fazer a implementação em VHDL.

5.3 Validação dos Módulos Básicos e Reutilizáveis

Esta seção apresenta a validação individual dos componentes fundamentais do sistema. Antes da integração nos blocos de maior complexidade, verificou-se o funcionamento lógico de cada módulo básico por meio de simulações. A seguir, são detalhados os resultados obtidos para os módulos `Clk_Decimator`, `Phase_Accumulator`, `Phase_to_Address` e `ROM`.

5.3.1 Clock Decimator (`Clk_Decimator`)

O módulo `Clk_Decimator` tem como função controlar a taxa de atualização das amostras do sistema, gerando um sinal de habilitação (`dec_clk_out`) a partir da divisão do clock principal. Esse sinal atua como referência temporal para o avanço dos endereços no módulo `Phase_Accumulator`, permitindo ajustar a duração total do pulso de RF conforme o fator de divisão configurado.

A Figura 25 apresenta o esquemático RTL do módulo gerado pelo Vivado. O

circuito implementa um contador controlado pelo clock e pelo sinal de reset, que gera o pulso `dec_clk_out` quando o valor do contador atinge o fator de divisão configurado.

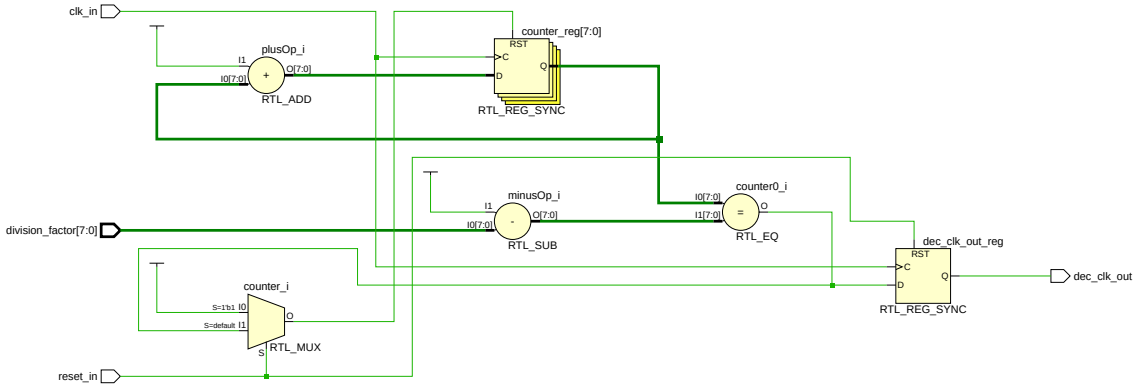


Figura 25: Esquemático RTL do módulo `Clk_Decimator` gerado pelo Vivado.

Para verificar o comportamento do módulo, foi desenvolvido um código de *test-bench*. O clock principal foi configurado com um período de 20 ns (50 MHz), e o fator de divisão (`division_factor`) foi alterado dinamicamente entre os valores 2, 5 e 10. A Figura 26 apresenta o resultado da simulação obtido no ambiente Vivado.

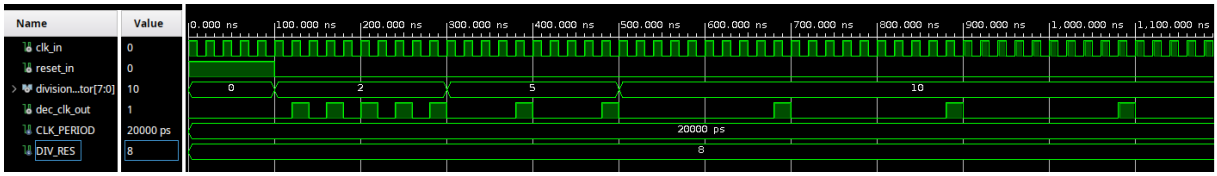


Figura 26: Resultado da simulação do módulo `Clk_Decimator` no Vivado.

Esses resultados validam o comportamento esperado do módulo, confirmando que o `Clk_Decimator` gera corretamente o pulso de habilitação de acordo com o fator de divisão configurado.

5.3.2 Phase Accumulator (`Phase_Accumulator`)

O módulo `Phase_Accumulator` é responsável por gerar os endereços que acessam as amostras armazenadas na `ROM`, controlando a taxa com que as amostras são percorridas. Esse módulo atua como um contador que acumula incrementos definidos pelo valor de `tuning_word`, permitindo ajustar a frequência da portadora digital gerada.

A Figura 27 apresenta o esquemático RTL do módulo, gerado automaticamente pelo Vivado a partir do código em VHDL. O circuito é composto por registradores, mul-

tiplexadores e operadores aritméticos de soma e módulo, que implementam o comportamento de um acumulador controlado pelo sinal de clock e de habilitação. O sinal `dec_clk_in`, proveniente do módulo `Clk_Decimator`, determina o instante em que o valor do acumulador é atualizado.

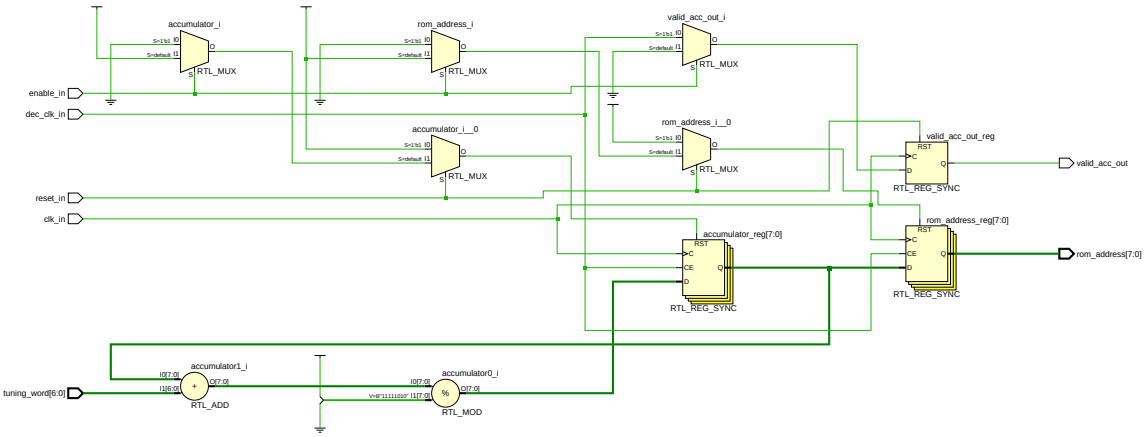


Figura 27: Esquemático RTL do módulo `Phase_Accumulator` gerado pelo Vivado.

No testbench utilizado para validação, o clock principal foi definido com um período de 20 ns (50 MHz), e o `tuning_word` foi configurado com um incremento de fase igual a 4. O sinal `dec_clk_in` foi utilizado para controlar o instante de atualização do acumulador, simulando o comportamento sincronizado com o clock desacelerado fornecido pelo módulo `Clk_Decimator`.

A Figura 28 apresenta o resultado da simulação. É possível obervar que o endereço de saída (`rom_address`) é incrementado de acordo com o valor de `tuning_word`, o endereço avança em passos de quatro posições a cada pulso de `dec_clk_in`. Foram testados diferentes valores de `tuning_word` de forma que ao aumentar esse parâmetro, a taxa de variação de `rom_address` também aumenta proporcionalmente, demonstrando o correto funcionamento do acumulador e sua capacidade de controlar a frequência de leitura da memória.

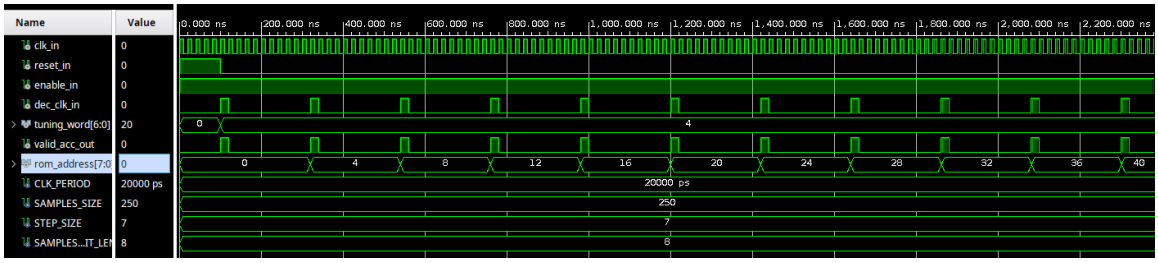


Figura 28: Resultado da simulação do módulo `Phase_Accumulator` no Vivado.

Por meio dessa simulação, foi possível validar o comportamento esperado do módulo,

verificando que `rom_address` é incrementado corretamente a cada pulso do `dec_clk_in` e pelo valor de `tuning_word`. O sinal `valid_acc_out` é ativado corretamente sempre que uma nova atualização do acumulador ocorre, servindo como sinal de sincronização para os módulos subsequentes.

5.3.3 Phase to Address (`Phase_to_Address`)

O módulo `Phase_to_Address` é responsável por converter o valor acumulado de fase, gerado pelo módulo `Phase_Accumulator`, em endereços equivalentes aos quatro quadrantes de uma senoide completa, a partir de uma `ROM` que armazena apenas um quarto de período da onda senoidal. Dessa forma, o módulo aplica operações de espelhamento e inversão de sinal sobre o mesmo conjunto de amostras, otimizando o uso de memória sem comprometer a fidelidade da forma de onda gerada.

Além de reconstruir a senoide completa, o módulo também gera os endereços correspondentes ao sinal cosseno, adicionando uma defasagem de 90° ao endereço de fase. Assim, são produzidos simultaneamente os índices de seno (`sin_index`) e cosseno (`cos_index`), juntamente com os sinais de controle `sin_negate` e `cos_negate`, que indicam a inversão de sinal necessária para cada quadrante.

A Figura 29 apresenta o esquemático RTL do módulo gerado pelo Vivado. Observa-se a presença de operadores de soma e subtração (`RTL_ADD` e `RTL_SUB`), multiplexadores (`RTL_MUX`) e comparadores (`RTL_LT`), que implementam a lógica de seleção de quadrante e a defasagem de 90° entre os endereços de seno e cosseno. Os registradores de saída (`RTL_REG_SYNC`) garantem a sincronização dos sinais `sin_index`, `cos_index`, `sin_negate`, `cos_negate` e `valid_phase_out` com o clock principal.

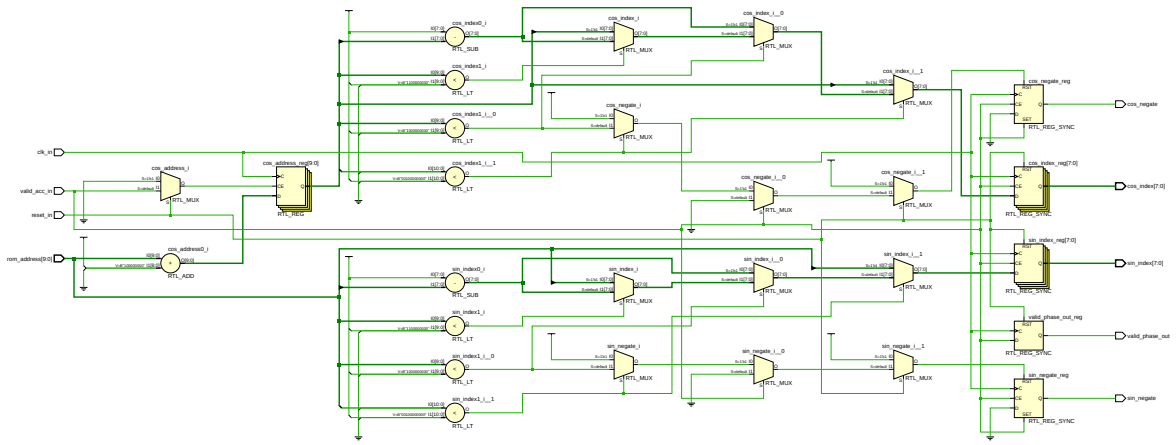


Figura 29: Esquemático RTL do módulo `Phase_to_Address` gerado pelo Vivado.

A simulação foi realizada por meio de um *testbench*, que varreu sequencialmente todos os endereços de fase (`rom_address`) de 0 a 4095. O sinal `valid_acc_in` permaneceu

ativo durante a execução, permitindo a atualização contínua dos endereços de seno e cosseno. As Figuras 30, 31 e 32 apresentam os resultados da simulação obtidos no ambiente Vivado, correspondentes a diferentes regiões do ciclo da onda.

Na Figura 30, correspondente ao primeiro quadrante, observa-se que o sinal `sin_index` cresce de forma direta, realizando a leitura sequencial dos endereços. Já o `cos_index` inicia com os endereços percorridos de maneira decrescente, esse comportamento reflete a defasagem de 90° entre as duas ondas. Ambos os sinais de negação permanecem inativos neste quadrante.

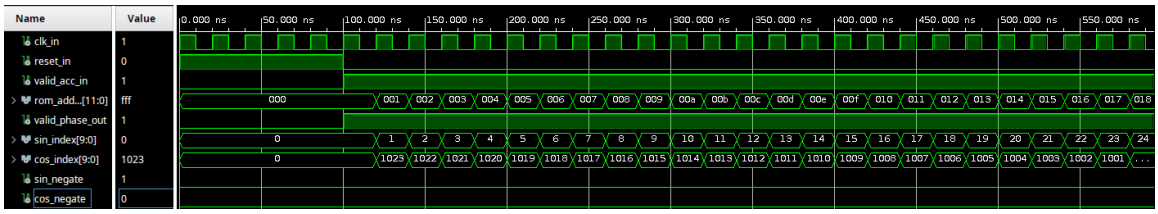


Figura 30: Resultado da simulação do módulo `Phase_to_Address` no primeiro quadrante.

A Figura 31, ilustra a transição entre o primeiro e o segundo quadrante. Nessa região, os endereços gerados passam a ser invertidos, o `sin_index` passa a decrescer (leitura espelhada do seno) enquanto o `cos_index` assume contagem crescente. Além disso, o sinal `cos_negate` está ativo, indicando que as amostras de cosseno correspondentes devem ser invertidas em sinal.

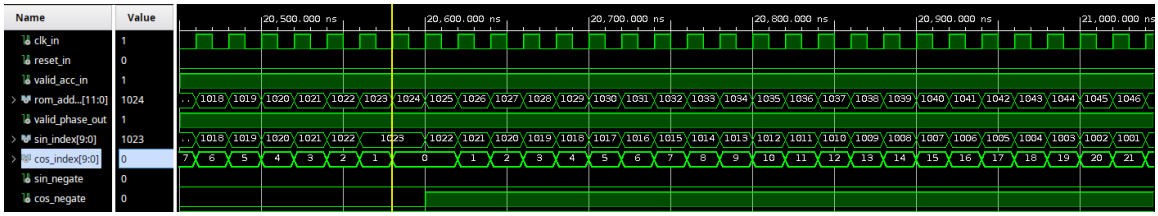


Figura 31: Resultado da simulação do módulo `Phase_to_Address` na transição do primeiro para o segundo quadrante.

A Figura 32 corresponde a transição entre o segundo e o terceiro quadrante. É possível observar que o `sin_index` começa a crescer novamente, mas o sinal `sin_negate` é ativado, indicando que as amostras devem ser negativas. O `cos_index` começa a decrescer e o sinal `cos_negate` permanece ativo.

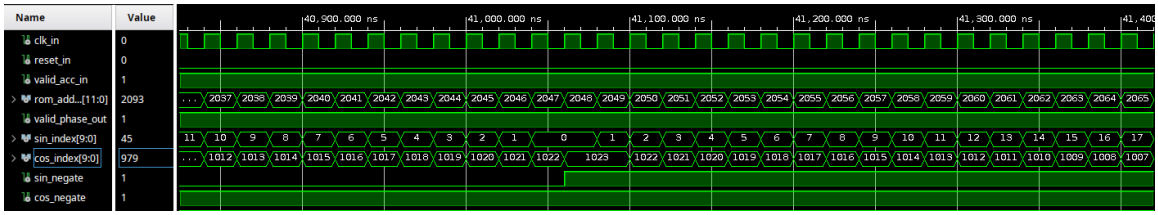


Figura 32: Resultado da simulação do módulo `Phase_to_Address` na transição do segundo para o terceiro quadrante.

Por fim, a Figura 33 mostra a transição do terceiro para o quarto quadrante, no qual o `sin_index` volta a apresentar leitura decrescente e o `cos_index` crescente, o sinal `sin_negate` permanece ativo e o `cos_negate` é desativado.

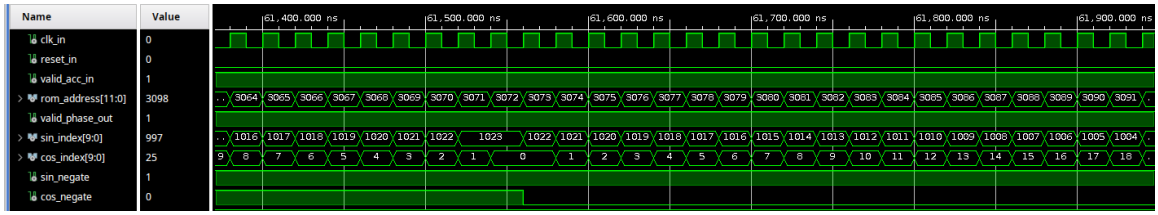


Figura 33: Resultado da simulação do módulo `Phase_to_Address` na transição do terceiro para o quadrante.

Os resultados da simulação confirmam o funcionamento esperado do módulo, que realiza corretamente o mapeamento dos quatro quadrantes e a defasagem entre seno e cosseno. A alternância entre leitura direta e espelhada, combinada com a inversão de sinal, assegura a geração precisa das formas de onda completas. Observa-se também que o sinal `valid_phase_out` é acionado sempre que ocorre uma atualização dos endereços, indicando a validade das saídas geradas pelo módulo.

5.3.4 ROM (Read-Only Memory)

O módulo `ROM` tem como função armazenar e disponibilizar as amostras utilizadas pelo sistema digital, tanto para a geração das portadoras quanto para a reprodução do pulso de RF. O conteúdo da memória é carregado a partir de um arquivo externo de texto, contendo os valores quantizados em ponto fixo. Nesta etapa, buscou-se validar a leitura correta e saída dos dados, garantindo que o conteúdo do arquivo externo fosse interpretado e disponibilizado conforme esperado.

A Figura 34 apresenta o esquemático RTL do módulo, gerado pelo Vivado. Observa-se que o circuito é composto por um bloco de memória e registradores de saída, sincronizados com o sinal de clock. O módulo possui duas portas de saídas independentes (`rom_data_1` e `rom_data_2`), permitindo o acesso simultâneo a dois endereços distintos, recurso utilizado no bloco de integração `NCO`.

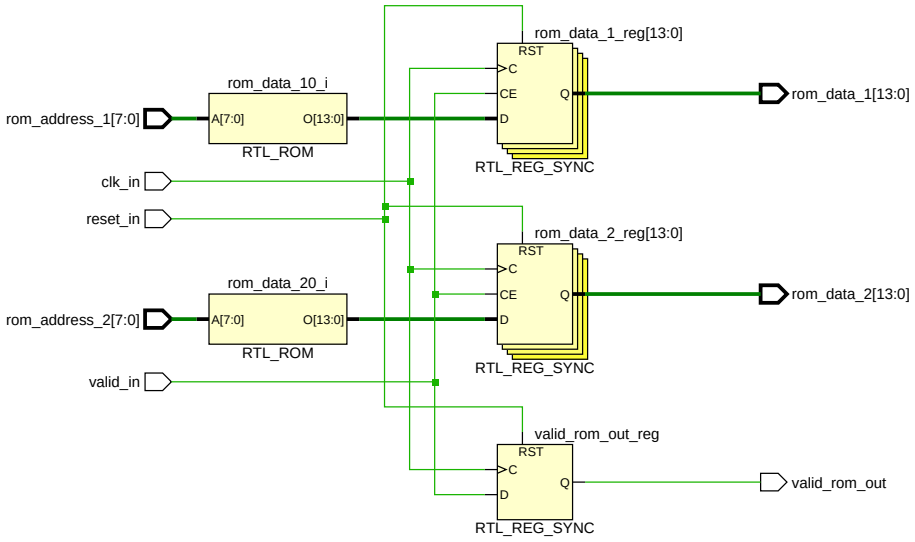


Figura 34: Esquemático RTL do módulo `ROM` gerado pelo Vivado.

Para validação, foi desenvolvido um código de testbench, configurado para ler o arquivo `pulse_sinc_hamming_testvec.txt`, que contém os valores de um pulso do tipo *sinc*. Durante a simulação, os endereços de leitura foram incrementados sequencialmente de 0 até 249, enquanto as saídas `rom_data_1` e `rom_data_2` apresentaram os valores correspondentes às amostras armazenadas no arquivo.

A Figura 35 mostra o resultado da simulação, onde é possível observar o comportamento do sinal reconstruído a partir das amostras no arquivo txt lidos pelo módulo. O formato da onda confirma que o conteúdo do arquivo foi lido e gerado corretamente.

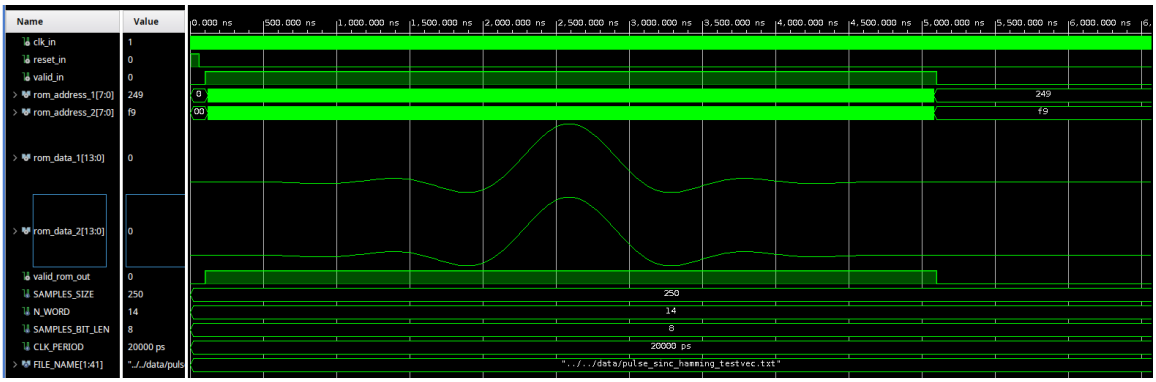


Figura 35: Resultado da simulação do módulo `ROM`, mostrando a leitura sequencial das amostras do pulso.

Durante o teste, ambas as saídas da `ROM` (`rom_data_1` e `rom_data_2`) apresentaram o mesmo comportamento, validando o funcionamento correto das duas portas de leitura.

5.4 Validação dos Módulos de Integração

Após a validação dos blocos básicos, realizou-se à validação dos módulos de maior complexidade, responsáveis pela geração e controle do pulso de RF. As simulações a seguir demonstram o funcionamento integrado dos módulos `Pulse_Gen`, `NCO` e `IQ_Modulation`.

5.4.1 Gerador de Pulso (`Pulse_Gen`)

O módulo `Pulse_Gen` integra os blocos `Clk_Decimator`, `Phase_Accumulator` e `ROM`, sendo responsável por controlar a geração digital do pulso de RF. O sinal de entrada `clk_in` é desacelerado pelo bloco `Clk_Decimator`, de acordo com o fator de divisão definido em `division_factor`, determinando a taxa de leitura das amostras armazenadas. Em seguida, o bloco `Phase_Accumulator` gera os endereços de leitura da `ROM`, sincronizados com o sinal de clock desacelerado, enquanto o módulo `ROM` fornece as amostras do pulso a partir de um arquivo de texto externo.

A Figura 36 apresenta o esquemático RTL do módulo, no qual é possível identificar a interligação entre os blocos de controle e geração para a geração do pulso *sinc*.

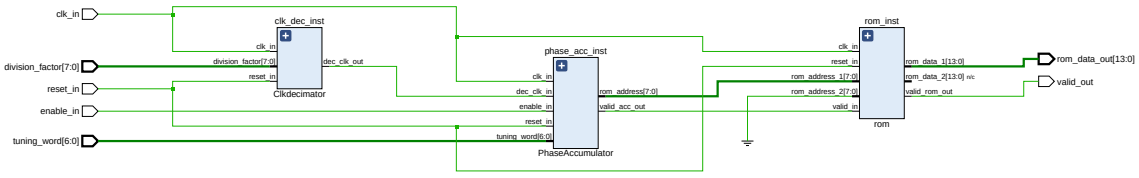


Figura 36: Esquemático RTL do módulo `Pulse_Gen` com integração dos blocos `Clk_Decimator`, `Phase_Accumulator` e `ROM`.

A simulação foi realizada utilizando um fator de divisão de 400, de modo que o módulo `ROM` percorre as 250 amostras do arquivo de entrada, resultando em um pulso com duração aproximada de 1 ms. A Figura 37 mostra o resultado da simulação, onde observa-se a forma do pulso do tipo *sinc* sendo reproduzido no sinal `rom_data_out`. O valor destacado em amarelo, na parte superior direita da imagem, mostra a duração do pulso com um valor de aproximadamente 1.004,06 μs. Desconsiderando o tempo inicial para o processo de estímulos para verificar o reset, o valor medido corresponde a cerca de 1 ms, confirmando a duração esperada para o pulso.

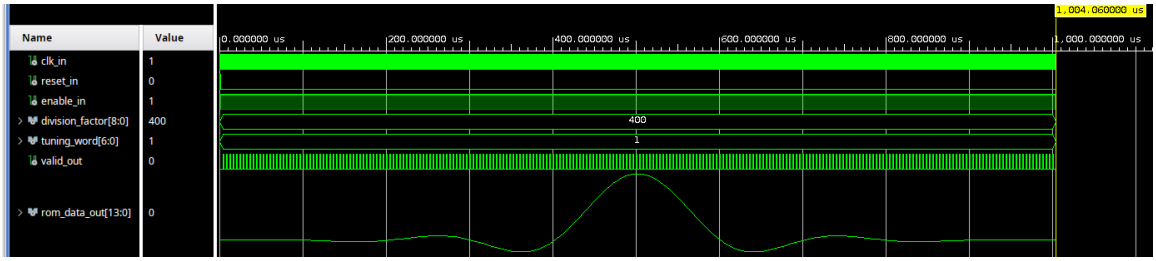


Figura 37: Resultado da simulação do módulo `Pulse_Gen` mostrando a geração do pulso *sinc* com duração controlada.

Além disso, o sinal gerado foi comparado com um vetor de referência obtido a partir da simulação em Python, onde o mesmo pulso foi interpolado para possuir a mesma quantidade de amostras da simulação em VHDL. A Figura 38 mostra o pulso gerado (`rom_data_out`) e o pulso de referência (`s_ref_pulse`). O sinal (`dif_pulse`) indica a diferença entre o sinal de saída e o vetor de teste, que permanece zero ao longo de toda a duração do pulso.

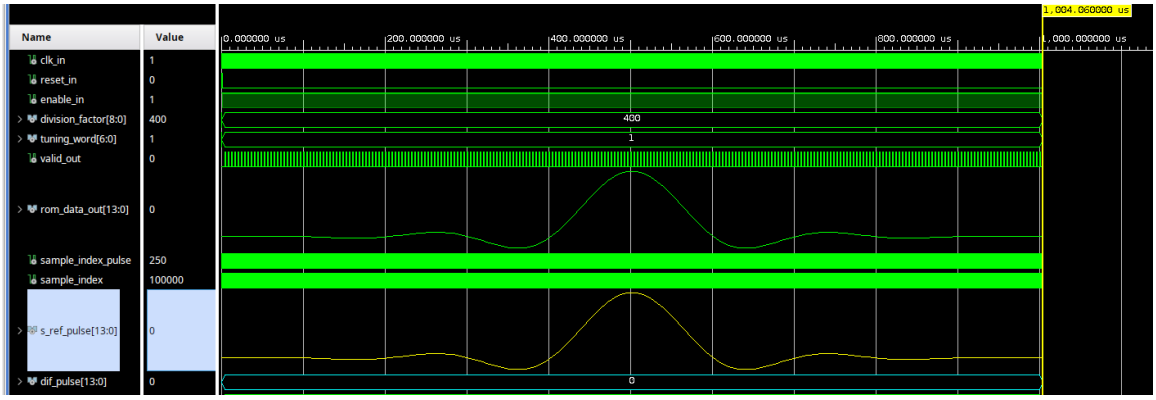


Figura 38: Comparação entre o pulso gerado em `VHDL` e o vetor de teste obtido em Python.

Os resultados demonstram a correta reprodução da envoltória do pulso, validando a leitura sequencial dos dados da `ROM` controlada pelo divisor de clock. A diferença igual a zero entre o pulso gerado e o vetor de referência mostra a correta implementação em `VHDL` em comparação com a simulação realizada em ambiente Python.

5.4.2 Gerador das Portadoras (`NCO`)

O módulo `NCO` é responsável pela geração das portadoras senoidal e cossenoidal. Ele integra os blocos `Clk_Decimator`, `Phase_Accumulator`, `Phase_To_Address` e `ROM`, operando de forma semelhante ao módulo `Pulse_Gen`, porém com a adição da lógica responsável por mapear os endereços de fase para as regiões correspondentes das ondas senoidal e cossenoidal.

A Figura 39 apresenta o esquemático RTL do módulo, mostrando a interligação entre os blocos responsáveis pelo controle de fase, endereçamento e leitura das amostras. Observa-se que a estrutura é semelhante ao diagrama mostrado na Figura 15.

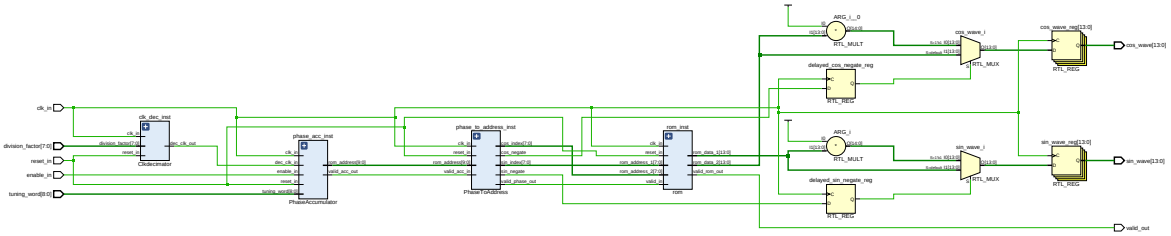


Figura 39: Esquemático RTL do módulo `NCO` com integração dos blocos `Clk_Decimator`, `Phase_Accumulator`, `Phase_to_Address` e `ROM`.

A Figura 40 mostra a simulação do módulo com `tuning_word` igual a 40, correspondente à geração de uma frequência de aproximadamente 1 MHz. Observa-se que as formas de onda senoidal e cossenoidal são geradas de forma contínua e com defasagem de 90°. O sinal `valid_out` é ativado sempre que uma nova amostra é disponibilizada na saída, indicando o instante em que os dados estão válidos.

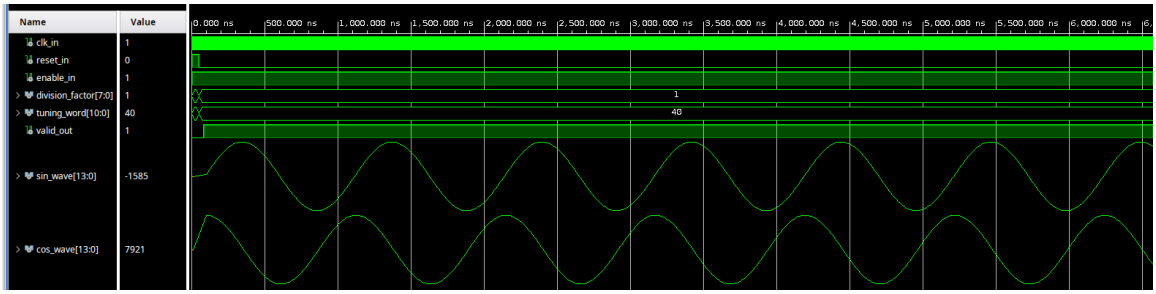


Figura 40: Simulação do módulo `NCO` mostrando a defasagem de 90° entre as ondas senoidal e cossenoidal.

A Figura 41 apresenta a variação das formas de onda conforme o valor de `tuning_word` é alterado durante a simulação. É possível observar o aumento da frequência de saída à medida que o valor de `tuning_word` é alterado, confirmando o comportamento esperado de um oscilador digital controlado numericamente.

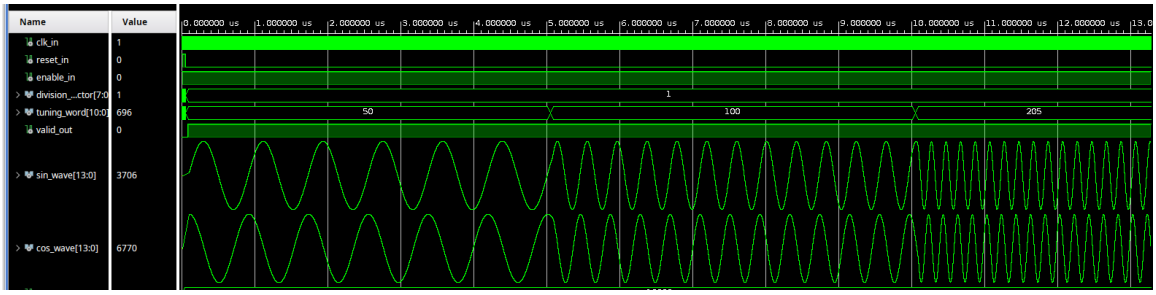


Figura 41: Variação da frequência das portadoras conforme a mudança de `tuning_word`.

Para validar a coerência das saídas geradas, foram utilizados vetores de testes gerados em Python. A comparação entre as ondas senoidais e o vetor de teste é apresentada na Figura 42, em que o sinal `dif_sin` representa a diferença entre os valores obtidos na simulação em VHDL e o vetor de referência. O modo de exibição analógico foi utilizado para permitir uma melhor visualização da diferença entre os sinais, considerando o grande número de amostras. Observa-se que a diferença se mantém próxima de zero ao longo de todo o ciclo, indicando a equivalência entre os sinais.

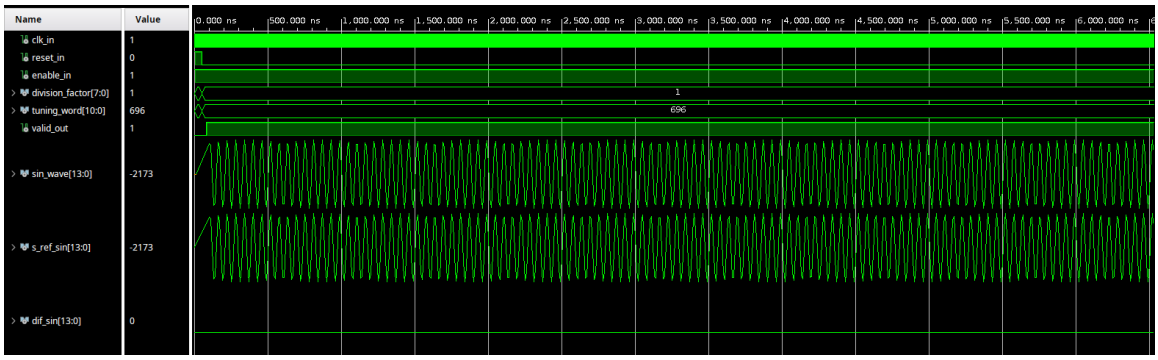


Figura 42: Comparação entre a onda senoidal gerada em VHDL e o vetor de teste gerado em Python.

A Figura 43 apresenta o mesmo procedimento para o cosseno, onde também é possível observar uma diferença próxima de zero entre o sinal gerado e o vetor de referência.

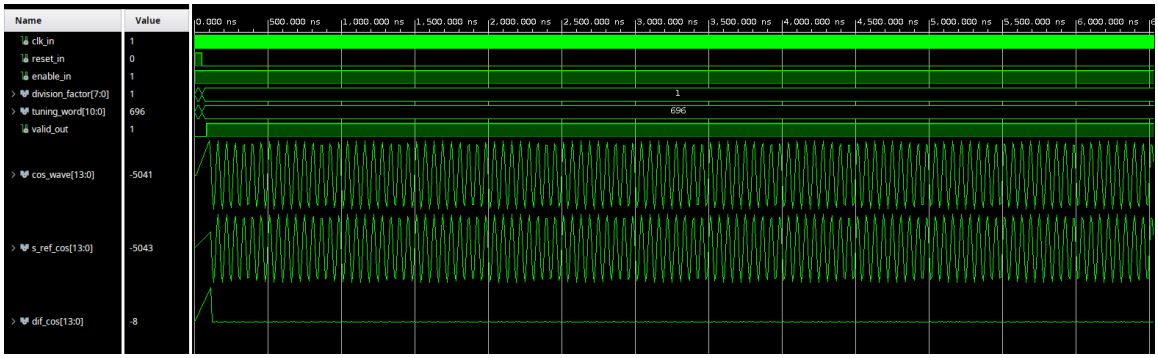


Figura 43: Comparação entre a onda cossenoidal gerada em `VHDL` e o vetor de teste gerado em Python.

Por fim, foram gerados arquivos txt com as amostras da saída `cos_wave` para diferentes valores de tuning_word (40, 204, 410, 696, 819, 1024) correspondendo as frequências de 1, 5, 10, 17, 20 e 25 MHz. Os dados foram reconstruídos em Python para análise, a Figura 44 ilustra o plot de cada arquivo, com os diferentes valores de frequência gerados.

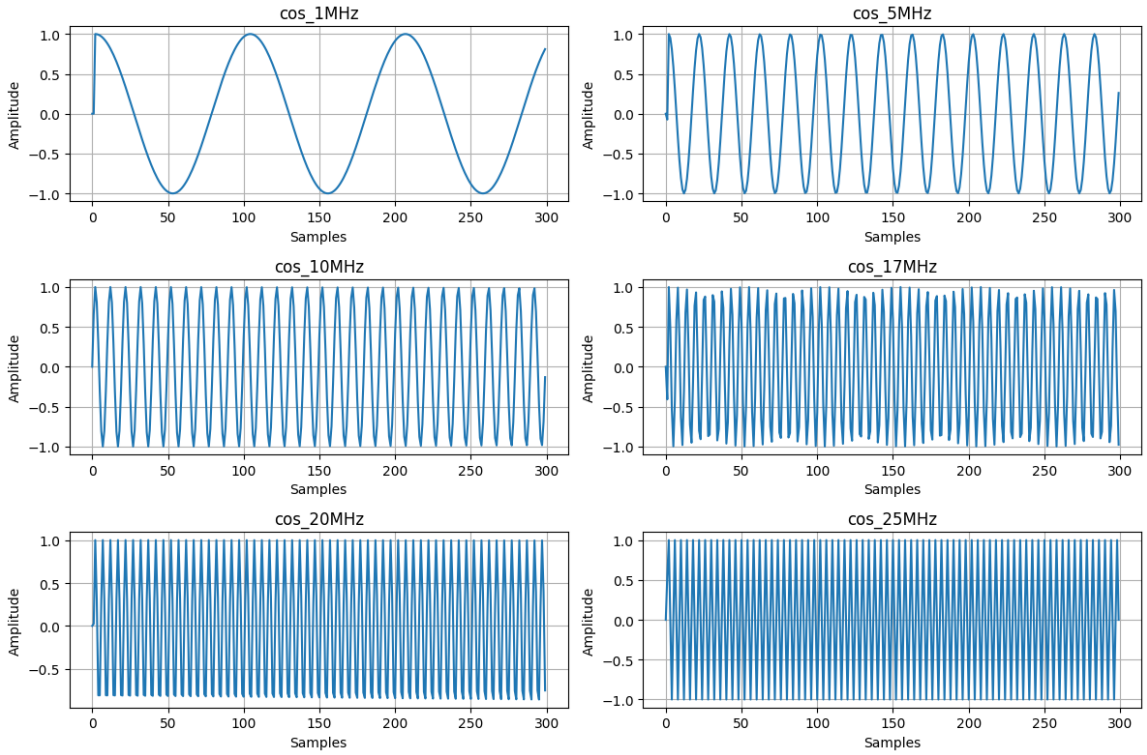


Figura 44: Formas de onda cossenoidais reconstruídas em Python para diferentes valores de `tuning_word`.

Para verificar se as saídas estavam sendo geradas na frequência correspondente ao valor de `tuning_word`, calculou-se a FFT para cada um dos arquivos contendo as amostras

(Figura 45). Os espectros obtidos mostram que os picos de frequência coincidem com as frequências esperadas para cada valor de `tuning_word`.

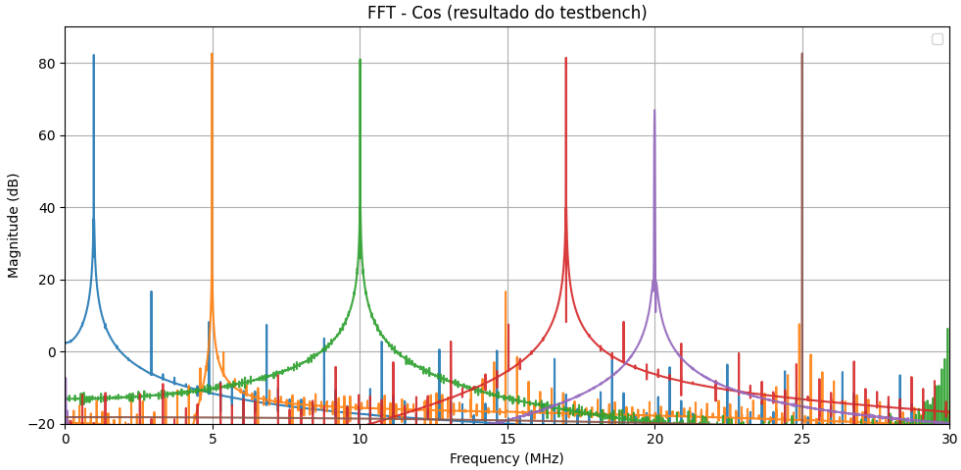


Figura 45: Espectros de magnitude das ondas cossenoidais geradas pelo `NCO`, mostrando picos nas frequências correspondentes aos valores de `tuning_word`.

O módulo `NCO` apresentou o comportamento de acordo com as simulações, validando a aplicação da técnica do NCO para o controle digital da frequência das portadoras conforme a alteração do parâmetro `tuning_word`. Também foi possível observar a correta geração das ondas a partir da otimização que utiliza apenas um quarto de onda do período do seno, reduzindo o uso da memória.

5.4.3 Modulação IQ (`IQ_Modulation`)

O módulo `IQ_Modulation` tem como função integrar o pulso gerado pelo bloco `Pulse_Gen` com as portadoras senoidal e cossenoidal provenientes do bloco `NCO`, realizando a etapa de modulação digital do sinal de RF. A arquitetura foi desenvolvida de modo a validar a integração entre os módulos responsáveis pela geração do pulso e das portadoras, permitindo verificar o comportamento do sistema de transmissão de forma completa.

O módulo realiza a multiplicação ponto a ponto do pulso pelas duas portadoras, resultando em duas saídas, o `real_pulse`, correspondente à multiplicação do pulso pelo cosseno, e o `imaginary_pulse`, correspondente à multiplicação do pulso pelo seno.

O esquema RTL do módulo é mostrado na Figura 46, onde é possível identificar as instâncias dos blocos `Pulse_Gen` e `NCO` e a lógica de multiplicação das saídas.

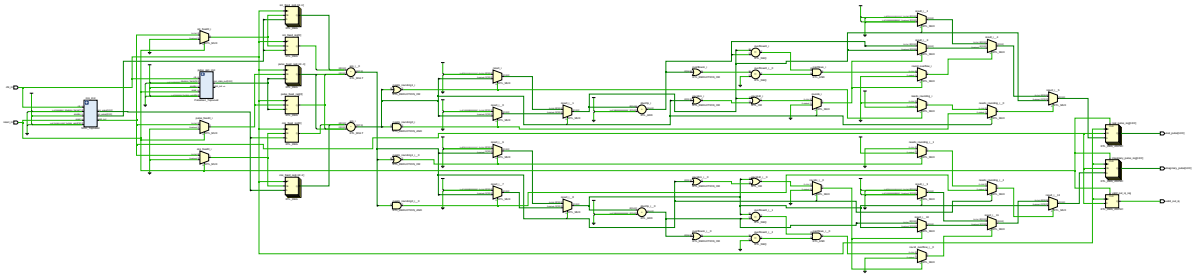


Figura 46: Esquemático RTL do módulo `IQ_Modulation` com integração dos blocos `Pulse_Gen` e `NCO`.

A Figura 47 apresenta o resultado da simulação, mostrando as duas saídas do módulo, o pulso modulado pela portadora cossenoidal (`real_pulse`) e o pulso modulado pela portadora senoidal (`imaginary_pulse`). É possível observar a envoltória do pulso *sinc* controlando a amplitude das portadoras, confirmando o funcionamento da modulação. O sinal `valid_out_iq` é ativado sempre que novas amostras válidas são disponibilizadas na saída.

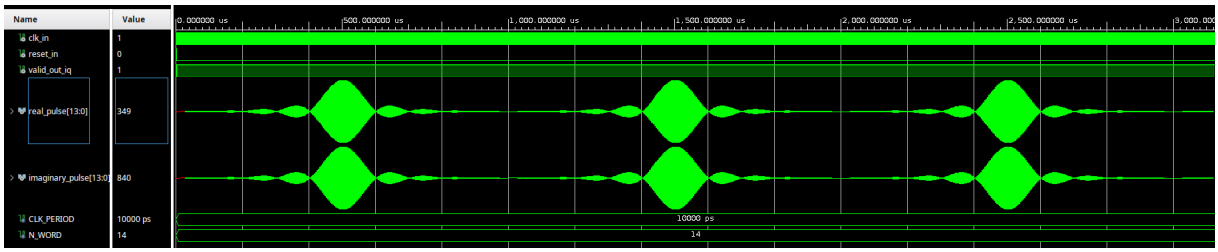


Figura 47: Simulação do módulo `IQ_Modulation` mostrando as saídas `real_pulse` e `imaginary_pulse`.

A Figura 48 mostra um recorte ampliado da simulação, destacando a portadora oscilando dentro da envoltória do pulso. O resultado demonstra a sincronização entre as saídas do `NCO` e o sinal do `Pulse_Gen`, evidenciando o funcionamento correto da multiplicação e da integração entre os blocos.

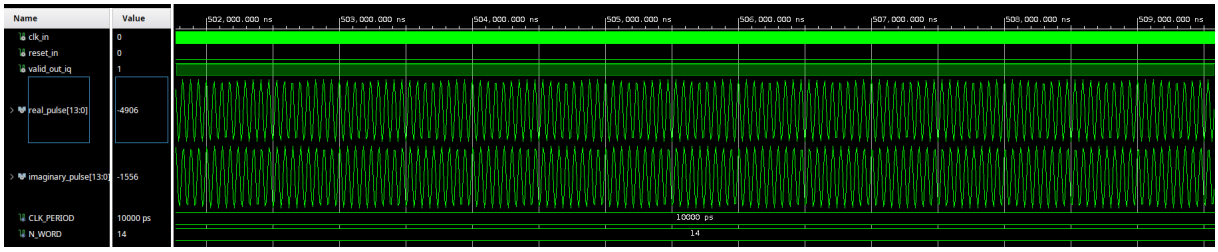


Figura 48: Simulação com zoom da portadora modulada pela envoltória do pulso.

Para validar o resultado da modulação, os valores da saída `real_pulse` foram exportados em formato de texto e reconstruídos em ambiente Python. A Figura 49 mostra o pulso modulado reconstruído.

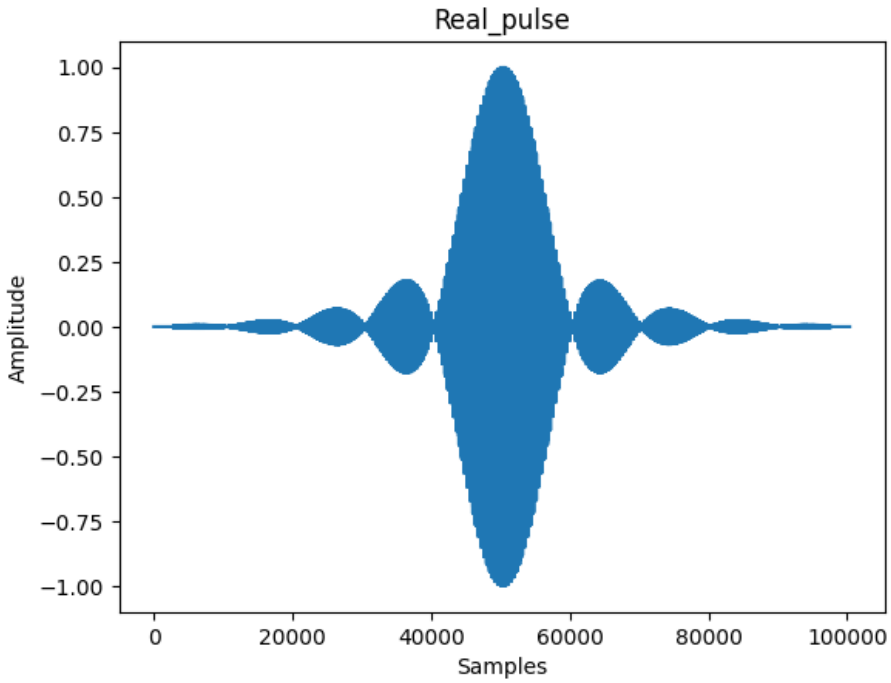


Figura 49: Pulso reconstruído em Python a partir dos dados da saída `real_pulse`.

A Figura 50 apresenta espectro de magnitude do sinal reconstruído, na qual observa-se um pico centrado em aproximadamente 17 MHz, valor correspondente à frequência da portadora definida pelo `tuning_word` igual a 696. Esse resultado confirma que o processo de modulação foi corretamente implementado e que a frequência gerada está de acordo com o esperado para o clock de 100 MHz utilizado na simulação.

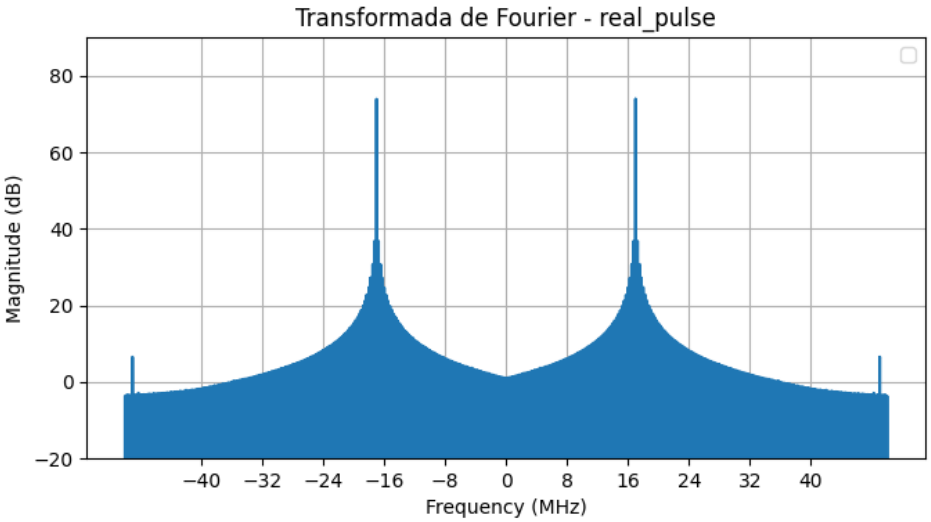


Figura 50: Espectro em frequência do pulso modulado (`real_pulse`), mostrando um pico em aproximadamente 17 MHz.

Analisando o espectro do pulso de RF modulado, centrado em aproximadamente 17 MHz, foi possível medir a largura de banda gerada, que foi de aproximadamente 10,67 kHz, valor próximo ao obtido para o pulso interpolado descrito na Seção 5.1 de 10,55 kHz. Esse resultado demonstra que o processo de modulação funcionou da forma esperada, fazendo corretamente a multiplicação da envoltória pelas portadoras, resultando em pulsos modulados nas frequências definidas por `tuning_word`.

Por fim, o módulo `IQ_Modulation` foi sintetizado no ambiente Vivado para avaliar a utilização de recursos lógicos e de memória da FPGA Xilinx Zynq 7020 (presente na placa Red Pitaya). A Figura 51 mostra que os resultados de síntese apresentaram uma baixa porcentagem dos recursos lógicos e de processamento da FPGA.

Utilization **Post-Synthesis** | Post-Implementation

Graph | **Table**

Resource	Estimation	Available	Utilization %
LUT	196	17600	1.11
FF	91	35200	0.26
BRAM	1	60	1.67
DSP	2	80	2.50
IO	31	100	31.00
BUFG	1	32	3.13

Figura 51: Relatório de Utilização de Recursos da FPGA (Pós-Síntese) para o módulo `IQ_Modulation`. As siglas representam os principais blocos internos da FPGA: LUT (Look-Up Table), FF (Flip-Flop), BRAM (Block RAM), DSP (Digital Signal Processor), IO (entradas e saídas) e BUFG (Buffer de clock).

5.5 Validação Experimental com Osciloscópio

Após a validação em simulação, foi realizada a verificação experimental do sistema implementado na FPGA. A Figura 52 apresenta a montagem experimental utilizada, composta pela placa Red Pitaya conectada ao osciloscópio, permitindo a visualização dos pulsos modulados gerado pelo sistema digital.

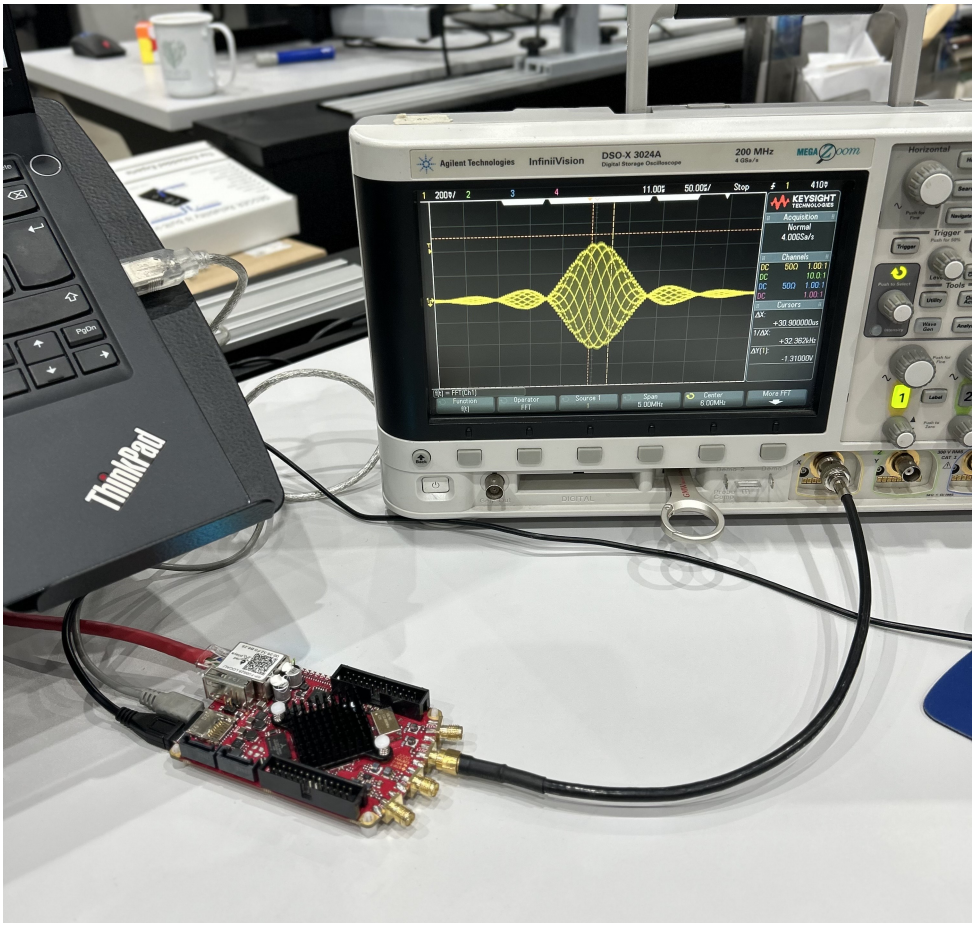


Figura 52: Montagem experimental utilizada para validação do pulso gerado na FPGA.

O experimento consistiu na observação dos pulsos modulados para diferentes valores de `tuning_word`, correspondentes às frequências de 1, 5, 10, 17, 20 e 25 MHz, considerando um clock de 100 MHz. A Figura 53 mostra os pulsos modulados observados no osciloscópio para essas configurações.

Os sinais exibem o formato característico do pulso *sinc* modulando a portadora, sendo possível ajustar a escala no osciloscópio e visualizar a portadora oscilando sob a envoltória do pulso. Essa observação confirma o funcionamento adequado da etapa de modulação implementada na FPGA.

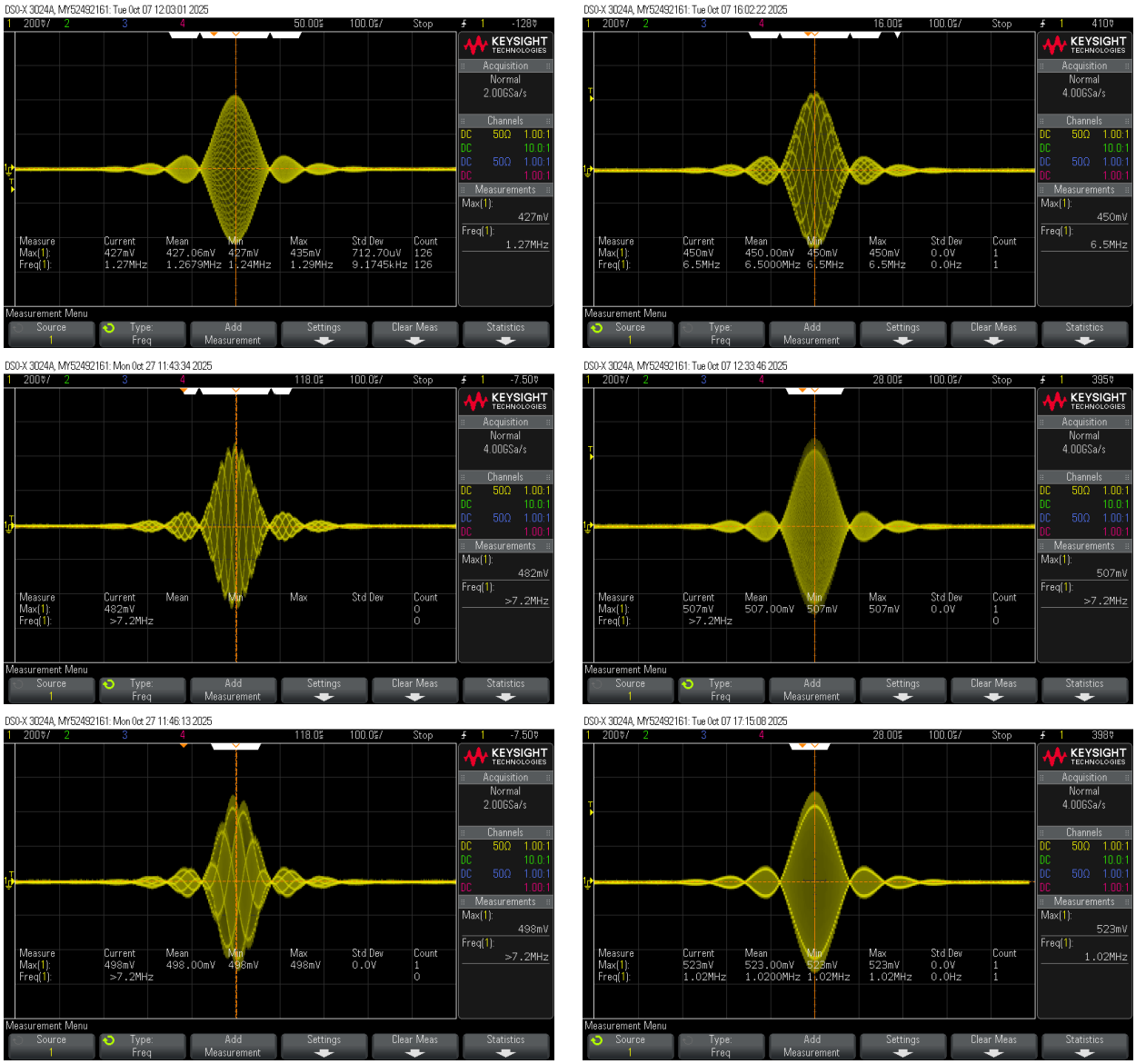


Figura 53: Pulsos modulados observados no osciloscópio para os valores de `tuning_word` de 41, 205, 410, 696, 819, 1024.

A Figura 54 apresenta os espectros (calculados via função FFT do osciloscópio) dos pulsos observados, onde é possível identificar o valor do pico de frequência em cada caso.

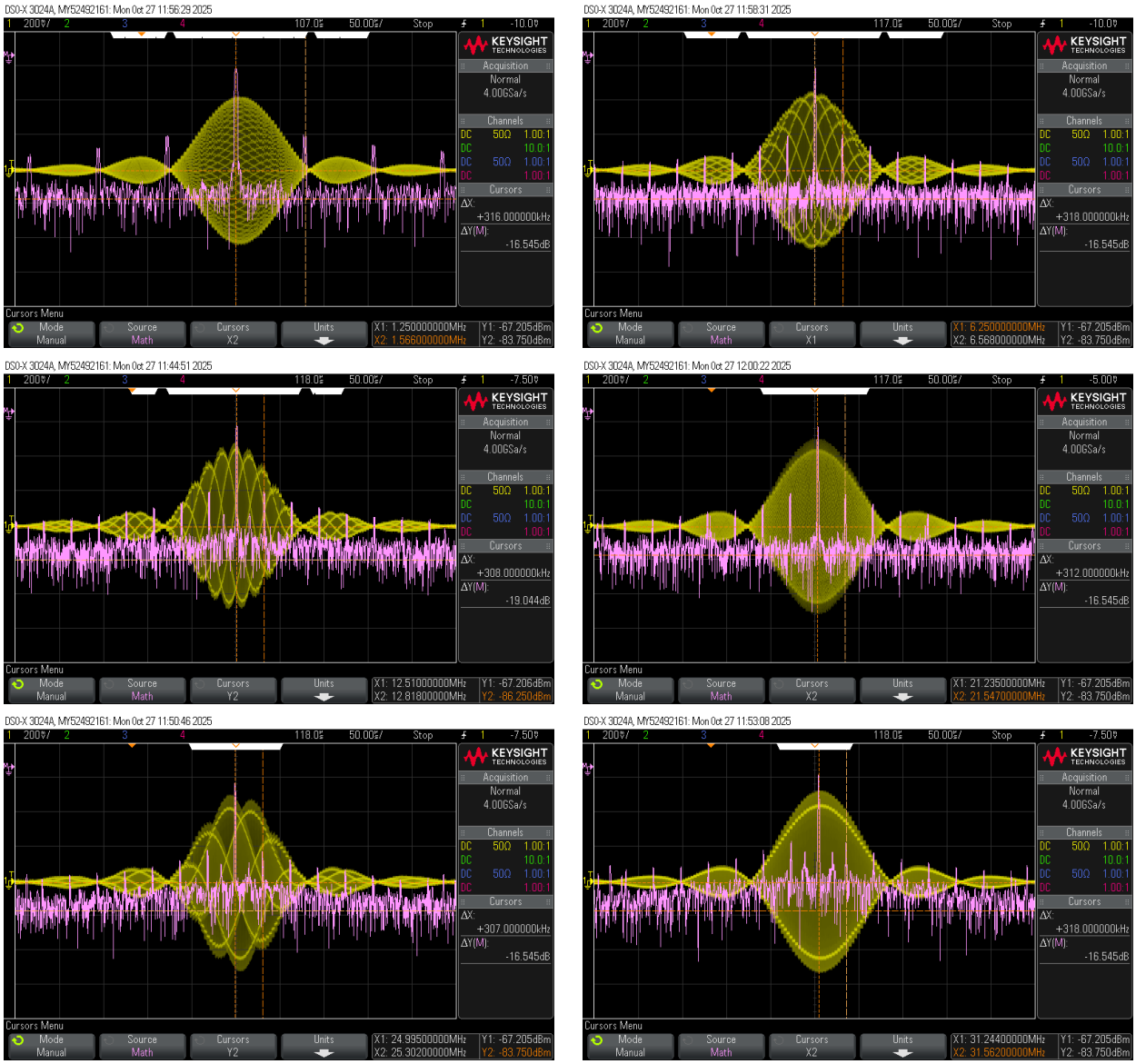


Figura 54: Espectros dos pulsos modulados observados no osciloscópio para os valores de `tuning_word` de 41, 205, 410, 696, 819, 1024.

A análise dos picos espectrais para cada pulso mostrou que os picos de frequência observados correspondiam a valores ligeiramente superiores aos esperados, indicando que o sistema estava operando com um clock de 125 MHz. A Tabela 1 mostra a relação entre os valores de `tuning_word`, as frequências teóricas esperadas para clocks de 100 MHz e 125 MHz, e as frequências efetivamente medidas no osciloscópio.

`tuning_word`	f_{out} (100 MHz)	f_{out} (125MHz)	f_{medido} (MHz)
41	1,00	1,25	1,25
205	5,00	6,25	6,25
410	10,01	12,51	12,51
696	16,99	21,24	21,23
819	19,99	24,99	24,99
1024	25,00	31,25	31,24

Tabela 1: Comparação entre valores teóricos e medidos para diferentes `tuning_word`.

Com base nessa análise, concluiu-se que o clock de operação real da placa era de 125 MHz. Dessa forma, foi gerado um novo arquivo `.bit` ajustando o valor de `tuning_word` para 558, de modo a obter um pulso na frequência de interesse do sistema de 17 MHz. A Figura 55 mostra o pulso modulado observado no osciloscópio.

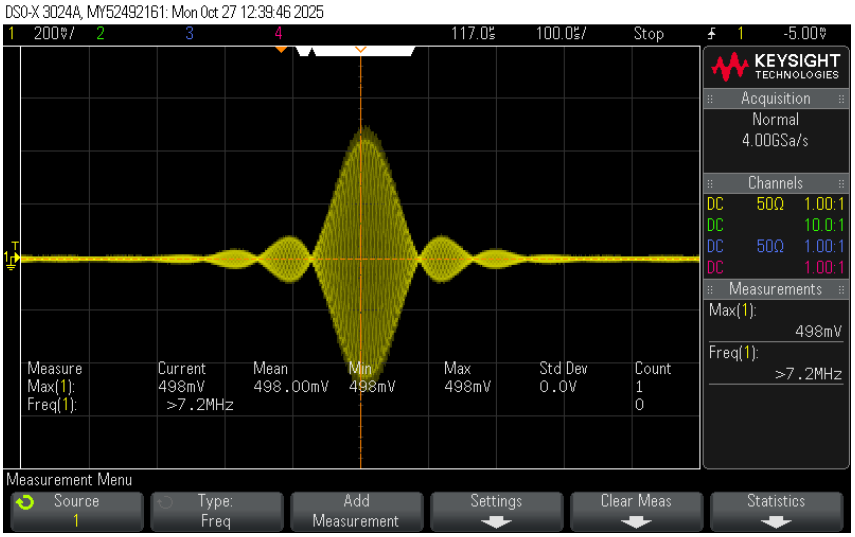


Figura 55: Pulso modulado observado no osciloscópio para `tuning_word` = 558, correspondente à frequência de 17 MHz.

A Figura 56 mostra o espectro exibido no osciloscópio, evidenciando o pico em aproximadamente 17 MHz, confirmando o comportamento esperado para o novo valor de `tuning_word`.

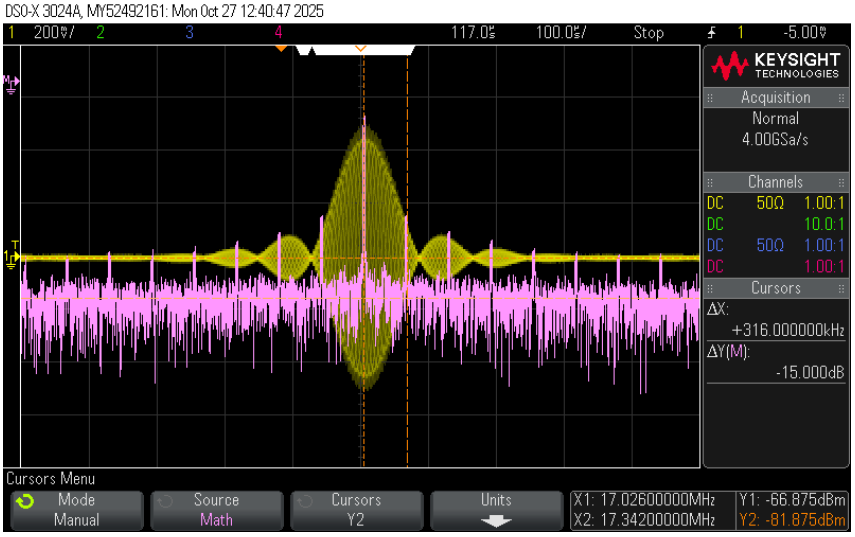


Figura 56: Espectro (FFT) do pulso modulado observado no osciloscópio para `tuning_word` = 558, com pico em aproximadamente 17 MHz.

Para análise detalhada, os dados do pulso foram exportados do osciloscópio em formato `.csv`, considerando o tamanho máximo de aquisição do equipamento (64.516 amostras). Esses dados foram posteriormente reconstruídos em ambiente Python, conforme mostrado na Figura 57, permitindo a visualização do pulso obtido experimentalmente.

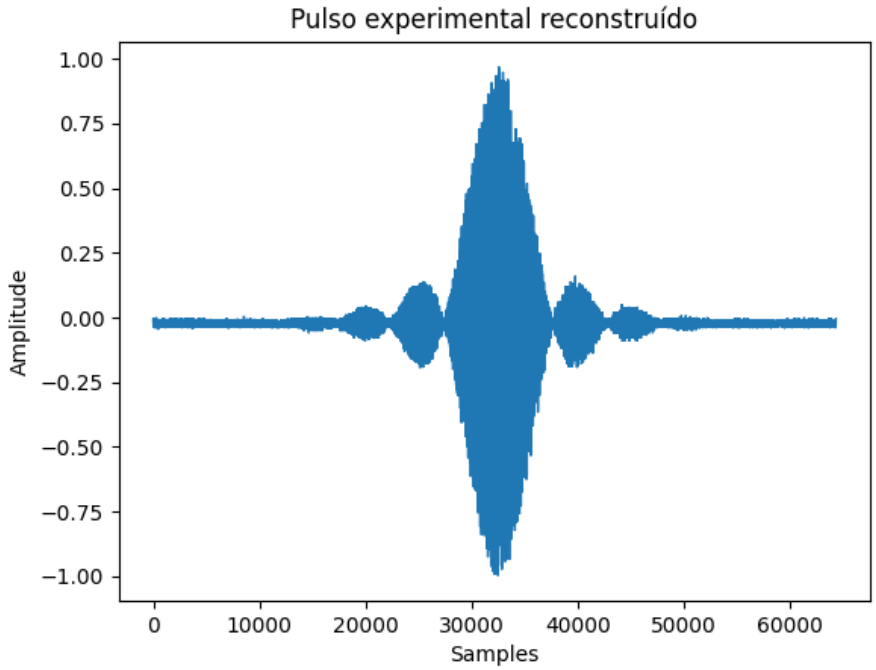


Figura 57: Pulso reconstruído em Python a partir dos dados adquiridos no osciloscópio.

A Figura 58 apresenta a magnitude da DFT do pulso reconstruído, utilizada para a análise da largura de banda.

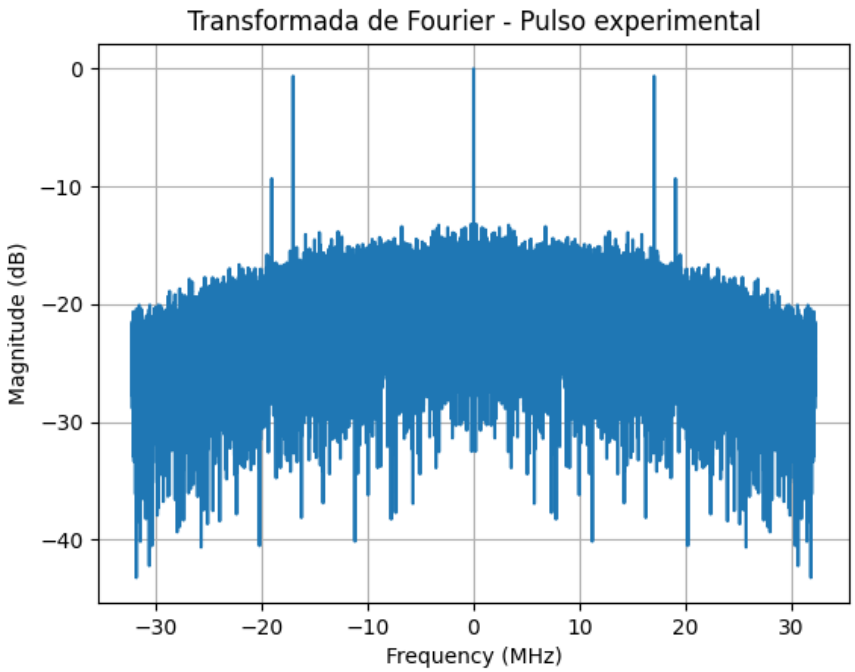


Figura 58: Espectro de magnitude do pulso reconstruído em Python a partir dos dados adquiridos no osciloscópio.

A largura de banda medida foi de aproximadamente 14,7 kHz em -10 dB. Considerando que o clock real da placa é de 125 MHz, esperava-se que esse valor fosse diferente da largura de banda obtida nas simulações (10,67 kHz para 100 MHz), devido à alteração na duração do pulso. Para efeito de comparação, foi gerado um novo pulso em Python com as mesmas amostras, mas considerando o clock de 125 MHz. A largura de banda obtida neste caso foi de aproximadamente 13,15 kHz, valor mais próximo ao obtido experimentalmente.

A diferença pode ser atribuída à limitação de amostragem do osciloscópio (64.516 pontos, contra 100.000 pontos do pulso original) e à dificuldade de capturar exatamente um pulso completo dentro da janela de aquisição do equipamento.

Portanto, a validação experimental confirmou o funcionamento da implementação em FPGA, apresentando pulsos modulados na frequência de interesse e de acordo com os resultados simulados em VHDL. Além disso, a análise espectral do pulso reconstruído em Python permitiu validar a largura de banda próxima da simulada.

6 CONCLUSÃO

Este trabalho teve como objetivo o desenvolvimento e a validação de um gerador digital de pulsos de RF em FPGA para um protótipo de MRI de 0,4 T, sevindo como uma etapa inicial para a criação de um console conceitual no CNPEM. Os resultados obtidos tanto nos módulos individuais quanto nos módulos de integração demonstraram um comportamento de acordo com o esperado. O TopLevel `IQ_Modulation` executou corretamente a multiplicação da envoltória pelas portadoras, mostrando que a implementação pode ser utilizada para modulação IQ no sistema de transmissão em RM. Além disso, os resultados comprovaram a correta implementação da técnica de DDS, validando o controle digital da frequência das portadoras de acordo com o parâmetro `tuning_word`.

Os testes em bancada, realizados com a placa Red Pitaya e um osciloscópio, confirmaram o funcionamento da implementação em hardware, apresentando pulsos modulados nas frequências de interesse e de acordo com os resultados simulados em VHDL no Vivado. A análise espectral do pulso reconstruído em Python permitiu medir a largura de banda do sinal transmitido, que apresentou valor próximo ao obtido em simulação, com uma variação que pode ser atribuída às limitações de amostragem dos dados exportados do osciloscópio.

Este trabalho contribuiu para a capacitação tanto na teoria dos equipamentos de RM quanto na geração de sinais digitais para transmissão de pulsos de RF. Os próximos passos incluem a otimização dos resultados obtidos e a investigação sobre a frequência de clock da Red Pitaya. Posteriormente, pretende-se implementar o sistema de geração e controle das formas de onda de gradientes e o de recepção dos sinais de RM, de modo que o console conceitual seja completamente desenvolvido no CNPEM. Esses avanços servirão como prova de conceito para a futura criação de um console de MRI desenvolvido integralmente no Centro, de forma que possa contribuir para a autonomia tecnológica nacional e a redução da desigualdade no acesso a exames de RM no país.

REFERÊNCIAS

[1] Alessandro A Mazzola. "Ressonância magnética: princípios de formação da imagem e aplicações em imagem funcional". Em: *Revista Brasileira de Física Médica* 3.1 (2009), pp. 117–129.

[2] João Flávio De Freitas Almeida, Samuel Vieira Conceição e Virgínia Silva Magalhães. "An optimization model for equitable accessibility to magnetic resonance imaging technology in developing countries". en. Em: *Decision Analytics Journal* 4 (set. de 2022), p. 100105. ISSN: 27726622. DOI: 10.1016/j.dajour.2022.100105. URL: https://linkinghub.elsevier.com/retrieve/pii/S2772662222000443 (acesso em 16/08/2025).

[3] Donald B. Plewes e Walter Kucharczyk. "Physics of MRI: A primer". en. Em: *Journal of Magnetic Resonance Imaging* 35.5 (mai. de 2012), pp. 1038–1054. ISSN: 1053-1807, 1522-2586. DOI: 10.1002/jmri.23642. URL: https://onlinelibrary.wiley.com/doi/10.1002/jmri.23642 (acesso em 16/08/2025).

[4] Camila De Almeida Costa Alencar et al. "Computed tomography and magnetic resonance imaging in Brazil: an epidemiological study on the distribution of equipment and frequency of examinations, with comparisons between the public and private sectors". en. Em: *Radiologia Brasileira* 57 (2024), e20230094en. ISSN: 1678-7099. DOI: 10.1590/0100-3984.2023.0094-en. URL: http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0100-39842024000100203&tlng=en (acesso em 16/08/2025).

[5] Eduardo H.M.S.G. De Figueiredo, Arthur F.N.G. Borgonovi e Thomas M. Doring. "Basic Concepts of MR Imaging, Diffusion MR Imaging, and Diffusion Tensor Imaging". en. Em: *Magnetic Resonance Imaging Clinics of North America* 19.1 (fev. de 2011), pp. 1–22. ISSN: 10649689. DOI: 10.1016/j.mric.2010.10.005. URL: https://linkinghub.elsevier.com/retrieve/pii/S1064968910000747 (acesso em 24/09/2025).

[6] Centro Nacional de Pesquisa em Energia e Materiais (CNPEM) et al. *Projeto de Pesquisa e Desenvolvimento em Ressonância Magnética Nuclear no CNPEM: Estudos Preliminares*. pt. Rel. técn. Centro Nacional de Pesquisa em Energia e Materiais (CNPEM), 2023.

[7] Donald W. McRobbie et al., ed. *MRI from picture to proton*. en. 2. ed., 4. print. Cambridge: Cambridge University Press, 2010. ISBN: 978-0-521-86527-2 978-0-521-68384-5.

[8] Matt A. Bernstein, Kevin Franklin King e Xiaohong Joe Zhou. *Handbook of MRI pulse sequences*. en. Amsterdam Boston: Academic Press, 2004. ISBN: 978-0-12-092861-3.

[9] Vlad Negnevitsky et al. "MaRCoS, an open-source electronic control system for low-field MRI". en. Em: *Journal of Magnetic Resonance* 350 (mai. de 2023), p. 107424. ISSN: 10907807. DOI: 10.1016/j.jmr.2023.107424. URL: https://linkinghub.elsevier.com/retrieve/pii/S1090780723000599 (acesso em 29/10/2025).

[10] Zesong Jiang et al. *A Low-Field Magnetic Resonance Signal Transmission and Reception Processing Platform*. en. arXiv:2409.08671 [physics]. Set. de 2024. DOI: 10.48550/arXiv.2409.08671. URL: http://arxiv.org/abs/2409.08671 (acesso em 29/10/2025).

[11] Suma Anand. "OCRA: A Low-Cost, Open-Source FPGA-Based MRI Console Capable of Real-Time Control". PhD Thesis. Massachusetts Institute of Technology, 2018.

[12] Clarissa Zimmerman Cooley et al. "Design and implementation of a low-cost, tabletop MRI scanner for education and research prototyping". en. Em: *Journal of Magnetic Resonance* 310 (jan. de 2020), p. 106625. ISSN: 10907807. DOI: 10.1016/j.jmr.2019.106625. URL: https://linkinghub.elsevier.com/retrieve/pii/S1090780719302642 (acesso em 29/10/2025).

[13] T. Schilcher. *RF Applications in Digital Signal Processing*. Rel. técn. CERN-2008-012. CERN, 2008. URL: https://cds.cern.ch/record/1100538.

[14] Preeti Hemnani et al. "FPGA Based RF Pulse Generator for NQR/NMR Spectrometer". en. Em: *Procedia Computer Science* 93 (2016), pp. 161–168. ISSN: 18770509. DOI: 10.1016/j.procs.2016.07.196. URL: https://linkinghub.elsevier.com/retrieve/pii/S1877050916314363 (acesso em 29/10/2025).