INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Síntese Digital Direta

Rui Tiago Ferro Henrique

Mestrado em Engenharia de Telecomunicações e Informática

Orientador:
Dr. Francisco António Bucho Cercas, Professor Catedrático,
ISCTE-IUL

Co-Orientador:
Dr. Nuno Manuel Branco Souto, Professor Associado (com Agregação),
ISCTE-IUL

Novembro, 2022

TECNOLOGIAS
E ARQUITETURA

Síntese Digital Direta

Rui Tiago Ferro Henrique

Mestrado em Engenharia de Telecomunicações e Informática

Orientador:
Dr. Francisco António Bucho Cercas, Professor Catedrático,
ISCTE-IUL

Co-Orientador:
Dr. Nuno Manuel Branco Souto, Professor Associado (com Agregação),
ISCTE-IUL

Novembro, 2022

Agradecimentos

Primeiramente, gostaria de expressar o meu agradecimento aos meus orientadores, Professor Francisco Cercas e Professor Nuno Souto. Ao Professor Francisco Cercas, que introduziu este campo de investigação, fornecendo os comentários e feedback necessários. Ao Professor Nuno Souto, que ajudou a superar todas as minhas dificuldades. Ambos foram essenciais para me manter motivado e aumentar o meu interesse pela pesquisa deste tema. Esta dissertação foi possível elaborar com a ajuda dos professores, sendo um privilégio trabalhar a vosso lado.

Gostaria de agradecer à minha namorada, Ana Catarina Dionísio, aos meus pais, Ana Luísa Henrique e Fernando Ferro, e aos meus tios, Clara e Tito Salvador, pois sem eles nada disto teria sido possível. Agradeço todo o incentivo e apoio incondicional. Obrigado por me terem sempre incentivado ao longo deste percurso académico. Quero agradecer dar um especial agradecimento à minha mãe e namorada por todo o auxílio dado ao longo do meu percurso escolar e na realização desta dissertação. Apesar de não estar fisicamente presente, agradeço ao meu irmão, por todo o carinho, amor e amizade, que será sempre uma figura de referência na minha vida e espero desejá-lo orgulhoso com a conclusão desta etapa.

Quero agradecer à instituição que permitiu desenvolver a nível técnico e profissional, o ISCTE – Instituto Universitário de Lisboa, assim como os professores que marcaram o meu percurso académico.

Gostaria de agradecer a todos os amigos e colegas por me ouvirem e aconselharem a sempre tentar o meu melhor. Eles são muito importantes e agradeço muito toda a ajuda e amizade que me deram.

Resumo

Os projetos tradicionais de sintetizadores de frequência de elevada largura de banda utilizam um circuito fechado de fase (PLL). Um sintetizador digital direto, ou na terminologia inglesa, *Direct Digital Dynthesis* (*DDS*) oferece muitas vantagens significativas sobre as abordagens PLL como, por exemplo, um tempo de estabilização rápido, resolução de frequência sub-Hertz, resposta de comutação de fase contínua e baixo ruído de fase. Embora o princípio do DDS seja conhecido há muitos anos, o DDS não desempenhou um papel dominante na geração de frequência de banda larga até recentemente. Os DDSs iniciais eram limitados a produzir frequências estreitamente espaçadas com pequena largura de banda, devido a limitações da lógica digital e das tecnologias de conversão D/A. As vantagens recentes nas tecnologias de circuitos integrados (CI) trouxeram um progresso notável nesta área. Ao programar um DDS, é possível adaptar as larguras de banda de canal, formatos de modulação, salto de frequência e taxas de dados. Este é um passo importante em direção a um "software-rádio" que pode ser usado em vários sistemas. Um DDS pode ser aplicado no modulador ou demodulador nos sistemas de comunicação. O objetivo desta pesquisa foi encontrar um frontend ideal para um transmissor, concentrando-se nas implementações de circuito do DDS, mas a pesquisa também inclui a interface para circuitos de banda base e aspetos de design de nível de sistema de sistemas de comunicação digital.

Palavras-Chave: Síntese Digital Direta, Acumulador de Fase, Conversor Fase/Amplitude, DAC, Gerador de Sinais

Abstract

Traditional high-bandwidth frequency synthesizer designs utilize a phase closed loop (PLL). A direct digital synthesizer, or in English terminology, Direct Digital Dynthesis (DDS) offers many significant advantages over PLL approaches such as a fast-settling time, sub-Hertz frequency resolution, continuous phase switching response and low phase noise. Although the principle of DDS has been known for many years, DDS has not played a dominant role in broadband frequency generation until recently. Early DDSs were limited to producing closely spaced frequencies with low bandwidth, due to limitations of digital logic and D/A conversion technologies. Recent advances in integrated circuit (IC) technologies have brought remarkable progress in this area. When programming a DDS, it is possible to adapt channel bandwidths, modulation formats, frequency hopping and data rates. This is an important step towards “radio software” that can be used on multiple systems. A DDS can be applied in the modulator or demodulator in the communication systems. The purpose of this research was to find an ideal frontend for a transmitter, focusing on the circuit implementations of DDS, but the research also includes the interface to baseband circuits and system-level design aspects of digital communication systems.

Keywords: Direct Digital Synthesis, Phase Accumulator, Phase/Amplitude Converter, DAC, Signal Generator

Índice

Índice de Figuras

Lista de Acrónimos

DDS	Direct Digital Synthesis
PA	Phase Accumulator
LUT	Look Up Table
DAC	Digital-to-Analog Converter
PLL	Phase Locked Loop
VCO	Voltage-controlled oscillator
PT	Phase Truncation
FCW	Frequency Control Word
ROM	Read only memory
ADC	Analog-to-Digital Converter
DNL	Differential Nonlinearity
RF	Radio frequency
AM	Amplitude Modulation
PM	Phase Modulation
SNR	Signal-Noise Ratio
SFDR	Spurious-free dynamic range
ECG	Eletrocardiograma
ULA	Unidade Lógica e Aritmética
DSP	Digital Signal Processing
FPU	Floating Point Unit
INL	Integral Nonlinearity

1. Introdução

Este capítulo destina-se à apresentação do tema da dissertação, concretamente quanto à motivação e enquadramento do mesmo e respetivas questões de investigação. São, também, definidos e detalhados os objetivos do trabalho em desenvolvimento, bem como o processo de investigação aplicado na elaboração da dissertação.

1.1. Enquadramento e Motivações

O sintetizador digital direto (DDS) é uma técnica popular para síntese de frequência. A ideia principal do DDS é usar blocos de processamento de dados digitais como um meio de gerar um sinal de saída ajustável em frequência e fase referenciado a uma fonte de relógio de precisão de frequência fixa. Os produtos DDS atuais, competitivos em termos de custo, de alto desempenho, funcionalmente integrados e de tamanho pequeno, estão a tornar-se rapidamente uma alternativa às soluções tradicionais de sintetizadores analógicos ágeis de frequência.

A topologia de um sintetizador digital direto convencional consiste em cinco elementos principais: um relógio de acionamento, um acumulador de fase (PA), uma tabela de consulta (LUT), um conversor digital-analógico (DAC) e um filtro de reconstrução. Foi apresentado por Tierney, Rader e Gold em 1971. [1]

O *Phase Locked Loop* (PLL) oferece uma largura de banda ajustável, devido ao uso de um divisor programável. Comparado com outras técnicas de sintetização de frequência, como o *Voltage-controlled oscillato*r (VCO), o *DDS* oferece resolução de frequência muito maior, comutação de canal de frequência mais rápida com fase contínua, faixa mais ampla de frequências e capacidade de modulação mais flexível. Atualmente, os sistemas *DDS* são competitivos, de alto desempenho e integrados à funcionalidade, sendo amplamente utilizados no campo das telecomunicações e sendo uma alternativa a algumas soluções tradicionais de sintetizadores analógicos. As aplicações do DDS incluem geração de sinal, osciladores locais, geradores de função, misturadores moduladores e sintetizadores de som.

De seguida listam-se algumas vantagens dos sistemas DDS:

- Estes dispositivos apresentam uma resolução de ajuste de micro-hertz da frequência de saída e capacidade de ajuste de fase de subgrau;

- O DDS tem uma “velocidade de salto” extremamente rápida na frequência de saída de sintonia (ou fase), saltos de frequência contínua de fase sem anomalias de tempo de ajuste de loop relacionado com o *overshoot* ou *undershoot;*
- A arquitetura digital DDS elimina a necessidade de ajustes manuais do sistema associados ao envelhecimento de componentes e desvio de temperatura em soluções de sintetizadores analógicos;
- A interface de controle digital da arquitetura DDS facilita um ambiente onde os sistemas podem ser controlados remotamente e otimizados minuciosamente, sob controle do processador;
- Quando utilizado como um sintetizador de quadratura, o DDS oferece correspondência e controle incomparáveis das saídas sintetizadas I e Q.

O objetivo principal é estudar o princípio da síntese digital direta, projetando e construindo um gerador genérico de funções arbitrárias que possa ser utilizado para fins de pesquisa na área de telecomunicações e com as possíveis aplicações mencionados e com uma frequência máxima tão alta quanto possível.

1.2. Questões de Investigação

- Quer-se gerar frequências muito elevadas ou prefere-se ter uma frequência razoável, mas muitas funções diferentes implementadas, ou ambas?
- Vai-se usar um processador rápido e com muitos bits, ou uma combinação de um processador mais normal (para o controle) e de alguma lógica dedicada para a parte mais rápida para o DDS propriamente dito?
- Que dispositivos existem já no mercado?

1.3. Objetivos

- Estudar o princípio de um gerador de frequência usando o princípio do DDS;
- Identificar o maior número possível de características e funções, de modo a projetar este dispositivo para ser um instrumento útil e geral para fins de pesquisa e laboratório;
- Escolher um processador adequado que permita trabalhar com frequências altas e úteis para aplicações em comunicações de rádio e satélite, até a banda de GigaHertz, se possível;

- Para implementar o instrumento, primeiramente será usada uma *breadbord* e posteriormente desenhado um circuito impresso próprio.

1.4. Estrutura da dissertação

A dissertação está dividida em seis capítulos:

No capítulo 2, os conceitos gerais da síntese digital direto são descritos, assim como a sua estrutura e todos os blocos para a produção de um sinal analógico. As suas técnicas são descritas e analisadas.

No capítulo 3, é analisada a implementação deste sistema sem memória, assim como a análise do espetro do sinal de saída, bem como a intercalação do conversor digital para analógico.

No capítulo 4, é efetuada uma descrição da montagem experimental. As especificações principais do hardware, bem como a implementação dos blocos funcionais desta técnica são incluídos, tais como o Teensy 4.1, o módulo de geração, de controlo e o DAC.

No capítulo 5, a simulação e as limitações experimentais são analisadas. Os resultados simulados e práticos são apresentados, analisados e discutidos.

No capítulo 6, as conclusões são apresentadas e os pontos de trabalho futuros são descritos.

2. Conceitos Gerais

2.1. DDS - Uma Visão Geral

A síntese digital direta é um método de produção de uma forma de onda analógica, geralmente uma onda sinusoidal, onde é gerado um sinal variável no tempo em formato digital, executando a conversão de digital para analógico. As operações destes dispositivos são, maioritariamente, digitais, portanto, pode existir uma comutação rápida entre as frequências de saída, com uma resolução de frequência fina e operação num amplo espectro de frequências.

Esta abordagem introduz uma frequência de fonte estável, tendo um relógio de referência para definir os tempos onde os valores das amostras digitais sinusoidais são produzidos, estas amostras são convertidas do formato digital para o analógico, sendo suavizadas por um filtro de reconstrução para produzir sinais de frequência analógica. O DDS consiste num acumulador de fase, ou, *phase accumulator* (PA) e uma tabela de pesquisa sinusoidal, *Lookup Table* (LUT). A entrada para o acumulador de fase é uma palavra de controlo de frequência, que determina a periodicidade do PA, a cada ciclo de relógio, o PA é atualizado para a palavra de controlo de frequência, onde a sua saída está conectada à tabela de pesquisa. A saída do LUT é convertida num sinal analógico usando um conversor digital para analógico.

O tamanho do *LUT* depende do comprimento do *PA* de *n* bits. Se o comprimento do *n* for grande, então o LUT será bastante grande, não sendo desejável, uma vez que isso diminui a velocidade do DDS, tendo um maior consumo de energia. Para reduzir o tamanho do LUT, adota-se a técnica de truncamento de fase, *Phase Truncation* (PT), como esta técnica é originada por parte da fase gerada, esta é truncada dando origem a um espectro de saída. Como o DDS funciona como um jitter do sistema digital, onde o jitter é definido como uma variação abrupta e indesejada de uma ou mais características do sinal, como, por exemplo, o intervalo entre pulsos sucessivos, a amplitude de ciclos sucessivos ou a frequência e fase de ciclos sucessivos, verificando a presença de ruído no espectro de saída.

2.2. Arquitetura de DDS de saída

O diagrama de blocos básico de um sintetizador de frequência digital direto é mostrado em [2]

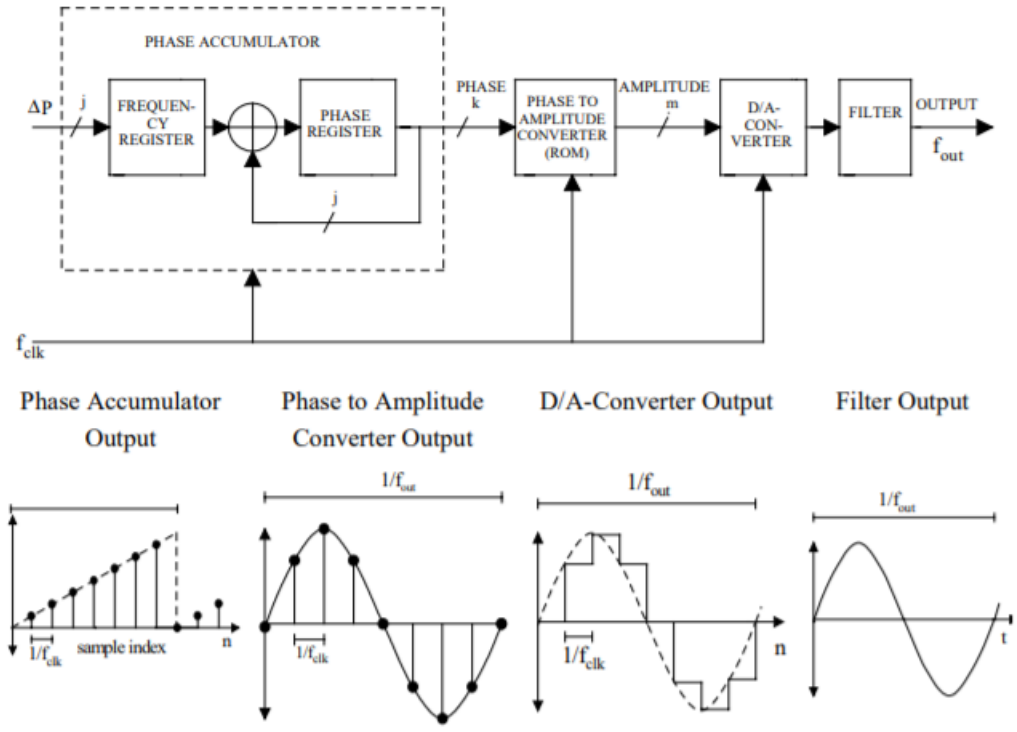


Figura 1 - Diagrama de blocos simplificado do sintetizador digital direto e o fluxo de sinal no DDS

Os principais componentes de um DDS são:

- Acumulador de Fase;
- Conversor de Fase-Amplitude, ou seja, uma tabela de consulta sinusoidal;
- Conversor Digital-Analógico;
- Filtro.

Um DDS produz uma onda sinusoidal numa determinada frequência, esta frequência depende da frequência do relógio de referência, f_{clk}, o número binário programado no registrador de fase (palavra de controle de frequência, M) e o comprimento do acumulador de *n*-bits. O número binário no registador de fase fornece a entrada principal para o acumulador de fase.

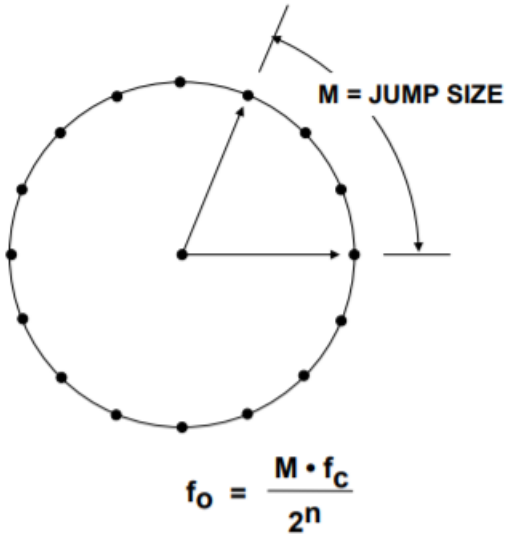
Se uma tabela de consulta de seno for usada, o acumulador de fase calcula um endereço de fase, ou seja, um ângulo para a tabela de consulta, emitindo o valor digital da amplitude que

corresponde ao seno deste ângulo de fase para o DAC. O conversor, por sua vez, transformará esse número num valor correspondente de tensão ou de corrente analógica, quando se pretende gerar uma onda sinusoidal de frequência fixa, um valor constante (o incremento de fase, determinado pelo número binário *M*) é adicionado ao acumulador de fase a cada ciclo de relógio. Se o incremento de fase for grande, o acumulador de fase passará rapidamente pela tabela de pesquisa do seno, gerando uma onda sinusoidal de alta frequência. Se o incremento de fase for pequeno, o acumulador de fase realizará um maior número de passos, gerando uma forma de onda mais lenta. [3] [4]

O coração deste sistema é o acumulador de fase, onde o conteúdo é atualizado a cada ciclo de relógio, sempre esta tabela é atualizada, o número digital, que se encontra armazenado no registador de fase, é adicionado o número no registador do acumulador de fase. Se o número no registo de fase for 00…01, e o conteúdo inicial do acumulador de fase for 00…00. O acumulador de fase é atualizado em 00...01 em cada ciclo de relógio. Caso o acumulador tenha 32 bits de largura, são necessários 232 ciclos de relógio, antes que o acumulador de fase retorne a 00...00 e o ciclo volte a repetir.

A saída do acumulador de fase serve como endereço para uma tabela de consulta de conversão de fase em amplitude. Cada endereço na *LUT* corresponde a um ponto de fase na onda sinusoidal de 0° a 360°. O *LUT* contém as informações de amplitude digital correspondentes para um ciclo completo de uma onda sinusoidal. O *LUT* mapeia as informações de fase do acumulador numa palavra de amplitude digital, acionando o DAC. Para n=32 e M=1, o acumulador de fase percorre cada uma das 232 saídas possíveis. A frequência da onda sinusoidal de saída correspondente é igual à frequência de relógio dividida por 232.

Caso o M=2, o registador do acumulador de fase roda duas vezes mais rápido e a frequência de saída é dobrada. Para um acumulador de fase de *n* bits, existem 2^n pontos de fase possíveis. A palavra digital no registrador de fase, M, representa a quantidade do acumulador de fase que é incrementada a cada ciclo de relógio. [4]



n	Number of Points = 2^n
8	256
12	4,096
16	65,536
20	1,048,576
24	16,777,216
28	268,435,456
32	4,294,967,296
48	281,474,976,710,656

Figura 2 – Roda de Fases

Se f_{clk} é a frequência do relógio, então a frequência da onda sinusoidal de saída é igual a:

$$f_{out} = \frac{M * f_{clk}}{2^n} \quad (1)$$

A equação acima é conhecida como a "equação de ajuste" do DDS. A resolução de frequência do sistema é obtida da seguinte forma,

$$\frac{f_{clk}}{2^n} \quad (2)$$

Num sistema DDS prático, todos os bits fora do acumulador de fase não são passados para a *LUT*, mas são truncados, deixando apenas os primeiros 13 a 15 bits mais significativos, reduzindo o tamanho da *LUT*, não afetando a resolução de frequência. O truncamento de fase adiciona apenas uma quantidade pequena, mas aceitável de ruído de fase à saída final. A resolução do DAC é normalmente 2 a 4 bits menor que a largura da tabela de pesquisa, um DAC de N-bits perfeito adiciona ruído de quantização à saída. [5]

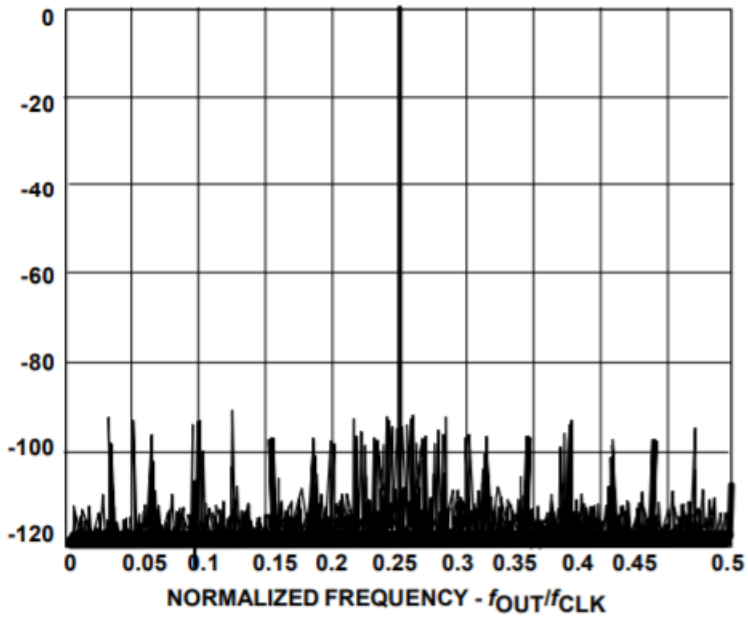


Figura 3 - Frequência Normalizada *[6]*

A figura acima mostra o espectro de saída calculado para um acumulador de fase de 32 bits com um truncamento de fase de 15 bits. O valor de M foi escolhido, de modo que a frequência de saída fosse ligeiramente deslocada de 0,25 vezes da frequência de relógio, os efeitos causados pelo truncamento de fase e a resolução finita do DAC estão todos, pelo menos, $-90\,dB$. Este desempenho excede muito o de qualquer DAC de 12 bits disponível comercialmente e é adequado para a maioria dos aplicativos. [6]

O sistema DDS básico é extremamente flexível e possui alta resolução. A frequência pode ser alterada instantaneamente sem descontinuidade de fase, alterando o conteúdo do registo M. No entanto, os sistemas DDS práticos primeiro requerem a execução de uma sequência em série ou de um carregamento de bytes para obter a nova palavra de frequência num registo de buffer interno que precede o registo M de saída paralela, minimizando a contagem dos pinos do pacote, sempre que a nova palavra é carregada no registador do buffer, o delta do registador de fase na saída paralela é cronometrado, alterando assim todos os bits simultaneamente. O número de ciclos de relógio necessários para carregar o registador de buffer determina a taxa máxima, na qual a frequência de saída pode ser alterada.

2.3. Equação de Ajuste de Frequência

Uma onda sinusoidal é, geralmente, expressa como $a(t) = \sin(\omega t)$, que não é linear e não é fácil de gerar, exceto através da construção de peças. No entanto, a informação angular é linear, porque o ângulo de fase gira em um ângulo fixo para cada unidade. Assim, a taxa angular depende da frequência do sinal descrito como $\omega = 2\pi f$, onde ω é a frequência angular.

Sabendo que a fase de uma onda sinusoidal é linear e depende de um período de relógio de referência, com frequência, f_{clk}, a rotação de fase, Δ_p para esse período pode ser determinada por,

$$\Delta_p = w * \Delta_t \tag{3}$$

Onde, Δ_p corresponde a uma mudança de fase na onde sinusoidal e Δ_t corresponde a uma pequena mudança no tempo, isolando a frequência angular, obtém-se o seguinte,

$$w = \frac{\Delta_p}{\Delta_t} = 2\pi f \tag{4}$$

O acumulador de fase sincronizado com f_{clk}, gera a sequência de valor de fase, onde Δ_t é a quantidade mínima de mudança,

$$f_{clk} = \frac{1}{\Delta_t} \tag{5}$$

Resolvendo a equação (4) e substituindo a frequência de relógio de referência para o período referido na equação (5), especifica-se a frequência do sinal de saída da seguinte maneira,

$$f_{out} = \frac{\Delta_p * f_{clk}}{2\pi} \tag{6}$$

Finalmente, para um acumulador de *n-bit* para o sinal de saída terá a frequência especificada,

$$f_{out} = \frac{\Delta_p * f_{clk}}{2^n} \tag{7}$$

Onde, o Δ_p é o incremento de fase da palavra e f_{clk} é a frequência de relógio, n é o comprimento do acumulador. Este valor de fase, Δ_p, é gerado usando o módulo 2^n como propriedade de overflowing de um n-bit PA. A taxa de overflow da frequência de saída dada na equação 7, Δ_p, é um inteiro, no entanto, a frequência de resolução é determinada definindo $\Delta_p = 1$,

$$\Delta_f = \frac{f_{clk}}{2^n} \tag{8}$$

Esta técnica funciona como uma técnica de salto de ponto, ou seja, como uma localização da memória e uma interpolação constante do sinal armazenado, sendo executada a taxa de relógio constante. À medida que a frequência de saída do DDS aumenta, o número de amostras por ciclo de forma de onda diminui.

Na técnica anteriormente descrita, *N* pontos de dados cobrem um ciclo completo da forma de onda sinusoidal. Um bloco de *N* amostras é armazenado na memória *LUT*. O ponteiro de endereço do LUT é um "tamanho de passo **D**, módulo N (mod N, overflowing)" acumulador de fase. **D**, um inteiro positivo, é um FCW: Frequency Control Word. [7]

- **D = 1** fornece uma "cópia exata" da forma de onda armazenada (o ponteiro percorre cada endereço sequencialmente, ou seja, o ponteiro acede a cada entrada consecutiva na tabela da mesma forma que o PPC: síntese de ponto por relógio).
- Quando **D > 1**, o ponteiro "pulará" algum endereço, resultando num valor de frequência mais alto:

$$\frac{f_{out}}{f_{clk}} = \frac{D}{N} \tag{9}$$

Quando a frequência de saída, f_{out}, aumenta, o número de amostras por ciclo diminui.

2.4. Blocos de um DDS

Um DDS é um dispositivo de sinal misto, ou seja, possui blocos analógicos e digitais. Estes blocos são o registador de fase, o acumulador de fase, o conversor de fase para amplitude (ROM/LUT), o conversor digital para analógico e o filtro de reconstrução.

2.4.1. Acumulador de Fase

Sinais sinusoidais de tempo contínuo têm uma faixa de fase angular repetitiva de 0 a 360 graus. A implementação digital não é diferente, a função do contador permite que o acumulador de fase atue como uma roda de fase. Para entender esta função básica, considera-se a oscilação da onda sinusoidal como um vetor a girar em torno de um círculo de fase. Cada ponto designado na roda de fase corresponde ao ponto equivalente num ciclo de uma onda sinusoidal. À medida que o vetor gira em torno da roda, visualiza-se que o seno do ângulo gera uma onda sinusoidal de saída correspondente. Um salto do vetor em torno da roda de fase, a uma velocidade constante, resulta num ciclo completo da onda sinusoidal de saída. O acumulador de fase fornece os valores angulares igualmente espaçados que acompanham a rotação linear do vetor em torno da roda de fase. O conteúdo do acumulador de fase corresponde aos pontos do ciclo da onda sinusoidal de saída. [3]

O PA é um contador de módulo *M* que incrementa o seu número armazenado, cada vez que recebe um impulso proveniente do relógio. A magnitude do incremento é determinada pela palavra de entrada codificada em binário (*M*). Esta palavra forma o tamanho do passo de fase entre as atualizações do relógio de referência, definindo quantos pontos deve-se pular em redor da roda de fase. Quanto maior o tamanho do salto, mais rápido o acumulador de fase transborda e completa o equivalente a um ciclo de onda sinusoidal. O número de pontos de fase discretos contidos na roda é determinado pela resolução do PA (n-bits), que determina a resolução do sintetizador.

Por exemplo, para um acumulador de fase de n = 28 bits, *M* terá um valor de 0000...0001, o que faria o acumulador de fase transbordar após 228 ciclos de relógio de referência. Se o valor de M for alterado para 0111...1111, o acumulador de fase irá transbordar após apenas 2 ciclos de relógio de referência (o mínimo exigido por *Nyquist)*. Essa relação pode ser vista na equação básica de ajuste para a arquitetura DDFS:

$$f_{out} = \frac{M * f_{clk}}{2^n} \tag{10}$$

Qualquer alteração no valor de *M* introduz alterações imediatas e contínuas de fase na frequência de saída. Num DDS, nenhum tempo de estabilização de ciclo é incluído, como no caso de um PLL. À medida que a frequência de saída aumenta, o número de amostras por ciclo diminui.

Uma vez que a teoria de amostragem determina que duas amostras por ciclo são necessárias para reconstruir a forma de onda de saída, a frequência de saída fundamental máxima de um DDS é $\frac{f_{clk}}{2}$. No entanto, para aplicações práticas, a frequência de saída é limitada um pouco menos do que esse valor, melhorando a qualidade da forma de onda reconstruído, permitindo filtragem na saída. Ao gerar uma frequência constante, a saída do PA aumenta linearmente, de modo que a forma de onda analógica gerada é automaticamente uma rampa.

2.4.2. Conversor Fase para Amplitude

A tabela de ROM do DDS pode ser uma tabela de consulta de seno, convertendo a entrada da fase digital do acumulador para a amplitude de saída. A saída do acumulador representa a fase da onda, bem como um endereço para uma palavra, que é a amplitude correspondente da fase na *LUT*. Esta amplitude de fase da ROM LUT aciona o DAC para fornecer uma saída analógica, sendo denominado por conversor digital de fase para amplitude, ou na terminologia inglesa, *Phase-to-Amplitude Converte*r (PAC), podendo efetuar a transformação polar para retangular, tendo uma projeção do componente real ou imaginário no tempo, com um dispositivo de mapeamento de forma de onda (seno) - uma memória. Todas as técnicas de cálculo de informação como simples operação de tabela de pesquisa são modeladas aqui. A memória de pesquisa contém um ciclo da forma de onda a ser gerada. O tamanho da LUT é de 2^n palavras. *LUT* traduz informações de fase truncada, estando em formato digital, em amostras de formas de onda numéricas quantificadas.

Alguns sistemas DDS podem ser implementados com ou sem ROM. A utilização de uma ROM como um conversor de fase para amplitude tem algumas vantagens sobre outras arquiteturas sem este tipo de implementação, a simplicidade deste circuito torna o ROM fácil de implementar, as arquiteturas sem a implementação de memória têm uma maior precisão de bits. Com uma maior precisão de bits, a *ROM* torna-se muito grande, consome mais energia e o sistema fica mais lento. O ROM armazena os valores das amplitudes de fase, enquanto as arquiteturas sem ROM calculam as amplitudes de fase. Inerentemente, uma ROM LUT fornece uma melhor Impulsoious Free Dynamic Range do que qualquer outra arquitetura para a mesma largura de bits. [2] Num cenário ideal sem quantização de fase e amplitude, a sequência de saída da tabela de consulta é dada por,

$$sin\ (2\pi * \frac{P(i)}{2^n}) \tag{11}$$

Onde $P(i)$ é um valor de registo de fase, $i - bit$, no enésimo período de relógio. O período numérico da sequência de saída do acumulador de fase é definido como o valor mínimo de P_e para o qual $P(i) = P(i + P_e)$ para todo i. O período numérico da sequência de saída do acumulador de fase (em ciclos de relógio) é,

$$P_e = \frac{2^n}{MDC(\Delta_P, 2^n)} \tag{12}$$

O $MDC(\Delta_P, 2^n)$ representa o máximo divisor comum de Δ_P e 2^n. O período numérico da sequência de amostras recuperado da ROM sinusoidal terá o mesmo valor que o período numérico da sequência gerada pelo acumulador de fase. Portanto, o espectro da forma de onda de saída do DDS antes de uma conversão digital para analógico é caracterizado por um espectro discreto consistindo de pontos P_e. A saída ROM é apresentada ao conversor D/A, que desenvolve uma onda sinusoidal analógica quantificada. O espectro de saída do conversor D/A contém frequências $nf_{clk} \pm f_{out}$, onde n = 0, 1, …etc. As amplitudes desses componentes são ponderadas por uma função,

$$sinc(\frac{f}{f_{clk}}) \tag{13}$$

Este efeito pode ser corrigido por um filtro $sinc(^{f}/_{f_{clk}})$ inverso. O filtro localizado após o conversor D/A remove os componentes de amostragem de alta frequência, fornecendo uma saída de uma onda sinusoidal pura.

Como o DDS gera frequências próximas de $^{f_{clk}}/_{2}$, a primeira amostra (f_{clk} - f_{out}) torna-se mais difícil de filtrar, resultando numa banda de transição mais estreita para o filtro. A complexidade do filtro é determinada pela largura da banda de transição. Portanto, para manter o filtro simples, a operação do DDS é limitada a menos de 40% da frequência do relógio.

2.4.3. Conversor Digital para Analógico

Existem dois tipos básicos de conversores, digital-analógico (DAC) e analógico-digital (ADC), o objetivo é bastante simples. No caso dos DACs, é emitida uma tensão analógica que é uma proporção de uma tensão de referência, onde a sua proporção é baseada na palavra digital aplicada. No caso do ADC, é emitida uma representação digital da tensão analógica que é aplicada à entrada dos ADC, ou seja, a representação proporcional a uma tensão de referência.

A palavra digital é sempre baseada numa proporção ponderada binariamente. A entrada ou saída digital é organizada em palavras de larguras variadas, chamadas de bits, normalmente num intervalo de 6 bits a 32 bits. Num sistema de ponderação binária, cada bit vale metade do bit à sua esquerda e duas vezes o bit à sua direita. Quanto maior o número de bits na palavra digital, melhor será a resolução. Esses bits são normalmente organizados em bytes.

A resolução de um conversor é o número de bits em sua palavra digital. A precisão é o número daqueles bits que atendem às especificações. Por exemplo, um DAC pode ter 16 bits de resolução, mas só pode ser manipulado para 14 bits, ou seja, a precisão garantida do DAC não será mais do que 14 bits. Com processamento adicional, normalmente filtragem, muitas vezes a precisão pode ser melhorada.

O DAC normalmente terá o próprio conversor e uma coleção de circuitos de suporte embutidos no chip. Os primeiros DACs desenvolvidos ocorreram nos anos 70, estes primeiros exemplos eram sub-blocos do DAC. Por exemplo, o AD550, que era uma fonte de corrente de 4 bits com uma ponderação binária. Este bloco de fonte de corrente seria acoplado a uma parte separada, como o AD850, que continha uma matriz de resistências e chaves CMOS. Em conjunto, teriam a operação de um DAC básico. Com o avançar dos anos, estas funções foram integradas na mesma matriz, como circuitos digitais adicionais, de forma a armazenar a entrada digital. O objetivo atual é permitir ao microprocessador a escrita em vários DACs do sistema e este fosse atualizando ao mesmo tempo. A classificação de entrada também pode ser um registador de deslocamento, o que permite uma interface em série. [8]

No back-end, como a saída do DAC geralmente é uma corrente, um amplificador operacional é frequentemente adicionado para realizar a conversão de corrente para tensão (I/V). No front-end, uma referência de tensão é frequentemente adicionada.

As informações do tempo discreto e amplitude discreta da onda sinusoidal são alimentadas a um conversor digital para analógico para serem convertidas numa onda sinusoidal de amplitude contínua e tempo contínuo. Os DACs de direção atuais são a melhor escolha para aplicações de alta velocidade devido à sua velocidade de comutação rápida, podendo ser implementados em arquiteturas binárias ponderadas, codificadas por termômetro e segmentadas. A arquitetura segmentada combina as arquiteturas binárias ponderadas e codificadas por termômetro para aproveitar os benefícios de ambas as arquiteturas. Ele usa termômetro codificado para seus bits mais significativos (MSB) e binário ponderado para seus bits menos significativos (LSB).

Na arquitetura binária ponderada, cada fonte atual é o dobro da anterior. De acordo com o código de entrada digital, uma combinação das fontes de corrente será comutada para a saída. Esta arquitetura tem a vantagem de ser uma pequena área e baixo consumo de energia consumo. No entanto, sofre de não linearidade diferencial, na terminologia inglesa, *Differential Nonlinearity* (DNL) e a presença de glitches, degradando o seu desempenho dinâmico.

Por outro lado, a arquitetura codificada por termômetro tem mais complexidade e maior consumo de energia, mas melhora DNL, com menores falhas e pequenos erros de comutação. Nesta arquitetura, todas as fontes atuais são iguais. O código de entrada digital é primeiro alimentado a um descodificador de termômetro, e o código deste aciona os interruptores.

A arquitetura segmentada utiliza o termômetro codificado para seus bits mais significativos, que são mais responsáveis pelo desempenho dinâmico, e binário ponderado para seus bits menos significativos. Um descodificador fictício deve ser usado para a parte binária ponderada para compensar o atraso do decodificador do termômetro da parte decodificada do termômetro. Deve-se notar que no DDS, o desempenho dinâmico do DAC desempenha um papel significativo na pureza espectral do espectro de saída.

2.4.4. Filtro

O DDS é um sistema de amostragem, portanto, existirá um ruído nas frequências de $nf_{clk} \pm f_o$ do espectro de saída, com a frequência de saída e o relógio de amostragem. Como resultado da funcionalidade de retenção de ordem zero do DAC, a amplitude do ruído é ponderada pela função $sinc(\frac{f}{f_{clk}})$.

Para a maioria dos dimensionamentos, estes ruídos são indesejáveis. Para remover estes ruídos, um filtro por uma função inversa $sinc(\frac{f}{f_{clk}})$ denominada por filtro *anti-aliasing* é usado no final do sistema. Idealmente, este filtro deve ter resposta unitária sobre a largura de banda *Nyquist* e zero além disso.

No entanto, a projeção deste filtro não é prática, consequentemente, alguma percentagem da largura de banda disponível será inutilizável. Portanto, a frequência de saída sintetizada do DDS é geralmente limitada a menos de 3/8 da frequência de amostragem.

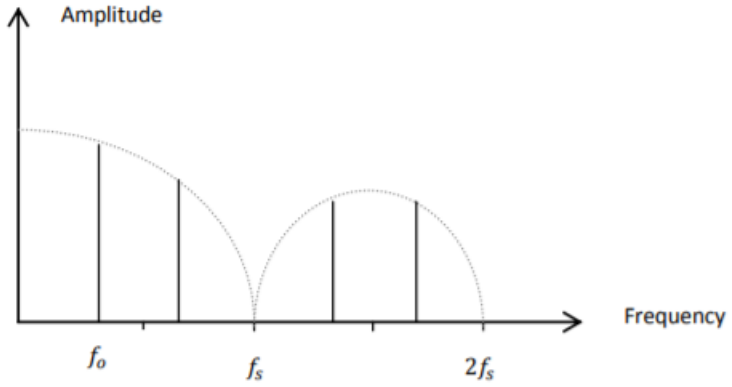


Figura 4 – Resposta de Frequência de um DDS

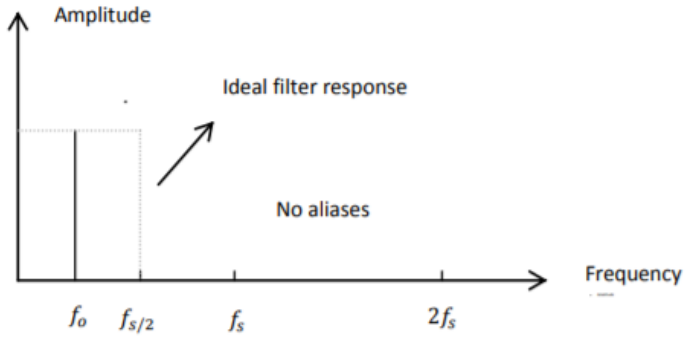


Figura 5 – Filtro Ideal de Anti-Aliasing

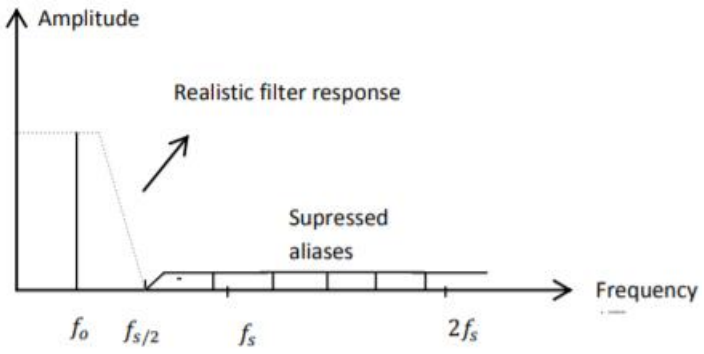


Figura 6 – Filtro Prático de Anti-Aliasing

3. Operações do DDS

3.1. Sintetizadores Digitais Diretos sem ROM

A *ROM Look Up Table* permite caracterizar a velocidade, potência e área dos *DDS*. Estes sistemas apresentam um grande consumo de energia e limitações em operações de alta frequência, embora tenham sido desenvolvidos mecanismos para comprimir a tabela de consulta ROM. [9] De modo a tentar solucionar estas limitações, foram desenvolvidas arquiteturas sem ROM.

3.1.1. Sintetizador digital direto usando um DAC com seno ponderado

O diagrama de blocos do DDS com um DAC com seno ponderado é demonstrado na Figura 8. Nesta arquitetura, o mapeamento da função seno e cosseno, bem como a conversão digital-analógica é efetuado num mesmo bloco, denominado por seno ponderado. A principal diferença entre este DAC e o linear, deve-se a este último apresentar as fontes de correntes idênticas entre si, sendo uma potência de dois ponderados. No DAC sinusoidal, as fontes de corrente são ponderadas de acordo com a amplitude da onda sinusoidal. [10] Nesta arquitetura, para cada fase da onda sinusoidal, o DAC ponderado comuta a respetiva corrente para a saída. Os dois bits mais significativos são usados para explorar a simetria de um quarto da onda sinusoidal. Inicialmente, estas arquiteturas usavam os DACs ponderados pelo seno do termômetro, no entanto, com o objetivo de reduzir o número de DACs, foram propostas técnicas de segmentação. [11] As técnicas de segmentação diferem conforme a linearidade do DAC, sendo mais complexos, quanto maior for a sua resolução. [9]

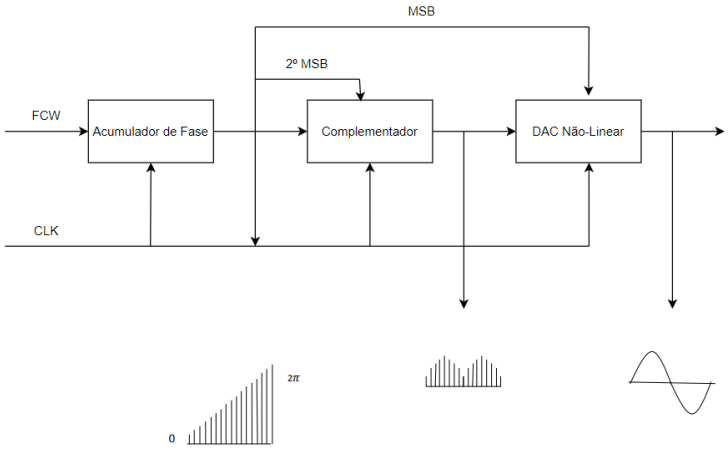


Figura 7 – Diagrama de Blocos do DDS com o DAC de seno ponderado *[10]*

3.1.2. Sintetizador digital direto usando triângulo para conversor de onda sinusoidal

O diagrama de blocos do DDS com a conversão de onda triangular para onda sinusoidal é ilustrado na figura 9.

Esta arquitetura utiliza o bit mais significativo, para explorar a simetria de metade de um ciclo de uma onda sinusoidal. Consequentemente, diminui o erro de truncamento. A saída do circuito complementar está diretamente conectada ao DAC Linear, esta produz uma onda triangular com a informação de fase analógica da onda sinusoidal, sendo efetuada a sua conversão, com uma metodologia de mapeamento sinusoidal analógico. Esta metodologia utiliza a aproximação parabólica, eletronicamente, isto significa que será usada a relação exponencial corrente-tensão dos transístores [9] [12]. A Figura 9 ilustra o esquema e a função de transferência de conversão de onda triangular para onda sinusoidal. [12] Esta arquitetura tem uma estrutura simples e de baixa potência, tendo uma precisão moderada na sua conversão. [9]

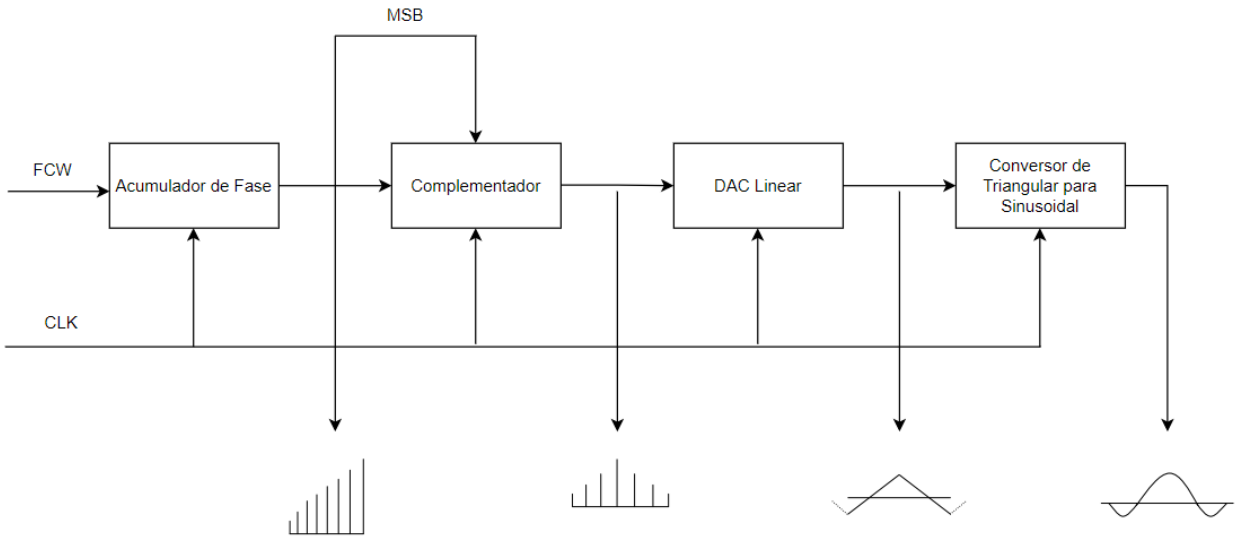


Figura 8 – Diagrama de blocos do DDS através da conversão de onda triangular para onda sinusoidal *[9]*

3.2. Análise de ruído do Espetro de Saída

O *DDS* pode ser aplicado em rádios, sistemas de instrumentação e de radar. Embora tenha impulsos grandes e imprevisíveis, causando incómodo em alguns projetos, estas inovações permitiram o melhoram do desempenho destes sistemas, onde os impulsos de pior caso têm uma menor probabilidade de surgimento. É necessário um planeamento cuidadoso da frequência, de modo a colocar os impulsos de pior caso fora da largura de banda de interesse, podendo ser filtradas com uma maior facilidade. A maior parte destas aplicações utiliza uma fração do seu espetro, sendo atenuada com filtros externos. A largura de banda necessária é considerada no seguinte intervalo $[0; 0.40 * f_s]$. A limitação de sub-Nyquist deve-se à banda de transição do filtro externo de rejeição de imagem. Algumas aplicações podem usar a banda da amostra, eliminado um estágio de *up-conversion*, a potência reduzida permita uma redução da relação Sinal-Ruído. O uso de amostras requer um filtro passa-banda em vez de um filtro passa-baixo. O sample-and-hold de ordem zero do DAC transmite uma atenuação de função $Sinc$ às amostras fundamentais e harmónicas do espetro DDS.

Um DDS tem quatro fontes principais de *impulso*:

- Relógio de referência;
- Truncamento no acumulador de fase;
- Erros de mapeamento de ângulo para amplitude;
- Não linearidades e ruído de quantização. A previsibilidade das frequências de impulso permite o desenvolvimento de um plano de frequência eficaz.

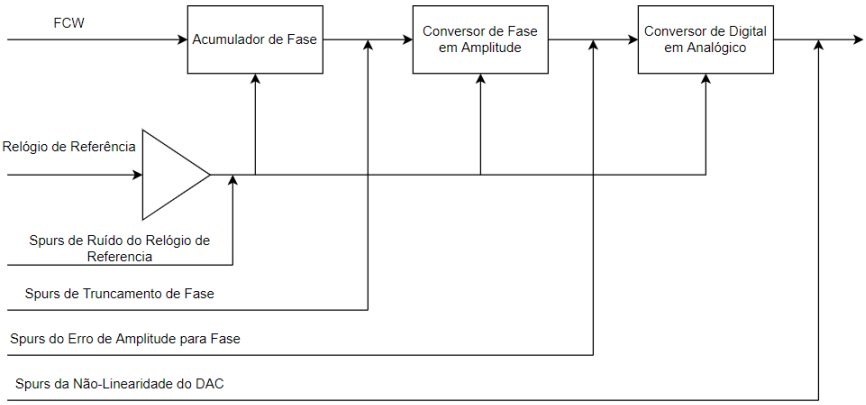


Figura 9 – Fontes de Impulso do DDS

3.2.1. Relógio de Referência

Um DDS funciona como um divisor de frequência de alta resolução com o relógio de referência de entrada e o DAC como saída. As características espectrais do relógio de referência impactam diretamente as da saída. [13] Embora o ruído de fase e os impulsos no Relógio de referência, também, apareçam na saída do DAC, devido à divisão de frequência a sua magnitude é menor. A melhoria, expressa em decibéis, é dado pela seguinte expressão [13]:

$$20 * \log(N) \tag{14}$$

Onde N corresponde à razão entre as frequências de entrada e saída. Por exemplo, a divisão de um relógio de $300\ MHz$ para 80 e $5\ MHz$ resultam numa diferença nos seus gráficos de ruído de fase de:

$$20 * log(\frac{80}{5}) = 24\ dB \tag{15}$$

O caminho do relógio de referência interno do DDS é o contribuinte dominante do ruído de fase do DDS. A modulação da amplitude do relógio gera *impulsos* no seu espectro de saída. [13]

Se uma portadora de RF de $400\ MHz$ com $10\%\ AM$ por um sinal de onda sinusoidal de 100 kHz for observada no relógio de referência com saídas DDS de 10,119 MHz, esse efeito pode ser demonstrado através da sobreposição do relógio de referência e o espectro de saída do DAC, tendo a redução da amplitude nos impulsos do relógio AM na saída do DDS. [13]

O cálculo de atenuação é determinado da seguinte forma,

$$20\ log\ \left(\frac{400}{10{,}119}\right) = 32\ dB \tag{16}$$

Prevê uma melhoria de 32 dB. A atenuação adicional do *impulso* pode dever-se ao facto da onda sinusoidal modulado no relógio de referência encontra-se num limitador ou circuito fechado na entrada DDS. O estágio limitador converte a onda sinusoidal numa onda quadrada e os impulsos AM são convertidos em impulsos PM (modulação de fase). A conversão AM para PM resulta numa atenuação adicional de impulsos, que depende das características do circuito limitador, mas normalmente é da ordem de $-6\ dB$. [13]

O relógio de referência impõe limites ao desempenho do DDS de maneiras que geralmente são reconhecíveis. O relógio de referência geralmente causa esses impulsos DDS que mantêm sua

relação com a portadora à medida que você altera a frequência de saída. Além disso, há algum grau de ruído na entrada de qualquer circuito. Um relógio de referência de alta taxa de variação gasta menos tempo percorrendo a região onde o ruído pode causar jitter.

3.2.2. Truncamento de Fase

Considerando um DDS com um acumulador de fase de 32 bits. Caso fossem mantidos todos os 32 bits, seria ocupada uma grande área de matriz, dissipando uma potência significativa. Truncando o valor do acumulador de fase, ou seja, passando apenas os bits mais significativos do acumulador para o conversor de fase para amplitude, reduz a dissipação de energia e a área de matriz, bem como a resolução de fase do mapeador de ângulo para amplitude. Os modelos de mecanismo de impulso de truncamento de fase do DDS como uma fonte de ruído somado a um sintetizador ideal. [13]

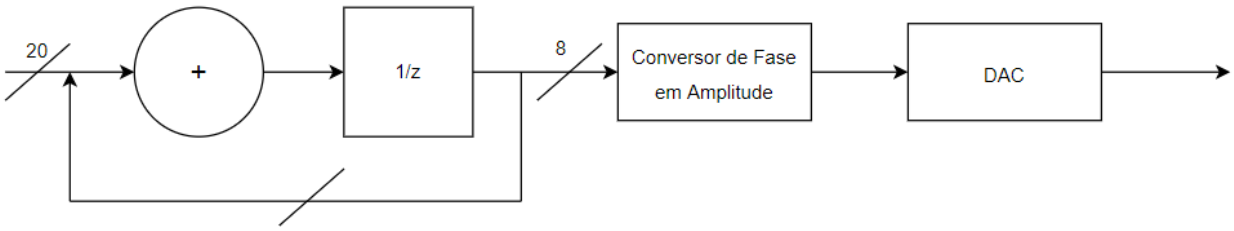


Figura 10 – Sintetizador Ideal

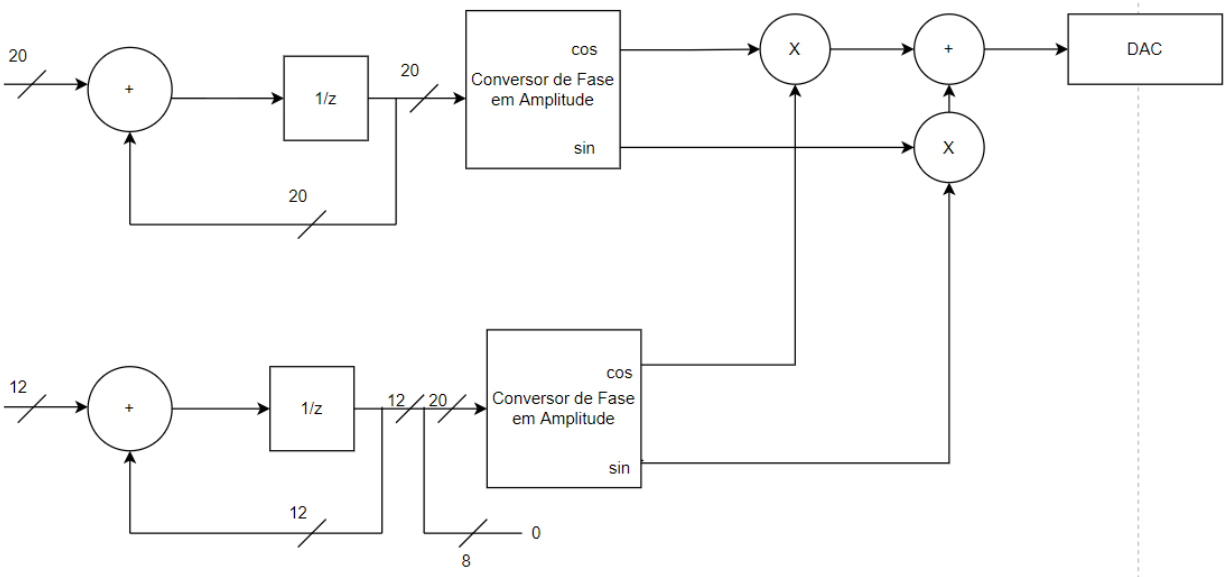


Figura 11 - Modelos de mecanismo de impulso de truncamento de fase do DDS como uma fonte de ruído somado

O exemplo demonstra uma diminuição do acumulador de fase de 20 bits para 8 bits. M é a largura da palavra de sintonia do acumulador de fase. Cada atualização do relógio de referência adiciona o valor de M à saída do acumulador. A saída é dividida em duas seções, a palavra de fase truncada, P, que é enviada ao mapeador, e os bits descartados, $D = M - L$. À medida que o valor

na seção descartada se acumula, este transborda para a palavra de fase truncada. Um efeito deste *overflow* é a produção de impulsos de modulação de fase. Um segundo efeito é que esses estouros mantêm a resolução de frequência total de T. Se nenhum bit na parte descartada for definido como lógico um, não ocorrerão *impulsos* de truncamento de fase. [13]

Quando a palavra de controle de frequência de entrada, FCW, é representada em M bits, o valor da fase do acumulador de fase é atualizado como,

$$\phi[n+1] = (\phi[n] + FCW) * mod(2^M) \tag{17}$$

Sendo truncado para um valor de fase L bits. Portanto, o erro de truncamento pode ser obtido através da equação dada por,

$$\psi[n+1] = (\psi[n] + R) * mod(2^{M-L}) \tag{18}$$

Onde R é o valor menos significativo de (M-L) bits da palavra de controlo da frequência,

$$R = FCW - \left[\frac{FCW}{2^{M-L}}\right] * 2^{M-L} \tag{19}$$

Por fim, a saída de cosseno do DDS é determinada por,

$$x[n] = \cos\left(\frac{2\pi*(\phi[n]-\psi[n])}{2^M}\right) \tag{20}$$

Efetuando a separação dos respetivos senos e cossenos de $\phi[n]$ e $\psi[n]$ obtém-se,

$$x[n] = \cos\left(\frac{2\pi*(\phi[n])}{2^M}\right) * \cos\left(\frac{2\pi*(\psi[n])}{2^M}\right) + \sin\left(\frac{2\pi*(\phi[n])}{2^M}\right) * \sin\left(\frac{2\pi*(\psi[n])}{2^M}\right) \tag{21}$$

Neste caso, o cosseno correspondente ao $\phi[n]$ só tem um componente de frequência desejada. O $\psi[n]$ é periódico, bem como $\phi[n]$, o período de $\psi[n]$ é dado por,

$$T(\psi[n]) = \frac{R}{2^{M-L}} \tag{22}$$

No caso de $L > 2$, o período de cos e sin é determinado por $\psi[n]$. Deste modo, o erro de truncamento periódico cria impulsos nas frequências harmónicas de $T(\psi[n])$. Se a palavra de controlo da frequência de entrada for decomposta como,

$$FCW = N * 2^{M-L} + R \tag{23}$$

O valor da fase truncada, L, é aumentado em N ou N+1, e a taxa de incremento em N+1 é dada por $T(\psi[n])$. O valor médio de incremento observado no valor de fase, L, é dado por,

$$T(\psi[n]) = N + \frac{R}{2^{M-L}} \tag{24}$$

Consequentemente, o acumulador de fase, M, convencional com o truncamento, L, pode ser decomposto de forma equivalente em dois acumuladores para L e (M-L) b. A atualização do acumulador (M-L) correspondente ao erro de truncamento é igual à equação 24.

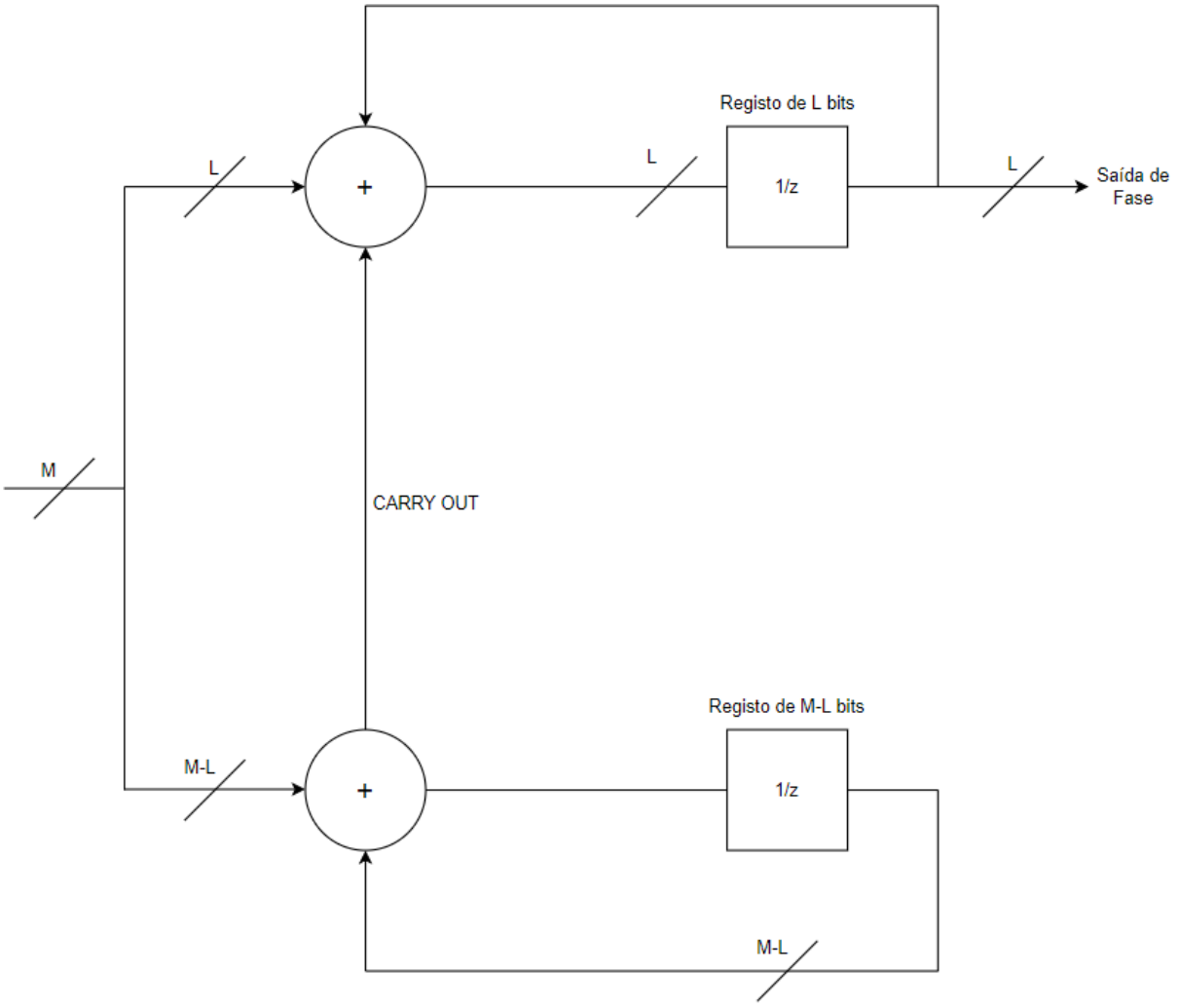


Figura 12 - Uma representação de acumulador de fase convencional com M bits truncados para L bits.

O acumulador, L, é incrementado em $N+1$ quando o acumulador, $(M-L)$, está em *Overflow*. Caso contrário, este é incrementado por N. Assim, o acumulador, $(M-L)$, pode ser simplesmente considerado como uma sequência geradora de $\{0,1\}$ controlada pela palavra de

controlo $(M - L)$ de R. Num acumulador de fase convencional, a sequência de $\{0,1\}$ do acumulador $(M - L)$ é periódico, porque o erro de truncamento é período. Por exemplo, num espetro de sequência gerada pela acumulação de bits, M-L, do acumulador de fase convencional. Os tons harmônicos contribuem para os impulsos na saída final. Se a periodicidade da sequência $\{0, 1\}$ for eliminada, a potência dos impulsos pode ser reduzida significativamente.

O impulso do truncamento de fase faz com que se passe de um acumulador de fase de 15 bits truncado para 9 bits, com uma resolução de DAC de 8 bits a 25 MHz e $FCW = 2^9 + 1$. Com $FCW = 1$, verifica-se que os impulsos de truncamento de fase, não têm alteração no nível de decibéis com a FCW. Portanto, os impulsos de truncamento de fase são proporcionais ao tamanho do *LSB* na palavra de fase e, portanto, não são problemáticos. Por outro lado, quando existe um acionamento da *PLL*, os impulsos de truncamento de fase dentro da largura de banda do loop do PLL serão amplificados em,

$$A = 20 * log(N)\ [dB] \tag{25}$$

Onde N é o fator de multiplicação do PLL. A saída da palavra de fase do acumulador deve ser acrescer, no mínimo, três bits que a resolução do DAC.

Pode-se calcular que a magnitude de pico do pior caso atribuível à largura da palavra de fase é $-6{,}02P\ dBc$, onde P é o número de bits na palavra de fase. Assim, uma resolução de fase de 12 a 19 bits produz uma magnitude de $-72\ a\ -114\ dBc$. [13]

O DAC de 8 bits tem resultados de cinco números efetivos de bits que resultam num intervalo de ruído de quantificação de $[30{,}36]\ dB$. A largura da palavra de sintonia tem uma fração eliminada por meio de truncamento, combinando com a frequência do relógio de referência para revelar o deslocamento de frequência das bandas laterais moduladas em fase de impulsos de truncamento. Pode-se obter esta frequência através da seguinte expressão, [13]

$$f_s = f_{ref} * \frac{M}{2^N} \tag{26}$$

Onde f_s é frequência de deslocamento de impulso, f_{ref} é a frequência de saída do relógio de referência, M é o valor decimal dos bits descartados e N é o número de bits descartados. Essas bandas laterais aparecem em ambos os lados da fundamental. Se a frequência de deslocamento for

maior que a frequência de saída, ou maior que a diferença entre a saída e as frequências de Nyquist, as bandas laterais estão em torno de DC, Nyquist ou ambas.

3.2.3. Sobre amostragem

A densidade espectral da portadora para o ruído varia com o FCW. A magnitude de impulso de pior caso é dada por $A = -6{,}02 * N$, onde N é o número de bits no registador de fase usado para endereçar a ROM. Uma melhoria na densidade espectral de saída pode ser observada reduzindo o FCW. Por exemplo, para um DDS com $N = 15\ bits$, $FCW = 1$ e $f_{clk}\ = 25$ MHz.

$$f_{out} = \frac{25000000}{2^{15}} * 1 = 762{,}93\ Hz \tag{27}$$

A frequência de Nyquist é obtida por,

$$f_s = 2 * f_{out} \tag{28}$$

A razão de sobre amostragem é definida como,

$$OSR = \frac{f_s}{2 * f_{out}} \tag{29}$$

Substituindo f_s e f_{out} tem-se,

$$OSR = \frac{2^N}{2 * FCW} \tag{30}$$

Onde se obtém,

$$OSR = \frac{2^{N-1}}{FCW} \tag{31}$$

Para FCW = 1, a taxa de sobre amostragem é 2^{14}. Para cada duplicação da taxa de sobre amostragem sobre o pior caso pode ser observada uma melhoria de $3\ dB$ no espectro de ruído de quantização dentro da banda. Neste caso será observada uma melhora de

$$A = \ 14 * 3 \ = \ 42\ dB \tag{32}$$

O ruído de quantização deve-se pelo DAC de 8 bits (5 ~ 6 bits efetivos) estar no valor de $-30\ dBc$, portanto, o ruído de quantização com $OSR = 2^{14}$ está abaixo da portadora por,

$$A = \ -30 - 3 * 14 = \ -72\ dBc \tag{33}$$

3.2.4. Conversão de Fase em Amplitude

Os projetos DDS podem implementar o bloco de conversão de fase-amplitude, reduzindo o a área da matriz e o consumo de energia, bem como a tabela de consulta de ROM. Pode ser feita uma abordagem algorítmica, aumentando a eficiência do hardware, mas as suas aproximações podem gerar níveis de impulso mais elevados. [13]

A conversão de fase para amplitude de uma onda sinusoidal amostrada no tempo é,

$$V_{a_i} = V_p * \sin(\varphi_i) \tag{34}$$

Onde V_{a_i} é amplitude de amostragem, V_p é metade da tensão de escala total do DAC e φ_i é o valor da palavra de fase da amostra.

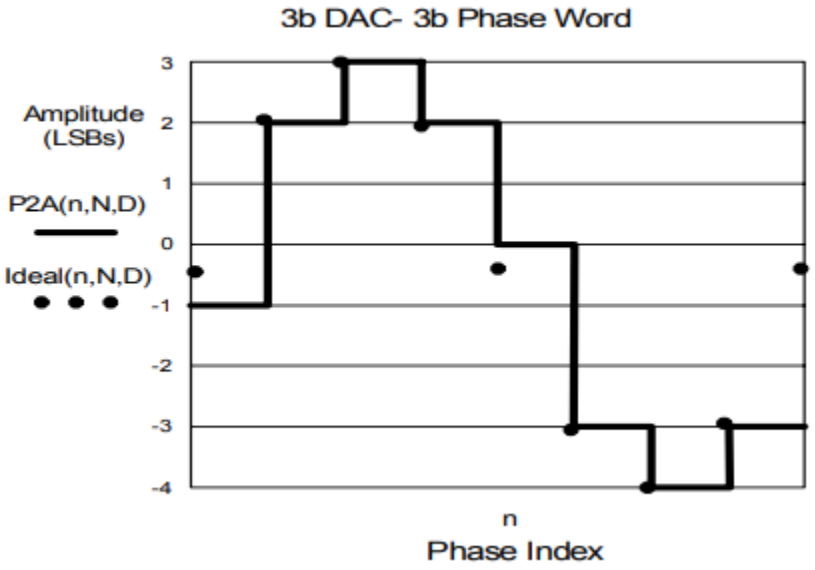


Figura 13 – DAC de 3 bits

Existe um erro residual, uma vez que a tensão de amostragem que o sistema não corresponde a um código DAC, onde o sistema selecionará um código mais próximo. Se a palavra de fase tiver poucos bits, o cálculo desta tensão pode efetuar a seleção do código DAC num intervalo maior, aumentando o número de erros. Quando existe uma maior retenção de bits, menor é o erro residual. Para garantir que todos os códigos DAC estejam disponíveis para o conversor fase-amplitude, uma deve-se definir uma palavra de fase com um intervalo de diferença entre de 3 bits com DAC. A transformada de Fourier de seno no domínio do tempo ilustra o gráfico espectral correspondente

aos impulsos de frequência. O erro pode ser considerado como o sinal modulador a atuar na onda sinusoidal. As localizações de frequência dos impulsos resultantes podem ser determinadas e suas amplitudes são aproximadas, embora estas estejam dependentes do design do sistema. Em alguns casos, a amplitude dos impulsos do pior caso tem um intervalo $[-12\ ,-24\ ]dB$ abaixo do nível de ruído de quantização do DAC. O ruído de quantização (SNR) é proporcional à resolução do DAC: [13]

$$SNR\ =\ 6.02 * N\ + 1.76 \tag{35}$$

Onde N é a resolução do DAC em bits.

Um gráfico espectral normalizado do DDS, baseado em tabela de consulta, apresenta, também, uma resposta de impulso que o sinal de erro de amplitude causa. Por exemplo, para uma palavra de ajuste de 15 bits, com uma palavra de fase de 9 bits, onde o relógio de referência é de $25M\ amostras/s$, palavra de ajuste, $FCW = 1$ e a resolução do DAC é de 8 bits, tem uma localização de impulsos devido ao ruído de quantização que depende da palavra de ajuste e da sua arquitetura. O nível de potência está abaixo do SNR esperado do DAC. Um DAC com maior resolução diminuiria as magnitudes destes impulsos. [13]

Em vez de realizar uma transformada de Fourier do sinal de erro de amplitude, consegue-se determinar a frequência dos impulsos mais destacados para uma dada palavra de sintonia. Este método usa uma palavra de ajuste de teste com apenas um conjunto de bits. O espectro resultante consiste na portadora de teste e os seus impulsos, cujos deslocamentos e espaçamentos relacionam-se harmonicamente com a frequência da portadora. A região espectral onde residem estes impulsos é, [13]

$$f_s =\ f_{ref} * \frac{\pi}{2^{b+n}} \tag{36}$$

Onde b representa a localização do bit único declarado na palavra de ajuste e n é a resolução do DAC em bits.

Esta análise pode ser efetuada através da medição e registo a frequência do relógio de referência, f_{ref}, com uma tolerância de $\pm 1\ Hz$. A partir do MSB da palavra de ajuste, destaca-se apenas o b-ésimo bit. O b-ésimo bit é definido como.

$$bth \geq n + 8 \tag{37}$$

Se a largura do DAC for de 10 bits, define-se o décimo oitavo bit ou maior contagem do MSB. [13]

1. Calcula-se a frequência da palavra de sintonia, f_c, f_{ref} e a palavra de sintonia. Utiliza-se a equação 2 para localizar a região de frequência do impulso destacado, que é definido para medir. O espaçamento entre esses conjuntos de impulsos é duas vezes a frequência da palavra de sintonia. Efetua-se a medição e registo da frequência de cada impulso individual no conjunto de pior caso, divide-se individualmente cada uma das frequências do impulso de pior caso definido pela palavra de ajuste. Os resultados provavelmente serão números ímpares consecutivos.
2. Estes resultados estão relacionados harmonicamente com a palavra de afinação. Quando é feita a mudança da palavra de afinação, é feita a previsão das localizações dos impulsos correspondentes multiplicando a nova palavra de afinação por cada valor obtido
3. Termos maiores que o valor de Nyquist, $f_{ref}/2$, serão melhor. Para localizar o melhor ponto, deve-se obter um valor de produto maior que o valor de Nyquist, mas menor que f_{ref}, para isso efetua-se a substração de f_{ref} pelo produto. Através da diferença obtem-se o melhor valor. Para localizar este valor, deve-se obter um produto maior que f_{ref}, efetuando a sua divisão e analisando o resto do quociente. Se o resto estiver abaixo de 0.5, é efetuada a multiplicação por f_{ref}. Caso contrário, é necessário efetuar a substração por um e, por fim, multiplicar esse valor a f_{ref}.

3.2.5. Ruído de quantificação e Não-Linearidades do DAC

O ruído e a distorção de quantização do DAC determinam a relação Sinal-Ruído, SNR. Pode-se calcular uma aproximação de primeira ordem de SNR efetuando a razão entre a potência do ruído de quantização, na largura de banda de Nyquist, e a potência na fundamental. Como resultado, o SNR é proporcional à resolução do DAC em bits. Este cálculo SNR descreve um DAC ideal.

Os DACs reais têm não linearidades devido a incompatibilidades de processo e dimensionamento de peso de bit imperfeito. As características de comutação não ideais também adicionam distorção e não linearidade. Os impulsos do DAC devem-se a características de comutação não ideais, que, juntamente com a não linearidade na função de transferência, aparecem como harmónicas de ordem inferior da fundamental. O ruído de quantização e as não-linearidades do DAC produzem uma resposta que consiste nos impulsos da onda fundamental harmonicamente relacionados. Esta relação é a chave para entender como prever a localização da frequência dos impulsos dominantes.

O DAC é um sistema de amostragem de tempo. Como resultado, os harmônicos da portadora, o relógio de referência e as harmónicas do relógio de referência criam vários produtos de soma e diferença. A relação matemática definida destes produtos torna possível a previsão das localizações dos impulsos. Harmônicos fora da primeira zona Nyquist são mapeados, novamente, para a primeira zona Nyquist.

Por exemplo, um DDS sintonizado em $25\,MHz$, com um relógio de referência de $100\,000\,000\ amostras/s$ gera harmônicos ímpares de baixa ordem. Uma vez que os harmónicos excedem a frequência de Nyquist, estes retornam à primeira zona de Nyquist de maneira previsível, através do produto de diferença. Este DDS tem um DAC de 14 bits. O SFDR (faixa dinâmica livre de impulsos) dentro da largura de banda de $4\ MHz$ é melhor que $-73\ dBc$.

Ao aumentar a sobre amostragem, eleva o f_{ref} para $400\,000\,000\ amostras/seg$ elimina os produtos do terceiro, quinto e sétimo harmônicos dentro da primeira zona de Nyquist.

Existem bastantes benefícios ao executar o DDS e o comparador num simples sub-harmónico do relógio de referência. Estes benefícios permite um jitter reduzido e um filtro de reconstrução mais simples. Como f_{ref} e f_c estão relacionados por uma razão inteira, o ruído de quantificação do DAC e os impulsos causados por outras fontes de erro têm influência sobre os harmónicos de

f_c. Nestes casos, os harmónicos não produzem jitter, porque são coerentes em fase com a portadora.

3.3. Intercalação do DAC

A intercalação mostrou impactos na expansão da largura de banda utilizável de conversores analógicos para digitais. Esta intercalação dos ADC permitiu aumentar o desempenho de alta taxa de amostragem, o mesmo sinal de entrada é alimentado para todos os sub-ADCs, que são amostrados em várias fases do relógio.

No entanto, devido à resposta de retenção de ordem zero dos conversores digitais para analógicos, essa abordagem não é tão direta. A intercalação pode ser agrupada do seguinte modo:

- Intercalação de dados;
- Intercalação de retenção. [14]

É necessário analisar as limitações de desempenho dos ADC para altas taxas de amostragem e largura de banda elevada.

3.3.1. Limitações do DAC para desempenho de alta frequência

O desempenho de alta frequência dos DAC apresenta as seguintes limitações: [14]

1. O DAC tem um comportamento de retenção de ordem zero no domínio do tempo e, consequentemente, uma resposta de frequência $Sinc$ com um nulo localizado na frequência de relógio, resultando numa distorção de amplitude para sinais gerados em alta frequência.

2. De acordo com o teorema de Nyquist, a largura de banda aplicável dos conversores digital para analógico é limitada a $\frac{f_{clk}}{2}$.

3. A expressão para a distorção de terceira ordem é igual a [15]:

$$HD3 = [\frac{R_L * N}{4|Z_o}]^2 \tag{38}$$

Com uma impedância de saída, Z_o, igual a [15]:

$$Z_o = R_O \,||(\frac{1}{jwC_O}) \tag{39}$$

Consequentemente, pode-se observar que o aumento da frequência resultará na degradação da distorção. O requisito no $HD3$ dará um requisito no Z_o e, consequentemente, no C_o. Quando a frequência aumenta, o valor necessário de C_o diminuirá e será mais difícil de conseguir.

3.3.2. DACs intercalados

O diagrama de blocos do DAC *Hold Interleaving* é constituído por ser um DAC intercalados de espera, os mesmos dados são usados para todos os sub-DACs, cada um com relógio na versão deslocada do relógio de amostragem, o que significa que, cada dado digital é amostrado por cada fase do relógio de amostragem [16]. Esta técnica levará ao cancelamento e redução dos picos do DAC escolhendo corretamente os deslocamentos de fase do relógio de amostragem para cada DAC. No entanto, esta abordagem não oferece vantagens significativas para aplicações de banda larga.

3.3.3. DACs de intercalação de dados

Esta abordagem foi desenvolvida para suprimir as imagens de *Nyquist* para que as frequências produzidas de *Nyquist* pudessem ser usadas sem requisitos rigorosos no filtro. [17] Esta supressão é feita tendo as imagens de *Nyquist* com um deslocamento de fase de 180º, e ainda, as frequências fundamentais com a mesma fase. No entanto, essa abordagem exige que a taxa de amostragem de cada DAC seja $N * f_{clk}$.

3.3.4. Data and Hold Interleaving DACs

Nesta abordagem, cada DAC é alimentado com as amostras intercaladas do sinal e em instantes de tempo diferentes. Consequentemente, as primeiras réplicas de imagem $N-1$ e a não linearidade são canceladas, e a operação de banda larga será alcançada. [14]

3.3.5. A abordagem Interleaving e Return to Zero

O diagrama de blocos do DAC de entrelaçamento de dados e de retenção, mostra que cada DAC trabalha em fases intercaladas do relógio e mantém a mesma saída durante todo o período do relógio. Ao adicionar um segundo DAC defasado do primeiro DAC, os componentes do domínio da frequência do primeiro e do segundo DAC podem ser escritos como as seguintes equações [14],

$$DAC_1: H(f) = sinc\left(\frac{\pi * f}{f_{clk}}\right) \sum_{k=-\infty}^{+\infty} h(f - k * F_s) \quad (40)$$

$$DAC_2: H(f) = sinc\left(\frac{\pi * f}{f_{clk}}\right) \sum_{k=-\infty}^{+\infty} h(f - k * F_s) * e^{-\frac{j*\pi*f}{f_{clk}}} \quad (41)$$

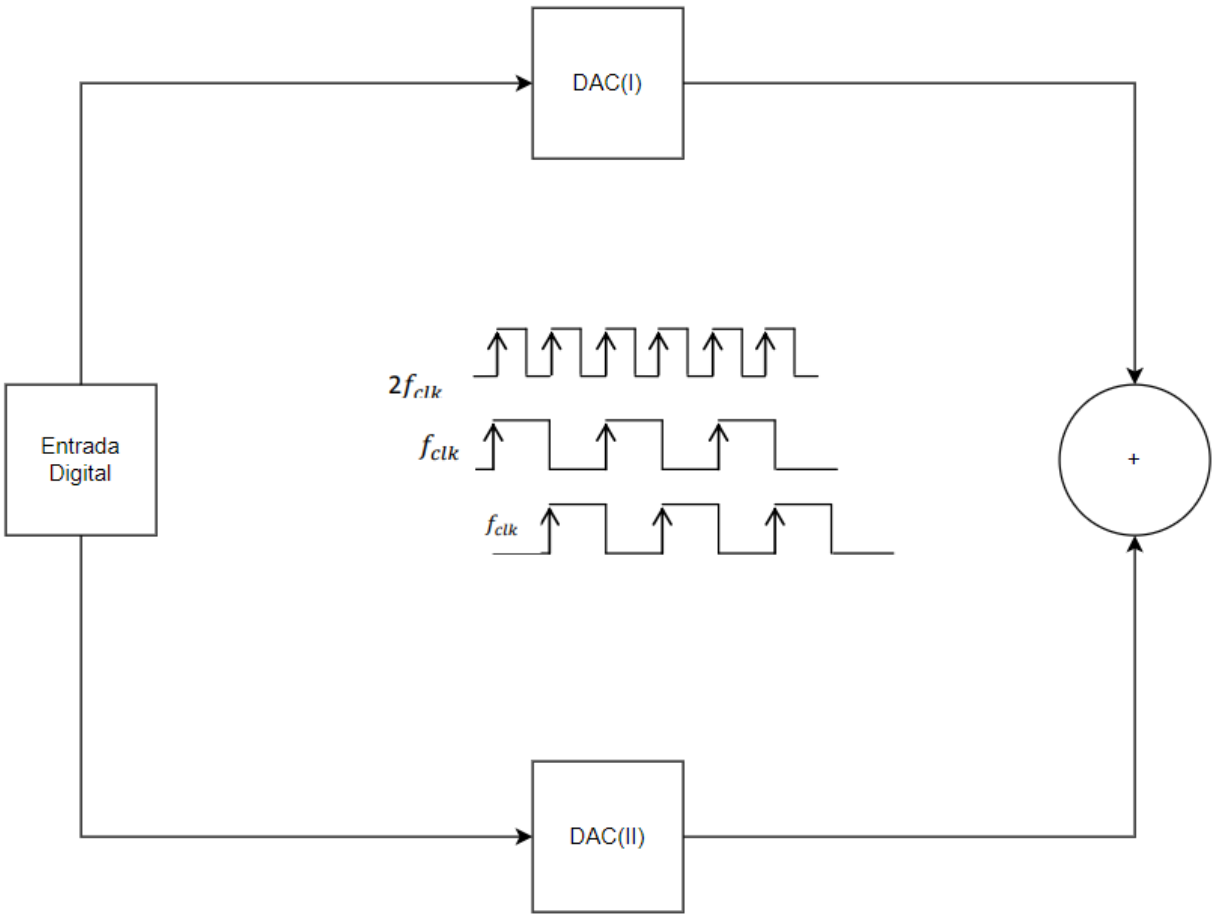


Figura 14 – Diagrama de blocos DAC Intercalados

Como pode ser visto nas fórmulas acima, os dois DACs têm o mesmo sinal para valores pares de k, enquanto têm sinais opostos para valores ímpares. Consequentemente, os valores ímpares de k cancelam quando as saídas dos dois DACs são somadas (técnica de retorno a zero). Portanto, o

espectro resultante será um único DAC a repetir duas vezes a taxa de amostragem, sendo possível gerar frequências de saída além da frequência *Nyquist* dos *DACs* individuais. No entanto, como a função de retenção de ordem zero de cada DAC não muda com esta técnica, estes terão um valor nulo em F_{clk}, que é a nova frequência de Nyquist, a largura de banda utilizável não apresenta melhorias. Para remover o valor nulo para $2 * F_{clk}$, é usada a técnica de retorno a zero como uma chave analógica. Com a técnica de retorno a zero, cada DAC fica ativo apenas por metade do ciclo de relógio, então a função $sinc$ terá seu valor nulo em $2 * F_{clk}$. Consequentemente, pode-se concluir que os dois DACs intercalados com a técnica de retorno a zero são equivalentes a um DAC que tem uma atuação superior, a duas vezes, da taxa de amostragem [14].

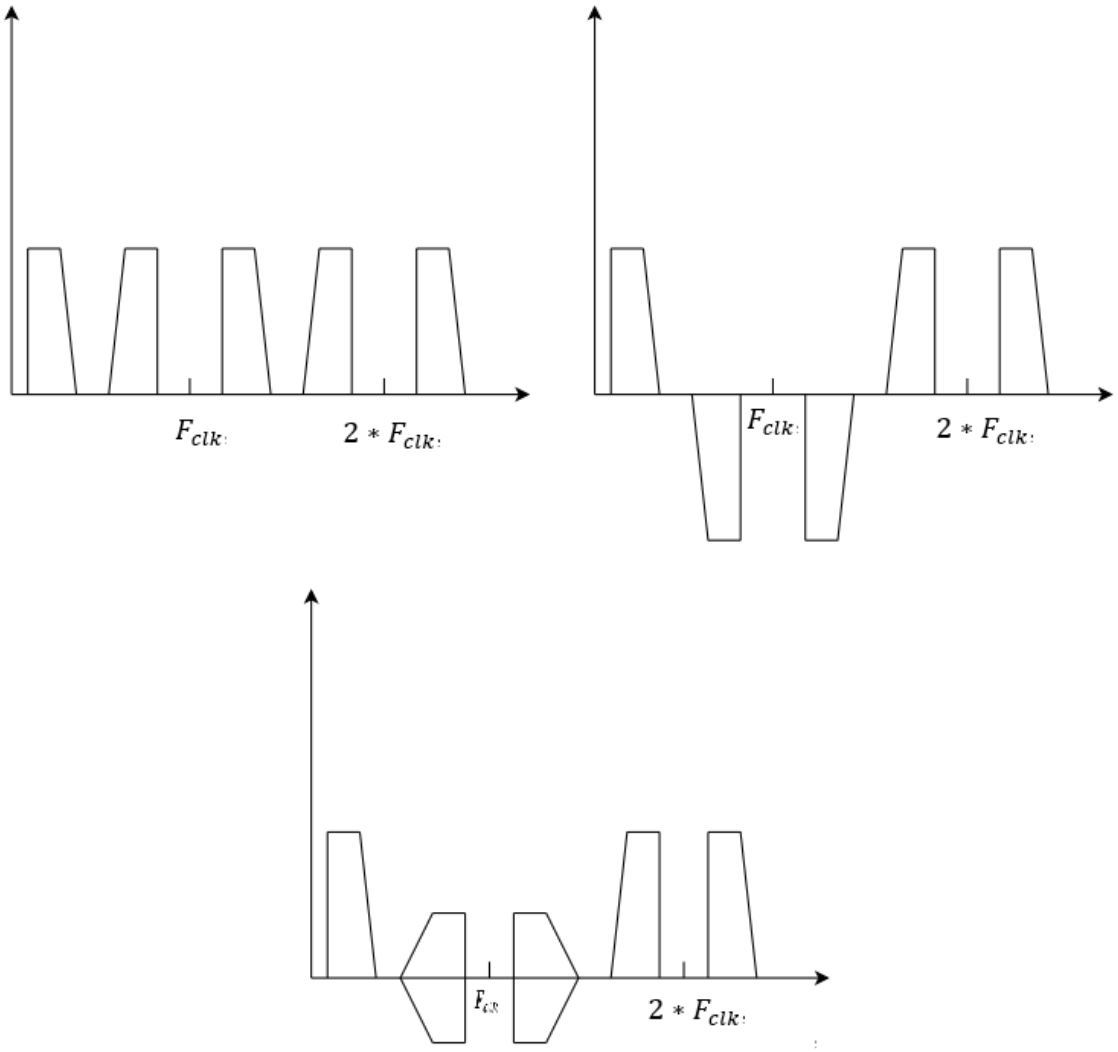


Figura 15 - Réplicas de imagem do primeiro, segundo DAC e réplicas de imagem dos DACs intercalados

Nos DACs intercalados no tempo, o alinhamento de tempo desempenha um papel importante. O cancelamento das imagens não ocorrerá corretamente, havendo picos indesejados dentro da banda de Nyquist, caso haja algum desvio do atraso de meia amostra ideal. Além disso, os dois DACs devem estar equilibrados na sua amplitude. O algoritmo de correção para equilíbrio de amplitude e alinhamento de tempo pode ser aplicado ao sistema pelo uso de filtros de calibração [14].

4. Arquitetura Experimental

4.1. Seleção do Microcontrolador

O gerador de sinais será implementado usando um microcontrolador, sendo que o software será constituído pelas seguintes partes: uma de comunicação, que é responsável pela receção dos dados (frequência e forma de onda) a serem gerados; outra onde estarão as tabelas das cinco formas de onda (sinusoide, quadrada, triangular, dente de serra e ECG); e, por último, a parte principal, que fica a coletar e a transferir os dados para o conversor D/A, de forma a ser obtida uma das ondas definidas nas tabelas.

Estes dispositivos, cada vez mais, apresentam uma maior capacidade de produzir e controlar com precisão formas de onda de várias frequências e perfis, são fornecidas várias fontes de frequências variáveis de baixo ruído de fase com bom desempenho para comunicações, ou simplesmente gerando um estímulo de frequência. Existem vários métodos para geração de frequências, desde técnicas baseadas em *Phase Locked Loop* (PLL) para síntese de frequência muito alta, até programação dinâmica de saídas do conversor digital-analógico (DAC) para gerar formas de onda arbitrárias em frequências mais baixas.

A técnica *DDS* está a ganhar aceitação rapidamente para resolver os requisitos da geração de frequência (ou forma de onda) em comunicações e aplicações industriais, porque os dispositivos de circuitos integrados com um chip único podem gerar formas de onda de saída analógica programáveis de forma simples e com alta resolução e precisão.

4.1.1. Estudo das Características dos Microcontroladores

Efetuou-se um estudo acerca dos diversos microcontroladores disponíveis no mercado, onde serão analisadas na tabela abaixo, as características como a sua velocidade de processamento, o número de bits, a sua resolução ADC, assim como a sua memória FLASH e SRAM, onde o custo, também, será uma característica determinante para a seleção do microcontrolador.

Tabela 1 – Características dos Microcontroladores disponíveis no Mercado

Nome	Microcontrolador	Velocidade CPU	Bits	Resolução ADC	Flash	SRAM	Preço
Arduino Uno	ATMega128P	16 MHz	8	10	32 KB	2 KB	€ 23,75
Arduino Mega	ATMega2560	16 MHz	8	10	256 KB	8 KB	€ 35,04
Arduino Nano	ATMega2560	16 MHz	8	10	26-32 KB	1-2 KB	€ 13,55
STM32L0	Cortex-M0	32 MHz	32	12	64 KB	20 KB	€ 17,90
Arduino MKR1000	Cortex-M0	48 MHz	32	12	256 KB	32 KB	€ 39,36
ESP8266	Tensilica Xtensa LX106	80 MHz	32	10	4 MB	160 KB	€ 4,31
Arduino Due	Cortex-M3	84 MHz	32	12	512 KB	96 KB	€ 59,93
STM32F2	Cortex-M3	120 MHz	32	12	1 MB	128 KB	€ 13,35
STM32F4	Cortex-M4	180 MHz	32	12	2 MB	384 KB	€ 13,35
ESP32	Tensilica Xtensa LX6	240 MHz	32	12	4 MB	320 KB	€ 14,70
Teensy® 4.1	ARM Cortex-M7	600 MHz	64	12	8 MB	1 MB	€ 24,44
RPI A+	-	700 MHz	64	12	SD Card	256 MB	€ 53,18
RPi Zero	-	1 GHz	64	12	SD Card	512 MB	€ 18,39
RPi 3B	-	1.2 GHz	64	12	SD Card	1 GB	€ 47,66

4.1.2. Seleção do Microcontrolador

O microcontrolador selecionado foi o Teensy® 4.1, devido às seguintes características:

- ARM Cortex-M7 em 600 MHz, mesmo seja utilizado $^{1}/_{4}$ da frequência de CPU, permite o desenvolvimento de formas de onda com uma elevada frequência.
- Unidade matemática de ponto flutuante, 64 e 32 bits;
- Flash de 7936K, RAM de 1024K (512K fortemente acoplado), EEPROM de 4K (emulado);
- Expansão de memória QSPI, locais para 2 chips extras de RAM ou Flash;
- Dispositivo USB 480 Mbit/s e host USB 480 Mbit/s;
- 55 pinos de entrada/saída digital, 35 pinos de saída PWM;
- 18 pinos de entrada analógica;
- 8 portas seriais, 3 SPI, 3 portas I2C;
- 2 portas de áudio digital I2S/TDM e 1 S/PDIF;
- 3 CAN Bus (1 com CAN FD);
- 1 porta de cartão SD nativa SDIO (4 bits);
- Ethernet 10/100 Mbit com DP83825 PHY;
- 32 canais DMA de uso geral;
- Aceleração criptográfica e gerador de números aleatórios;
- RTC para data/hora;
- Flex IO programável;
- Pipeline de Processamento de Pixel;
- Acionamento cruzado periférico;
- Gerenciamento de ligar/desligar. [18]

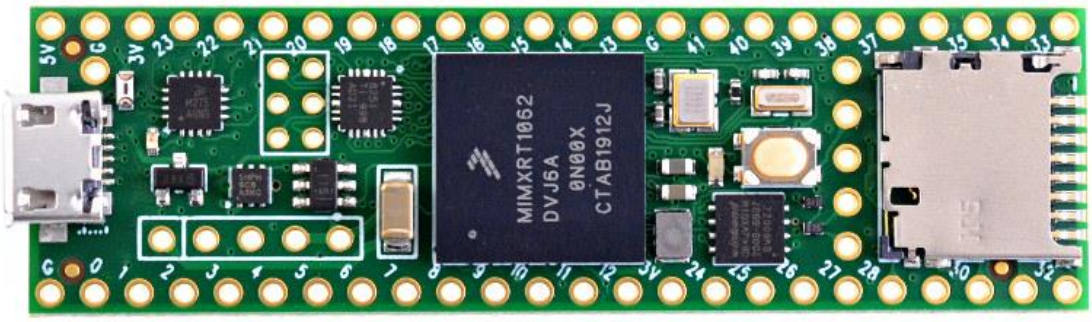


Figura 16 – Teensy 4.1

Em termos de Software, é possível desenvolver o código programável através da plataforma Arduino IDE com o complemento Teensyduino é o principal ambiente de programação para o Teensy. O Teensyduino inclui uma grande coleção de bibliotecas que são testadas e otimizadas para Teensy. outras bibliotecas podem ser instaladas manualmente ou pelo gerenciador de bibliotecas do Arduino. [18]

Em relação ao Processador, pode-se analisar as seguintes características:

- Performance - O ARM Cortex-M7 traz muitos recursos poderosos de CPU para uma verdadeira plataforma de microcontrolador em tempo real. O desempenho da CPU é muito mais rápido do que os microcontroladores típicos de 32 bits;
- Arquitetura Superescalar de Emissão Dupla - O Cortex-M7 é um processador de emissão dupla, o que significa que o M7 pode executar 2 instruções por ciclo de relógio a 600 MHz, para executar simultaneamente depende das instruções e dos registos de ordenação do compilador. Os benchmarks iniciais mostraram que o código C++ compilado pelo Arduino tende a atingir 2 instruções cerca de 40% a 50% do tempo enquanto executa um trabalho numericamente intensivo usando números inteiros e ponteiros;
- Unidade de ponto flutuante - A FPU executa matemática de precisão dupla de 32 bits e 64 bits em hardware. A velocidade de flutuação de 32 bits é aproximadamente a mesma velocidade da matemática inteira. A precisão dupla de 64 bits é executada na metade da velocidade do float de 32 bits;
- Memória fortemente acoplada - A memória é um recurso especial que permite ao Cortex-M7 acesso rápido de ciclo único à memória usando um par de barramentos de 64 bits. O barramento fornece um caminho de 64 bits para buscar instruções, o outro barramento é um par de caminhos de 32 bits, permitindo que o M7 execute até 2 acessos separados à memória no mesmo ciclo. Estes barramentos de velocidade extremamente alta são separados do barramento AXI principal do M7, que acede a outras memórias e periféricos;
- Cache - Dois caches de $32\ KB$, um para instruções e outro para dados, que são usados para acelerar o acesso repetitivo à memória de cache;
- Processamento de sinal digital - As instruções de extensão DSP aceleram o processamento de sinal, filtros e transformada de Fourier. A biblioteca de áudio faz uso dessas instruções DSP automaticamente. [18]

4.2. Implementação de um DDS

O gerador de sinais será implementado usando o microcontrolador do Teensy 4.1, ARM Cortex-M7, sendo que o software será constituído pelas seguintes partes: uma de comunicação, que é responsável pela receção dos dados (frequência e forma de onda) a serem gerados; outra onde estarão as tabelas das cinco formas de onda (sinusoide, quadrada, triangular, dente de serra e ECG); e, por último, a parte principal, que fica a coletar e a transferir os dados para o conversor D/A, de forma a ser obtida uma das ondas definidas nas tabelas.

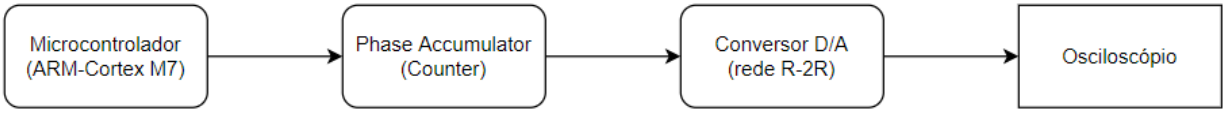


Figura 17 - Diagrama de Blocos do Gerador de Funções

4.3. Módulo de Geração

O gerador de sinais desenvolvido utiliza a técnica da síntese digital direta com uma palavra de ajuste de 24 bits e um conversor D/A de 8 bits, baseado numa rede R-2R.

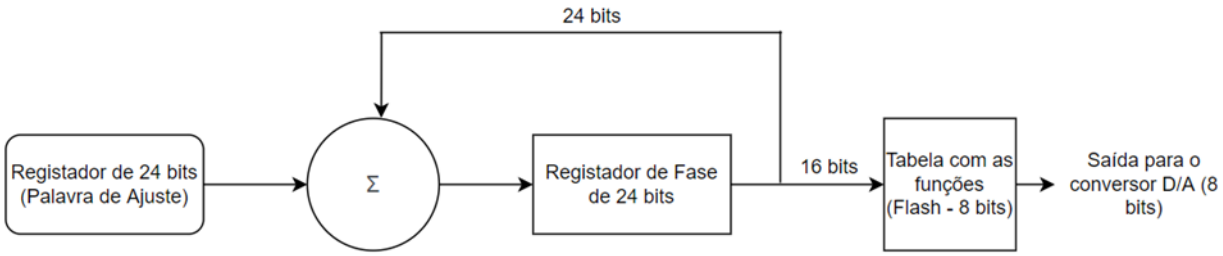


Figura 18 - Diagrama de Blocos do módulo de Geração

Por exemplo, considerando que se pretende $f_o = 20\ kHz$, e que se obtém o seguinte tamanho da palavra de ajuste:

$$2^n = 2^{24} = 16777216 \tag{43}$$

Sabendo que a frequência de Cristal, $f_{XTAL} = 600\ MHz$ e que, por exemplo, o modo do timer está configurado para um máximo de 200 contagens, obtém-se a seguinte f_{clk}:

$$f_{clk} = \frac{f_{XTAL}}{200} = 3\ MHz \tag{44}$$

Sabendo o valor da frequência de relógio, é possível determinar a palavra de ajuste de frequência, M.

$$M = \frac{f_o * 2^n}{f_{clk}} \tag{45}$$

Substituindo os valores determinados:

$$M = \frac{(20*10^3)* 2^{24}}{3*10^6} = 111848 \tag{46}$$

Para obter a menor frequência, deve-se considerar $M = 1$, ficando com a seguinte resolução de frequência:

$$f_o = \frac{1* 3*10^6}{2^{24}} = 178.81\ mHz \tag{47}$$

A frequência de relógio determinada, f_{clk}, é equivalente para que um circuito integrado DDS, com $n = 24\ bits$, tenha uma resolução de $9{,}536\ mHz$. Como o DDS é desenvolvido por software, leva 9 ciclos de instrução para realizar a soma no registador de fase, procurar o nível equivalente na tabela e enviar para o conversor D/A.

O acumulador de fase é composto pelo somador e pelo registador de fase. Como o gerador é construído, no microcontrolador, o acumulador de fase e o somador não existem fisicamente como um bloco isolado, sendo que a função do somador é efetuada através de instruções de adição na unidade lógica e aritmética (ULA) do microcontrolador.

4.4. Módulo de Controlo

Na figura abaixo, tem-se o diagrama de blocos mostrando o módulo de controlo:

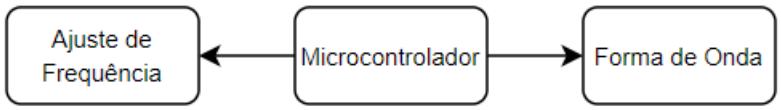


Figura 19 – Diagrama de Blocos do Módulo de Controlo

Como se verifica na figura, o módulo de controlo é fornecido pelo ajuste de frequência, que consiste num potenciómetro, onde à medida que se vai aumentando o seu valor de leitura, aumenta o período da onda gerada, diminuindo o valor da frequência gerada, no software está definido um valor máximo de contagem de 256.

Resumindo, conforme a expressão de f_{clk}.

$$f_{clk} = \frac{f_{XTAL}}{Valor\ de\ Contagem} \tag{48}$$

Quanto maior for o valor da contagem, menor vai ser a frequência de relógio.

Através um display touch 2.8'' compatível com Teensy 4.1, o utilizador define qual a forma de onda a ser visualizada, sendo que apresenta as seguintes opções:

1. Onda sinusoidal;
2. Onda Quadrada;
3. Onda Dente de Serra;
4. Onda Dente de serra inversa;
5. Onda triangular;
6. Onda semelhante a um Eletrocardiograma.

4.5. Conversor Digital-Analógico

O conversor digital para analógico, DAC, é um sistema que converte o sinal digital num sinal analógico correspondente. O Conversor Analógico para Digital (ADC) é um sistema que converte sinais analógicos em digitais.

O DAC tem múltiplas aplicações, como áudio, vídeo, mecânica e para fins de comunicação. A adequação de um DAC é determinada pelos seus parâmetros, entre os quais estão a resolução e frequência de amostragem, existindo várias arquiteturas de DAC como, por exemplo, o *DAC R-2R Ladder*.

O *R-2R Ladder* é um dos conversores DAC mais simples, utilizando apenas dois valores de resistência, sendo extensível a qualquer número de bits. Além disso, a impedância de saída é sempre igual a R, que é independentemente do número de bits, simplificando a filtragem e o design do circuito.

O número de níveis é igual a duas potências do número de bits, ou seja, 2^N. Neste projeto, 8 bits significam que haverá 256 níveis, podendo calcular a saída de tensão máxima por esta equação:

$$v_{o_{max}} = \frac{V_{high}}{2^N} * (2^N - 1) \tag{49}$$

A equação de saída de tensão para o próprio *R-2R DAC* é dada por:

$$v_o = \frac{V_{b_0}}{2^N} + \frac{V_{b_1}}{2^{N-1}} + \frac{V_{b_2}}{2^{N-2}} + \cdots + \frac{V_{bn}}{2^1} \tag{50}$$

Como está definido um *R-2R DAC* de 8 bits, o $N = 8$ bits, a tensão de saída pode ser calculada por:

$$v_o = \frac{V_{b_0}}{2^8} + \frac{V_{b_1}}{2^7} + \frac{V_{b_2}}{2^6} + \frac{V_{b_3}}{2^5} + \frac{V_{b_4}}{2^4} + \frac{V_{b_5}}{2^3} + \frac{V_{b_6}}{2^2} + \frac{V_{b_7}}{2^1} \tag{51}$$

Existem 8 entradas digitais que representam cada bit, estando no formato LSB. Foram dimensionadas resistências, no valor de $10\ k\Omega$, uma vez que eram as disponibilizadas, o valor mais próximo do dobro do valor destas resistências encontra-se no valor de $22\ k\Omega$, podendo obter, aproximadamente, 2R.

A avaliação de desempenho do DAC também é feita pelos diferentes tipos de erros descritos a seguir:

1. Não linearidade diferencial (DNL): É a diferença entre uma altura real do degrau e o valor ideal de LSB. Portanto, se a altura do degrau de um DAC é igual ao LSB, então o DNL é zero. Se DNL for maior que 1 LSB, então o conversor pode se tornar não monótono;
2. Não-linearidade integral (INL): Desvio dos valores na função de transferência real de uma linha reta de melhor ajuste;
3. Erro de ganho: Diferença entre a inclinação da curva ideal e a saída real do DAC;
4. Erro de Offset: É a diferença de tensão constante entre a saída ideal do DAC e a saída real;
5. Não Monótona: É a diminuição da tensão de saída com o aumento da entrada digital.

Este DAC pode apresentar limitações a altas frequências, nomeadamente, na ordem dos *Gigahertz*, não sendo possível calcular alguns parâmetros. A sua saída analógica pode seguir um ajuste de entrada linear muito bom para frequências altas, tendo uma resposta monótona para essas frequências, demonstrando uma boa linearidade e um bom desempenho de erro.

4.6. Descrição do Software de Controlo

O software de controlo é composto por duas rotinas como, as de "Gerar Forma de Onda" e "Ajustar Frequência".

A rotina de "Geração de Onda" é executada a cada milissegundo, verificando o retorno de cada uma das seis colunas, onde o utilizador selecionará qual a forma de onda pretendida. Na função principal, estão definidas as seguintes opções:

- Primeira opção, a forma de onda apresentada é a sinusoidal, tendo, inicialmente, uma frequência de $415\ Hz$, que poderá variar, conforme a variação da rotina de "Ajustar Frequência";
- Segunda opção, a forma de onda apresentada é a retangular, tendo, inicialmente, uma frequência de $415\ Hz$, que poderá variar, conforme a variação da rotina de "Ajustar Frequência";
- Terceira opção, a forma de onda apresentada é a dente de serra, tendo, inicialmente, uma frequência de $415\ Hz$, que poderá variar, conforme a variação da rotina de "Ajustar Frequência";
- Quarta opção, a forma de onda apresentada é a dente de serra inversa, tendo, inicialmente, uma frequência de $415\ Hz$, que poderá variar, conforme a variação da rotina de "Ajustar Frequência";
- Quinta opção, a forma de onda apresentada é triangular, tendo, inicialmente, uma frequência de $415\ Hz$, que poderá variar, conforme a variação da rotina de "Ajustar Frequência";
- Sexta opção, a forma de onda apresentada é, semelhante, a uma eletrocardiograma, tendo, inicialmente, uma frequência de $415\ Hz$, que poderá variar, conforme a variação da rotina de "Ajustar Frequência".

A rotina de "Ajustar Frequência", consiste num potenciómetro, que fará com que haja um aumento da frequência, fazendo com que haja uma diminuição do período da onda visualização, uma vez que as ondas se encontram definidas em tabelas, o software correrá um ciclo for, que corresponderá à leitura de cada um dos 256 bits, convertidos para uma forma de onda, o que mudará neste ciclo é o tempo de leitura de cada um destes bits, à medida que esta função "Ajuste Frequência" é utilizada, diminui o tempo de atraso da leitura de cada um dos bits, fazendo com que haja uma maior frequência.

4.7. Descrição do Software de Geração

O software de geração tem a função de receber os dados e a geração da onda.

A rotina de recebimento de dados entra em execução sempre que chega um pacote de dados e provoca a geração de uma interrupção, a função principal desta rotina é receber os 4 bytes do módulo de controlo, sendo que os três primeiros bytes correspondem à palavra de ajuste (24 bits) e o último byte corresponde à forma de onda a ser gerada.

A rotina de geração executa a função do acumulador de fase e enviar os dados coletados na tabela para o conversor D/A.

4.8. Esquema Elétrico

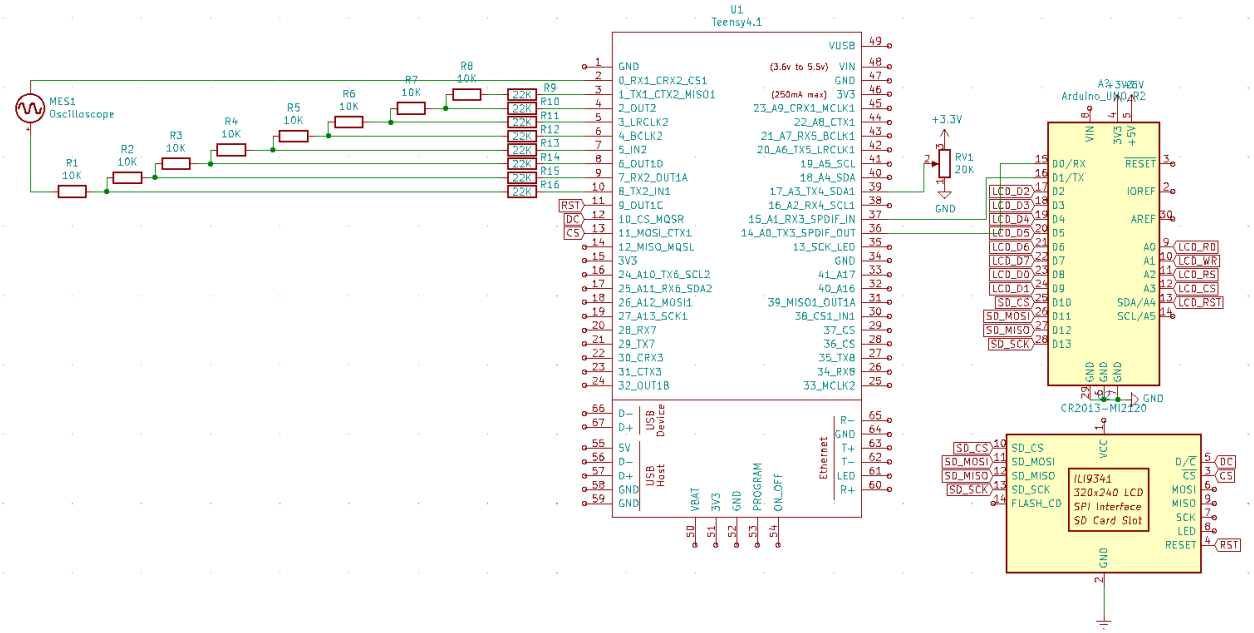


Figura 20 - Esquema Elétrico do Gerador de Sinais

4.9. Comunicação em série

Uma vez que não foi possível conectar o display ao Teensy 4.1, foi necessário adotar o método de comunicação em série entre o Teensy 4.1 e o Arduino UNO, onde este último será um intermediário na comunicação display-Teensy, o Arduino UNO é compatível com o display e através desta comunicação é possível efetuar a seleção da forma de onda, através do display touch.

A comunicação em série é um método de comunicação que utiliza uma ou duas linhas de transmissão para enviar e receber dados, e esses dados são enviados e recebidos continuamente por um bit de cada vez, para existir contenção de custos de material de fiação e equipamentos de retransmissão.

O Arduino usa comunicação em série para transferir dados entre o microcontrolador e o PC ou com qualquer outro microcontrolador. Um barramento em série é usado para esta comunicação que consiste em dois terminais, ao qual um serve para envio de dados e outro para recebimento de dados. Assim, todos os dispositivos que utilizam protocolo em série possuem dois pinos deste tipo de comunicação:

- RX Receiver;
- TX Transmitter.

É importante considerar que estes são específicos para o próprio dispositivo, ou seja, se for necessário comunicar entre os dois microcontroladores o pino RX do primeiro será conectado ao pino TX do segundo e da mesma forma o pino TX do primeiro com o pino RX do segundo.

Os LEDs TX e RX na placa piscam quando qualquer tipo de dado está a ser transmitido ou recebido usando a porta série. Os LEDs não piscam se a comunicação em série for feita através dos pinos 0(RX),1(TX) em sua placa. Esses dois pinos são designados para conectar seu próprio dispositivo serial, independentemente do cabo USB estar ou não conectado ou não. O Led TX a piscar significa que a placa está a enviar algo através da função Serial.print().

Infraestrutura de protocolos UART necessária para TX/RX

Se o utilizador pretende com qualquer dispositivo externo, existem alguns requisitos para estabelecer a conexão via série, como:

1: Pinos necessários – A infraestrutura geral UART exige os pinos RX/TX. RX para receção e TX para transmissão;

2: Estrutura de Pacotes – UART significa (Universal Asynchronous Receiver & Transmitter), é uma comunicação assíncrona porque não há compartilhamento de relógio comum entre dispositivos. Ambos os dispositivos onde a comunicação série é necessária devem concordar com a mesma estrutura em quais dados estão sendo enviados e com que velocidade os dados são enviados; isso ajudará o UART a amostrar os dados e converter dados brutos em pacotes de dados;

3: Taxa de transmissão – A mesma taxa de dados é obrigatória para compartilhar dados entre dois dispositivos UART, ambos os dispositivos devem ser configurados na mesma taxa de dados para receber e enviar a devida informação.

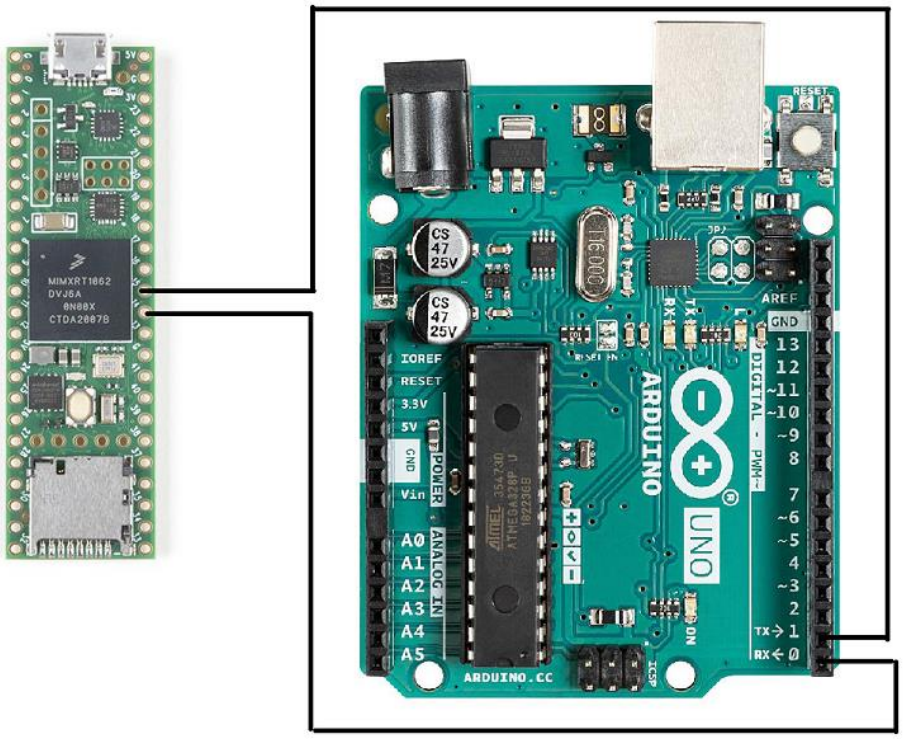


Figura 21 - Conexão Teensy 4.1 - Arduino

5. Simulação e Resultados Experimentais

Neste capítulo discute a implementação do modelo do DDS e os seus resultados de simulação. A modelagem e simulação de alto nível de um DDS foi realizada usando o Arduino IDE para entender a funcionalidade geral e o fluxo da entrada para a saída. O DDS também foi modelado usando um osciloscópio para uma visualização real das ondas. Nesta visualização. o Relógio de referência usado é um relógio jitter, para entender o efeito do relógio jitter no espectro de saída.

As simulações ajudam a aumentar a compreensão do projeto sob operações não ideais. As simulações servem como um protótipo para um projeto antes que ele seja realmente construído em hardware. Essas simulações são usadas para entender o efeito das características não ideais dos blocos de construção de um DDS em seu espectro de saída. A operação do DDS foi explicada nos capítulos anteriores. A implementação de um sintetizador digital direto consiste num *Phase Register (PR), Phase Accumulator (PA)* e *Look up Table (LUT)*. Este modelo foi desenvolvido para estudar a funcionalidade do DDS. O PR contém a palavra de sintonia de frequência. O bloco de atraso unitário junto com um somador e um loop de feedback representam o PA. Esses blocos de atraso unitário atuam como registrador. O LUT é implementado usando um LUT sinusoidal. A entrada do LUT é dimensionada usando o bloco de ganho.

A cada pulso de relógio, o conteúdo do PR é adicionado ao do PA. O PA gera os valores de fase da onda sinusoidal de saída. A saída do PA serve como endereço da LUT. Cada vez que o PA contém todos os elementos acumulados, a LUT emite valores amostrados da onda sinusoidal. Esta saída da LUT representa um ciclo da forma de onda sinusoidal, uma vez que a LUT contém valores amostrados de um ciclo da onda.

A taxa de *overflow* do PA depende do número de bits do *PA* e da palavra de sintonia de frequência. Quanto maior o tamanho da palavra de sintonia de frequência, mais rapidamente o *buffer* do PA fica cheio. A frequência de saída do *DDS* é diretamente proporcional à palavra de sintonia de frequência. Portanto, quanto maior a palavra de sintonia de frequência, maior é a frequência de saída e mais rápido o *overflow* do PA. Estas simulações ajudam a entender o fluxo do sinal pelo DDS e o overflow do PA e a relação da frequência de saída com o PA. A frequência da onda de saída depende da taxa de overflow do PA e da palavra de sintonia de frequência. Esta taxa de overflow depende da palavra de sintonia de frequência armazenada no registrador de fase.

5.1. Visualização das Ondas Geradas na Simulação

5.1.1. Simulação do Acumulador de Fase

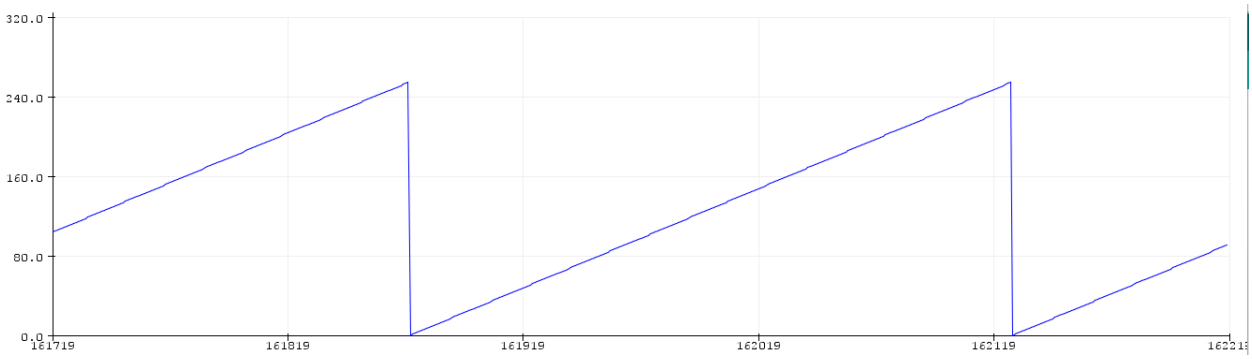


Figura 22 – Simulação do Acumulador de Fase~

5.1.2. Simulação de Onda Sinusoidal

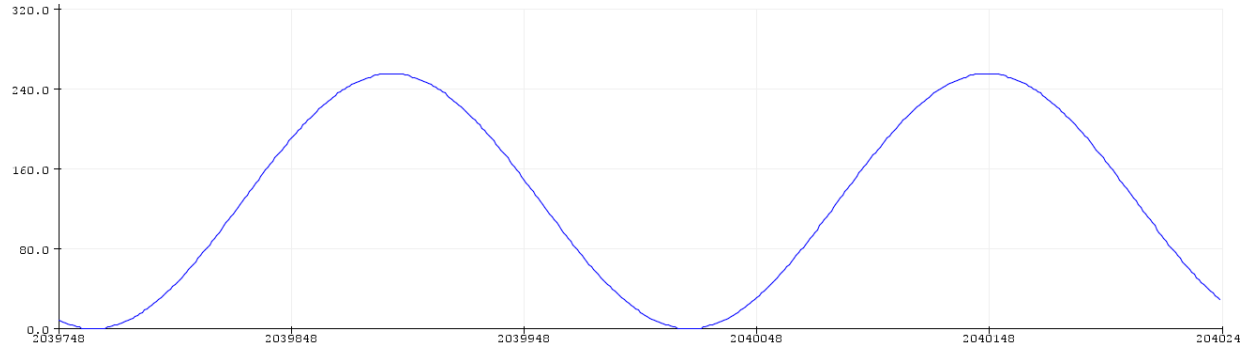


Figura 23 – Onda Sinusoidal Simulada

5.1.3. Simulação de Onda Quadrada

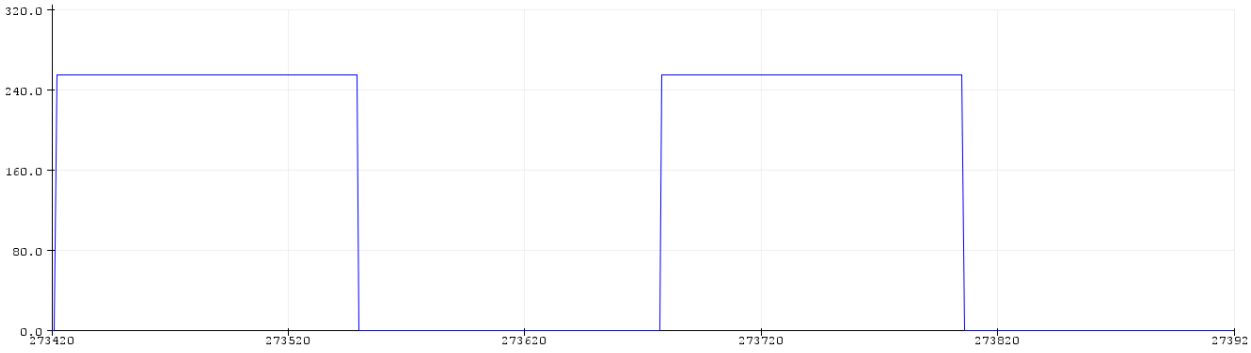


Figura 24 – Onda Quadrada Simulada

5.1.4. Simulação de Onda Dente de Serra

Figura 25 – Simulação de Onda Dente de Serra

5.1.5. Simulação de Onda Dente de Serra Invertida

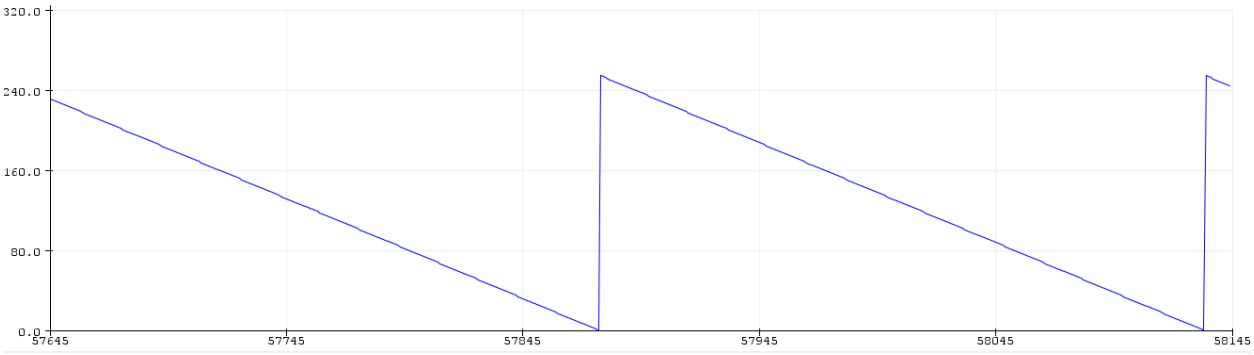


Figura 26 - Simulação de Onda Dente de Serra Invertida

5.1.6. Simulação de Onda Triangular

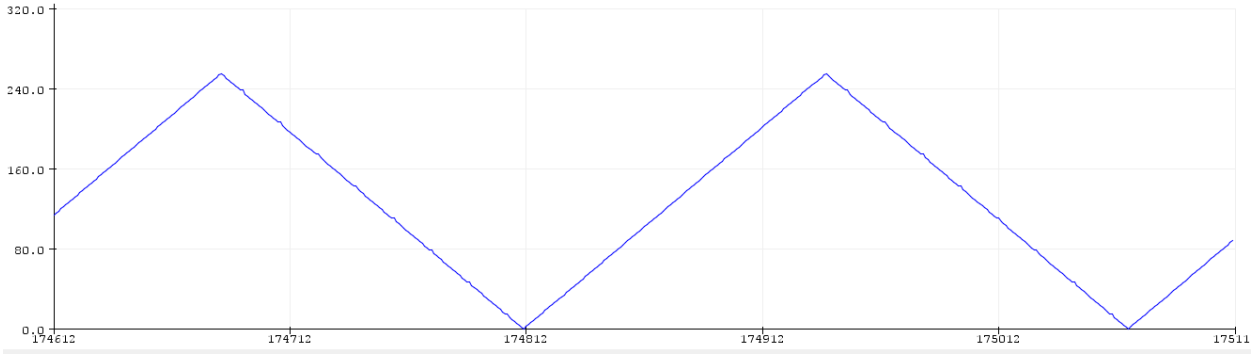


Figura 27 – Simulação de Onda Triangular

5.1.7. Simulação de Onda ECG

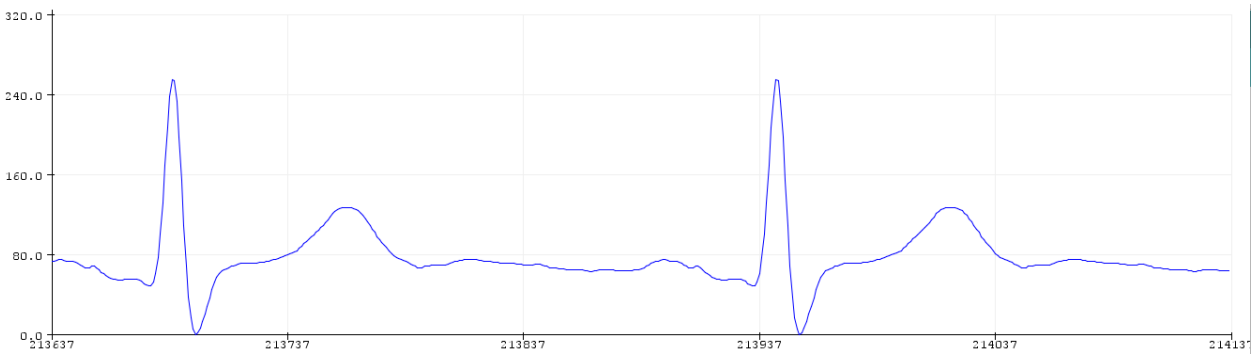


Figura 28 – Simulação de Onda ECG

5.2. Análise dos Resultados de Simulação

A simulação deste sistema será brevemente explicada. Conforme visto no diagrama de blocos desta implementação, serão usados dados de 8 bits, consistindo numa basicamente em uma memória de pesquisa, um acumulador, um contador e um conversor digital para analógico (*DAC*), os sinais são construídos através do contador do acumulador e no endereço de memória de consulta, que poderá ser em binário e decimal, as frequências são facilmente ajustadas.

A avaliação do desempenho deste sistema foi efetuada através da plataforma *Arduino IDE*, onde foi desenvolvido todo o software necessário para este sistema, onde foi analisado o comportamento do acumulador de fase, assim como, a forma virtualização de cada uma das ondas geradas. O diagrama de blocos desta simulação consistirá necessariamente na memória de pesquisa, acumulador e forma de medição. O bloco de memória de consultada apresenta vários componentes, tais como, o divisor, o multiplicador, conversor de graus para radianos e forma de onda. Através do software, é feita a conversão do byte de cada posição da memória para o seu valor de tensão.

A memória de consulta é usada para armazenar os dados da forma de onda, existindo vários tipos de amostra. Esta memória terá a informação de um ciclo da forma de onda desejada. Tal como informado anteriormente, foi utilizada uma *LM* de 8 bits.

Portanto, existem 256 amostras que contêm a informação de um ciclo de forma de onda sinusoidal. Cada amostra é um valor inteiro de 8 bits. Um único ciclo de forma de onda sinusoidal (360 graus) pode ser representado por 2^8 pontos.

Neste projeto, N = 2^8 = 256 amostras. Portanto, o ângulo para cada amostra é dado por,

$$°/amostra = \frac{360}{256} = 1{,}40 \tag{52}$$

Cada ponto, em graus, tem a sua conversão para radianos, efetuando a divisão por 180 graus no número multiplicado por π para dar o ângulo exato em radianos). Em radianos,

$$\frac{1{,}40}{180} * \pi = 0{,}024 \tag{53}$$

Sendo assim, efetuada o cálculo da função da onda para todos os ângulos em radianos.

O acumulador funciona como um divisor, um multiplicador e um conversor.

O acumulador tem a capacidade de contar com quase todos os números com um. Para este exemplo, o Acumulador é simulado com ciclo *for* tendo a opção de fazer a contagem por *Frequency Control Word (FCW)*. A contagem é representada para simular um contador de 8 bits. Quando o valor mais alto é alcançado, um contador atua como um acumulador.

O valor no acumulador é convertido numa matriz para facilitar a obtenção do bit mais alto e, em seguida, converter esse valor num inteiro. O bit mais alto destes 8 bits do acumulador é enviado para a memória, assim como o índice da matriz. O valor deste índice é um inteiro que representa uma amostra de 8 bits de uma forma de onda a ser enviada ao DAC.

O acumulador funciona da seguinte forma, tendo, por exemplo, N o número de contagens a serem realizadas e i o índice de iteração. O acumulador será executado por N vezes. O ângulo obtido é em graus, que é então convertido em radianos. A computação continua até ser obtido N amostras.

Através de PROGMEM, é feito o armazenamento dos dados na memória flash, programando esta, em vez de SRAM. A palavra-chave PROGMEM é um modificador de variável, ao qual deve ser usada somente com os tipos de dados definidos em *pgmspace.h*, esta variável indica ao compilador para colocar informação na memória flash, em vez de ser na SRAM, onde normalmente é passada essa informação.

Como o PROGMEM é um modificador de variável, não existe nenhuma regra especifica para este tipo de variável, assim o compilador aceita todos os tipos de definição abaixo. Embora o PROGMEM possa ser usado em uma única variável, só vale a pena se você tiver um bloco maior de dados que precisa ser armazenado, o que geralmente é mais fácil em uma matriz.

O bloco de medição é composto pelos sinais, onde o valor que precisa de ser armazenado devem estar conectados a este bloco, efetuando a medição de um único sinal, uma vez que é efetuada a conversão individual dos sinais.

Existem dois tipos de exibições como a memória de pesquisa, forma de onda da amostra enviada para o DAC e saída do gráfico de forma de onda DAC. Na exibição da memória LM, há um ciclo da forma de onda.

O eixo vertical é a amplitude, estando no intervalo $[-1; 1]$ e o eixo horizontal o número da amostra (0 a 1.000.000). Na exibição da forma de onda da amostra enviada ao DAC, este contém um sinal discreto, uma vez que se trata da conversão de discreto para analógico. Sendo assim, o eixo vertical continua a ser a amplitude e o eixo horizontal é o tempo. Na exibição da forma de onda da saída DAC, será um sinal analógico.

Neste software, a forma de onda vem numa variável PROGMEM, onde estão definidas as posições que irão ser amostradas, estando armazenada nesta variável. Portanto, esta variável poderá funcionar como um acumulador ou *Lookup Memory*. A partir do armazenado desta variável, é efetuado a criação de um ciclo for para que possa ser feita a conversão de cada das posições para ser obtida a forma de onda desejada, efetuando através de *pgm_read_word*, a leitura de uma palavra do espaço do programa com um endereço de 16 bits. Primeiramente, os dados consistem numa memória de endereço e depois são amostrados para que se possa obter a forma de onda desejada.

Além dos sistemas de armazenamento, o software de simulação tem outras vantagens, uma vez que os dados da nossa simulação estão em formato digital, os dados está à saída do DAC. Por fim, implementamos dados de 8 bits para a tabela de consulta. Ao fazer isso, o acumulador pode obter o endereço mais rapidamente, tornando o software mais simples. Numa grande maioria dos sistemas de geradores de sinal, um registo de fase de 24 bits é usado para máxima precisão. Destes, 16 bits são usados para escolher uma amostra da tabela de pesquisa. Portanto, nem todo endereço é usado. Esta estrutura é mais simples e permite uma recuperação mais rápida da informação.

5.3. Visualização das Ondas Geradas de Menor Frequência

5.3.1. Onda Sinusoidal de 415 Hz

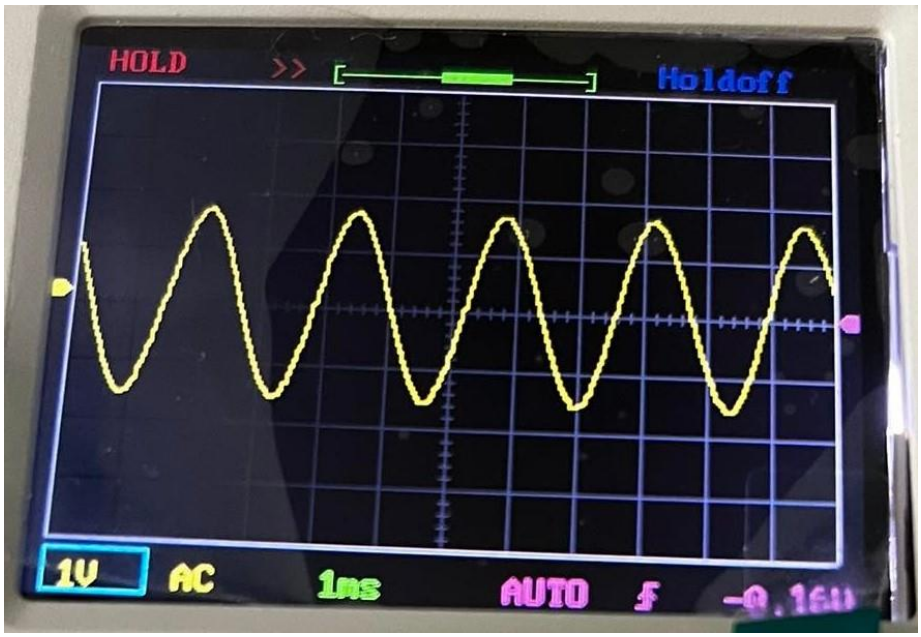


Figura 29 – Onda Sinusoidal de 415 Hz

5.3.2. Onda Quadrada de 415 Hz

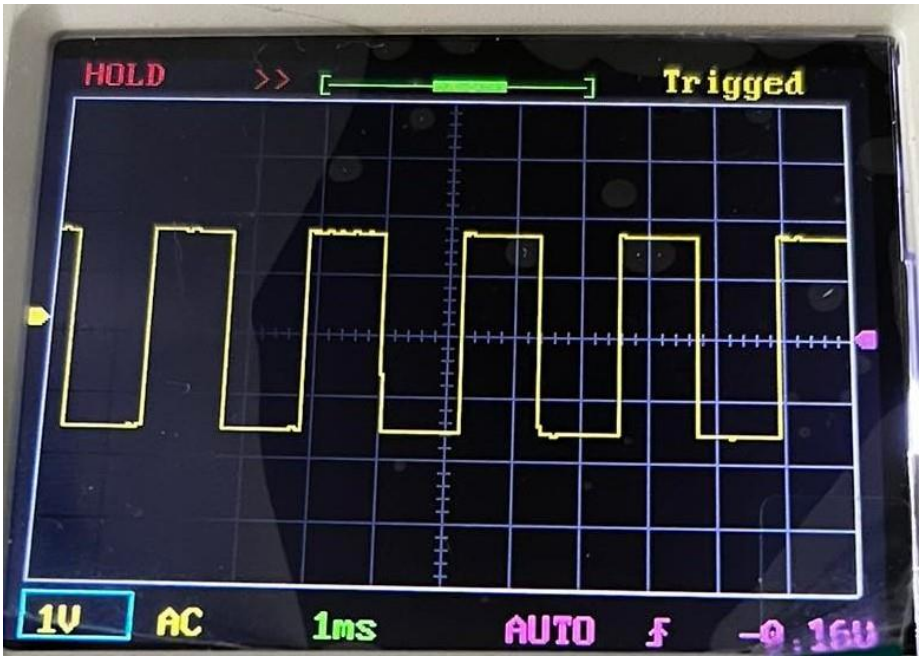


Figura 30 - Onda Quadrada de 415 Hz

5.3.3. Onda Dente de Serra de 415 Hz

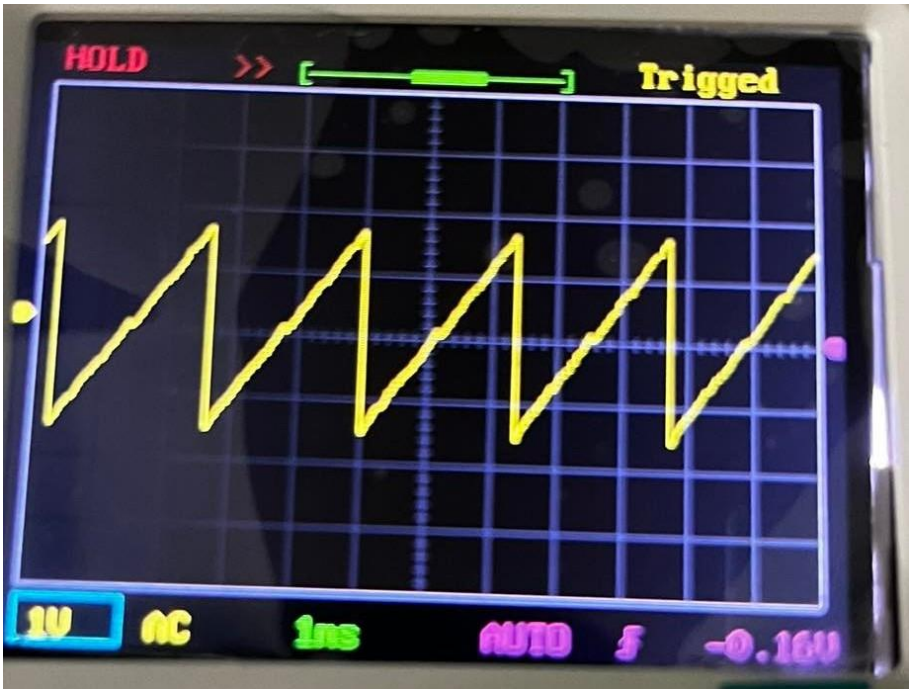


Figura 31 - Onda Dente de Serra de 415 Hz

5.3.4. Onda Dente de Serra Inversa de 415 Hz

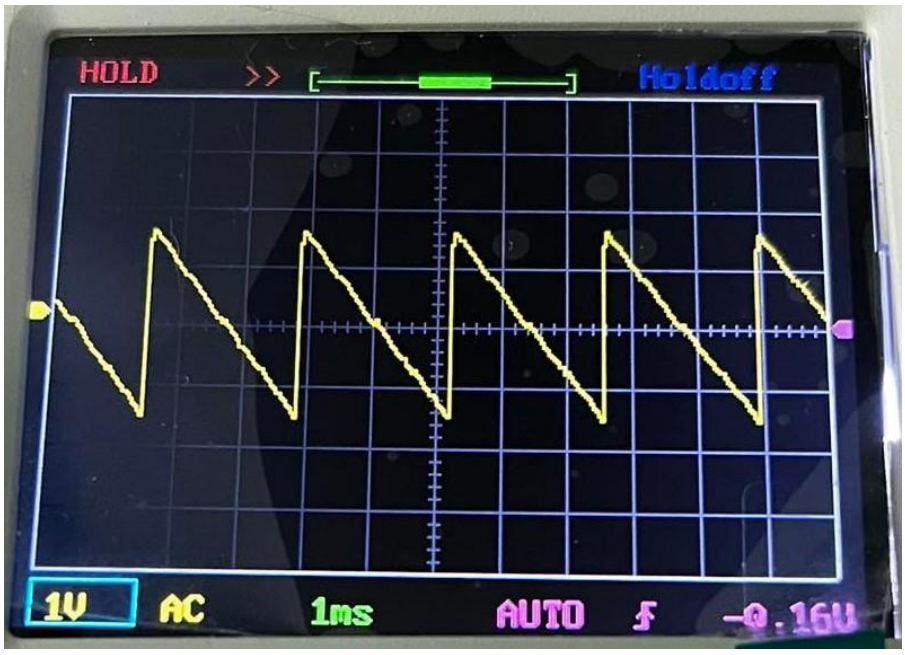


Figura 32 – Onda Dente de Serra Inversa de 415 Hz

5.3.5. Onda Triangular de 415 Hz

Figura 33 - Onda Triangular de 415 Hz

5.3.6. Onda Eletrocardiograma de 415 Hz

Figura 34 - Onda ECG de 415 Hz

5.4. Visualização das Ondas Geradas de Maior Frequência

5.4.1. Onda Sinusoidal de 17 kHz

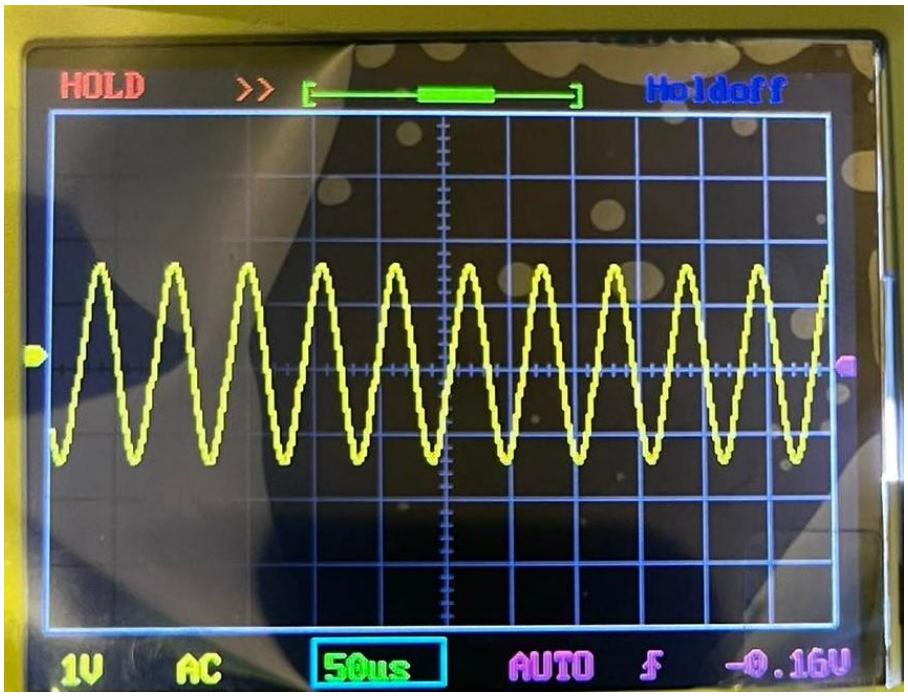


Figura 35 - Onda Sinusoidal de 17 kHz

5.4.2. Onda Quadrada de 17 kHz

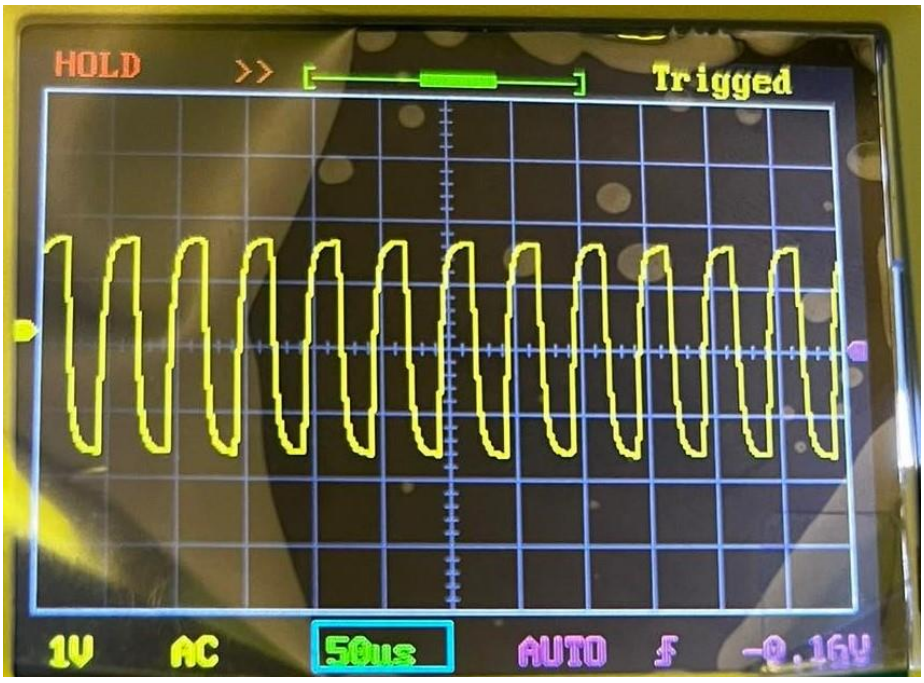


Figura 36 - Onda Quadrada de 17 kHz

5.4.3. Onda Dente de Serra de 17 kHz

Figura 37 - Onda Dente de Serra de 17 kHz

5.4.4. Onda Dente de Serra Inversa de 17 kHz

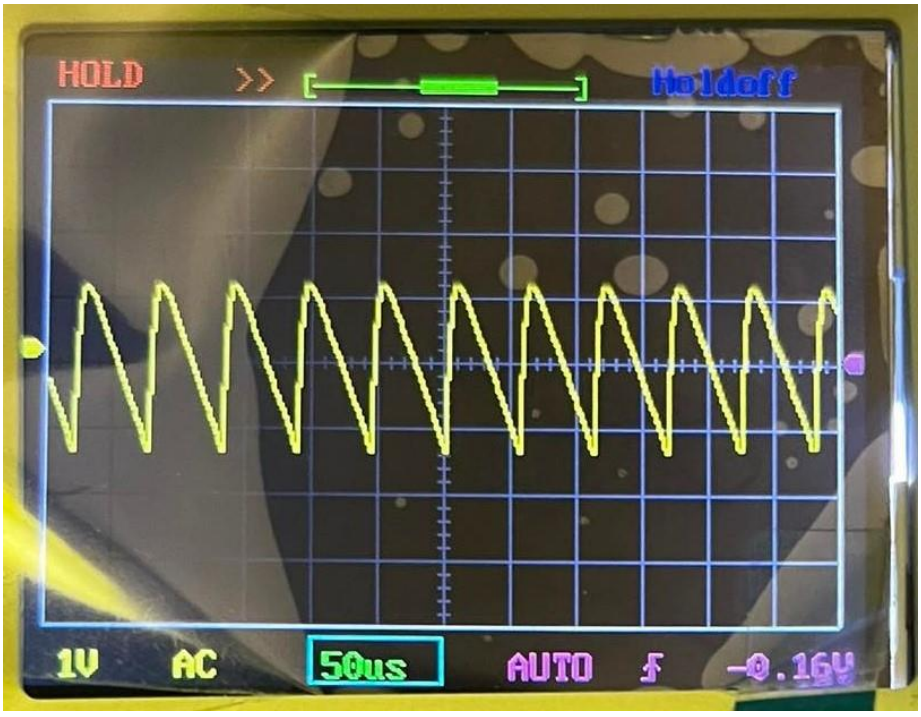


Figura 38 - Onda Dente de Serra Inversa de 17 kHz

5.4.5. Onda Triangular de 17 kHz

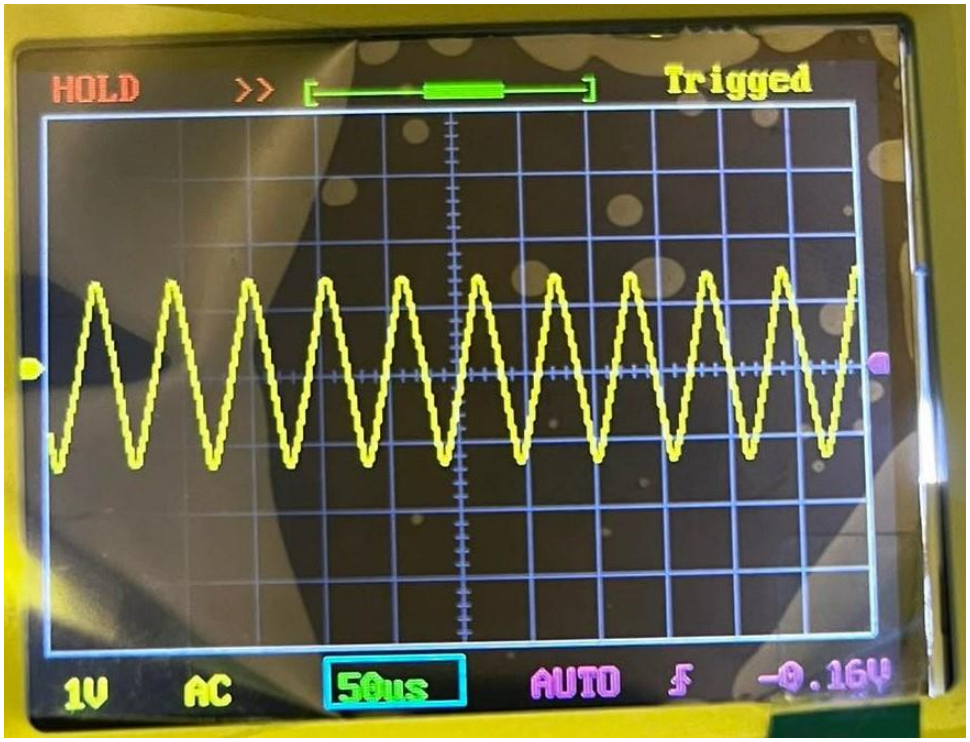


Figura 39 - Onda Triangular de 17 kHz

5.4.6. Onda ECG de 17 kHz

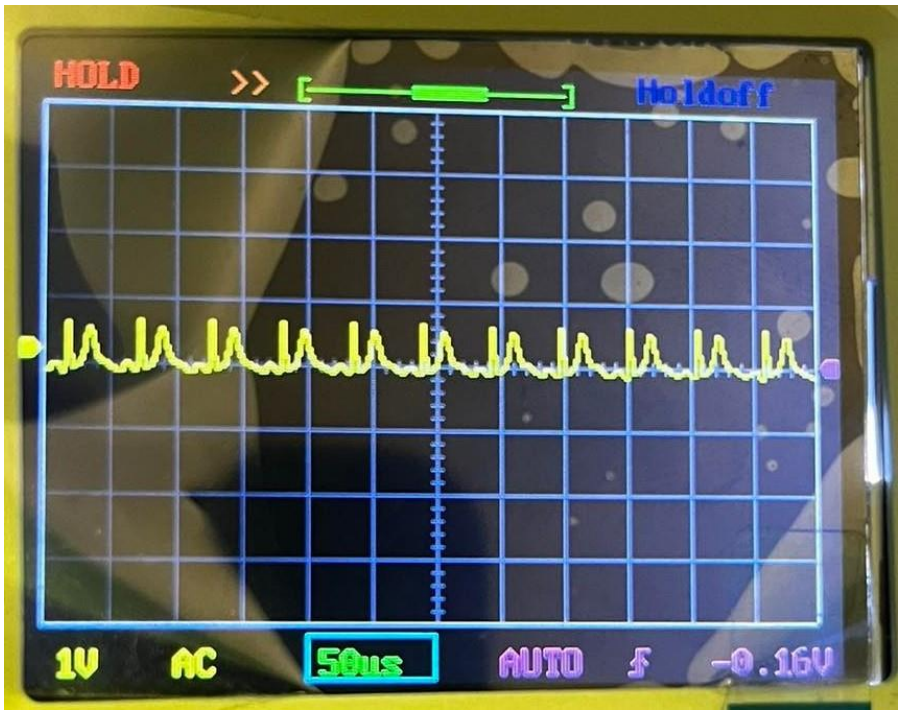


Figura 40 - Onda ECG de 17 kHz

5.5. Análise Experimental

O bloco de relógio representa o relógio de referência. O selecionador de relógio permite escolher qual o relógio interno que se pretende, sendo controlado pelo Teensy 4.1. Como esta técnica DDS não introduz instabilidade de frequência no sistema, a estabilidade na frequência é dominada pelo relógio interno.

O bloco provedor de relógio entrega o sinal de relógio ao sistema e também fornece uma saída externa para sincronização com outros sistemas. O sistema gera sinais de forma de onda sinusoidal, quadrada, triangular, dente de serra e ECG. Os sinais são gerados individualmente no Teensy 4.1, utilizando unicamente este dispositivo.

Como são geradas formas de ondas, é sempre necessário verificar internamente a tabela de consulta, sendo efetuada uma conexão diretamente a saída do acumulador de fase ao DAC.

Para avaliar o comportamento do DDS, foram analisadas as principais fontes de distorção que são:

- Truncamento do acumulador de fase;

- Conversor D/A interno;

- Interferência entre trilhas no projeto de PCB.

Para simular e validar o desempenho do sistema foi desenvolvido um programa de simulação. Os resultados da simulação mostraram que a frequência espúria devido ao truncamento do acumulador de fase estava acima da largura de banda do sistema (10 kHz), e pode ser filtrada adequadamente com os filtros mencionados na seção anterior. Esses filtros de reconstrução também reduzem a distorção harmônica causada pelo conversor D/A.

Para reduzir os efeitos do ruído de quantização, podem ser efetuadas diferentes tipos de medições no sistema, variando a relação entre a frequência de saída do DDS e a frequência de relógio de referência.

A Placa de Circuito Impresso foi projetada para reduzir a interferência entre as trilhas, é necessário ter em consideração:

- Inclusão do Ground, para facilitar uma placa de circuito impresso podem ser projetadas duas camadas, onde a camada inferior atua como Ground;

- Redução capacitiva de *crosstalk*. Para reduzir a interferência entre as trilhas, deve-se evitar trilhas paralelas, mantendo-as mais curtas possível. A geração de sinais, implica que as trilhas do relógio de referência para esses dispositivos sejam percorridas de igual modo, tendo o mesmo comprimento;

- Caminho de retorno de baixa impedância do relógio de referência. Trilhas cruzadas atrás da trilha do relógio de referência (camada inferior) devem ser evitadas para evitar a propagação da corrente de retorno por todo o circuito.

Por exemplo, para gerar uma determinada frequência e variar a frequência do Relógio de referência, conforme a configuração do DDS, é necessário selecionar a melhor frequência do relógio de referência, permitindo calcular a distorção harmônica total (THD) em diferentes frequências de relógio de referência. Ao reduzir a largura de banda do sistema implica uma redução de THD. Obtém-se uma melhoria quando a relação entre a frequência de relógio de referência e a frequência de saída é reduzida. Quando esta relação foi ajustada, o THD apresentou uma redução no seu valor.

Através da placa de teste e desenvolvimento de um programa de computador foi efetuado o controlo da medição. No software, a palavra de incremento de fase pode ser escrita em hexadecimal ou na frequência necessária. Neste último caso, o software calculará a palavra de incremento de fase correspondente. A palavra de incremento de fase e os outros sinais de controle são carregados na placa de teste através da conexão do Teensy 4.1.

Por exemplo, quando se pretende analisar o efeito das não linearidades estáticas do conversor D/A, onde a frequência de relógio é de 600 MHz e a frequência de saída é baixa, a distorção de ordem uniforme é reduzida devido ao design diferencial. O conversor D/A cumpre o requisito de linearidade estática de 8 bits. O pior caso do DDS próximo dos picos da portadora na banda larga, considerado a largura de banda *Nyquist*, ocorre quando a frequência de saída é sintonizada perto de $\frac{f_{clk}}{3}$.

Numa primeira avaliação dos resultados, após a resolução de pequenos problemas, são definidas algumas frequências. Quando se aumenta a frequência do sinal, este corre o risco de aumentar o seu ruído, tornando-o impercetível, este problema pode ser através da implementação de um filtro, com uma frequência de corte adequada para eliminar os ruídos provenientes.

O sinal é medido na saída do DAC R-2R Ladder, o seu sinal de saída é de 425 Hz a 20 kHz, sendo gerado e lido corretamente no display. Os resultados poderiam melhores ser obtendo um protótipo funcional de gerador baseado em DDS. Apesar da frequência CPU do Teensy 4.1 ser 600 MHz, devido a limitações de hardware, assim como software, o intervalo de frequências geradas é inferior ao intervalo inicialmente planeado, até 125 MHz, apesar de terem sido efetuadas correções a nível de software, tentando efetuar o *delay_us* com a melhor precisão possível, tal não foi possível, no entanto, foram desejadas as formas de ondas, inicialmente, planeadas, correndo como previsto.

No entanto, como se pretendia gerar frequências bastantes elevadas, estes sistemas podem aplicações de transmissores de sinais de rádio onde se requer uma alta qualidade do sinal. Para avaliá-lo, analisa-se o SFDR, pode-se analisar os seus gráficos de saída do sinal, numa banda de frequências elevadas, assim como o seu espetro. O SFDR piora à medida que a frequência aumenta.

A truncagem é uma fonte primordial de impureza espectral, de forma a reduzir o tamanho da tabela de pesquisa, tendo uma saída do acumulador de fase truncada, gerando um sinal de erro de truncamento que cria picos discretos no domínio da frequência.

A não linearidade do DAC contribui para as impurezas do espetro de saída, devendo à diferença entre a saída teórica e real do DAC, obtendo um resultado que não é perfeitamente linear, tendo a saída do DAC distorcida. Os quantificadores não-lineares consistem numa soma do sinal esperado com os seus harmónicos.

6. Conclusões e Trabalho Futuro

6.1. Conclusões Finais

O objetivo da presente tese é discutir sobre Síntese Digital Direta (DDS) e implementar um gerador de onda sinusoidal baseado em esta tecnologia.

Com o objetivo de implementar um microcontrolador de baixo custo, foi possível desenvolver um gerador de sinais sintetizados através desta tecnologia, com uma faixa de frequência que abrange desde 415 Hz a um máximo de 20 kHz, dependendo da forma de onda selecionada. Estes geradores têm vindo a ter uma maior presença no mercado, oferecendo melhorias substanciais de desempenho com um menor custo, em relação aos geradores de função analógicos convencionais. Como o custo dos diversos componentes, como RAMs, DACs, entre outros, diminui, tendo a vantagem do aumento da sua velocidade e resolução, o futuro pode ter a presença de geradores de funções baseados em DDS a substituir os geradores de sinais analógicos.

Estes sistemas apresentam algumas vantagens no seu uso, em comparação com outras formas de geração de sinais como, por exemplo, os circuitos osciladores, tendo as seguintes vantagens:

- Resolução elevada, gerando diversos valores de frequência com saltos, variando em unidades de μHz;
- Flexibilidade, podendo ser adicionadas formas de onda, ou modificar estas mesmas, apenas reprogramando o processador;
- Maior imunidade às variações do ambiente que poderiam afetar os circuitos analógicos;
- Estável, uma vez que a sua função de transferência possui apenas zeros, tendo um comportamento estável para toda faixa de frequência projetada;
- Troca de frequência rápida.

As contribuições do projeto incluem a construção de um protótipo funcional que pode ser utilizado para fins educacionais, entre outros. Em relação à motivação primária do projeto, trabalhar como oscilador local para transmissões de RF, algumas melhorias devem ser feitas.

A primeira melhoria a ser feita é o aumento das frequências geradas, uma vez que se pretendia gerar frequências elevadas, bem como a adição de um filtro com uma boa frequência de corte e

inclinação. O filtro atuará como um filtro *anti-aliasing*, atenuando a resposta das imagens que aparecem no espectro em $f_{clk} \pm f_{out}$.

6.2. Trabalho Futuro

A partir das conclusões tiradas acima, são propostos alguns tópicos de trabalho para investigação futura:

- Modulação em fase e vetor de fase em quadratura de radiofrequência;
- Projeto de um Sintetizador de Frequência Digital Direto sem ROM;
- Sintetizador Digital Direta para Aplicações Wireless;
- Um estudo completo do sistema experimental e seleção e teste de hardware para melhorar a força do sinal e testar as transmissões a distâncias;
- Análise detalhada de um Conversor Digital-Analógico Calibrado para obter uma melhor resolução e com taxas de conversão elevadas, permitindo atingir um desempenho de referência para comunicações via rádio ou satélite.

Bibliografia

[1] M. R. e. B. G. C. Tierney, “A digital frequency synthesizer,” IEEE Trans. Audio Electroacoust, 1971, pp. 48-57.

[2] J. V. Boston, “Direct Digital Synthesizers: Theory, Design And Applications,” 2001. [Online].

[3] “A Technical Tutorial on Digital Signal Synthesis,” [Online].

[4] B. S. SHRIMALI, “DIRECT DIGITAL FREQUENCY SYNTHESIZER,” [Online].

[5] H. N. e. S. H, “An analysis of output spectrum of direct digital,” em *Proceedings of the*, pp. 495-502.

[6] A. Devices, “Fundamentals of Direct Digital Synthesis (DDS),” [Online]. Available: https://www.analog.com/media/en/training-seminars/tutorials/MT-085.pdf.

[7] “DDS Technology,” [Online]. Available: http://www.hit.bme.hu/~papay/sci/DDS/start.htm.

[8] “CHAPTER 6: CONVERTERS,” [Online]. Available: https://www.analog.com/media/en/training-seminars/design-handbooks/Basic-Linear-Design/Chapter6.pdf.

[9] I. J. o. S.-S. Circuits, “A 5-GHz Direct Digital Frequency Synthesizer Using an Analog-Sine-Mapping Technique,” September 2011.

[10] I. J. o. S.-S. Circuits, “An 11-Bit 8.6 GHz Direct Digital Synthesizer MMIC With 10-Bit Segmented Sine-weighted,” 2010.

[11] I. ISCAS, “Segmented SineWave Digital-to-Analog Converters for Frequency Synthesizer,” 2001.

[12] I. ISLPED, “A Low-Power Direct Digital Frequency Synthesizer Using an Analogue-Sine-Conversion,” 2001.

[13] D. BRANDON, “DDS design,” 2004. [Online]. Available: https://www.edn.com/dds-design/.

[14] I. T. on, “Systematic Analysis of Interleaved Digital-to-analog Converters,” 2011.

[15] I. J. o. S.-S. Circuits, “A 12 Bit GS/s DAC With IM3 < -60 dBc Beyond 1 GHz in 65nm CMOS,” 2009.

[16] I. T. on, “ On the attenuation of DAC aliases through multiphase Clocking,” 2009.

[17] I. J. o. S.-S. Circuits, “ Parallel-path Digital-to-Analog Converters for Nyquist Signal Generation,” 2004.

[18] P. -.. E. P. C. A. Worldwide, “Teensy® 4.1 Development Board,” [Online]. Available: https://www.pjrc.com/store/teensy41.html.

Anexo A – Gerador de Funções

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
double sampling_period_us;
unsigned long microseconds;
const byte interruptPin = 17;

const PROGMEM byte R2RLookup_Sine[256] =
{
0x80,0x83,0x86,0x89,0x8c,0x8f,0x92,0x95,0x98,0x9c,0x9f,0xa2,0xa5,0xa8,0xab,0xae,
0xb0,0xb3,0xb6,0xb9,0xbc,0xbf,0xc1,0xc4,0xc7,0xc9,0xcc,0xce,0xd1,0xd3,0xd5,0xd8,
0xda,0xdc,0xde,0xe0,0xe2,0xe4,0xe6,0xe8,0xea,0xec,0xed,0xef,0xf0,0xf2,0xf3,0xf5,
0xf6,0xf7,0xf8,0xf9,0xfa,0xfb,0xfc,0xfc,0xfd,0xfe,0xfe,0xff,0xff,0xff,0xff,0xff,
0xff,0xff,0xff,0xff,0xff,0xff,0xfe,0xfe,0xfd,0xfc,0xfc,0xfb,0xfa,0xf9,0xf8,0xf7,
0xf6,0xf5,0xf3,0xf2,0xf0,0xef,0xed,0xec,0xea,0xe8,0xe6,0xe4,0xe2,0xe0,0xde,0xdc,
0xda,0xd8,0xd5,0xd3,0xd1,0xce,0xcc,0xc9,0xc7,0xc4,0xc1,0xbf,0xbc,0xb9,0xb6,0xb3,
0xb0,0xae,0xab,0xa8,0xa5,0xa2,0x9f,0x9c,0x98,0x95,0x92,0x8f,0x8c,0x89,0x86,0x83,
0x80,0x7c,0x79,0x76,0x73,0x70,0x6d,0x6a,0x67,0x63,0x60,0x5d,0x5a,0x57,0x54,0x51,
0x4f,0x4c,0x49,0x46,0x43,0x40,0x3e,0x3b,0x38,0x36,0x33,0x31,0x2e,0x2c,0x2a,0x27,
0x25,0x23,0x21,0x1f,0x1d,0x1b,0x19,0x17,0x15,0x13,0x12,0x10,0x0f,0x0d,0x0c,0x0a,
0x09,0x08,0x07,0x06,0x05,0x04,0x03,0x03,0x02,0x01,0x01,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x01,0x02,0x03,0x03,0x04,0x05,0x06,0x07,0x08,
0x09,0x0a,0x0c,0x0d,0x0f,0x10,0x12,0x13,0x15,0x17,0x19,0x1b,0x1d,0x1f,0x21,0x23,
0x25,0x27,0x2a,0x2c,0x2e,0x31,0x33,0x36,0x38,0x3b,0x3e,0x40,0x43,0x46,0x49,0x4c,
0x4f,0x51,0x54,0x57,0x5a,0x5d,0x60,0x63,0x67,0x6a,0x6d,0x70,0x73,0x76,0x79,0x7c
};

const PROGMEM byte R2RLookup_Square[256] =
```

```
{

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,
0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,
0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,
0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,
0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,
0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,
0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,
0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,
};
const PROGMEM byte R2RLookup_Sawtooth[256] =
{
0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,
0x10,0x11,0x12,0x13,0x14,0x15,0x16,0x17,0x18,0x19,0x1a,0x1b,0x1c,0x1d,0x1e,0x1f,
0x20,0x21,0x22,0x23,0x24,0x25,0x26,0x27,0x28,0x29,0x2a,0x2b,0x2c,0x2d,0x2e,0x2f,
0x30,0x31,0x32,0x33,0x34,0x35,0x36,0x37,0x38,0x39,0x3a,0x3b,0x3c,0x3d,0x3e,0x3f,
0x40,0x41,0x42,0x43,0x44,0x45,0x46,0x47,0x48,0x49,0x4a,0x4b,0x4c,0x4d,0x4e,0x4f,
0x50,0x51,0x52,0x53,0x54,0x55,0x56,0x57,0x58,0x59,0x5a,0x5b,0x5c,0x5d,0x5e,0x5f,
0x60,0x61,0x62,0x63,0x64,0x65,0x66,0x67,0x68,0x69,0x6a,0x6b,0x6c,0x6d,0x6e,0x6f,
0x70,0x71,0x72,0x73,0x74,0x75,0x76,0x77,0x78,0x79,0x7a,0x7b,0x7c,0x7d,0x7e,0x7f,
0x80,0x81,0x82,0x83,0x84,0x85,0x86,0x87,0x88,0x89,0x8a,0x8b,0x8c,0x8d,0x8e,0x8f,
```

```
0x90,0x91,0x92,0x93,0x94,0x95,0x96,0x97,0x98,0x99,0x9a,0x9b,0x9c,0x9d,0x9e,0x9f,

0xa0,0xa1,0xa2,0xa3,0xa4,0xa5,0xa6,0xa7,0xa8,0xa9,0xaa,0xab,0xac,0xad,0xae,0xaf,
0xb0,0xb1,0xb2,0xb3,0xb4,0xb5,0xb6,0xb7,0xb8,0xb9,0xba,0xbb,0xbc,0xbd,0xbe,0xbf,
0xc0,0xc1,0xc2,0xc3,0xc4,0xc5,0xc6,0xc7,0xc8,0xc9,0xca,0xcb,0xcc,0xcd,0xce,0xcf,
0xd0,0xd1,0xd2,0xd3,0xd4,0xd5,0xd6,0xd7,0xd8,0xd9,0xda,0xdb,0xdc,0xdd,0xde,0xdf,
0xe0,0xe1,0xe2,0xe3,0xe4,0xe5,0xe6,0xe7,0xe8,0xe9,0xea,0xeb,0xec,0xed,0xee,0xef,
0xf0,0xf1,0xf2,0xf3,0xf4,0xf5,0xf6,0xf7,0xf8,0xf9,0xfa,0xfb,0xfc,0xfd,0xfe,0xff
};
const PROGMEM byte R2RLookup_RevSawtooth[256] =
{
0xff,0xfe,0xfd,0xfc,0xfb,0xfa,0xf9,0xf8,0xf7,0xf6,0xf5,0xf4,0xf3,0xf2,0xf1,0xf0,
0xef,0xee,0xed,0xec,0xeb,0xea,0xe9,0xe8,0xe7,0xe6,0xe5,0xe4,0xe3,0xe2,0xe1,0xe0,
0xdf,0xde,0xdd,0xdc,0xdb,0xda,0xd9,0xd8,0xd7,0xd6,0xd5,0xd4,0xd3,0xd2,0xd1,0xd0,
0xcf,0xce,0xcd,0xcc,0xcb,0xca,0xc9,0xc8,0xc7,0xc6,0xc5,0xc4,0xc3,0xc2,0xc1,0xc0,
0xbf,0xbe,0xbd,0xbc,0xbb,0xba,0xb9,0xb8,0xb7,0xb6,0xb5,0xb4,0xb3,0xb2,0xb1,0xb0,
0xaf,0xae,0xad,0xac,0xab,0xaa,0xa9,0xa8,0xa7,0xa6,0xa5,0xa4,0xa3,0xa2,0xa1,0xa0,
0x9f,0x9e,0x9d,0x9c,0x9b,0x9a,0x99,0x98,0x97,0x96,0x95,0x94,0x93,0x92,0x91,0x90,
0x8f,0x8e,0x8d,0x8c,0x8b,0x8a,0x89,0x88,0x87,0x86,0x85,0x84,0x83,0x82,0x81,0x80,
0x7f,0x7e,0x7d,0x7c,0x7b,0x7a,0x79,0x78,0x77,0x76,0x75,0x74,0x73,0x72,0x71,0x70,
0x6f,0x6e,0x6d,0x6c,0x6b,0x6a,0x69,0x68,0x67,0x66,0x65,0x64,0x63,0x62,0x61,0x60,
0x5f,0x5e,0x5d,0x5c,0x5b,0x5a,0x59,0x58,0x57,0x56,0x55,0x54,0x53,0x52,0x51,0x50,
0x4f,0x4e,0x4d,0x4c,0x4b,0x4a,0x49,0x48,0x47,0x46,0x45,0x44,0x43,0x42,0x41,0x40,
0x3f,0x3e,0x3d,0x3c,0x3b,0x3a,0x39,0x38,0x37,0x36,0x35,0x34,0x33,0x32,0x31,0x30,
0x2f,0x2e,0x2d,0x2c,0x2b,0x2a,0x29,0x28,0x27,0x26,0x25,0x24,0x23,0x22,0x21,0x20,
0x1f,0x1e,0x1d,0x1c,0x1b,0x1a,0x19,0x18,0x17,0x16,0x15,0x14,0x13,0x12,0x11,0x10,
0x0f,0x0e,0x0d,0x0c,0x0b,0x0a,0x09,0x08,0x07,0x06,0x05,0x04,0x03,0x02,0x01,0x00,
};
```

```
const PROGMEM byte R2RLookup_Triangle[256] ={
0x00,0x02,0x04,0x06,0x08,0x0a,0x0c,0x0e,0x10,0x12,0x14,0x16,0x18,0x1a,0x1c,0x1e,
0x20,0x22,0x24,0x26,0x28,0x2a,0x2c,0x2e,0x30,0x32,0x34,0x36,0x38,0x3a,0x3c,0x3e,
0x40,0x42,0x44,0x46,0x48,0x4a,0x4c,0x4e,0x50,0x52,0x54,0x56,0x58,0x5a,0x5c,0x5e,
0x60,0x62,0x64,0x66,0x68,0x6a,0x6c,0x6e,0x70,0x72,0x74,0x76,0x78,0x7a,0x7c,0x7e,
0x80,0x82,0x84,0x86,0x88,0x8a,0x8c,0x8e,0x90,0x92,0x94,0x96,0x98,0x9a,0x9c,0x9e,
0xa0,0xa2,0xa4,0xa6,0xa8,0xaa,0xac,0xae,0xb0,0xb2,0xb4,0xb6,0xb8,0xba,0xbc,0xbe,
0xc0,0xc2,0xc4,0xc6,0xc8,0xca,0xcc,0xce,0xd0,0xd2,0xd4,0xd6,0xd8,0xda,0xdc,0xde,
0xe0,0xe2,0xe4,0xe6,0xe8,0xea,0xec,0xee,0xf0,0xf2,0xf4,0xf6,0xf8,0xfa,0xfc,0xfe,
0xff,0xfd,0xfb,0xf9,0xf7,0xf5,0xf3,0xf1,0xef,0xef,0xeb,0xe9,0xe7,0xe5,0xe3,0xe1,
0xdf,0xdd,0xdb,0xd9,0xd7,0xd5,0xd3,0xd1,0xcf,0xcf,0xcb,0xc9,0xc7,0xc5,0xc3,0xc1,
0xbf,0xbd,0xbb,0xb9,0xb7,0xb5,0xb3,0xb1,0xaf,0xaf,0xab,0xa9,0xa7,0xa5,0xa3,0xa1,
0x9f,0x9d,0x9b,0x99,0x97,0x95,0x93,0x91,0x8f,0x8f,0x8b,0x89,0x87,0x85,0x83,0x81,
0x7f,0x7d,0x7b,0x79,0x77,0x75,0x73,0x71,0x6f,0x6f,0x6b,0x69,0x67,0x65,0x63,0x61,
0x5f,0x5d,0x5b,0x59,0x57,0x55,0x53,0x51,0x4f,0x4f,0x4b,0x49,0x47,0x45,0x43,0x41,
0x3f,0x3d,0x3b,0x39,0x37,0x35,0x33,0x31,0x2f,0x2f,0x2b,0x29,0x27,0x25,0x23,0x21,
0x1f,0x1d,0x1b,0x19,0x17,0x15,0x13,0x11,0x0f,0x0f,0x0b,0x09,0x07,0x05,0x03,0x01};
const PROGMEM byte R2RLookup_ECG[256] ={
73,74,75,75,74,73,73,73,73,72,71,69,68,67,67,67,
68,68,67,65,62,61,59,57,56,55,55,54,54,54,55,55,
55,55,55,55,54,53,51,50,49,49,52,61,77,101,132,
169,207,238,255,254,234,198,154,109,68,37,17,5,
0,1,6,13,20,28,36,45,52,57,61,64,65,66,67,68,68,
69,70,71,71,71,71,71,71,71,71,72,72,72,73,73,74,
75,75,76,77,78,79,80,81,82,83,84,86,88,91,93,96,
98,100,102,104,107,109,112,115,118,121,123,125,
126,127,127,127,127,127,126,125,124,121,119,116,
113,109,105,102,98,95,92,89,87,84,81,79,77,76,75,
```

```
74,73,72,70,69,68,67,67,67,68,68,68,69,69,69,69,

69,69,69,70,71,72,73,73,74,74,75,75,75,75,75,75,
74,74,73,73,73,73,72,72,72,71,71,71,71,71,71,71,
70,70,70,69,69,69,69,69,70,70,70,69,68,68,67,67,
67,67,66,66,66,65,65,65,65,65,65,65,65,64,64,63,
63,64,64,65,65,65,65,65,65,65,64,64,64,64,64,64,
64,64,65,65,65,66,67,68,69,71,72,73
};
char func = ' ';
double final_freq;
int freqInitial;
bool stringComplete = false;  // whether the string is complete

void setup(){
  Serial3.begin(9600);
  Serial.println("START");
  cli();
  pinMode(interruptPin, INPUT_PULLUP);
  pinMode(18,INPUT);
  DDRD = 0b11111111;
  attachInterrupt(digitalPinToInterrupt(interruptPin), change_frequency, RISING);
  sei();
}

void loop(){
while(Serial3.available()) {
  char func = Serial3.read();
  if (func == '1') {
   wave = 1;
```

```
  }

  if (func == '2') {
   wave = 2;
  }
  if (func == '3') {
   wave = 3;
  }
  if (func == '4') {
   wave = 4;
  }
  if (func == '5') {
   wave = 5;
  }
  if (func == '6') {
   wave = 6;
  }
  Serial.println(func);
}
switch(wave){
  case 1:
    for(int i=0; i<256; i++){
      PORTD = pgm_read_word(&R2RLookup_Sine[i]);
      _delay_us(sampling_period_us);
      }
    break;
  case 2:
    for(int i=0; i<256; i++){
      PORTD = pgm_read_word(&R2RLookup_Square[i]);
      _delay_us(sampling_period_us);
```

```
    }
   break;
case 3:
    for(int i=0; i<256; i++){
       PORTD = pgm_read_word(&R2RLookup_Sawtooth[i]);
       _delay_us(sampling_period_us);
    }
    break;
case 4:
    for(int i=0; i<256; i++){
       PORTD = pgm_read_word(&R2RLookup_RevSawtooth[i]);
       _delay_us(sampling_period_us);
    }
    break;
case 5:
    for(int i=0; i<256; i++){
       PORTD = pgm_read_word(&R2RLookup_Triangle[i]);
       _delay_us(sampling_period_us);
    }
    break;
case 6:
    for(int i=0; i<256; i++){
       PORTD = pgm_read_word(&R2RLookup_ECG[i]);
       _delay_us(sampling_period_us);
    }
    break;
default:
    break;
}
}
```

```
void change_frequency(){
  double max_freq = F_CPU/5;
  double samples = analogRead(18);
  double variate_samples = 1024.00 - samples;
  Serial.println(samples);
  final_freq = max_freq/variate_samples;
  Serial.println(final_freq);
  sampling_period_us = (1000000.0*(1.0/final_freq));
  Serial.println(sampling_period_us,9);
}
```