

Instrumentação A

Relatório

Afinador automático para violão baseado em transdutor piezoeletrico

Davi Nachtigall Lazzarotto, Lucas Fernando Santolin e Lucas Nedel Kirsten

Universidade Federal do Rio Grande do Sul, Departamento de Engenharia Elétrica, Curso de Engenharia Elétrica, Instrumentação A, Prof. Dr. Alexandre Balbinot

E-Mails: davi.nl@hotmail.com (D.N.L); lucas.santolin@ufrgs.br (L.F.S); lucasnkir@gmail.com (L.N.K.)

Data Início: 05/10/2018; Data Final: 14/12/2018

Resumo. O projeto apresentado nesse relatório tem o objetivo de facilitar a tarefa de afinação de um violão. Ele é baseado em um sensor piezoelettrico acoplado ao instrumento, que mede a vibração do mesmo quando uma corda é tocada. O sinal do sensor é condicionado e enviado ao conversor AD de um microcontrolador, que recebe este sinal e aplica um algoritmo de detecção de frequência para medir se ele está afinado. A frequência medida serve como parâmetro de entrada de um *fuzzy*, que calcula o ângulo de rotação de um motor de passo que atua sobre a tarraxa do violão. Na análise do sistema, foi constatado que a incerteza da detecção de frequência é menor para as três cordas mais graves. A performance do sistema como um todo foi razoável para a quarta e quinta corda, enquanto que para as cordas mais agudas, sobretudo a primeira, sua capacidade de afinação é reduzida.

Abstract. The purpose of the project hereby presented is to ease the task of tuning a guitar. It is based off of a piezoelectric transducer which is coupled to the inside of the guitar, thus being able to measure the vibrations produced by a resonant string. The signal produced by the transducer is conditioned and routed to the input of a microcontroller, which will use a frequency detection algorithm to determine if the guitar is in tune. This frequency is then used as an input for a fuzzy controller which generates an optimal shaft rotation angle for a stepper motor to act on the tuning peg. Through the analysis of this system it was discovered that the uncertainty of the frequency detection is smaller for the lower frequency strings. Overall, while the system's performance is acceptable for the fourth and fifth strings, it is unsatisfactory for the higher frequency ones.

Palavras Chave: Violão, Afinação automática, Transdutor piezoelettrico

1. Introdução

Um dos fatores primordiais para que uma melodia seja bem executada em instrumentos musicais é que o mesmo esteja corretamente regulado conforme algum esquema de afinação. Em instrumentos de cordas, se faz necessária a realização do processo de afinação de forma frequente para garantir a qualidade do som. Sem a ajuda de nenhum equipamento, esse processo é realizado pelo instrumentista, que ajusta a frequência de uma corda girando a tarraxa à qual ela está conectada. Esse processo é repetido para todas as cordas, e o instrumentista confia na sua capacidade auditiva para perceber o momento em que a frequência da corda em processo de afinação é igual à uma frequência de referência, por exemplo gerada por um diapasão. Contudo, já há uma grande variedade de opções de afinadores no mercado capazes de reduzir o trabalho do instrumentista e agregar confiabilidade à afinação do seu instrumento. Os modelos incluem os afinadores já acoplados ao instrumento, afinadores que afinam captando o som da nota tocada com um microfone e afinadores que afinam com relação a vibração que chega ao mesmo no contato com o instrumento.

A maioria dos violões modernos já possuem acoplado internamente ao seu tampo algum tipo de sensor de vibração, tratando-se, na maioria das vezes, de sensores piezoelétricos. Isso se deve, principalmente, à estruturação física dos violões e pelo fato dos sensores piezoelétricos serem sensíveis às vibrações mecânicas produzidas pelo movimento das cordas na estrutura do violão. Assim, os sensores piezoelétricos podem ser usados tanto em violão com corda de nylon, quanto em violões com corda de aço - diferentemente dos sensores magnéticos utilizados nas guitarras (os captadores) que só são sensíveis as variações de vibração devido a variação do campo magnético provocado pelo movimento das cordas de aço perto dos ímãs que os compõem. Contudo, esses sensores em sua maioria se restringem a apenas captar o som gerado e não fazer quaisquer tipo de processamento em cima do mesmo. Com o avanço da eletrônica e dos sistemas embarcados torna-se possível, então, que tais sensores também adquiram outras finalidades, como a de um afinador.

Os violões mais populares são compostos por 6 cordas e cada uma possui uma afinação diferente da outra em relação a sua nota fundamental. A afinação padrão utilizada é a Em11. Contudo, há diversas outras afinações possíveis de serem utilizadas, como a Drop D, Open G e Dadgad (NIGRO, 2018).

O projeto tem então o objetivo de trazer uma alternativa para músicos inexperientes, que não têm boa percepção auditiva de frequências, mas mesmo assim necessitam de uma afinação precisa no seu instrumento, principalmente àqueles cujo violão não tenha o sistema de afinação acoplado. O afinador automático será, dessa forma, capaz de afinar a corda atuando sobre a tarraxa a partir da vibração detectada pelo transdutor piezoelétrico, afinando o violão automaticamente a partir do momento que o usuário encaixe o atuador na tarraxa e faça vibrar a corda.

O relatório está estruturado da seguinte maneira: primeiro são apresentados os fundamentos teóricos necessários para a compreensão do sistema, em seguida é apresentada a metodologia utilizada para desenvolver o sistema, e por fim, os resultados obtidos e uma análise dos mesmos. Por último, é feita uma conclusão sobre a realização de todo o trabalho,

2. Fundamentos teóricos

2.1. Afinação padrão para violões

A afinação padrão dos violões é tal que ao se tocar todas as cordas soltas o acorde resultante seja o Em11. A Tabela 1 mostra, então, a frequência de cada uma das cordas do violão na afinação padrão, identificando o número da corda (em ordem crescente da corda mais aguda até a mais grave) e o notação formal da nota na escala musical.

Tabela 1. Notas da afinação padrão de um violão.

Corda	Nota	Frequência fundamental	Notação Formal
1	Mi	330Hz	E4
2	Si	247Hz	B3
3	Sol	196Hz	G3
4	Ré	146Hz	D3
5	Lá	110Hz	A2
6	Mi	82Hz	E2

Fonte – NIGRO, 2018.

2.2. Sensor piezoelétrico

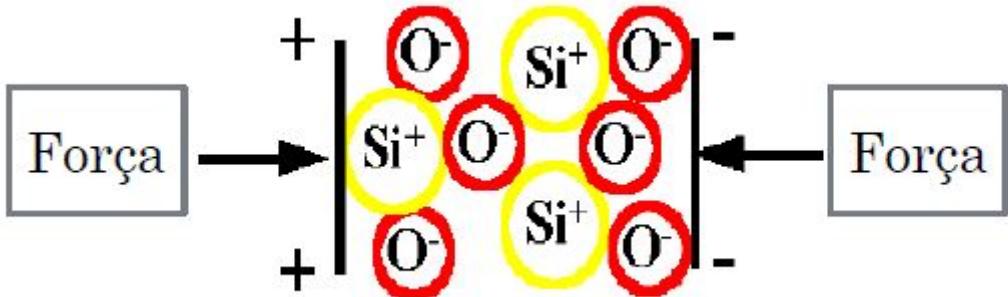
Sensores piezoelétricos são sensores capazes de fornecer uma tensão proporcional a uma deformação aplicada a eles. Nesse projeto, um sensor piezoelétrico é utilizado para medir a vibração no corpo do violão.

2.2.1. Princípios de funcionamento

Conforme Balbinot (2007), o efeito piezoelétrico, propriedade presente nos materiais utilizados nesse tipo de sensor, foi primeiramente observado no cristal de quartzo por Jacques e Pierre Curie em 1880. Esse efeito consiste no surgimento de um potencial elétrico no material quando ele é submetido a uma deformação.

O princípio desse fenômeno pode ser explicado pensando-se na estrutura molecular do material. Quando uma deformação de algum tipo (tração, compressão ou torção) é aplicada ao material, uma quantidade de cargas é deslocada em direção às superfícies do material. Em superfícies opostas, se acumulam cargas de sinais opostos, gerando uma diferença de potencial. Um exemplo gráfico desse princípio é dado na figura 1.

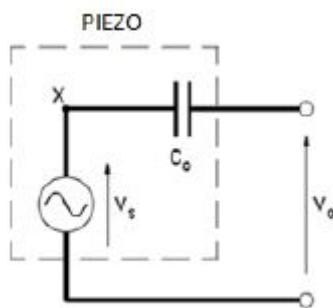
Dois dos materiais utilizados na área de sensores são o PZT (Titanato Zirconato de Chumbo), uma cerâmica que exibe propriedades piezoelétricas quando submetida a altas tensões elétricas de polarização, e o PVDF (Polifluoreto de Vinilideno), fabricado na forma de filme piezoelétrico. Sensores baseando-se nesses dois materiais foram testados no contexto do projeto.

Figura 1. Acumulação de cargas em um material piezoelétrico.

Fonte – BALBINOT, 2018

2.2.2. Circuito condicionador

Para o condicionamento do sinal do sensor, é necessário conhecer o seu modelo de circuito equivalente. Existem dois modelos que podem ser utilizados dependendo da aplicação. O primeiro é uma fonte de tensão em série com uma capacitância. O segundo é uma fonte de cargas em paralelo com uma capacitância e com uma resistência. Esse segundo modelo é útil principalmente em frequências mais altas, onde não pode ser desprezada a resistência interna do sensor. Para a aplicação deste projeto, o primeiro modelo é suficiente, o qual é apresentado na figura 2.

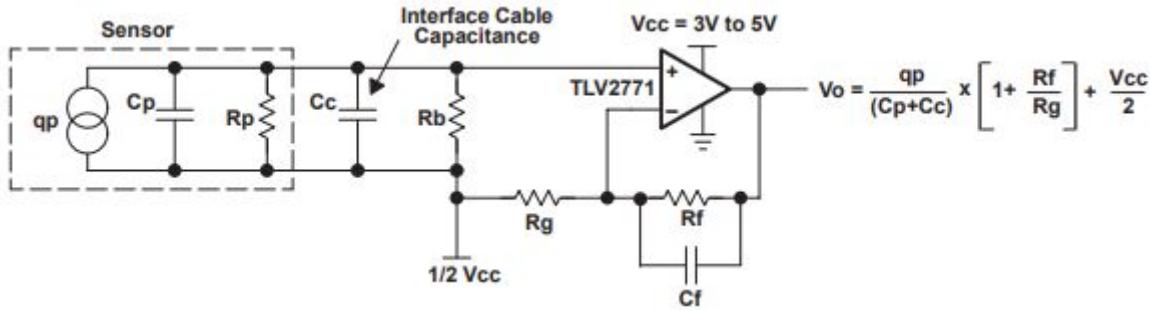
Figura 2. Circuito equivalente do piezoelétrico desprezando resistência interna.

Fonte – BALBINOT, 2018

Segundo o manual da Texas Instruments (2000) sobre condicionamento de sinais de sensores piezoelétricos, existem dois tipos de amplificadores para esses tipos de sensores: modo tensão e modo carga. O modo tensão é indicado quando o condicionador está perto do sensor, e portanto é o modo utilizado no projeto. A topologia do circuito é indicada na figura 3.

Esse circuito possui um ganho dado pela razão entre R_f e R_g . Além disso, ele funciona como um filtro passa-banda, com frequência de corte nas baixas frequências f_L e nas altas frequências f_H dadas pelas equações 1 e 2.

$$f_L = \frac{1}{2\pi(R_p//R_b)(C_p//C_c)} \quad (1)$$

Figura 3. Amplificador modo tensão

Fonte – Texas Instruments, 2000

$$f_L = \frac{1}{2\pi R_f C_f} \quad (2)$$

2.3. Sinais discretos

Um sinal discreto corresponde a uma sequência indexada de números reais ou complexos. Por conta disso, um sinal discreto é uma função de uma variável com valor inteiro. Para se obter a discretização de um sinal contínuo (ou analógico) é necessário que esse seja amostrado através de um conversor analógico/digital (ADC).

De acordo com Hayes (1998), o processo de amostragem de um sinal analógico se refere a: (1) conversão analógica/digital (A/D), (2) quantização do sinal amostrado e (3) codificação do sinal quantizado. A conversão analógica/digital amostra valores do sinal analógico a intervalos constantes de tempo, denominado período de amostragem, T_s , sendo seu inverso a frequência de amostragem ($F_s = 1/T_s$). O sinal convertido digitalmente terá a seguinte relação com o sinal analógico no domínio do tempo:

$$x[n] = x(n \cdot T_s). \quad (3)$$

E no domínio da frequência a relação será:

$$X_d(e^{j\omega}) = \frac{1}{T_s} \sum_{k=-\infty}^{\infty} X_a \left(j \frac{\omega}{T_s} - j \frac{2\pi k}{T_s} \right), \quad (4)$$

onde X_d é o sinal digitalizado e X_a é o sinal analógico nos domínios da frequência digital. É possível observar que o sinal digital será periódico no domínio frequência com período 2π , repetindo-se sob a frequência de amostragem. Dessa forma, é necessário que a amostragem sempre seja feita utilizando uma F_s no mínimo duas vezes maior que a maior componente do sinal analógico. Tal condição é chamada critério de Nyquist, ou teorema da amostragem. Caso o teorema da amostragem não for respeitado, o sinal discretizado sofrerá o efeito de *aliasing*, que corresponde à sobreposição das altas frequências sob o sinal. Tal efeito modifica o sinal original e é incorrigível no domínio discreto. Por conta disso, é necessário sempre se projetar filtros passa-baixas capazes de atenuar os harmônicos de mais alta ordem do sinal a ser discretizado.

A quantificação do sinal passado pela conversão A/D trata-se de um sistema não linear cujo propósito é transformar o sinal amostrado de entrada em um dos valores finitos possíveis dentro da memória do

sistema que armazenará a informação. Para um quantificador de 3bits, por exemplo, é possível armazenar 8 valores distintos em memória ($2^3 = 8$). O tamanho do passo do quantificador é definido como:

$$\Delta = \frac{X_{max}}{2^B}, \quad (5)$$

em que X_{max} é o valor máximo de entrada do ADC e B é o número de bits utilizado na quantização. Como os valores amostrados são restringidos a uma faixa de valores finitos, há uma diferença entre o sinal analógico e suas amostras. Tal diferença resulta no erro de quantização. Seu valor satisfaz a seguinte desigualdade:

$$-\frac{\Delta}{2} \leq e[n] \leq \frac{\Delta}{2}. \quad (6)$$

Por fim, o codificador é o sistema que converte o sinal elétrico de saída do quantizador no valor correspondente em bits a ser armazenado.

2.4. Autocorrelação

Conforme Gajic (2003), a correlação é uma operação muito similar à convolução, mas com significado físico completamente diferente. A correlação pode ser tanto operada em apenas um sinal (autocorrelação) ou entre dois diferentes sinais (correlação cruzada). Fisicamente, a autocorrelação indica o quanto a energia do sinal é distribuída ao longo do sinal. Aplicações típicas da autocorrelação se dão em radares, sonares, satélites e em comunicação sem fio.

A autocorrelação de um sinal discreto, $x[n]$, é dada por:

$$R_{xx}[k] = \sum_{n=-\infty}^{\infty} x[n] \cdot x[n-k], \quad (7)$$

onde k é qualquer inteiro entre $-\infty \leq k \leq \infty$.

Caso o sinal $x[n]$ seja periódico com período T , os valores de pico de sua autocorrelação estarão localizados em:

$$k = \pm F_s \cdot T, \quad (8)$$

em que F_s é a frequência cujo o sinal $x[n]$ foi amostrado. Ou seja, o período do sinal $x[n]$ pode ser obtida por:

$$T = \frac{1}{F_s} = \frac{\Delta k}{F_s}, \quad (9)$$

onde Δk é a distância entre dois picos do sinal de autocorrelação.

2.5. Critério de Chauvenet

Uma amostra de dados pode conter valores espúrios ou duvidosos, que podem constituir erros graves. Para excluí-los judiciosamente emprega-se o critério de Chauvenet a uma amostra de n eventos. O critério baseia-se em identificar o maior desvio da amostra, o que implica em calcular o desvio padrão de cada evento em relação a média (SCHNEIDER, 2007). Dessa forma, define-se uma dispersão aceitável ao redor do valor médio do conjunto de amostras de uma mesma população. O critério de eliminação pode estar relacionado ao desvio padrão máximo aceitável usando a função distribuição de probabilidade Gaussiana (BALBINOT, 2018). A Tabela 2 contém a relação do máximo desvio aceitável para cada conjunto de número de dados.

Tabela 2. Valores do máximo desvio aceitável para diferentes conjuntos de dados.

Número de dados	Máximo desvio aceitável
3	1,38
4	1,54
5	1,65
6	1,73
7	1,80
8	1,87
9	1,91
10	1,96

Fonte – BALBINOT, 2018.

O critério para rejeição das amostras se baseia, então, em calcular o valor de desvio (*tau*) para cada amostra e eliminá-la da base de dados caso seu valor seja maior que o máximo desvio aceitável. O desvio de cada amostra é dado por:

$$\tau = \left| \frac{X_i - \bar{X}}{\sigma} \right|, \quad (10)$$

onde X_i é a i-ésima amostra, \bar{X} é a média dos dados e σ é o desvio padrão dos dados. Tal descarte de dados só pode ser utilizado uma vez no banco de dados e, após o descarte das amostras rejeitadas, uma nova média e desvio padrão devem ser calculados sobre os dados.

2.6. Projeto de filtros passa-baixas

Segundo Karki (2000), a função de transferência de um filtro passa-baixas de segunda ordem pode ser expressa na forma

$$H(f) = -\frac{K}{\left(\frac{f}{FSF \cdot f_c}\right) + \frac{1}{Q} \cdot \frac{jf}{FSF \cdot f_c} + 1}, \quad (11)$$

onde f é a frequência variável, f_c é a frequência de corte do filtro, FSF é o fator de escala da frequência e Q é o fator de qualidade. Os valores de FSF e Q são tabelados para cada estágio do filtro e variam de acordo com a topologia adotada para o mesmo. Em filtros Butterworth, o valor de FSF é 1 para todos os estágio, e o valor de Q pode ser obtido através da Tabela 3. Filtros de topologia Butterworth possuem equação otimizada para que haja ganho quase constante na banda de passagem, atenuação de -3dB na frequência de corte e, acima da frequência de corte, atenuação de -20dB/década/ordem.

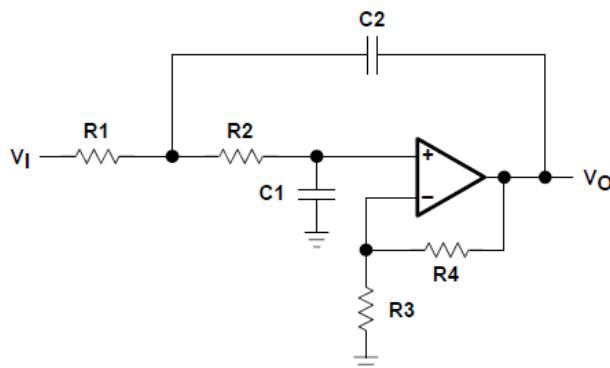
Utilizando a arquitetura de projeto Sallen-Key, como apresentada na Figura 4, para um estágio de filtro passa-baixas, os valores dos parâmetros do filtro são dados por:

- Ganho do filtro: $K = \frac{R3+R4}{R3}$;
- Frequência de corte do filtro: $f_c = \frac{1}{2\pi\sqrt{R1R2C1C2}}$;
- Fator de qualidade: $Q = \frac{\sqrt{R1R2C1C2}}{R1C1+R2C1+R1C2(1-K)}$.

Tabela 3. Valores de Q para cada estágio de um filtro Butterworth de até sexta ordem.

Ordem do filtro	Estágio 1	Estágio 2	Estágio 3
2	0.7071		
3	1		
4	0.5412	1.3065	
5	0.6180	1.6181	
6	0.5177	0.7071	1.9320

Fonte – modificado de KARKI, 2000.

Figura 4. Arquitetura Sallen-Key para um estágio de filtros passa-baixas.

Fonte – KARKI, 2000.

2.7. Lógica fuzzy

A lógica *fuzzy* baseia-se no fato de que uma afirmação pode ser parcialmente verdadeira e, portanto, pode possuir um certo grau de verdade entre completamente falso e completamente verdadeiro. Isso permite com que uma afirmação possua um grau de pertinência no intervalo [0,1]. Desse modo, essa lógica busca preencher o vazio da lógica booleana (verdadeiro ou falso apenas) trabalhando com incertezas, conceitos vagos e com a verdade parcial dos fenômenos naturais de maneira sistemática e rigorosa, permitindo que máquinas (computadores) possam processar tais informações de maneira mais inteligente (BALBINOT, 2018).

No contexto de sistemas de controle, um sistema de lógica *fuzzy* fornece uma metodologia formal para representar, manipular e implementar o conhecimento heurístico de um ser humano sobre como controlar um sistema (PASSINO e YURKOVICH, 1998). Nesses sistemas há, então, um foco maior no uso de regras que ditam como a planta deve ser controlada, diferentemente de sistemas de controles convencionais onde a base para o controle é dada por equações diferenciais ordinárias.

2.7.1. Princípios de Funcionamento

Conforme representado na Figura 5, um controlador fuzzy é composto de quatro blocos principais: interface de "fuzzificação", base de conhecimentos, lógica de tomada de decisões e interface de "defuzzificação". Fuzzificação é o processo de decomposição da entrada e da saída de um sistema em um ou

mais conjuntos *fuzzy* (SIMOES, 2003). Cada um desses conjuntos é representado por uma função de pertinência que atribui um grau de pertinência no intervalo [0,1] a cada um dos seus elementos, sendo os valores 0 e 1 atribuídos a elementos que não pertencem e pertencem totalmente ao conjunto, respectivamente. Este processo permite que sejam atribuídas variáveis linguísticas aos conjuntos, facilitando assim a construção e aplicação das regras que expressam o sistema.

A base de conhecimento do controlador é um conjunto de regras no formato "*Se-Então*" que ditam a melhor maneira de controlar o sistema. Essa base deve ser construída através da consulta de especialista com vasto conhecimento prático no controle do processo, do estudo da dinâmica da planta e/ou através da compreensão do sistema de maneira experimental. Em conjunto com a lógica de decisões, as regras presentes na base de conhecimento dão origem ao mapeamento dos conjuntos de entrada com os conjuntos de saída.

A lógica de decisões do controlador avalia quais são as regras relevantes em um dado instante e utiliza uma estratégia para determinar os valores *fuzzy* da saída do sistema. O processo de defuzzificação, por sua vez, tem por objetivo atribuir um valor numérico à saída do controlador *fuzzy* (e à entrada da planta) com base na variável linguística inferida pela lógica de decisões. Para tal, diferentes métodos podem ser utilizados e estes serão discutidos após o projeto do controlador. Portanto, o controlador *fuzzy* presente no sistema em malha fechada da Figura 5 faz uso das informações de saída do processo, comparando-as com um sinal de referência e gerando um sinal de controle para que os objetivos de performance do processo sejam atingidos.

2.7.2. Projeto de Controlador Fuzzy

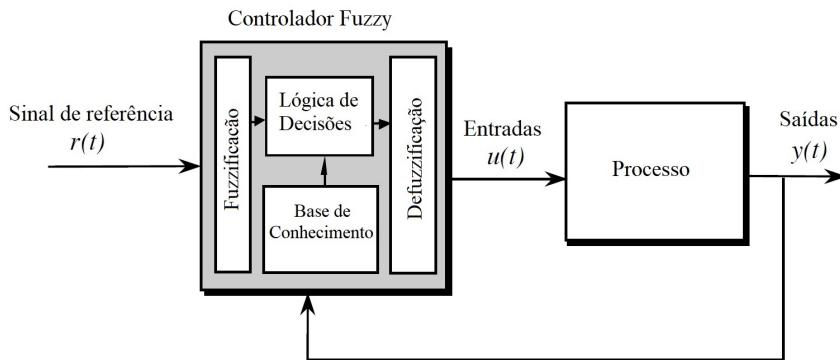
De acordo com Simoes (2003) existem seis passos básicos que são necessários para que um controlador *fuzzy* seja capaz de processar entradas e inferir saídas:

1. Identificar as variáveis de entrada do sistema e suas respectivas faixas de operação, atribuindo nomes (variáveis linguísticas a cada uma delas).
2. Identificar as variáveis de saída do sistema e suas respectivas faixas de operação, atribuindo nomes (variáveis linguísticas a cada uma delas).
3. Criar as funções de pertinência para cada uma das entradas e saídas do sistema.
4. Construir a base de conhecimento que irá ditar a operação do controlador.
5. Decidir como serão executadas as ações através da atribuição de pesos a cada uma das regras.
6. Combinar as regras e "defuzzificar" a saída.

2.8. Tom musical e cents

Segundo Oxenham (2012), o ouvido humano é capaz de perceber sons de frequências numa faixa de aproximadamente 20Hz a 20kHz. Quando um som é harmônico, isto é, apresenta uma componente de frequência fundamental e componentes de frequências múltiplas da fundamental, percebemos o som como uma nota musical, em oposição ao som percebido como ruído, por exemplo (HELMHOLTZ,

Figura 5. Diagrama de blocos de um sistema de controle fuzzy em malha fechada.



Fonte – Adaptado de PASSINO e YURKOVICH, 1998.

1877). Cientificamente, a frequência de ondas vibratórias é normalmente medida em Hz. No entanto, o valor em Hz de um som não reflete o modo em que humanos percebem o som.

Salvo exceções, a maior parte das pessoas não é capaz de perceber a frequência de um som sem um referencial. Ao invés disso, percebem a razão de duas frequências, que corresponde a um intervalo musical (SCHMIDT-JONES, 2013). Por exemplo, quando duas frequências tem uma razão de 2, o intervalo correspondente é uma oitava. Já quando essa razão é 1.5, o intervalo é uma quinta perfeita.

Segundo O'Donnell (2004), na música ocidental, uma oitava é dividida em 12 intervalos de mesma razão, que são chamados de semitonos. Esse sistema, chamado de temperamento igual, estabelece que a razão entre duas notas separadas por um semitom seja sempre a mesma e igual a $2^{1/12}$. Sendo o padrão da nota lá da quarta escala estabelecido em 440Hz, é possível a partir daí calcular a frequência de todas as notas.

Muitas vezes, em instrumentos com cordas, a frequência do som de cada corda tem que ser periodicamente ajustada, pois com o tempo ocorre um desvio da frequência original. Esse desvio é comumente calculado como uma razão de frequências, adequando-se à escala de temperamento igual. Para isso, existe a escala de cents. Um cent equivale a um centésimo de semitom, e equivale a uma razão de $2^{1/1200}$ (SCHMIDT-JONES, 2013). Dessa forma, o desvio em cents de uma frequência f_2 em relação a uma frequência de referência f_1 é dado pela equação 12.

$$\Delta_{cents} = 1200 \cdot \log_2 \left(\frac{f_2}{f_1} \right) \quad (12)$$

Em afinadores, a frequência da corda medida é muitas vezes exibida em cents ao invés de Hz, sendo utilizada como f_1 a frequência padrão de afinação para aquela corda. Quanto menor a diferença em cents medida, mais afinada está a corda.

2.9. Revisão de Metrologia

O sistema desenvolvido foi caracterizado metrologicamente pela sua: resolução, sensibilidade, incerteza-padrão e erro de medição. De acordo com o VIM (2012), tais grandezas são caracterizadas da seguinte forma:

- Resolução: menor variação da grandeza medida que causa uma variação perceptível na indicação correspondente.
- Sensibilidade: Quociente entre a variação duma indicação dum sistema de medição e a variação correspondente do valor da grandeza medida.
- Incerteza-padrão: Incerteza de medição expressa na forma dum desvio-padrão.
- Erro de medição: Diferença entre o valor medido duma grandeza e um valor de referência.

Conforme Balbinot (2018), a resolução de entrada do sistema é calculada como:

$$RE\% = 100 \cdot \frac{\Delta x_{min}}{FS_e}, \quad (13)$$

onde Δx_{min} é a menor variação do sinal de entrada e FS_e é o fundo de escala da entrada. E a resolução de saída é dada por:

$$RS\% = 100 \cdot \frac{\Delta x_{min}}{FS_s}, \quad (14)$$

onde Δy_{min} é a menor variação do sinal de saída e FS_s é o fundo de escala da saída.

A sensibilidade de um sistema é dada por:

$$S_{x_k} = \frac{\partial f}{\partial x_k}, \quad (15)$$

em que f é a função de saída do sistema e x_k é a k-ésima variável de entrada.

A avaliação da incerteza-padrão do tipo A é estimada através da dispersão dos dados obtidos por medidas diretas. A métrica usada é o desvio padrão da média:

$$\sigma(\bar{x}_i) = \frac{\sigma(x_i)}{\sqrt{n}}, \quad (16)$$

onde $\sigma(x_i)$ é o desvio-padrão da i-ésima variável e n é o número de amostras para n medições repetidas do mensurado.

Por fim, o erro de medição pode ser caracterizado através do erro absoluto, dado por:

$$\Delta x = x - x_v, \quad (17)$$

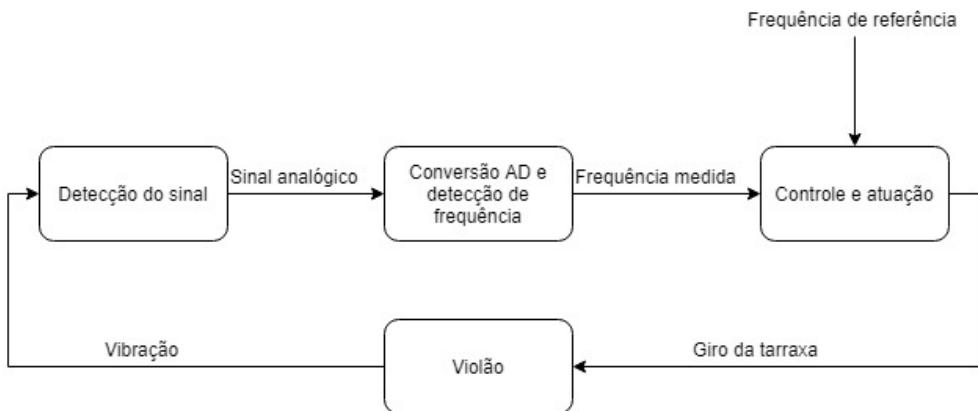
sendo x o valor medido e x_v o valor verdadeiro, ou valor convencional verdadeiro.

3. Metodologia experimental

Baseando-se nos objetivos definidos na introdução, foram identificadas três tarefas principais necessárias para a auto-afinação do violão. A primeira consiste na medição do sinal vibratório do violão, a qual se baseia em sensores piezoelétricos. A segunda consiste na detecção da frequência de vibração. Para isso, será utilizado um dispositivo digital, o que gera a necessidade de um conversor AD. A terceira é o controle e a atuação do giro da tarraxa do violão, através de um motor acoplado à mesma. Esse giro produz uma alteração na frequência de vibração, que será sentida pelo sensor de vibração, o que configura um sistema de malha fechada. O princípio está ilustrado no diagrama de blocos da Figura 6.

É possível ainda detalhar cada um desses grandes blocos quanto ao modo de funcionamento e quanto aos componentes e estágios que serão necessários para a construção do sistema. Os blocos menores, contidos nos blocos gerais do projeto, estão apresentados na figura 7.

Figura 6. Diagrama de blocos do sistema utilizado no projeto.



Fonte – Própria, 2018.

3.1. Escolha do sensor piezoelétrico

Sensores piezoelétricos de dois tipos foram adquiridos: cerâmica e filme piezoelétrico. O primeiro consiste em um disco fino e rígido de PZT, enquanto que o segundo é um filme retangular flexível de PVDF. Para um teste qualitativo, os dois foram acoplados ao corpo do violão para verificação da tensão produzida nos terminais.

No teste do sensor cerâmico, foi verificado que o sinal pode chegar a uma amplitude da ordem de centenas de mV, no momento em que uma corda é tocada. Já no teste do filme piezoelétrico, não foi possível detectar um sinal de amplitude considerável comparativamente ao ruído.

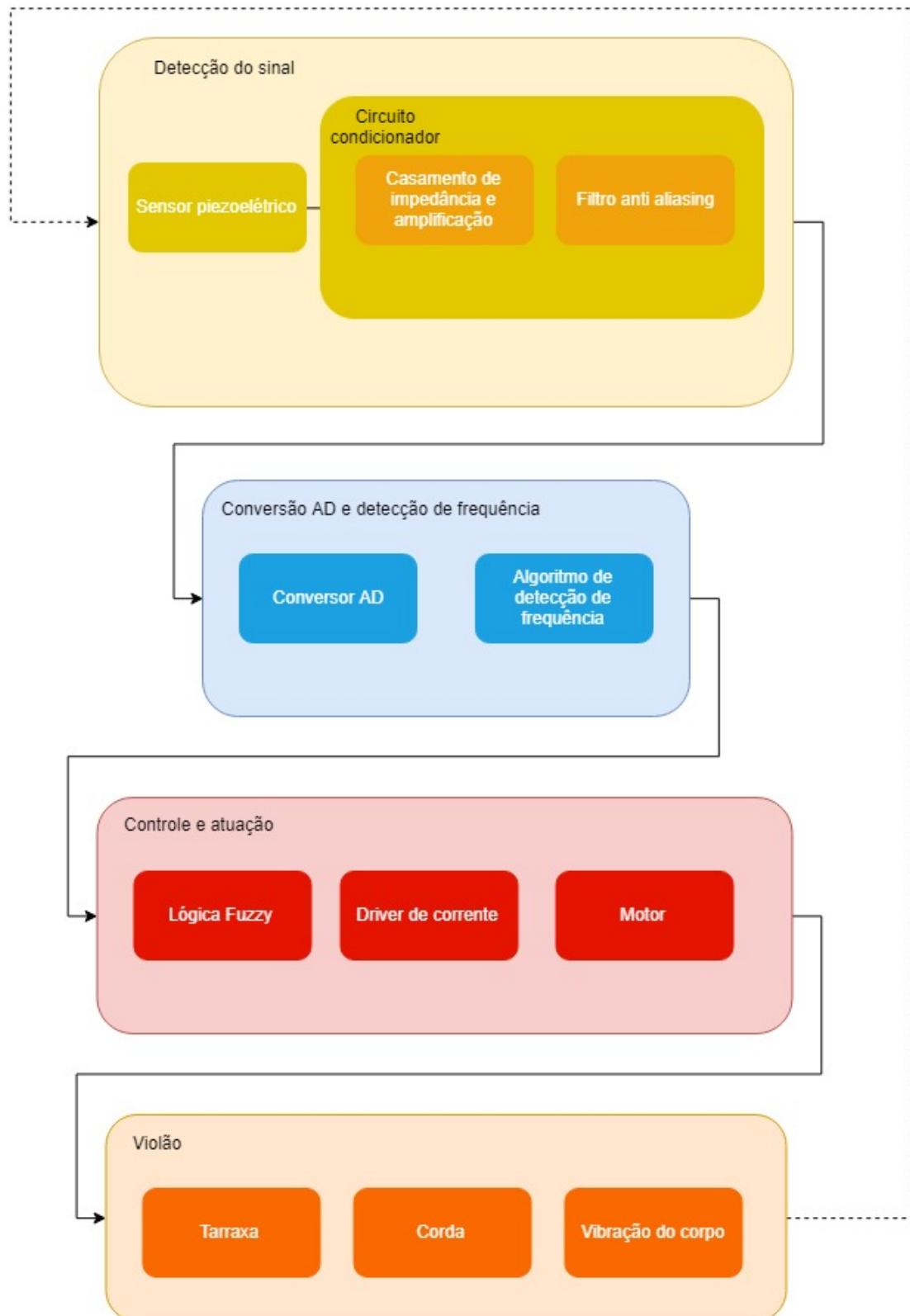
Constatou-se que isso se deve à rigidez do material. Como o filme piezoelétrico é muito menos rígido, o esforço sobre ele para uma mesma deformação é muito menor, o que causa um sinal menor na sua saída. O filme piezoelétrico seria ideal para aplicações onde há grande deformação. Portanto, decidiu-se pela utilização do sensor cerâmico.

3.2. Acoplamento do sensor ao violão

O sensor piezoelétrico foi acoplado no interior do violão. Como o objetivo é que a vibração do violão seja captada pela deformação sofrida pelo sensor, buscou-se que ele fosse colocado em um ponto onde a vibração tem maior intensidade. Para isso, foram testados três pontos diferentes, conforme indicado na figura 8.

As posições 1 e 2 foram escolhidas pela sua proximidade com a ponte, uma vez que é através dessa peça que a vibração é transmitida das cordas para o corpo do violão. Já a posição 3 foi escolhida após uma análise empírica dos pontos de maior vibração no tampo do violão.

Em cada um dos pontos, um sensor foi colado com cola silicone. Após uma verificação do sinal de tensão nos terminais de cada sensor enquanto que uma corda do violão vibrava, constatou-se que a amplitude do sinal elétrico era maior para o sensor na posição 2. Portanto, essa foi a posição escolhida.

Figura 7. Detalhamento dos blocos constituintes do sistema.

Fonte – Própria, 2018.

Figura 8. Foto do violão com as três posições testadas para os sensores piezoelétricos



Fonte – Própria, 2018

3.3. Circuito condicionador

Segundo os estágios do circuito que já foram apresentados, o circuito condicionador tem duas funções. A primeira é fazer a interface com o sensor piezoelétrico e amplificar a tensão por ele gerada. A segunda é filtrar as altas frequências, com o intuito de reduzir o ruído presente na medição e evitar o efeito de *aliasing*, possibilitando a digitalização do sinal. Além disso, é necessário que o sinal de saída esteja contido na faixa para a qual o conversor AD do kit de desenvolvimento Teensy esteja configurado. Para que essas funções sejam desempenhadas corretamente, é também preciso uma alimentação de tensão estável e precisa, o que deve, do mesmo modo, ser garantido pelo circuito. Os diferentes blocos do circuito são mostrados a seguir.

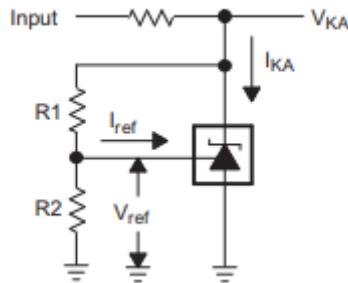
3.3.1. Circuito de alimentação

Para estabilizar a alimentação do circuito, foi escolhido o uso de o regulador de tensão do tipo shunt programável TL431A. Esse componente é capaz de fornecer uma tensão de alimentação com precisão de 1% programável pelo valor de dois resistores, tendo o valor mínimo de 2,495V. Segundo o datasheet do produto, o circuito necessário para o seu setup é indicado na figura 9. A tensão no nó V_{KA} é dada pela expressão da equação 18.

$$V_{KA} = V_{ref} \left(1 + \frac{R_1}{R_2} \right) + I_{ref} R_1 \quad (18)$$

Para a alimentação dos amplificadores operacionais (modelo TL074), é desejável uma tensão da ordem de 10V. Para isso, buscou-se uma razão R_1/R_2 igual a 3. Como I_{ref} pode variar de $2\mu A$ a $4\mu A$, conforme o datasheet, foi escolhido R_1 na ordem dos milhares de ohms, para que a influência da corrente de referência seja pequena comparada à incerteza.

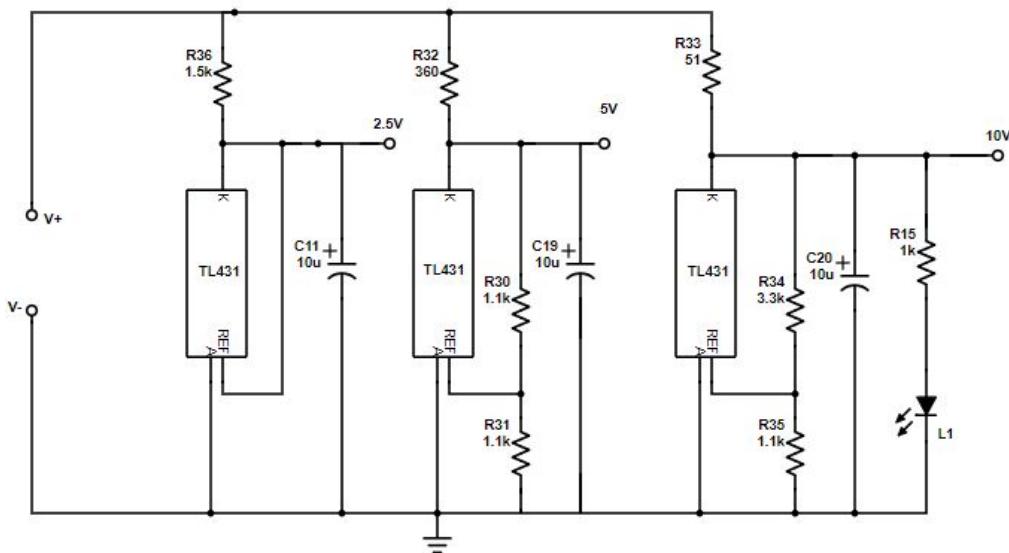
O valor calculado para a tensão V_{KA} é então 9,98V, que pode ser aproximado para 10V devido à faixa de incerteza de aproximadamente $\pm 0,1$.

Figura 9. Esquema do circuito de tensão de referência.

Fonte – Texas Instruments, 2018

Para o estágio inicial do condicionador, também é necessária uma tensão de referência de 5V, que foi obtida com outro regulador de tensão tendo $R_1 = R_2$. Além disso, foi utilizada uma tensão de referência de 2,5V para o ADC, obtida conectando-se V_{ref} em V_{KA} .

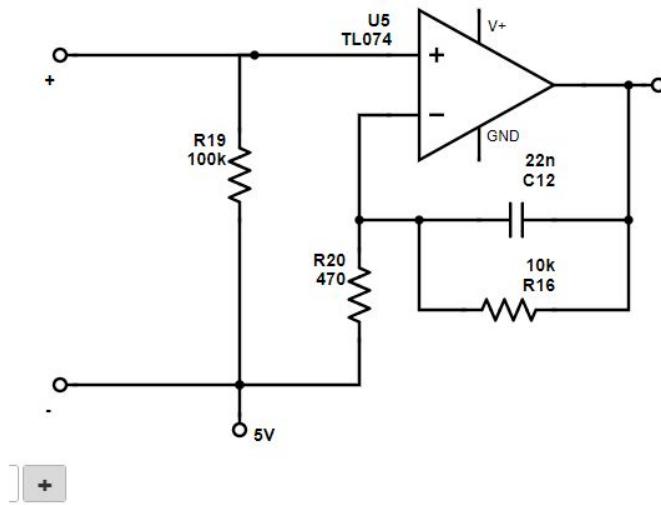
Para esses três reguladores, foram colocados capacitores em paralelo para estabilização da tensão, e um LED para indicar que a alimentação está ligado. Eles foram projetados para que $1mA < I_{KA} < 100mA$, conforme o datasheet do componente. O circuito final de alimentação é indicado na figura 10. Os resistores que regulam o nível de tensão da saída têm precisão de 1%.

Figura 10. Circuito de alimentação do condicionador.

Fonte – Própria, 2018

3.3.2. Amplificação e filtragem

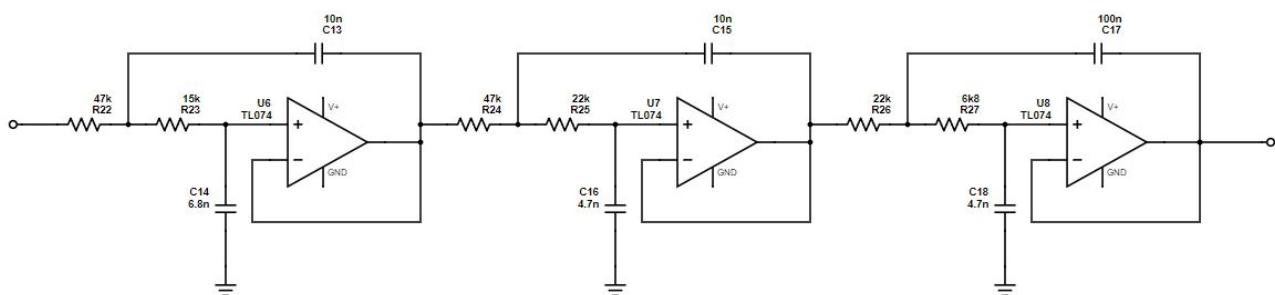
O estágio inicial do condicionador, que se conecta ao sensor, é necessário para casamento de impedância e amplificação do sinal. Baseando-se no circuito da figura 3, foi projetado o seguinte circuito.

Figura 11. Estágio inicial do condicionador.

Fonte – Própria, 2018

O ganho calculado para ele foi de $10000/470 = 21.3$ e foi escolhido empiricamente pela análise da amplitude do sinal de saída, em comparação com a tensão necessária na entrada do conversor AD. A frequência de corte nas altas frequências desse filtro é $1/(2\pi \cdot 10k \cdot 22n) = 723Hz$, e é próxima da frequência de corte do filtro anti-aliasing. A frequência de corte em baixas frequências depende dos parâmetros internos do piezoelétrico e da capacitância dos cabos, e por isso é mais difícil de ser calculada. No entanto, não foi percebido efeito atenuante nas frequências de interesse mais baixas (pouco menores que 80Hz), e portanto a atenuação em baixas frequências foi desconsiderada.

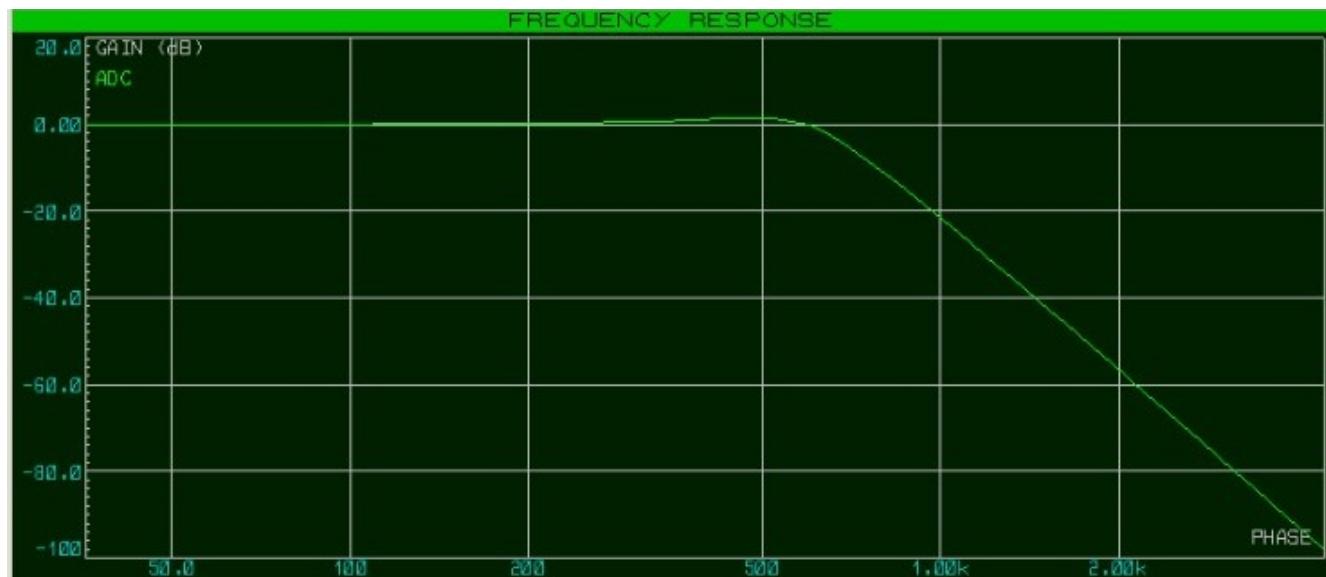
O circuito anti-aliasing foi projetado com o intuito de evitar o efeito aliasing na conversão AD, bem como para diminuir o ruído em altas frequências do sinal do filtro. Ele foi projetado com a arquitetura Salen-Key para sexta ordem, e está mostrado na figura 12.

Figura 12. Esquemático do filtro anti-aliasing desenvolvido.

Fonte – Própria, 2018.

O circuito desenvolvido foi então simulado utilizando o software Proteus. A simulação baseou-se em obter a resposta em frequência do filtro *anti-aliasing*, que se observa na figura 13. Como se pode ver, ele atenua fortemente para frequências acima da frequência de corte, que está próxima de 600Hz.

Figura 13. Resposta frequencial simulada do condicionador.



Fonte – Própria, 2018.

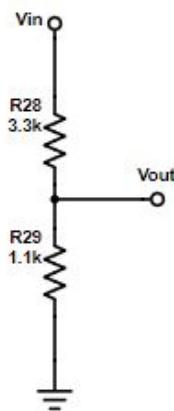
O nível DC do sinal na saída desse filtro é o mesmo da saída do amplificador, ou seja, 5V. No entanto, é necessário que ele esteja contido numa faixa $0V < V_{sinal} < V_{refADC}$, sendo a tensão de referência no máximo igual a 3,3V. Então, a tensão de referência escolhida foi de 2,5V, produzida no circuito de alimentação conforme já mencionado. Para máxima excursão de sinal dentro dessa faixa, é necessário colocar o nível DC em 1,25V. Para isso, é utilizado um divisor de tensão conforme a figura 14. Os resistores escolhidos para o divisor tem precisão de 1%.

3.4. Motor de Atuação

A escolha do motor é uma parte importante do desenvolvimento do projeto, pois, caso não realizada com cuidado, esta pode dificultar o processo de atuação e afetar o resultado final da afinação. É importante que o motor gire de maneira lenta o suficiente para que não haja risco de que as cordas se quebrem devido ao excesso de tensão, porém o tempo de afinação também deve ser levado em consideração. Desse modo, um motor com rotação entre aproximadamente 20 RPM e 40 RPM deve ser escolhido.

Os requisitos de torque das tarraxas de um violão foram medidos por Bocanegra (2005) e generalizados pelo grupo. Foi concluído então que para que o motor seja capaz de atuar em todas as tarraxas do violão, este deve produzir um torque mínimo de 0,283Nm. Portanto, as opções de motores que podem ser capazes de atender as especificações do projeto são motores de corrente contínua (CC), motores de passo e motores servo modificado para rotação de 360 graus.

A fim de preencher os requisitos de torque, rotação e fornecer uma certa precisão ao processo de afinação foi escolhido o motor de passo bipolar NEMA 17 17HS4401. Para o acoplamento do motor

Figura 14. Divisor de tensão para ajuste DC.

Fonte – Própria, 2018

com a tarraxa do violão foi utilizado o sistema de acoplamento um "encordoador" de violões. O motor e o sistema de acoplamento adaptado estão representados na Figura 15. As especificações do motor, fornecidas pelo fabricante, encontram-se na tabela 4.

Figura 15. Motor 17HS4401 e sistema de acoplamento à tarraxa.

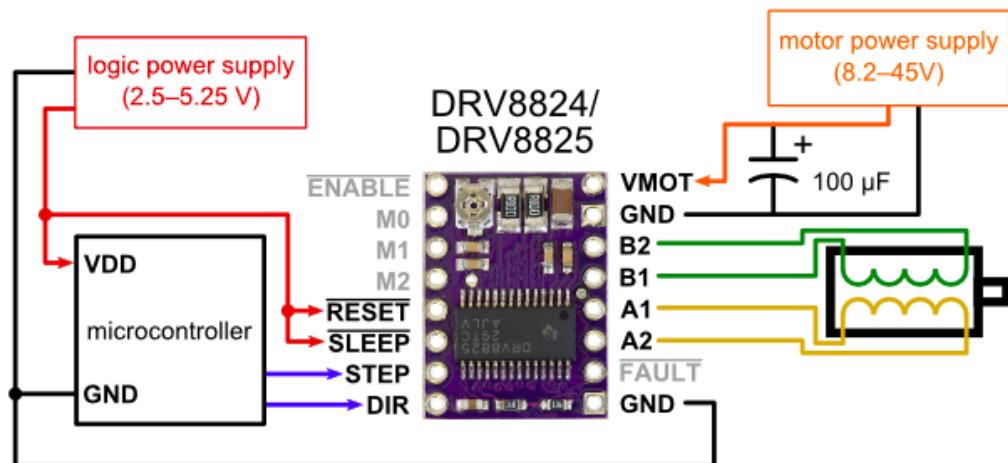
Fonte – Própria, 2018.

Devido a alta demanda de corrente por parte do motor e seu funcionamento, se fez necessária a escolha de um *driver* para motor de passo. Foi escolhido uma placa de circuito comercial baseada no DRV8825, um *driver* de alta corrente que permite que o motor de passo seja controlado bidirecionalmente através da emissão de pulsos. Sua utilização permitiu que a corrente nas bobinas do motor fosse limitada a 0,3A/fase, sua velocidade máxima fosse fixada em 100 passos/seg e sua aceleração em 100 passos/seg². A figura 16 representa as conexões do *driver*.

Tabela 4. Especificações do motor 17HS4401.

Característica	Especificação
Tipo	Bipolar de 4 fios
Ângulo de Passo	1.8°
Torque Estático	40 Ncm
Corrente Nominal/fase	1.7 A
Resistência de fase	$1.5 \pm 10\% \Omega$

Fonte – Adaptado de MOTIONKING, 2018.

Figura 16. Diagrama de conexões entre o motor de passo, o *driver* e o microcontrolador.

Fonte – (POLOLU, 2018).

3.5. Teensy 3.6

Para a elaboração do trabalho, escolheu-se utilizar o kit de desenvolvimento Teensy 3.6, desenvolvido pela PJRC. Nesse kit, é utilizado o núcleo de microcontrolador Cortex-M4F, que permite executar tarefas de alta performance, incluindo processamento via *hardware* para ponto flutuante de 32bits. As especificações mais relevantes do Teensy 3.6 se encontram na Tabela 5.

3.6. Algoritmos

3.6.1. Amostragem

A amostragem do sinal foi feita com uma frequência de amostragem de 50kHz e com uma janela de 15000 amostras (300ms). O algoritmo responsável pela amostragem se encontra dentro da interrupção do *timer* que é acionado com um período de $T_s = 1/F_s = 20\mu s$. A janela de amostras só começa a ser preenchida caso seja amostrado um sinal cujo valor mínimo esteja acima de um determinado *threshold*, afim de que se evite o processamento em trechos sem sinal. Quando a janela é preenchida, uma *flag* é disparada informando que o algoritmo de detecção da frequência fundamental pode ser executado sob as amostras.

Tabela 5. Especificações técnicas do Teensy 3.6.

Característica	Especificação
Processador	MK66FX1M0VMD18
Núcleo do processador	Cortex-M4F
Velocidade de processamento	180Mhz
Memória Flash	1024kbytes
RAM	256kbytes
EEPROM	4096bytes
E/S digitais	58 pinos
Tensão de saída	3.3V
Corrente de saída	50mA
Conversores (ADCs)	2
Resolução dos ADCs	12bits
Tensão de entrada máxima	3.3V
Timers	19 (6 com 16bits)

Fonte – modificado de PJRC, 2018.

3.6.2. Detecção da frequência fundamental

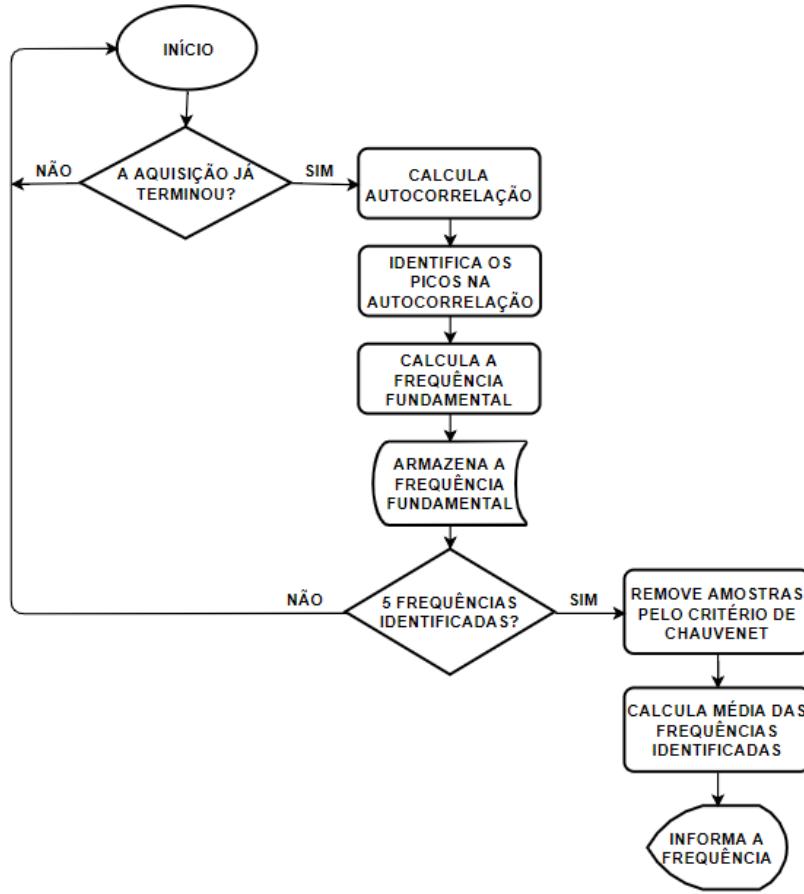
O algoritmo desenvolvido para a detecção da frequência fundamental da nota tocada tem como base a operação de autocorrelação com o uso do critério de Chauvenet para o descarte das amostras. Ele consiste, portanto, em: (1) identificação da frequência fundamental da corda que está sendo tocada através da autocorrelação; (2) armazenamento de 5 amostras de frequências fundamentais identificadas; (3) uso do critério de Chauvenet para descarte de amostras; (4) média das amostras restantes como estimador da frequência fundamental da nota. O fluxograma da Figura 17 apresenta com mais detalhes as etapas do algoritmo.

3.6.3. Controlador fuzzy

O processo de atuação no motor de passo foi controlado através do uso de um controlador *fuzzy*. Para tal, foi projetado um sistema em malha fechada como o representado de maneira geral na Figura 5, onde o processo a ser controlado é a afinação do violão. A variável de entrada escolhida foi o *erro* de frequência, dado em *cents*. A referência para o cálculo do erro varia com a corda, visto que é a frequência de afinação da mesma. A variável de saída escolhida para o sistema foi o número de passos que o motor deve girar, visto que a natureza sequencial de processamento do microcontrolador torna impossível que a realização da leitura de frequência e a atuação aconteçam simultaneamente. O controlador foi projetado utilizando o *Fuzzy Logic Designer* do Matlab.

Seguindo o modelo apresentado por Rahnamai, Cox e Gorman (2007), foram definidas três funções de pertinência para entrada: frequência baixa (*freqBaixa*), afinado (*Afinado*) e frequência alta (*freqAlta*). Já para a saída, foram definidas outras três funções de pertinência: rápido anti-horário (*RapAntHorario*), zero e rápido horário (*RapHorario*). As funções de pertinência da entrada são dadas pelas equações 19,

Figura 17. Fluxograma do algoritmo para detecção da frequência fundamental.



Fonte – Própria, 2018.

23 e 21, enquanto que as da saída são dadas pelas equações 22, 23 e 24. Estas estão representadas graficamente na Figura 18.

$$\mu_{freqBaixa}(x) = \begin{cases} 1 & x < -100 \\ \frac{-10-x}{90} & -100 \leq x \leq -10 \\ 0 & x > -10 \end{cases} \quad (19)$$

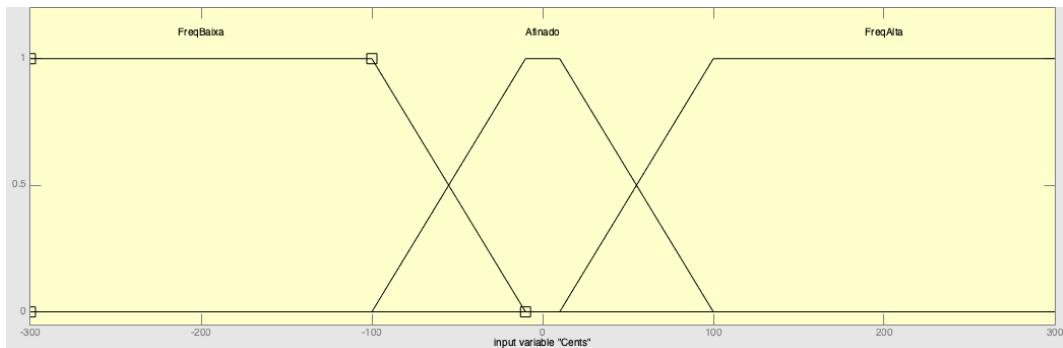
$$\mu_{afinado}(x) = \begin{cases} \frac{x+100}{90} & -100 \leq x \leq -100 \\ 1 & -10 < x \leq -10 \\ \frac{100-x}{90} & 10 < x \leq -100 \end{cases} \quad (20)$$

$$\mu_{freqAlta}(x) = \begin{cases} 0 & x < 10 \\ \frac{100-x}{90} & -100 \leq x \leq -10 \\ 1 & x > 100 \end{cases} \quad (21)$$

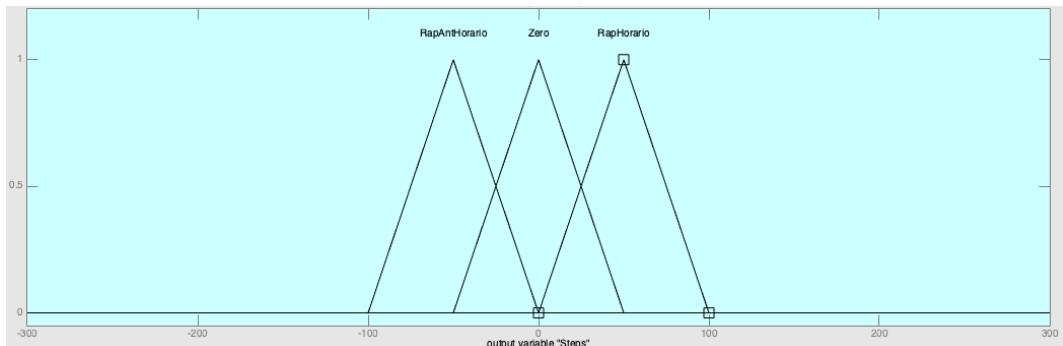
$$\mu_{rapAntHorario}(x) = \begin{cases} \frac{x+100}{50} & -100 \leq x \leq 50 \\ \frac{-x}{50} & 50 < x \leq 0 \end{cases} \quad (22)$$

$$\mu_{afinado}(x) = \begin{cases} \frac{x+50}{50} & -50 \leq x \leq 0 \\ \frac{50-x}{50} & 0 < x \leq 50 \end{cases} \quad (23)$$

$$\mu_{rapHorario}(x) = \begin{cases} \frac{x}{50} & 0 \leq x \leq 50 \\ \frac{100-x}{50} & 50 < x \leq 100 \end{cases} \quad (24)$$

Figura 18. Funções de pertinência do sistema.

(a) funções de pertinência da entrada.



(b) funções de pertinência da saída.

Fonte – Própria, 2018.

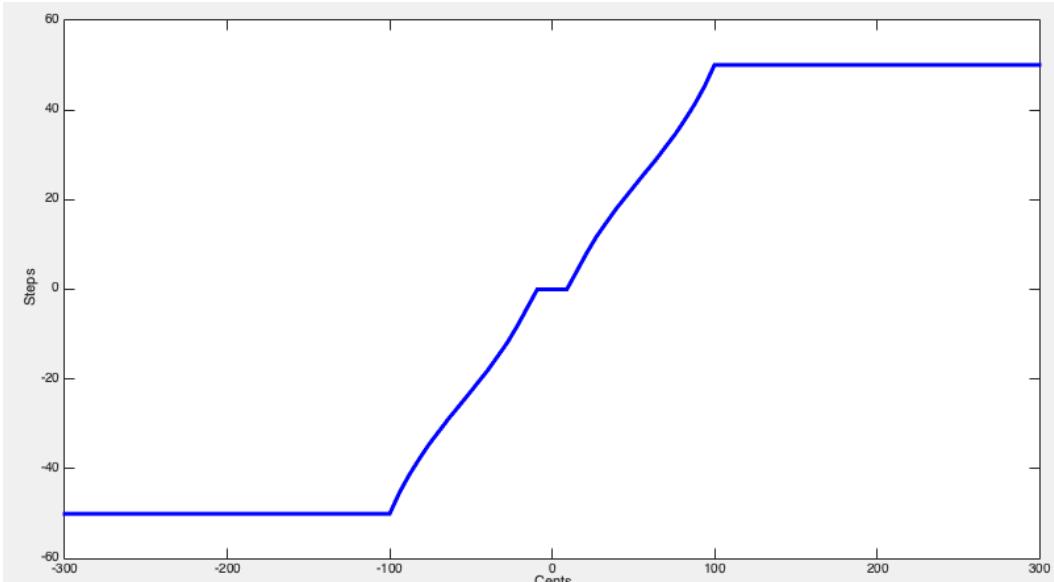
Para o controlador projetado foi escolhido o método de inferência Mandani com operadores *MAX-MIN*. O processo de *defuzzificação* é dado pelo método do centroide. A partir das funções de pertinência foi criado um conjunto de três regras descritos na Tabela 6. O processo de refinamento dos parâmetros do controlador se deu de maneira experimental visto que a geração de um modelo matemático para cada uma das cordas é um processo complexo e demorado. A resposta do sistema para o conjunto de regras é dada na Figura 19.

Em um processo de afinação ideal a função de pertinência *afinado* (Figura 18(a)) possuiria um formato triangular, visto que o valor de afinação é único e bem definido. Assim, o motor só deixaria de atuar no sistema quando o valor exato de afinação fosse atingido (*erro = 0*). No entanto, devido a incerteza e variabilidade do processo de medição de frequência, fez-se necessário adicionar uma "zona morta" para este valor de ± 10 cents. Desse modo, há ausência de uma saída de atuação para o motor quando a entrada se encontra nessa faixa, e isso é observado claramente na Figura 19.

Tabela 6. Conjunto de regras fuzzy para o controlador.

SE	ENTÃO
<i>freqBaixa</i>	<i>RapHorario</i>
<i>afinado</i>	<i>zero</i>
<i>freqAlta</i>	<i>rapAntHorario</i>

Fonte – Própria, 2018.

Figura 19. Resposta do controlador fuzzy para o conjunto de regras especificado.

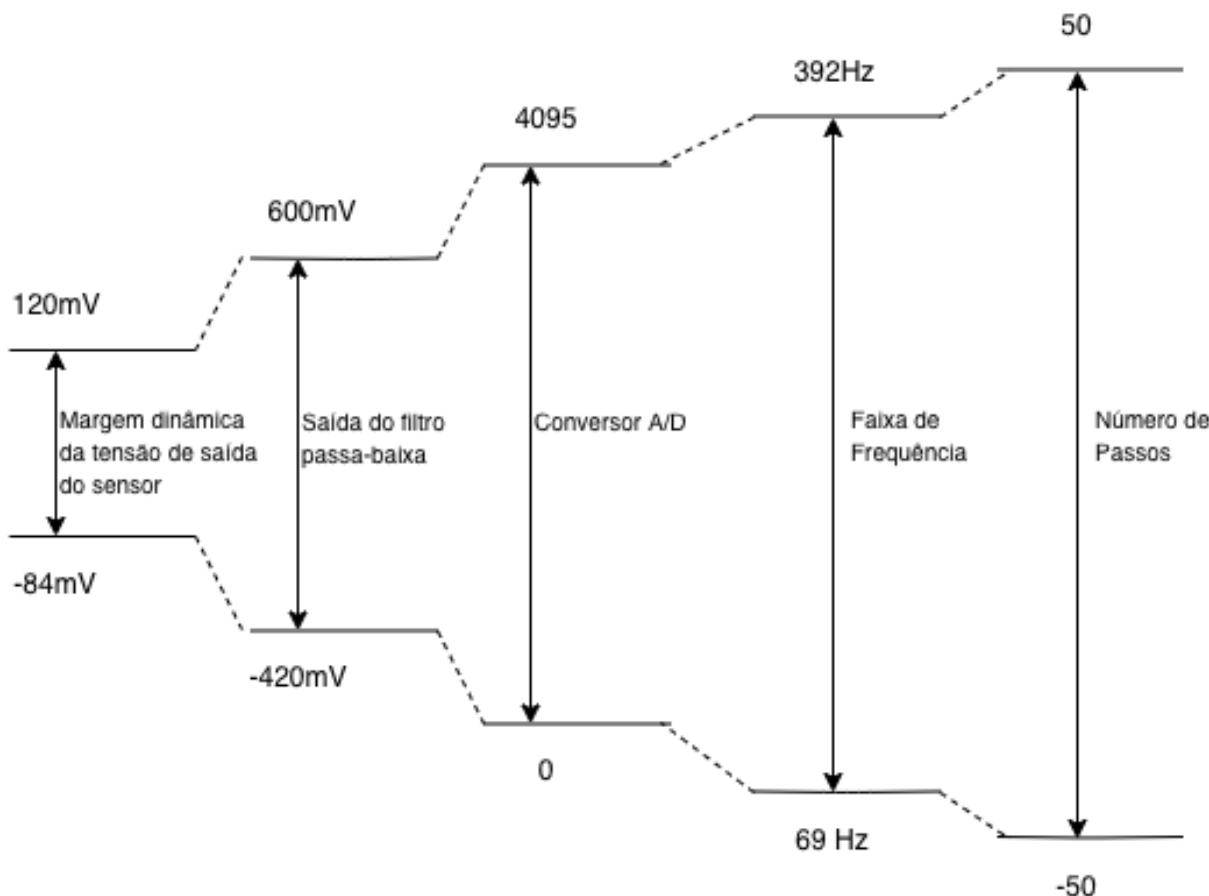
Fonte – Própria, 2018.

3.7. Cadeia de medidas

Para representar todas as etapas envolvidas no processo de aquisição do sinal, detecção da frequência, controle e atuação, foi feita uma cadeia de medidas. Essa cadeia está representada na figura 20. A entrada do sistema é definida como a tensão no sensor piezoelettrico, uma vez que não seria possível para o grupo, com os aparelhos de medição disponíveis, determinar a vibração no tampo do violão. Além disso, não há um grande interesse nesse valor para o escopo do projeto, uma vez que o valor medido não é a grandeza em si e sim a sua frequência.

3.8. Layout em PCB

Uma vez projetado todo o circuito condicionador, escolhido o Teensy 3.6 como microcontrolador utilizado, o motor e o seu respectivo driver, pôde-se fazer o design do circuito impresso. O layout utilizado ocupou uma área de 10cmx10cm, conforme a figura 21.

Figura 20. Cadeia de medidas do sistema.

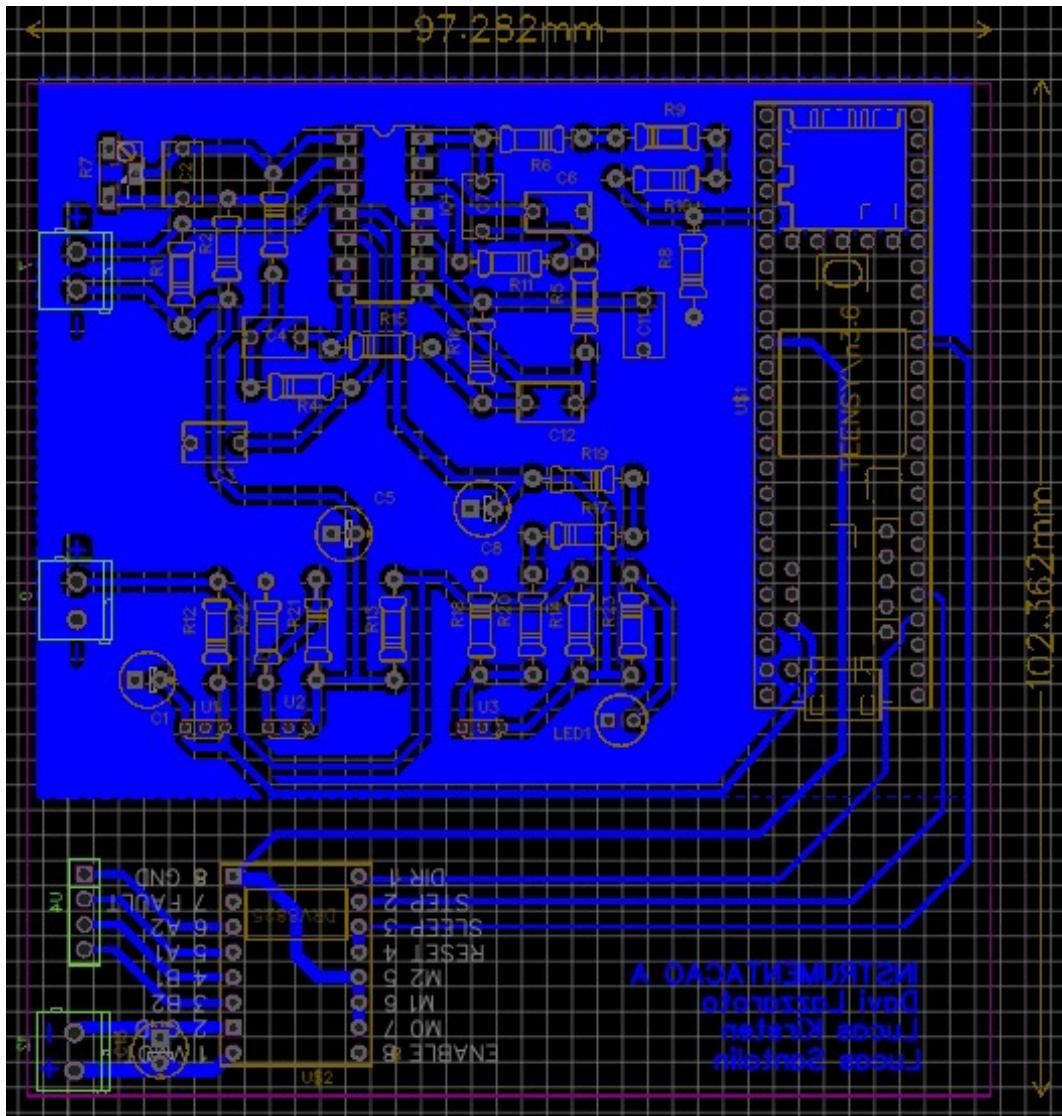
Fonte – Própria, 2018.

3.9. Caracterização do sistema

Foram projetados três testes para a caracterização do sistema. O primeiro foi apenas para o circuito anti-aliasing, e consistiu no levantamento de sua resposta em frequência. Para isso, foi utilizado um gerador de frequências colocado na entrada do circuito, somado com um nível DC de 5V para excursão em torno da metade da alimentação dos amplificadores operacionais. Utilizou-se também um osciloscópio para observação do sinal da saída e medição de sua amplitude e frequência. O teste foi iniciado com a frequência do sinal na entrada em aproximadamente 10Hz, e foi-se aumentando gradualmente até o ponto onde a amplitude do sinal na saída não pudesse mais ser detectada devido à atenuação do filtro. Para cada frequência, foi medida a amplitude tanto na entrada do filtro quanto na saída, para o cálculo da razão entre a amplitude na entrada e na saída, usada para estimar o ganho do filtro naquela frequência.

O segundo teste consistiu na determinação da dispersão dos dados de frequência medidos pelo sistema constituído pelo sensor, condicionador e detector de frequência. Para isso, foi conectado o sensor

Figura 21. Layout da placa de circuito impresso.



Fonte – Própria, 2018

acoplado no violão na entrada do circuito. Em seguida, o violão foi afinado com um afinador utilizado como referência de acordo com a afinação padrão. Então, foi feito um grande número de medidas de frequência para cada corda. Foi calculada a média e o desvio padrão para a medição de frequência de cada corda, para estimativa da incerteza de medição. Essa incerteza é calculada pelo desvio-padrão da média. Para cada valor medido, foi também calculado o desvio em número de cents. Para esses valores também foram calculados a média e o desvio-padrão.

O último ensaio de caracterização consistiu num teste do sistema inteiro, tanto a medição de frequência como a atuação. Para esse teste, o violão foi conectado à entrada do condicionador e suas cordas desafinadas. A seguir, foi realizada a medição de frequência para cada corda, com o fuzzy ativado e o motor acoplado à tarraxa. Com o ambiente preparado, a corda foi tocada por um dos integrantes do grupo, e o motor acoplado à tarraxa foi afinando gradualmente a corda, até o momento onde fossem realizadas três medições seguidas em que a corda fosse considerada afinada. Nesse momento, a frequência

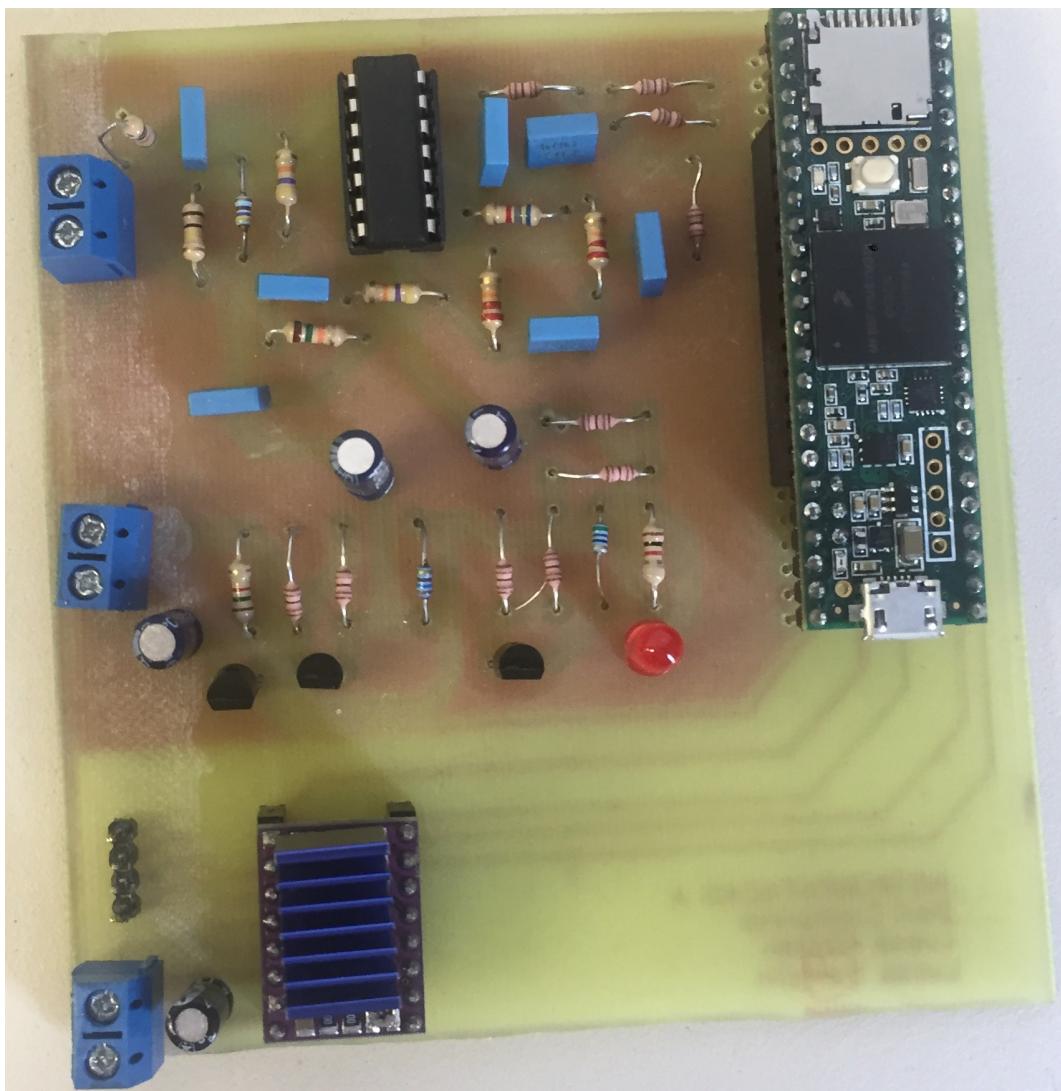
da corda era medida com o afinador de referência, determinando então o erro de afinação. Os resultados para os três ensaios são apresentados na próxima seção.

4. Resultados e Discussões

4.1. Circuito desenvolvido

Depois do processo de fabricação do circuito impresso, foi obtida a placa mostrada nas figuras 22 e 23.

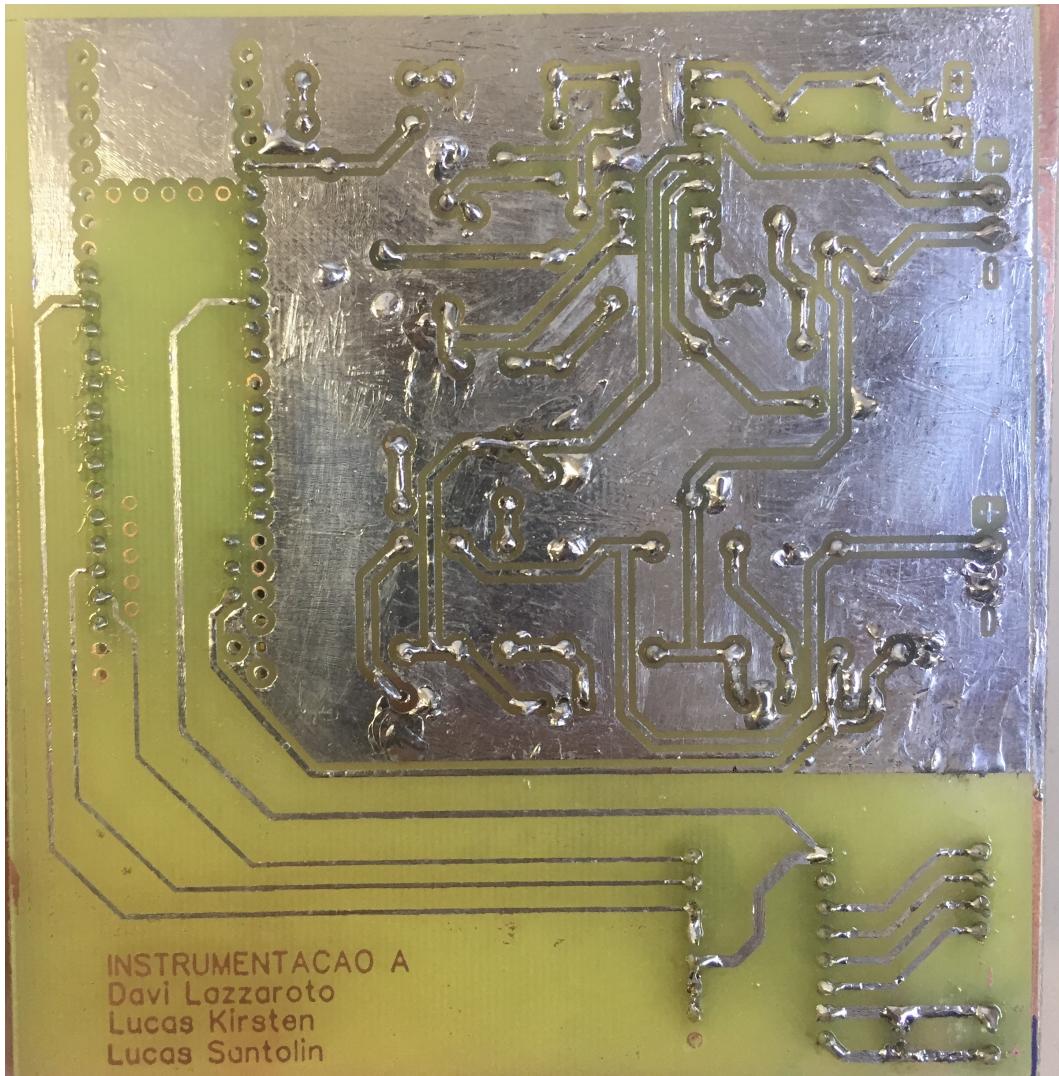
Figura 22. Placa em circuito impresso vista de cima.



Fonte – Própria, 2018

Conectando-se o violão na entrada do circuito condicionador e observando o sinal na entrada do conversor AD com um osciloscópio, observa-se o sinal mostrado na figura 24. O modelo do osciloscópio utilizado na medição é Agilent Techonologies DSO 3102A. A acquisição foi realizada no momento em que foi tocada a corda Lá no violão. Observa-se que a tensão pico a pico vai a aproximadamente 1V, e o tempo de sustentação do sinal é grande para essa corda, havendo sinal detectável até cinco segundos

Figura 23. Placa em circuito impresso vista de baixo.



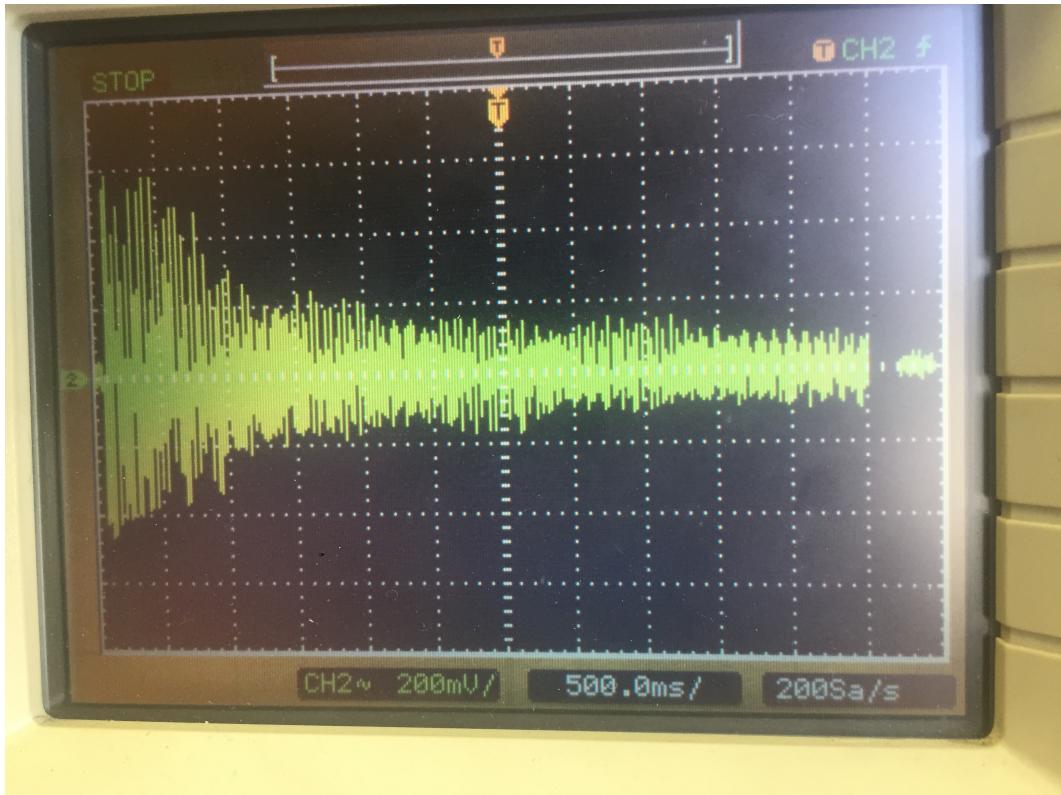
Fonte – Própria, 2018

depois do momento em que a corda é tocada.

Para a caracterização do filtro anti-aliasing foi utilizado o mesmo modelo de osciloscópio. Para a geração do sinal de entrada, foi utilizado um gerador de funções modelo Simpson 420. A resposta em frequência experimental do filtro é mostrada na figura 25. Comparando essa resposta com a figura 13, observa-se que a resposta de frequência experimental ficou muito próxima da simulada, com a mesma frequência de corte e a mesma inclinação na faixa de rejeição.

4.2. Caracterização do sistema

A Figura 26 mostra os valores da resolução de entrada do sistema em relação às frequências detectáveis. É possível observar, portanto, o comportamento não linear de tal grandeza, de forma que ela aumenta a medida que o valor de frequência fundamental a ser detectada também aumenta. Entretanto, para a maior frequência ser afinada (330Hz), seu valor está na ordem dos 0.55%, que se converte num valor de 2.18Hz e um desvio em cents de 11.4.

Figura 24. Sinal medido na saída do condicionador para a nota Lá.

Fonte – Própria, 2018

A relação entre o valor mensurado e o valor de afinação (entrada e saída) do sistema é ilustrada na Figura 27. Através dessa figura, verifica-se uma sensibilidade do sistema de $S = 1.01\text{Hz}/\text{Hz}$ (o que era esperado por se tratar de um sistema linear).

Na Figura 28 é apresentado os histogramas das medições feitas com as cordas afinadas. Os valores da média e da incerteza-padrão tipo A calculados para os valores mensurados de cada corda são apresentados, então, na Tabela 7. A incerteza-padrão é calculada utilizando o desvio-padrão da média como estimador. Como é possível observar, a incerteza na medida é maior para as cordas mais agudas (1,2,3) e menor para as cordas mais graves. Além disso, observa-se que a média das medidas é bem próxima ao valor da afinação para as cordas graves. No entanto, para as cordas mais agudas esse valor difere muito.

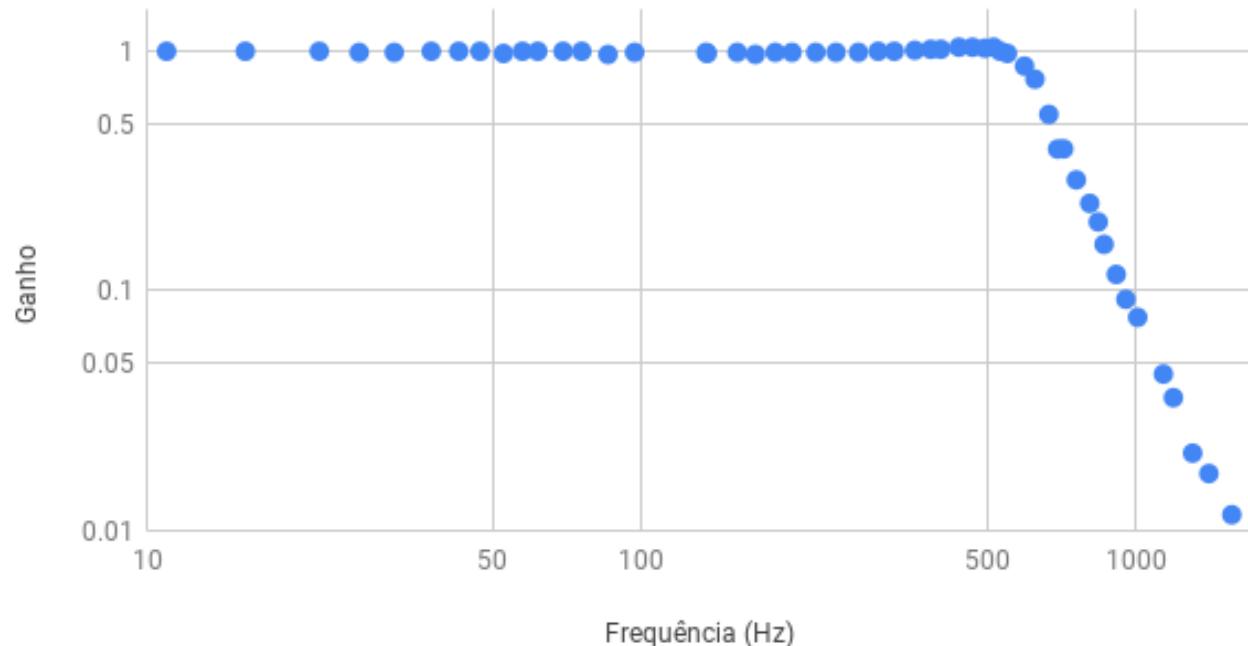
Tabela 7. Valores de incerteza-padrão para cada corda.

Corda	Medida [Hz]	Afinação [Hz]
1	337.6 ± 0.71	329.6
2	251.33 ± 0.23	246.9
3	198.1 ± 0.22	196.0
4	147.3 ± 0.060	146.8
5	110.4 ± 0.12	110.0
6	82.5 ± 0.073	82.4

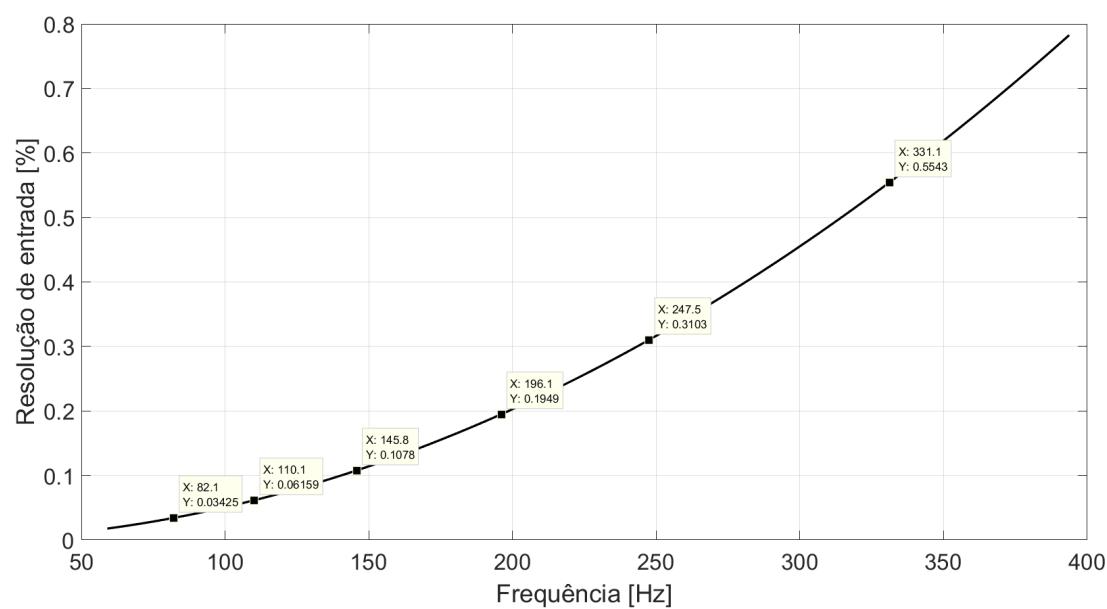
Fonte – Própria, 2018.

Figura 25. Resposta em frequência experimental do filtro anti-aliasing.

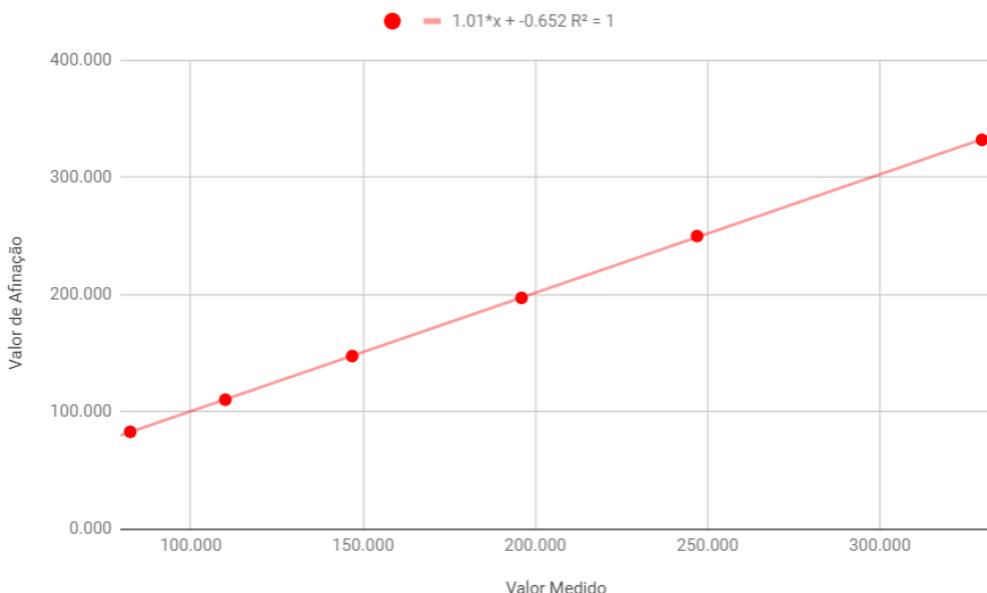
Resposta em frequência do filtro



Fonte – Própria, 2018

Figura 26. Resolução de entrada.

Fonte – Própria, 2018

Figura 27. Relação entre o valor medido e o valor de afinação.

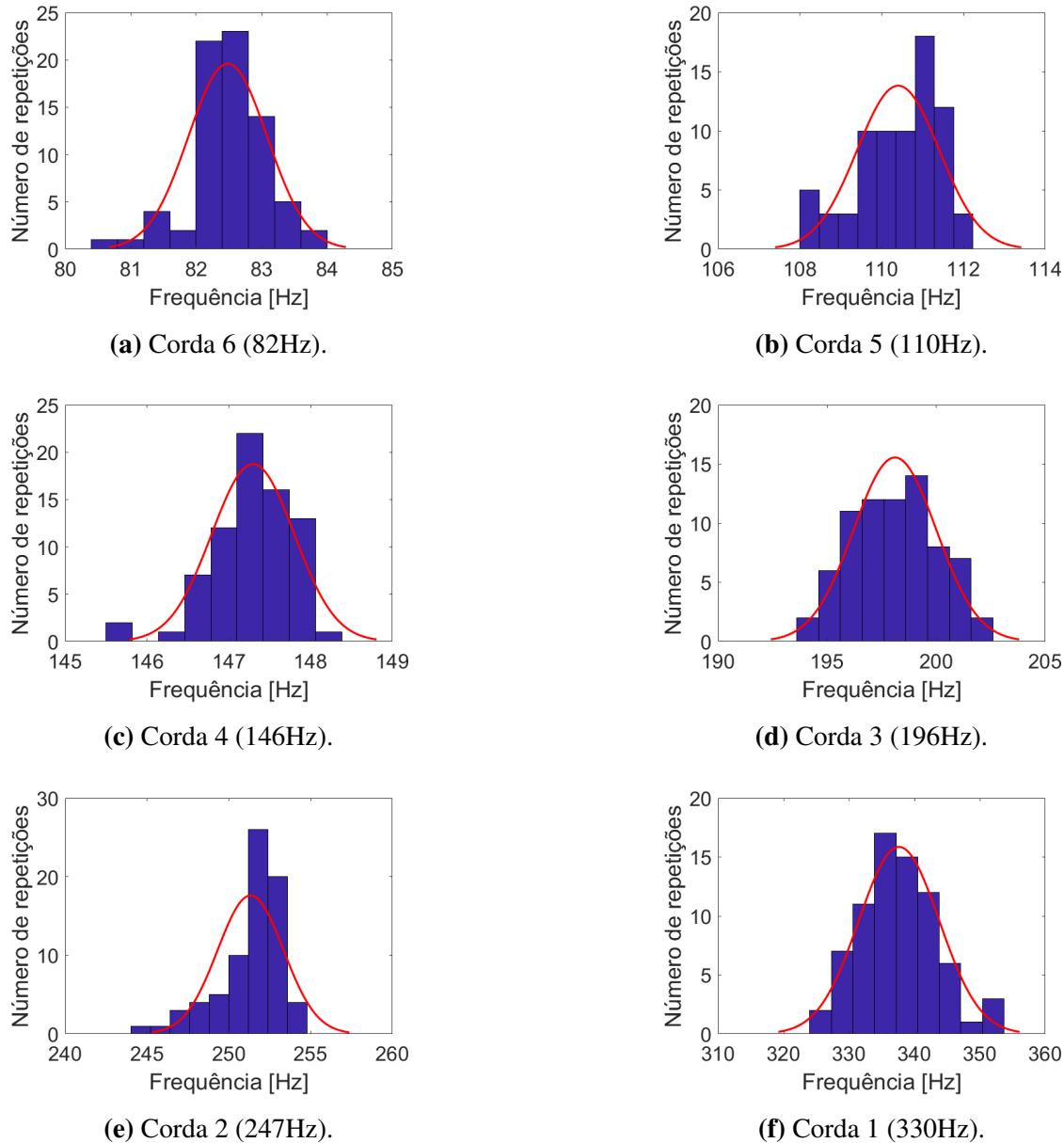
Fonte – Própria, 2018

Por fim, a tabela 8 mostra os valores obtidos ao sistema ser afinado pelo sistema proposto em comparação com o afinador do violão utilizado para 4 repetições, com a corda inicialmente desafinada de forma aleatória. É possível verificar que não há indicação do erro para 2 repetições com a corda 1, pois nessas repetições o sistema não convergiu para a afinação e, portanto, atribuiu-se que o sistema não foi capaz de afinar tal corda para aquela repetição. Além disso, percebe-se que o sistema converge melhor para as cordas 4 e 5, cujas quais resultaram em erros absolutos menores que 10 cents para todas as repetições. Pode-se então afirmar que a resposta do sistema foi satisfatória para essas duas cordas, se utilizarmos uma tolerância de ± 10 cents de desvio. Com esse mesmo critério, a corda 3 apresentou um resultado um pouco abaixo do proposto, enquanto que a 2 e a 6 estiveram mais abaixo ainda. O maior problema, no entanto, acontece na corda 1, na qual a frequência da corda não converge para um valor fixo. Isso se deve à alta dispersão das medidas de frequência para essa corda.

Tabela 8. Erro na afinação com o sistema proposto.

Corda	Erro absoluto [cents]				
	-23.33	-20	-	-	-
1	-23.33	-20	-	-	-
2	-30	-30	-6.67	-6.67	
3	-3.33	0	-13.33	-13.33	
4	0	-6.67	3.33	-6.67	
5	0	0	-10	-3.33	
6	-30	-10	-16.67	-10	

Fonte – Própria, 2018.

Figura 28. Histogramas das medições com as cordas afinadas.

Fonte – Própria, 2018.

5. Conclusões

A partir de uma proposta inicial genérica (auto-afinação de um violão baseada no efeito piezoelettrico), foi possível, através de técnicas de instrumentação e eletrônica, definir as bases do projeto. Essas bases são os blocos principais do sistema, tais como sensor, condicionador, conversor AD, algoritmo de detecção de frequência, controlador, e atuador.

Seguindo essa ideia geral, foi realizado o projeto de cada bloco, o que se traduziu na escolha do material do sensor (cerâmica PZT), na topologia do circuito condicionador, na utilização do kit de desenvolvimento Teensy, na escolha da lógica fuzzy e aprofundamento teórico sobre o mesmo, e finalmente na utilização de um motor de passo na atuação.

Os detalhes de cada um desses blocos foram então definidos e o circuito condicionador projetado, simulado e construído, o algoritmo de detecção testado e finalmente o sistema inteiro foi ajustado e colocado em prática.

Na caracterização do sistema, verificou-se que a qualidade da medição é boa para as cordas mais graves, sendo a incerteza-padrão baixa e a média das medições próxima do valor real. No entanto, para as cordas mais agudas, a incerteza cresce e a média se afasta do valor real. Isso ocorre devido à grande dispersão das medições nessa faixa.

A avaliação da performance do sistema como um todo mostrou que a afinação das cordas 4 e 5 é realizada dentro da tolerância, e para as outras cordas a afinação foge da faixa previamente estabelecida. O principal problema foi verificado na corda mais aguda, na qual o sistema nem sempre converge para uma frequência fixa.

Referências

- [1] NIGRO, MÁRCIO. O “mistério” das afinações alteradas. Disponível em: <<http://www.marcionigro.com.br/buraconigro/afinacoes/>>. Acesso em: 04 de outubro de 2018.
- [2] BECCHI, Natan Gabriel. *Sistema Automatizado para Afinação de Violão Elétrico*. Lajeado, 2017.
- [3] SERAFINA, Guilherme Nunes. *Afinador automático digital para guitarras baseado no método do espectro do produto harmônico*. Porto Alegre, 2015.
- [4] RADAELLI, Rodrigo Alexandre. *Guitarra Auto-Afinável*. Porto Alegre, 2010.
- [5] MACERA, Telos Galante. *Afinador Digital para Violão e Guitarra Elétrica Implementado em FPGA*. São Carlos, 2013.
- [6] ZÖLZER, Udo (Ed.). DAFX: digital audio effects. John Wiley & Sons, 2011.
- [7] OPPENHEIM, Alan V. Discrete-time signal processing. Pearson Education India, 1999.
- [8] HAYES, Monson H. Schaum’s outline of digital signal processing. McGraw-Hill, Inc., 1998.
- [9] SCHNEIDER, Paulo. Notas de aula: Incertezas de Medição e Ajuste de dados - UFRGS, 2007. Disponível em: <<http://www.ufrgs.br/medterm/areas/area-i/Incertezaedicao.pdf>>. Acesso em: 20 de novembro de 2018.
- [10] BALBINOT, Alexandre. Notas de aula: Introdução a propagação de incertezas - UFRGS, 2018.
- [11] GAJIC, Zoran. Linear Dynamic Systems and Signals. Prentice Hall, 2003. KARKI, Jim. Active low-pass filter design. Texas Instruments Application Report, 2000.
- [12] PJRC. Teensy USB Development Board. Disponível em: <<https://www.pjrc.com/teensy/>>. Acesso em: 20 de novembro de 2018.
- [13] BALBINOT, Alexandre, BRUSAMARELLO, Valner João. Instrumentação e Fundamentos de Medidas, Vol 2 - 2007.

- [14] BALBINOT, Alexandre. Notas de aula: Principais Transdutores e Condicionadores Geradores: Piezoelétrico e Piroelétrico - UFRGS, 2018.
- [15] Texas Instruments Incorporated. Signal Conditioning Piezoelectric Sensors. 2000. Disponível em: <https://www.ti.com/lit/an/sloa033a/sloa033a.pdf?fbclid=IwAR0QzwV3edNgyy740ecHA1w3L-bQ1xwXA_K8kfZtrnqIK8gEH0x8x1EEfA>. Acesso em: 20 de novembro de 2018.
- [16] PASSINO, Kevin M., YURKOVICH, Stephen. Fuzzy Control. Addison-Wesley, 1997.
- [17] BALBINOT, Alexandre. Notas de aula: Apresentação da Disciplina e conceitos básicos - UFRGS, 2018.
- [18] SIMOES, Marcelo Godoy. *Introduction to Fuzzy Control**. 2003.
- [19] INMETRO, E. SENAI. Vocabulário internacional de termos fundamentais e gerais de metrologia. 1a. ed, Brasília: SENAI/DN, 2012.
- [20] MOTIONKING. HB Stepper Motor Catalog. Disponível em: <http://www.svaltera.ua/catalogs/knowledge-base/brands/motionking/HB_Stepper_Motor_E.pdf>. Acesso em: 25 de novembro de 2018.
- [21] TEXAS INSTRUMENTS. DRV8825 Stepper Motor Controller IC, 2010. Disponível em <<http://www.ti.com/lit/ds/symlink/dr8825.pdf>>. Acesso em: 25 de novembro de 2018.
- [22] POLOLU.DRV8825 Stepper Motor Driver Carrier, High Current. Disponível em <DRV8825 Stepper Motor Driver Carrier, High Current>. Acesso em: 25 de novembro de 2018.
- [23] RAHNAMAI K., COX B. and GORMAN K., "Fuzzy Automatic Guitar Tuner,"NAFIPS 2007 - 2007 Annual Meeting of the North American Fuzzy Information Processing Society, San Diego, CA, 2007, pp. 195-199.
- [24] OXENHAM, Andrew J. Pitch Perception. Journal of Neuroscience, 2012.
- [25] HELMHOLTZ, Hermann L. F. Sensations of Tone as a Physiological Basis for the Theory of Music. 4ed, 1877. Tradução por Alexander J. Ellis. 3ed, London e New York, 1895.
- [26] O'DONNEL, Michael. Perceptual Foundations of Sound. 2004. Disponível em: <http://people.cs.uchicago.edu/~odonnell/Scholar/Work_in_progress/Digital_Sound_Modelling/lectnotes/node4.html>. Acesso em: 13 de dezembro de 2018.
- [27] SCHMIDT-JONES, Catherine. Sounds, Physics and Music. Connexions, Rice University, Houston, Texas, 2013. Disponível em: <<http://cnx.org/content/col10261/1.1/>>. Acesso em: 13 de dezembro de 2018.
- [28] Texas Instruments. TL431/TL432 Precision Programmable Reference. 2018. Disponível em: <<http://www.ti.com/lit/ds/symlink/tl431.pdf>>. Acesso em: 13 de dezembro de 2018.

Apêndice

A. CÓDIGOS

```

#include <ADC.h>
#include <stdlib.h>
#include <Fuzzy.h>
#include <FuzzyRule.h>
#include <FuzzyComposition.h>
#include <FuzzyRuleConsequent.h>
#include <FuzzyOutput.h>
#include <FuzzyInput.h>
#include <FuzzyIO.h>
#include <FuzzySet.h>
#include <FuzzyRuleAntecedent.h>
#include <AccelStepper.h>

Fuzzy *fuzzy = new Fuzzy();           // objeto do fuzzy
ADC *adc = new ADC();                // objeto do adc
IntervalTimer amostrador;           // objeto do timer
AccelStepper stepper(1, 3, 2);      // 3 - pinStep; 4 - pinDirection.

const float Fs = 50000;
const float Ts = 1000000.0 / Fs; //uS

const int TAM = 10000;
const int PEAKS = 1;

const float afinacao[6] = {329.627556913, 246.941650628, 195.997717991, 146.832383959, 110, 82.4068892282};

//*****FREQUENCIAS*****
// START: lag mimo para detectar frequencia
// LAG: lag maximo para detectar frequencia
void getFreqIdx(float* x, int START, int LAG, int* idx) {
    float r[LAG - START] = {0}, der[LAG - START - 1] = {0}, pk[PEAKS] = {0};

    for (int i = START; i < LAG; i++) {
        //autocorrela o
        r[i - START] = 0.0;
        for (int j = 0; j < TAM; j++) {
            if (j > i) r[i - START] += x[j] * x[j - i];
        }
        //Serial.print("r["); Serial.print(i); Serial.print("] = "); Serial.println(r[i]);
        //derivada
        if (i - START > 0) der[i - START - 1] = r[i - START] - r[i - START - 1];
        // indice dos picos
        if (i - START > 2) {
            //Serial.print("der["); Serial.print(i-1); Serial.print("] = "); Serial.println(der[i-1]);
            if (r[i - START - 2] > pk[0] && (der[i - START - 3] > 0 && der[i - START - 1] < 0)) { //verifica se h pico
                for (int k = PEAKS - 1; k >= 0; k--) { //varre do maior ao menor pico
                    if (r[i - START - 2] > pk[k]) { //encontrou posicao do pico
                        for (int l = 1; l <= k; l++) { //realoca a posicao dos picos
                            pk[l - 1] = pk[l];
                            idx[l - 1] = idx[l];
                        }
                        pk[k] = r[i - START - 2];
                        idx[k] = i - 2;
                        break;
                    }
                }
            }
        }
    }
}

// ****CALCULOS ESTATISTICOS*****
// m dia
float mean(float* x, int SIZE) {
    float total = 0.0;
    for (int i = 0; i < SIZE; i++) total += x[i];
    return total / SIZE;
}

// desvio padrao
float deviation(float M, float* x, int SIZE) {
    float total = 0.0;
    for (int i = 0; i < SIZE; i++) total += (x[i] - M) * (x[i] - M);
    return sqrt(total / (SIZE - 1));
}

```

```

// crit río de Chauvenets
float chauvenets(float tau, float* x, int SIZE) {
    float M = mean(x, SIZE);
    float S = deviation(M, x, SIZE);

    float t = 0.0; int n = 0;
    if (S <= 0.0) return x[0]; //std=0 : todas as amostras s o iguais
    for (int i = 0; i < SIZE; i++) {
        if ((abs(x[i] - M) / S) < tau) {
            t += x[i];
            n++;
        }
    }
    return t / n;
}
//*********************************************************************


// *****AMOSTRAGEM*****
volatile uint16_t sample;
float amostras[TAM] = {0.0};
volatile float total = 0.0;
volatile int it = 0;
volatile int amostrou = 0, janelaValida = 0;

float thresh = 3000;
void getSample() {
    sample = (uint16_t)adc->analogRead(A18, ADC_1);
    if (sample > thresh) janelaValida = 1;
    //sample=(uint16_t)(sine(2*pi*it*147/Fs)+1)*2048;
    if (it < TAM && janelaValida) {
        amostras[it] = (float)(sample - 2048) / 4096; /*hann(it);
        it++;
    }
    if (it >= TAM - 1) {
        amostrou = 1;
        janelaValida = 0; total = 0; it = 0;
    }
}
// *****


// *****CONFIGURA O ADC,DAC e TIMER*****
void configADC() {
    // ADC_1
    pinMode(A18, INPUT);
    adc->setReference(ADC_REFERENCE::REF_EXT, ADC_1); //REF_EXT
    adc->setSamplingSpeed(ADC_SAMPLING_SPEED::VERY_HIGH_SPEED, ADC_1);
    adc->setConversionSpeed(ADC_CONVERSION_SPEED::VERY_HIGH_SPEED, ADC_1);
    adc->setResolution(12, ADC_1);
    adc->setAveraging(8, ADC_1);
}

// inicia o timer para fazer as amostragens
void initTimer() {
    amostrador.begin(getSample, Ts);
    amostrador.priority(0);
}

// *****


//*********************************************************************CONFIGURACAO DO FUZZY*****/
void configFuzzy(void)
{
    FuzzyInput *erro = new FuzzyInput(1); // Entrada do sistema

    //Fuzzy Sets (funcões de pertinencia) da entrada.
    FuzzySet *negativo = new FuzzySet(-300, -300, -100, -10); //fc saturacao esquerda.
    erro->addFuzzySet(negativo); // Adiciona a entrada.
    FuzzySet *zeroIn = new FuzzySet(-100, -10, 10, 100); //fc triangular centrada em zero.
    erro->addFuzzySet(zeroIn); // Adiciona a entrada.
    FuzzySet *positivo = new FuzzySet(10, 100, 300, 300); // fc saturacao direita.
    erro->addFuzzySet(positivo); // Adiciona a entrada.

    fuzzy->addFuzzyInput(erro); // Adiciona a entrada ao objeto.

    FuzzyOutput *rpm = new FuzzyOutput(1); // saida do sistema.

    //Fuzzy Sets da saida
    FuzzySet *rapAntHorario = new FuzzySet(-100, -50, -50, 0); // fc triangular centrada em -1.
    rpm->addFuzzySet(rapAntHorario); // adiciona a saida
    FuzzySet *zeroOut = new FuzzySet(-50, 0, 0, 50); // fc triangular centrada em zero
    rpm->addFuzzySet(zeroOut); // adiciona a saida
    FuzzySet *rapHorario = new FuzzySet(0, 50, 50, 100); // fc triangular centrada em 1
    rpm->addFuzzySet(rapHorario); // adiciona a saida

    fuzzy->addFuzzyOutput(rpm); // Adiciona a saida ao objeto

    //Criando as regras fuzzy.
}

```

```

// 1. Se Erro = negativo entao rpm = rapanthorario
FuzzyRuleAntecedent *ifErroNeg = new FuzzyRuleAntecedent(); //Adicionando a premissa
ifErroNeg->joinSingle(negativo); // Adicionando o set correspondente
FuzzyRuleConsequent *thenRPMRapHorario = new FuzzyRuleConsequent(); // Adicionando a consequencia
thenRPMRapHorario->addOutput(rapHorario); // Adicionando o set correspondente

// Instanciar objeto FuzzyRule
FuzzyRule *fuzzyRule01 = new FuzzyRule(1, ifErroNeg, thenRPMRapHorario);

fuzzy->addFuzzyRule(fuzzyRule01); // Adicionando o FuzzyRule ao objeto

// 2. Se Erro = Positivo ENTAO rpm=rapHorario
FuzzyRuleAntecedent *ifErroPos = new FuzzyRuleAntecedent(); // Adicionando a premissa
ifErroPos->joinSingle(positivo); // Adicionando o set correspondente
FuzzyRuleConsequent *thenRPMRapAntHorario = new FuzzyRuleConsequent(); // Adicionando a Consequencia
thenRPMRapAntHorario->addOutput(rapAntHorario); // Adicionando o set correspondente

//Instanciar o objeto FuzzyRule da segunda regra
FuzzyRule *fuzzyRule02 = new FuzzyRule(2, ifErroPos, thenRPMRapAntHorario);

fuzzy->addFuzzyRule(fuzzyRule02); // Adicionando ao objeto Fuzzy.

// 3. Se Erro = zeroIn entao RPM = zeroOut
FuzzyRuleAntecedent *ifErroZeroIn = new FuzzyRuleAntecedent(); // Adicionando A premissa.
ifErroZeroIn->joinSingle(zeroIn); // Adicionando o Set correspondente
FuzzyRuleConsequent *thenRPMZeroOut = new FuzzyRuleConsequent(); // Adicionando a consequencia
thenRPMZeroOut->addOutput(zeroOut); // Adicionando o set conrrrespondente

// Instanciar o objeto FuzzyRule da terceira regra
FuzzyRule *fuzzyRule03 = new FuzzyRule(3, ifErroZeroIn, thenRPMZeroOut);

fuzzy->addFuzzyRule(fuzzyRule03); // adicionando a ultima regra ao objeto fuzzy
}

/*
void setup() {
  Serial.begin(2000000);
  stepper.setMaxSpeed(100);
  stepper.setAcceleration(100.0);
  //sinaliza inicio das amostragens (pisca o led 3 vezes) e mat m aceso
  pinMode(LED_BUILTIN, OUTPUT);
  for (int j = 0; j < 8; j++) {
    if (j % 2 == 0) digitalWrite(LED_BUILTIN, LOW);
    else digitalWrite(LED_BUILTIN, HIGH);
    delay(200);
  }

  configADC();
  Serial.println("ADC configurado ... ");
  initTimer();
  Serial.println("TIMER configurado ... ");
  configFuzzy();
  Serial.println("FUZZY configurado ... ");
}

int idx[PEAKS] = {0};
float fund[5] = {0.0}; int f = 0;
int corda = 6; float minFreq = 48, maxFreq = 150;
long iTime;
long fTime;
float fAmostra, cent;
int girarMotor=0;
int afinado = 0;
float freq;
void loop() {
  if (Serial.available()) {
    noInterrupts();
    corda = Serial.parseInt() - 1;
    if (corda == 0 || corda == 1 || corda == 2 || corda == 3 || corda == 4 || corda == 5) {
      if (corda==5 || corda == 4 || corda == 3) thresh = 3000;
      else thresh = 2800;
      Serial.print("\nCorda:"); Serial.println(corda + 1);
      freq = afinacao[corda];
      minFreq = freq/1.189207115; maxFreq = freq*1.189207115;
      // reinicia os calculos
      afinado=0; amostrou = 0; f = 0; it = 0;
    }
    interrupts();
  }
  if (amostrou) {
    //Serial.print("amostrado..."); Serial.print(f+1); Serial.print("/5");
    noInterrupts();

    iTime = micros();
    getFreqIdx(amostras, (int)Fs / maxFreq, (int)Fs / minFreq, idx);
  }
}

```

```

fTime = micros();

// Serial.print("TEMPO: "); Serial.print((fTime - iTime) / 1000.0); Serial.println("ms");

//fAmostra = (float)Fs / idx[0];

amostrou = 0;

fund[f++] = (float)Fs/idx[0];
//Serial.print("f="); Serial.println((float)Fs/idx[0]);
if (f>=5) {
    fAmostra = chauvenets(1.65, fund, 5); Serial.println(fAmostra);
    cent = 1200*log(fAmostra/freq)/log(2);
    girarMotor=1;
    // curva de calibracao
    //fAmostra = 1.01*fAmostra-0.652;
    //Serial.print("FREQ "); Serial.print(fAmostra); Serial.println("Hz");
    //Serial.print("CENTS "); Serial.println(cent);
    f = 0; it = 0;
}
interrupts();
}

if (girarMotor && afinado<3) {
    girarMotor=0;
    noInterrupts();
    float err = cent;
    //Serial.print("Erro:");
    //Serial.println(err);
    fuzzy->setInput(1, err);           // seta a entrada que tem ID 1 (erro)
    fuzzy->fuzzify();                // executa a fuzzificacao
    float sps = fuzzy->defuzzify(1); // retorna a saida em stepsPerSeconds/100.

    // verifica h 3 valores seguidos de sps=0 para corda afinada
    if ((int)sps==0) afinado++;
    else afinado=0;
    if (afinado==3) Serial.println("Corda afinada!");

    //Serial.print("SPS");
    //Serial.println(sps);
    stepper.move(round(sps));          // passa a variavel pra entrada do motor. Type casting pra int.
    //Serial.print("INTSPS");
    //Serial.println((int)sps);
    stepper.runToPosition();
    interrupts();
    Serial.println("-----\n");
}
}

```