# Aplicação de Tecnologias de IoT Para a Análise Remota de Motores Trifásicos Baseado na Aquisição de Corrente

Gabriel Alves de Amorim

CENTRO DE ENERGIAS ALTERNATIVAS E RENOVÁVEIS
UNIVERSIDADE FEDERAL DA PARAÍBA

João Pessoa, 2020

Gabriel Alves de Amorim

# Aplicação de Tecnologias de IoT Para a Análise Remota de Motores Trifásicos Baseado na Aquisição de Corrente

Monografia apresentada ao curso Engenharia Elétrica
do Centro de Energias Alternativas e Renováveis, da Universidade Federal da Paraíba,
como requisito para a obtenção do grau de Bacharel em Engenharia Elétrica

Orientador: Juan Moises Mauricio Villanueva

Abril de 2020

**Catalogação na publicação**
**Seção de Catalogação e Classificação**

A519a Amorim, Gabriel Alves de.
Aplicação de Tecnologias de IoT Para a Análise Remota de Motores Trifásicos Baseado na Aquisição de Corrente / Gabriel Alves de Amorim. - João Pessoa, 2020.
77 f. : il.

Orientação: Juan Villanueva.
TCC (Especialização) - UFPB/Cear.

1. Vetor de Park. 2. ESP32. 3. MQTT. 4. Python. 5. Manutenção Preditiva. I. Villanueva, Juan. II. Título.

UFPB/BC

CENTRO DE ENERGIAS ALTERNATIVAS E RENOVÁVEIS
UNIVERSIDADE FEDERAL DA PARAÍBA

Trabalho de Conclusão de Curso de Engenharia Elétrica intitulado ***Aplicação de Tecnologias de IoT Para a Análise Remota de Motores Trifásicos Baseado na Aquisição de Corrente*** de autoria de Gabriel Alves de Amorim, aprovada pela banca examinadora constituída pelos seguintes professores:

---

Prof. Dr. Juan Moises Mauricio Villanueva
CEAR/UFPB

---

Prof. Dr. Nady Rocha
CEAR/UFPB

---

Prof. Dr. Carlos Alberto de Souza Filho
CEAR/UFPB

---

Coordenador(a) do Departamento Engenharia Elétrica
Alexsandro José Virgínio dos Santos
CEAR/UFPB

João Pessoa, 13 de abril de 2020

Centro de Energias Alternativas e Renováveis, Universidade Federal da Paraíba
Campus I - Lot. Cidade Universitaria, PB, 58051-900
Fone: +55 (83) 3216-7136

*"Don't Panic." - Douglas Adams.*

# AGRADECIMENTOS

Agradeço neste trabalho de conclusão de curso minha mulher e companheira Caroline Costa que sempre esteve ao meu lado, me apoiando, ajudando e não me deixando desistir no meio dessa difícil jornada.

Agradeço também minha mãe por ter me criado sempre com responsabilidade e carinho, que permitiu eu chegar aonde eu estou hoje.

Agradeço ao meu Pai que me inspirou a seguir o caminho da engenharia, e que mesmo longe já me ensinou muito.

Agrageço a Aldonso Martins, meu supervisor do estágio da Jeep que conseguiu me ensinar muito mais do que eu esperava aprender em tão pouco tempo, assim como me inspirou ainda mais a seguir o caminho o qual estou seguindo.

Agradeço ainda a todos meus amigos e colegas que me fizeram companhia, nos bons e maus momentos, e fizeram parte do caminho que me levou a este momento.

Por fim agradeço meu professor e orientador Juan Maurício Vilanueva, que foi compreensivo e se disponibilizou a me ajudar o máximo para que eu pudesse concluir este trabalho.

# RESUMO

Com o advento da tecnologia da indústria 4.0, muitos sistemas de medição têm incorporado tecnologias digitais, tais como comunicação M2M, uso de ferramentas de IoT, processamento de sinais, inteligência artificial, entre outros. Neste cenário, este trabalho consiste na criação de um sistema de aquisição de dados com capacidade de comunicação WiFi para monitoramento das correntes trifásicas de um motor. A partir deste monitoramento, tem-se como objetivo a reconstrução das ondas e análise das correntes através da abordagem dos vetores de Park. Com a finalidade de ilustrar os resultados deste trabalho, foi utilizado um motor de 0,5 CV, aquisição de correntes usando sensores de efeito Hall e o uso do protocolo de comunicação Mensage Query Telemetry Transport (MQTT). A partir do processamento dos sinais de correntes foi possível ter uma ferramenta de tomada de decisões associadas a operação e diagnóstico de falhas do motor.

**Key-words:Vetor de Park, Esp32, Manutenção preditiva, MQTT, Python**

# ABSTRACT

With the advent of industry 4.0 technology, many measurement systems have been incorporating digital technologies, such as M2M communication, use of IoT tools, signal processing, artificial intelligence, among others. In this scenario, this work consists of the creation of a data acquisition system with WiFi communication capacity for monitoring the three-phase currents of an engine. Based on this monitoring, the objective is predictive analysis using the Park vectors approach. In order to illustrate the results of this work, a 0.5 CV motor was used, current acquisition using Hall effect sensors and the use of the MQTT communication protocol. From the processing of the current signals it was possible to have a decision-making tool associated with the operation and diagnosis of motor failures.



**Key-words:Park's Vector, Esp32, Predictive Maintence, MQTT, Python**

# LISTA DE FIGURAS

# LISTA DE ABREVIATURAS

A/D - Analógico/Digital

ADC - Analogic Digital Converter

Amp Op - Amplificador Operacional

CA - Corrente Alternada

CC - Corrente Continua

CI - Circuito Integrado

CPS – Cyber Phisical Systems

f.e.m - Força Eletro Motriz

IoT – Internet of Things

M2M - Machine to Machine

MCSA - Motor Current Signature Analisys

MQTT – Message Query Telemetry Transfer

## Sumário

# 1 INTRODUÇÃO

A palavra "revolução"significa uma mudança brusca e radical. A história do mundo é marcada por revoluções, onde a maioria delas foram causadas com a chegada de novas ideias, técnicas ou tecnologias que causaram mudanças enormes na economia, rotina ou maneira de se enxergar o mundo [1].

Como exemplo de revolução, temos a primeira grande mudança na maneira de viver do ser humano, que ocorreu por volta de 10.000 anos atrás, com o surgimento da agricultura e domesticação dos animais, isso marcou a transição do estilo de vida de caça para um estilo de vida assentado que permitiu o desenvolvimento da sociedade [1].

Após isso, o mundo passou por diversas revoluções que se somaram para formar a sociedade em que vivemos hoje, dentre elas, temos as revoluções industriais, que foram essenciais para definir o modelo socioeconômico atual. Até o momento existiram três revoluções industriais, e estamos atualmente presenciando a quarta.

A primeira Revolução Industrial foi transição dos métodos de produção artesanais para processos de produção mecanizados movidos a água ou vapor no período entre 1760 e 1840 [3]. Neste período, muitas pessoas tiveram que deixar de executar muitos trabalhos de maneira manual para aprender a operar máquinas que realizavam essas atividades com muito mais eficiência. Essas mudanças revolucionaram não só a economia, com o aumento da produtividade, mas também mudou a vida das pessoas em relação ao trabalho e ao quotidiano com o surgimento de novas invenções como a locomotiva e o telégrafo [2].

A segunda revolução industrial surgiu com o descobrimento da energia elétrica e do uso de petróleo como combustível. O mundo se tornou mais conectado, agora com carros e caminhões circulando pelas ruas, foi inventado o avião e até mesmo o telefone, mudando completamente os meios de comunicação entre as pessoas.

Já a terceira foi marcada pela mudança das tecnologias analógicas pela digital. Em meio a guerra fria os avanços tecnológicos foram enormes, possibilitando até a ida do homem à lua. Novas formas de gerar energia surgiram, como a nuclear, eólica e solar. Chegando até aos celulares e à internet, que são itens essenciais do cotidiano atual.

Hoje em dia estamos vivendo o início de uma nova revolução, a quarta revolução industrial, que deu origem a um novo conceito chamado indústria 4.0. Tal termo foi criado por um projeto alemão que tem o objetivo de implementar sistemas de produção mais inteligentes e eficientes, utilizando-se de novas tecnologias que permite com que as máquinas consigam, a partir de seus próprios dados, diagnosticar possíveis falhas, otimizar seus parâmetros de funcionamento ou até utilizar-se de inteligência artificial para executar tarefas mais complexas com alta eficiência e qualidade [4]. Através da implementação generalizada de sensores no ambiente de produção, os mundos físico e virtual fundem-

se, dando origem aos *Cyber Physical Systems* (*CPS*). Esses sistemas conectados através da *Internet of Things* (*IoT*) interagem entre si usando protocolos padrão baseados na internet, gerando dados para análise humana ou automática possibilitando a predição de falhas [2].

A partir de tal momento na história da humanidade, surgiu um ramo de estudo da indústria 4.0 voltado para a chamada manutenção preditiva, esse tipo de manutenção permite estipular quando um certo equipamento pode falhar e assim programar uma manutenção antes que a falha ocorra. Como na indústria, os equipamentos geralmente possuem um custo elevado, é essencial que as falhas sempre sejam evitadas.

No geral existem três tipos de manutenção, a corretiva, preventiva e preditiva [8]. A manutenção corretiva é a menos eficiente, porém a mais comum de ocorrer, pois nenhum equipamento dura para sempre, basicamente consiste em agir após uma quebra, consertando o equipamento e deixando-o na condição antes da quebra ocorrer. Os outros dois tipos de manutenção objetivam evitar ao máximo que ocorra a quebra, a preventiva realiza manutenções periódicas tentando sempre garantir as condições de base do equipamento, porém muitos fatores podem alterar o funcionamento de uma máquina, assim os períodos entre manutenções podem acabar ficando ou muito distantes, ocorrendo alguma quebra antes do período da próxima manutenção, ou muito curto gastando recursos desnecessários para aquele momento. Por fim a preditiva que é o foco deste trabalho, que consiste no constante monitoramento da máquina para executar a manutenção somente quando necessário, para isso o uso de sensores, principalmente os conectados à rede de internet, são primordiais para esse tipo de manutenção.

Um exemplo de equipamentos fundamentais para a indústria, que deve sempre estar funcionando, são os motores elétricos. Essas máquinas são responsáveis pelo funcionamento de elevadores, esteiras, robôs, transportadores e muitos outros itens primordiais para o funcionamento da empresa, e uma falha em qualquer equipamento desse pode levar a perdas enormes de produção. Além disso, muitas vezes eles se encontram em locais de difícil acesso para serem monitorados frequentemente por um funcionário. Por virtude disso, é importante a implementação de sensores capazes de diagnosticar o correto funcionamento de tais motores em tempo real, permitindo agir com priorização apenas nos motores que realmente seja necessário alguma intervenção, antes que ocorra alguma parada por quebra durante a produção.

## 1.1 Estado da arte

Existem diversos estudos voltados para o monitoramento de motores com o objetivo de análises para manutenção preditiva, alguns focando no uso de ferramentas manipuladas diretamente por manutentores enquanto outros partindo para uma abordagem mais atual

se relacionando com a indústria 4.0.

Em [5], Thomson apresenta o método de análise de assinatura de corrente ou Motor *Current Signature Analisys* (*MCSA*) em motores para a detecção de falhas. Neste trabalho, foi demonstrado como a variação da resposta em frequência das correntes podem ser associadas a diferentes tipos de falhas do motor. Entre as vantagens de realizar essa análise consiste na avaliação do motor por métodos não invasivos, ainda em pleno funcionamento do motor.

Em [6], Brito realizou um amplo estudo sobre o comportamento dos sinais de frequência do motor relacionados a cada tipo de falha. Com o auxilio de equipamentos de bancadas, o autor foi capaz de realizar diversos testes ao ponto de construir um banco de dados para aplicação em uma rede neural artificial para identificar as possíveis falhas do motor.

Em [7], WEIDLICH fez uma análise comparativa do uso de sensores ultrassônicos e de vibração para a inspeção da lubrificação de motores. Esta abordagem foi realmente eficaz porém apenas para obtenções de alta frequência (até 25kHz), perdendo eficácia para sinais abaixo de 20kHz.

Em [3], Borlido fez um estudo de todo sistema organizacional da indústria atual, explicando os pilares da indústria 4.0, e fazendo um paralelo dos tipos de manutenção existentes e demonstrando qual a vantagem de se utilizar das ferramentas criadas na quarta revolução industrial para a manutenção preditiva.

Em [4], Borges desenvolveu uma aplicação experimental para análise de temperatura de um motor monofásico utilizando IoT. Foi utilizado um microcontrolador com capacidade de comunicação sem fio (ESP32) conectado a um sensor de temperatura para aquisitar os dados do motor e enviar via um servidor web um arquivo de texto com os dados. Foi desenvolvido um programa para o processamento e validação dos dados, e a apesar das dificuldades de comunicação demonstrou-se uma abordagem viável de monitoramento.

De acordo as referências apresentadas no estado da arte, foi evidenciado o crescente interesse no monitoramento de variáveis de motores para diagnóstico de falhas, assim como o uso das ferramentas da indústria 4.0 para análise preditiva dos mesmos, porém ainda é escasso trabalhos focados no desenvolvimento de ferramentas IoT para realizar as coletas de dados para essas análises.

## 1.2 Definição do Problema

O primeiro passo para implementar sistemas inteligentes de manutenção preditiva é a definição de que variáveis devem ser obtidas, em seguida atribuir os sensores e sistemas

para obter e tratar esses dados. Existem diversas empresas focadas em criar soluções para este problema, desenvolvendo sensores inteligentes, interfaces mais avançadas e softwares cada vez mais robustos, porém quanto mais características são adicionadas por itens externos mais caro fica a solução. Assim também, quando é levado em consideração a quantidade de máquinas a serem monitoradas, se torna inviável uma solução pronta para cada uma. Outro empecilho também é muitas vezes as particularidades de cada solução, todas as empresas criam um produto pronto, para ser vendido como um único produto para qualquer empresa que possua uma necessidade parecida, porém cada empresa possui suas individualidades, que podem acabar não sendo completamente atendidas, tendo que se adequar com a solução sem poder realizar nenhuma modificação.

Voltando o foco para os motores, uma variável que geralmente é monitorada para a abordagem de manutenção preditiva é a vibração. Sensores de vibração são capazes de detectar quando há alguma anomalia na frequência de vibração do equipamento, e quando isso ocorre pode indicar alguma problema na máquina. Porém como a indústria está muito sujeita a ruídos, esta vibração pode acabar sendo causada por outros fatores que não estão relacionadas com as características de falhas do motor que esta sendo monitorado.

## 1.3 Premissas e Hipóteses

A premissa de partida deste trabalho é o estudo do uso das correntes do motor para a identificação de falhas, utilizando-se do modelo matemático os vetores de Park. Para tanto, foi estabelecido como hipótese se seria possível criar um protótipo IoT capaz de monitorar as correntes trifásicas de um motor, envia-las por um protocolo de rede, e reconstruir em um outro dispositivo as formas de onda para análise? Poderia ainda com a utilização da transformada de Park demonstrar graficamente se o motor está funcionando devidamente ou se há alguma anomalia? Com a abordagem de análise de frequência seria possível identificar qual a falha com este sistema?

## 1.4 Objetivo geral

O objetivo deste trabalho é desenvolver um sistema de aquisição das correntes trifásicas de um motor, por meio da aplicação de tecnologias de IoT e processamento digital de sinais, com a finalidade de desenvolver uma ferramenta de tomada de decisões para o diagnóstico de falhas de motores.

## 1.5 Objetivos específicos

- Aquisitar os dados de corrente, condicionar o sinal, e organizar para envio.
- Configurar protocolo de comunicação para enviar os dados.

- Desenvolver um algoritmo para receber, processar e analisar os dados.

## 1.6 Estrutura da monografia

Este trabalho de conclusão de curso está dividido em 5 capítulos, o foi o já abordado contendo a introdução ao assunto, junto com a apresentação do problema e solução a ser estudada.

O segundo capítulo consiste no desenvolvimento teórico necessário para a realização deste trabalho, começando com o estado da arte para apresentar os trabalhos no meio acadêmico relacionados ao assunto. Ainda neste capítulo será desenvolvido os conceitos essenciais para o sistema de aquisição de dados proposto. O primeiro é o condicionamento do sinal, em seguida o tratamento para envio e o protocolo MQTT utilizado, por fim a base teórica do tratamento de vetor de Park que será usado para o tratamento dos dados.

O terceiro capítulo demonstrará como as ferramentas apresentadas no capítulo anterior foram utilizadas para o desenvolvimento do projeto, apresentando o protótipo do circuito de condicionamento de sinal, a programação do microcontrolador para aquisição e envio dos dados via protocolo MQTT, e o algoritmo para aquisição dos dados e tratamento baseado na transformada de Park e na assinatura da corrente com o objetivo de facilitar o diagnostico da saúde do motor.

O quarto capítulo será responsável por apresentar os resultados obtidos pelos testes de cada etapa do protótipo, assim como a avaliação de cada etapa para por fim no capítulo cinco efetuar as conclusões obtidas por este trabalho de conclusão de curso.

# 2 CONCEITOS GERAIS E REVISÃO DA LITERATURA

Nesta seção será explorado os conceitos teóricos necessários para entender a base do protótipo a ser desenvolvido, demonstrando a teoria para cada etapa do sistema de aquisição de dados de corrente do motor trifásico. De forma a ilustrar as etapas do projeto, na Figura 1 é apresentado o fluxograma com as etapas a serem abordadas e desenvolvidas:

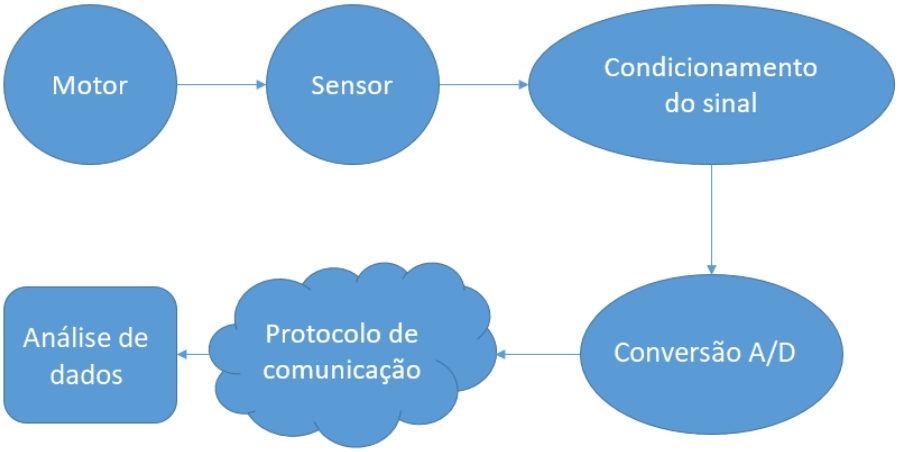


**Figura 1: Fluxograma do projeto.**

## 2.1 Motores de indução trifásicos

A máquina que será monitorada neste trabalho é o motor de indução trifásico, este tipo de motor atualmente constituí uma boa parte dos motores da industria. Este equipamento é composto por diversos elementos como pode ser visto na Figura 2, mas o principal é o rotor e estator [10].

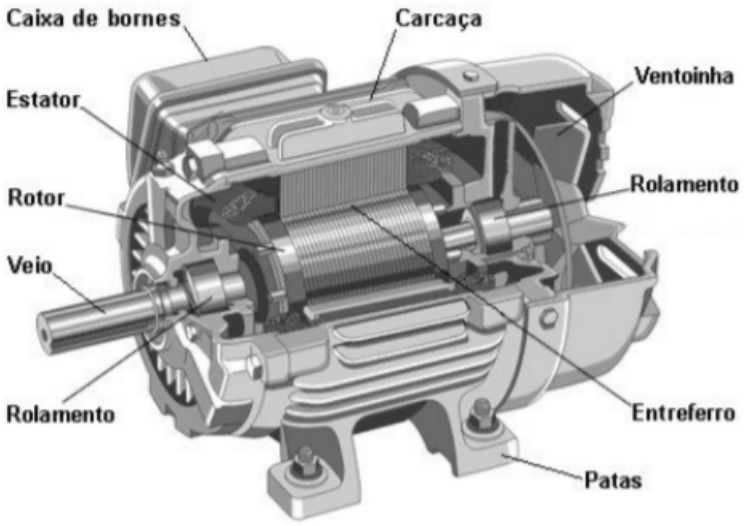


**Figura 2: Componentes de um motor trifásico [10]**

.

O estator é composto por três enrolamentos dispostos de forma que exista um ângulo de $120^{o}$ entre eles, dessa forma quando alimentado pela rede elétrica, alimentando cada rolamento com uma fase da rede, um campo magnético girante é criado, este campo percorre o rotor, gerando uma variação de fluxo em seus condutores conforme a Figura 3, gerando de acordo com a lei de Faraday uma força eletromotriz induzida (f.e.m). Nos condutores do rotor é induzido uma corrente que de acordo com a lei de Lenz, possuem sentido oposto a que as criou, gerando um campo magnético que se opõe ao campo magnético do estator, como o campo do estator é girante, e os polos do estator e rotor são opostos, o rotor gira tentando acompanhar o campo do estator [10].

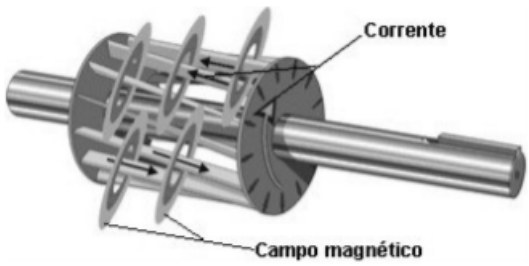


**Figura 3: Influência das correntes trifásicas [10].**

## 2.2 Transformada de Park

Os motores trifásicos, por apresentarem três fases de tensão e corrente, muitas vezes se tornam complicados de se equacionar para estudo, por esse motivo é usual a aplicação das transformadas de Park, a qual diagonaliza as matrizes circulares simétricas

da modelagem dessas máquinas. Em termos físicos, essa transformada muda o sistema trifásico para um bifásico com mesmas características [22]. Os vetores equivalentes podem ser observados na Figura 4. Nesse vetor está sendo ilustrado a transformada de Park, os vetores $f_{as}$, $f_{bs}$ e $f_{cs}$, são as componentes de tensão ou corrente do estator, enquanto $f_{ar}$, $f_{br}$ e $f_{cr}$ são as componentes do rotor. A aplicação da matriz de transformação permite a transformada desses três vetores do estator e do rotor para dois vetores para cada, sendo $S_d$ e $S_q$ as componentes de Park do estator e $R_d$ e $R_q$ as componentes de Park do rotor.

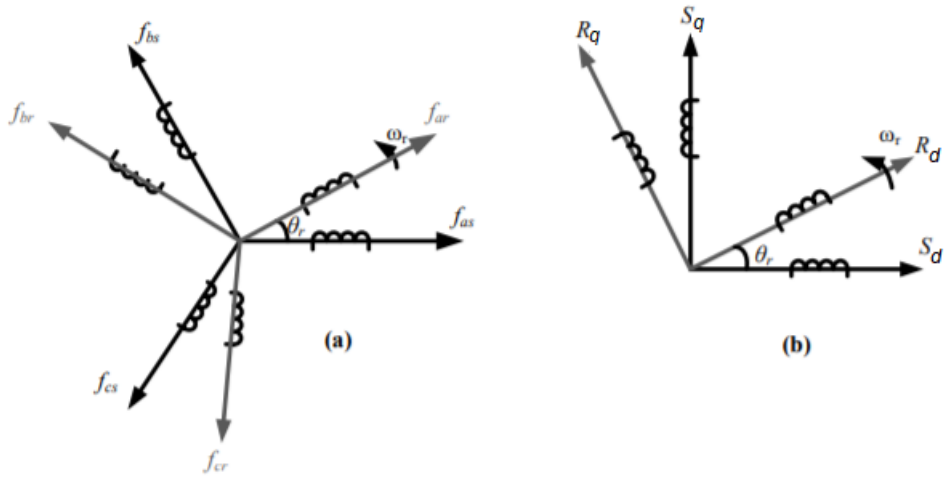


**Figura 4: (a) Sistema trifásico (b) Sistema bifásico equivalente [22]**

A matriz que permite a transformação pode ser observada na equação (1), é escolhido o termo $\sqrt{\frac{2}{3}}$ multiplicando a matriz para a conservação de potência do sistema:

$$\begin{bmatrix} i_d \\ i_q \end{bmatrix} = \sqrt{\frac{2}{3}} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} i_{s1} \\ i_{s2} \\ i_{s3} \end{bmatrix} \tag{1}$$

Em que se comparando com a figura 4:

$i_{s1} = f_{as}$

$i_{s2} = f_{bs}$

$i_{s3} = f_{cs}$

$i_d = S_d$

$i_q = S_q$

Resolvendo para obter $i_d$ e $i_q$ obtemos as equações (2) e (3)

$$i_d = \sqrt{\frac{2}{3}} i_{s1} - \frac{1}{\sqrt{6}} i_{s2} - \frac{1}{\sqrt{6}} i_{s3} \tag{2}$$

$$i_q = \frac{1}{\sqrt{2}} i_{s2} - \frac{1}{\sqrt{2}} i_{s3} \tag{3}$$

A análise dos vetores de Park de corrente permite verificar o mecanismo de excentricidade da transformada, dessa forma ao fazer um gráfico $i_q = i_d$ de um motor em correto funcionamento deve ser gerado uma circunferência centrada na origem dos eixos [24], como pode ser visto na figura 5.

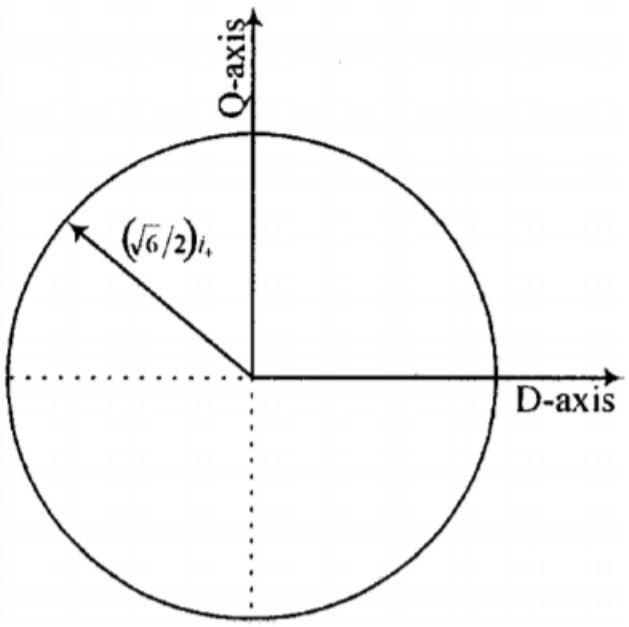


**Figura 5: Representação do vetor de Park em funcionamento normal [24]**

.

## 2.3 Obtenção dos dados

Para entender e compreender qualquer grandeza da natureza é necessário monitorar as variáveis relacionadas a ela, por isso para qualquer sistema de aquisição de dados o primeiro passo é obter esses dados, para isso é necessário a utilização de sensores.

O termo sensor é empregado para designar dispositivos sensíveis a grandezas mensuráveis como temperatura, pressão, tensão, corrente, entre outros [9]. Os sensores permitem a medição direta de variáveis como as citadas previamente, assim como podem ser usados para inferir indiretamente uma variável a partir da medição de uma ou mais grandezas, como definir a potência elétrica consumida de circuito a partir das medições de tensão e corrente.

Os sensores geralmente são construídos para possibilitar a conversão da grandeza a qual ele é sensível para uma nova mensurável, podendo ser analógicos ou digitais. Sensores analógicos possuem uma saída contínua como ilustrado na Figura 6, e tentam replicar na

saída o comportamento da entrada, mesmo que em escala diferente, podendo assumir qualquer valor dentro do intervalo de seu funcionamento. Os sensores digitais por sua vez possuem uma saída binária de zero ou um Como demonstrado na Figura 7, podendo identificar por exemplo a presença de uma peça ou ativações de um *encoder* [9].

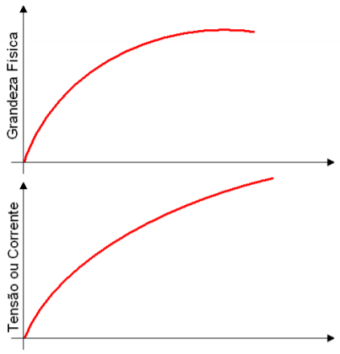


**Figura 6: Saída de um sensor analógico [9].**

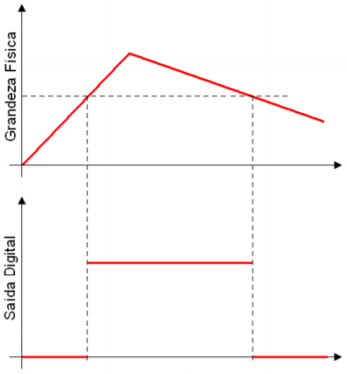


**Figura 7: saída de um sensor digital [9].**

Como o interesse do presente estudo é o monitoramento do motor trifásico, pode-se aprofundar a discussão para sensores de correntes, além disso, é interessante que o monitoramento ocorra enquanto o motor está funcionando, então se mostrou preferível buscar um sensor de corrente não invasivo. O tipo de sensor que atende estas características é um sensor de corrente, não invasivo baseado em efeito Hall. Tal sensor funciona como um transformador de corrente, amplamente utilizado para medir correntes de forma não invasiva como visto na Figura 8. O procedimento de medição consiste em selecionar o condutor que se deseja medir a corrente, inserir ele por dentro de um núcleo magnético que age como o primário do transformador, essa corrente induz uma força eletromagnética no núcleo. Nele um enrolamento de fio condutor é posicionado ao redor do núcleo, agindo como o enrolamento secundário do transformador, dessa forma quando uma corrente é aplicada no condutor primário, uma corrente de menor valor é induzida no secundário,

esta corrente ao passar por um resistor de referência, gera uma tensão na saída do sensor [11].

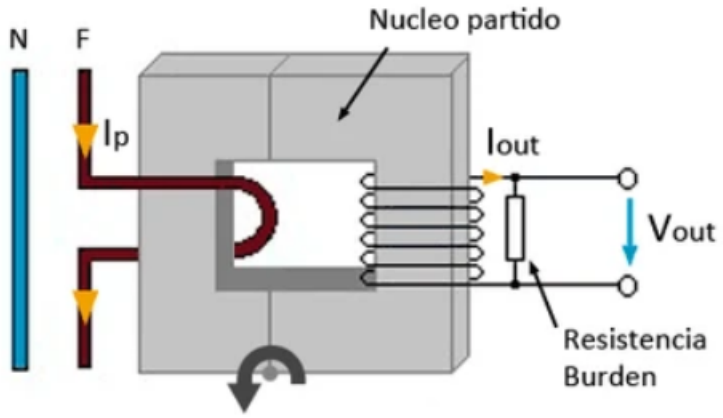


**Figura 8: Sensor baseado em um transformador de corrente.**

## 2.4 Condicionamento do sinal

O passo inicial para criação de um sistema de aquisição da dados é o condicionamento do sinal, as variáveis de interesse reais nem sempre podem ser analisadas ou até mesmo obtidas de maneira direta, na verdade na maioria das vezes o sinal apresenta ruídos indesejados ou amplitudes e níveis fora da faixa utilizada pelo equipamento. Por este motivo a eletrônica conta com diversas configurações de circuitos capazes de modificar o sinal de maneira com que ele mantenha as características que devem ser analisadas, porém agora nas condições ótimas de funcionamento dos dispositivos de aquisição. Como por exemplo, uma onda de baixa amplitude e ruidosa pode ser convertida em uma onda de amplitude dos níveis do equipamento e com baixa relação de sinal ruído (SNR), mantendo sua frequência e forma de onda para análise.

A seguir será apresentado algumas configurações de circuitos condicionadores de sinais explicando seu princípio de funcionamento para entender a função de cada etapa do circuito final do sistema de aquisição de dados.

## 2.5 Condicionamento da alimentação

Para amplificar um sinal de corrente alternada a partir de um Amp Op, é necessário alimenta-lo com uma tensão negativa e uma positiva, porém muitos locais oferecem apenas uma alimentação DC, a qual não possuí componente negativa. Para isso existem configurações de circuitos que permitem a criação de um terra virtual, permitindo o tratamento do dado alternado apenas no plano positivo.

Para isso é utilizada a configuração de *buffer*, onde a tensão aplicada na entrada é a mesma na saída. O ganho é unitário, a diferença está na impedância da entrada, que é altíssima e a da saída muito baixa. A impedância de entrada muito alta torna essa configuração muito interessante pois não carrega elementos primários, a saída é limitada para os valores de saída do Amp Op que não é muito alta, porém como nesta etapa o interesse é apenas usar as tensões como referência, a corrente baixa não será um problema. Podemos verificar a configuração na figura 9:

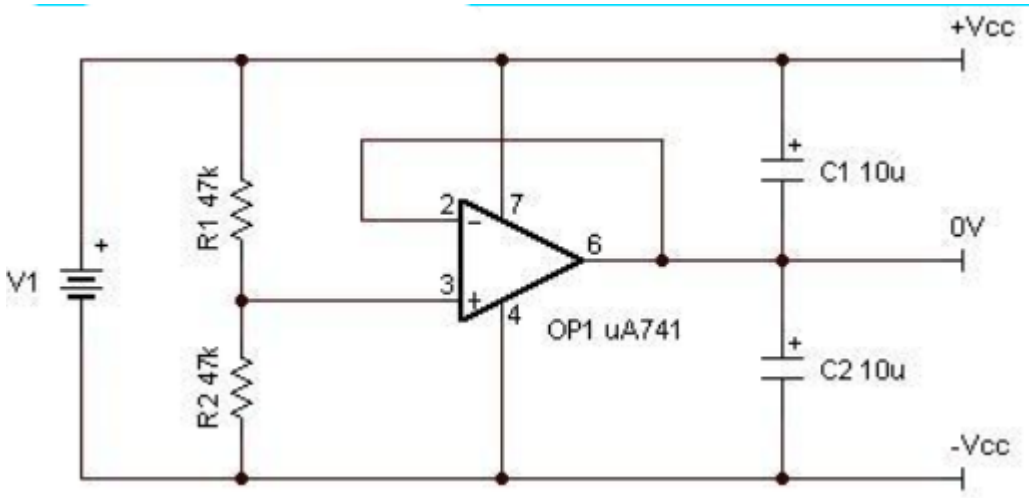


**Figura 9: Fonte simétrica com alimentação simples.**

Analisando a figura podemos ver que o divisor resistivo divide a tensão de entrada no meio, o *buffer* com alta impedância passa esse nível de tensão para a saída gerando um terra virtual com as tensões garantidas pela carga dos capacitores, com essa configuração a saída do Amp Op é tida como a tensão de referência, o terra real do sistema é visto como a tensão negativa com valor de $-\frac{Vcc}{2}$, e a tensão positiva é vista como $\frac{Vcc}{2}$.

## 2.6 Amplificador de instrumentação

O problema mais comum na leitura de um dado físico, é que muitas vezes os dados de interesse são emitidos em escalas muito baixas, sendo assim, se os equipamentos de medição pudessem apenas demonstrar as grandezas físicas como elas realmente são no mundo não seriamos capazes de observar seu comportamento. Por este motivo é necessário amplificar o sinal existente para uma escala que seja possível a observação e análise do sinal, e sabendo a relação usada para a amplificação pode-se saber também qual o real valor da grandeza.

Para a instrumentação eletrônica existe uma configuração de amplificador de instrumentação muito comum para aplicações que desejam uma alta precisão, que é o amplificador de instrumentação ilustrado na Figura 10 que replica o CI LH0036, este tipo

de amplificador age de maneira diferencial, onde as duas saídas do sinal são aplicadas na entrada do circuito e subtraídas, onde uma das entradas pode ser considerada a referência do sinal, outra vantagem desse tipo de amplificador é a possibilidade aplicação do ganho em duas etapas, facilitando o projeto para encontrar o ganho desejado utilizando-se de valores de resistores comerciais, além de poder deixar o ganho circuito em função apenas da variação de um resistor de referencia ou potenciômetro resistivo $R_g$ [12].

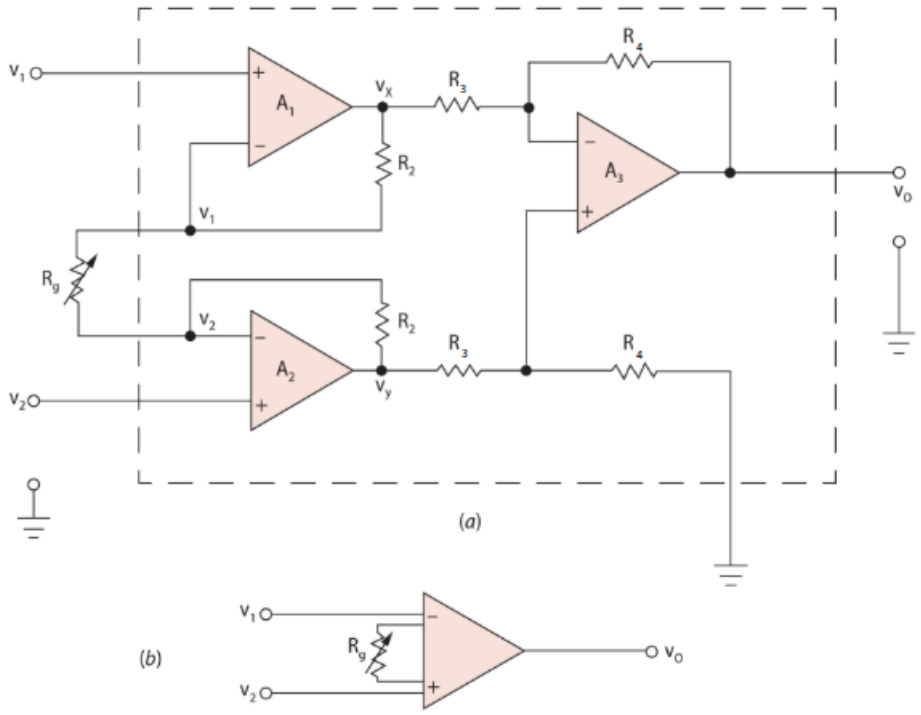


**Figura 10: Amplificador de instrumentação (a) e Amplificador de instrumentação encapsulado (b) [12].**

A equação que descreve o comportamento do circuito da Figura 10.a é dada pela equação 4:

$$V_o = \frac{R_4}{R_3}(1 + 2\frac{R_2}{RG})(V_2 - V1) \tag{4}$$

Nesta equação podemos ver que a entrada considerada para o sistema é a diferença entre as duas tensões de entrada, passando assim por dois estágios de ganho representados nas equações (5) e (6):

$$G1 = \frac{R_4}{R_3} \tag{5}$$

$$G2 = 1 + 2\frac{R_2}{RG} \tag{6}$$

O ganho G1 é definido na hora de projetar o amplificador, tende a sempre deixar

esses valores fixos, deixando a variação de ganho apenas para o G2.

A versão final deste amplificador tende muitas vezes a colocar um potenciômetro no lugar do RG, permitindo controlar o ganho do total do amplificador com a variação de apenas um resistor, um amplificador de instrumentação usado desta maneira pode ser verificado na Figura 10.b.

## 2.7 Filtros

Uma importante parte do condicionamento de sinais é a filtragem. Os filtros elétricos nos permite definir a frequência de interesse e permite eliminar ou atenuar os sinais desnecessários. Os filtros podem ser classificados quanto a sua faixa de passagem como filtros passa-baixas, passa altas, passa-faixas e rejeita faixas, seus comportamentos podem ser visto na Figura 11.[21]

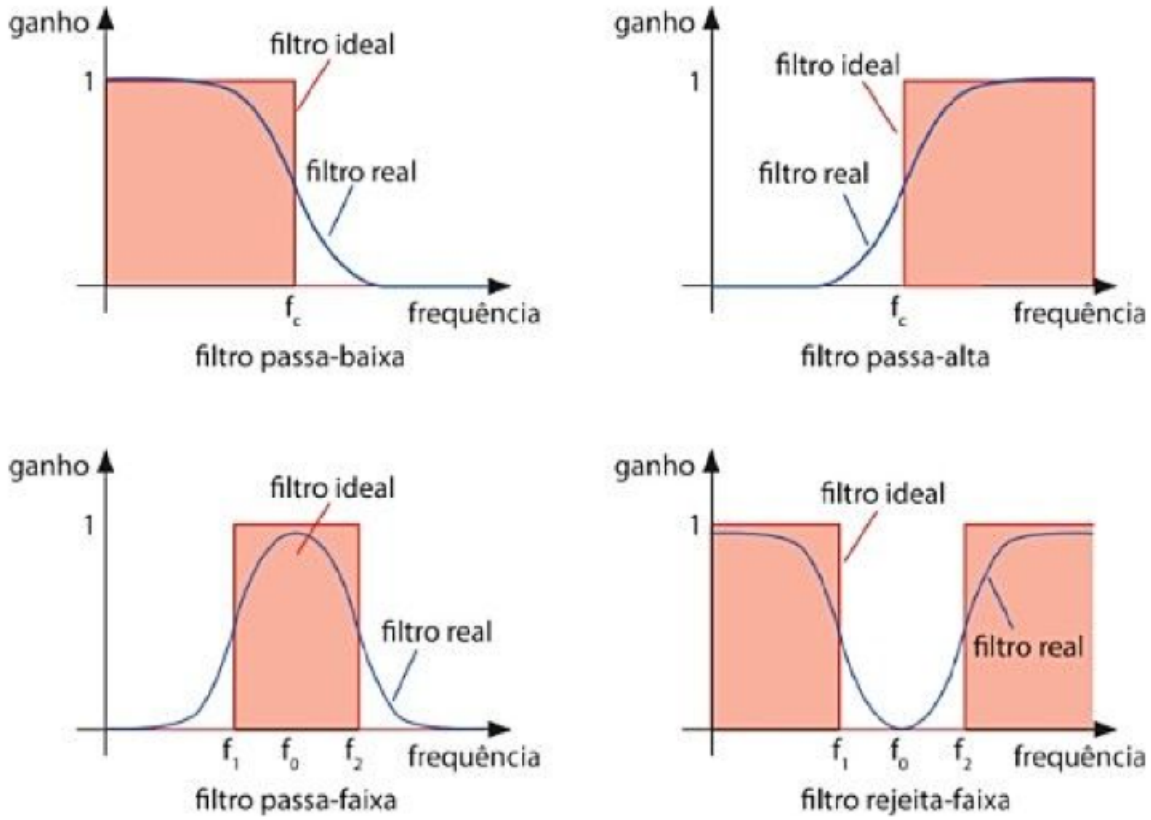


**Figura 11: Resposta dos filtros [21].**

Os filtros passa-baixas permitem apenas a passagem de baixas frequências, atenuando todas as frequências acima de um certo valor chamado frequência de corte.

Os filtros passa-altas atuam como o inverso dos passa baixa, estes permitem a passagem apenas de frequências acima da frequência de corte.

Os filtros passa-faixas possuem duas frequências de corte, permitindo apenas a passagem das frequências entre elas.

Os filtros rejeita-faixas possuem duas frequências de corte, permitindo apenas a passagem das frequências fora do intervalo entre as duas frequências.

O filtro de interesse para este trabalho é o filtro passa-baixas, será desenvolvido um filtro passivo RC, ou seja composto apenas por um resistor e capacitor, a montagem desse tipo de filtro pode ser visto na Figura 12. Este filtro foi selecionado por ser um filtro de primeira ordem, com resposta plana na banda passante, e com atenuação de -20 dB/dec para sinais com frequências acima da frequência de corte. Assim também este filtro passa-baixa permitirá limitar a banda no processo de aquisição de sinais.

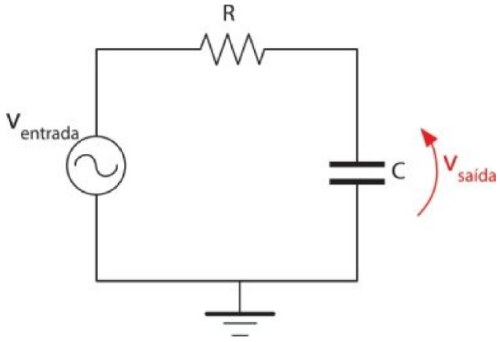


**Figura 12: Montagem de um filtro passa-baixas RC [21].**

Esse filtro é projetado de forma que o capacitor funcione como uma chave aberta para baixas frequências e como uma chave fechada para altas como pode ser observado na Figura 13 [21].

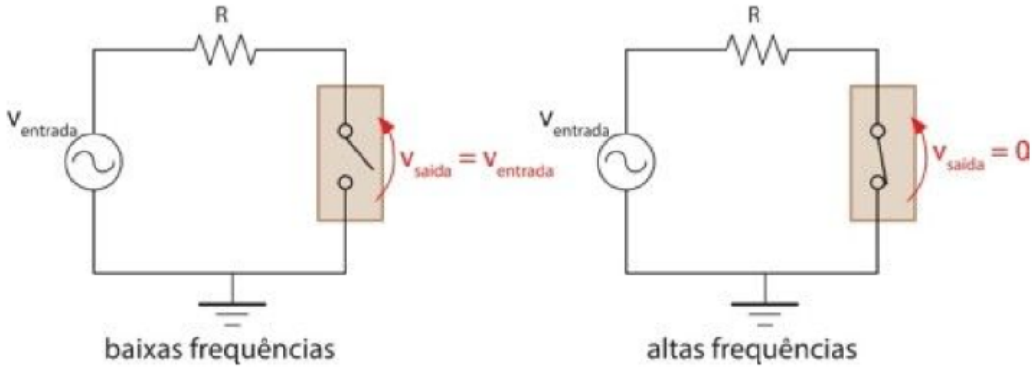


**Figura 13: Circuito equivalente ao funcionamento do filtro passa-baixas. [21]**

A relação do resistor e do capacitor do circuíto é o que define a frequência de corte

do filtro, o cálculo dessa frequência é dada pela equação (7):

$$f_c = \frac{1}{2\pi * RC} \tag{7}$$

Dessa forma é possível projetar os valores de R e C que satisfazem a frequência de corte desejada para o filtro.

## 2.8 Princípio da superposição

Outro problema comum na aquisição dos sinais é a localização da referência, para a aquisição usando um microcontrolador, por exemplo, não é possível a identificação de um sinal negativo, fazendo-se necessário muitas vezes deslocar o zero do sinal, isso é possível ser executado utilizando-se do princípio da superposição.

O teorema da superposição é uma consequência do princípio da linearidade dos circuitos, ele menciona que, em um circuito linear com duas ou mais fontes, é possível em qualquer ponto do circuito calcular a corrente ou tensão de formas isoladas (Como se as demais fontes não existissem), que o resultado da variável naquele ponto seria a soma algébrica do resultado obtido por todas as outras fontes [17]. A representação desse principio pode ser visto na Figura 14.

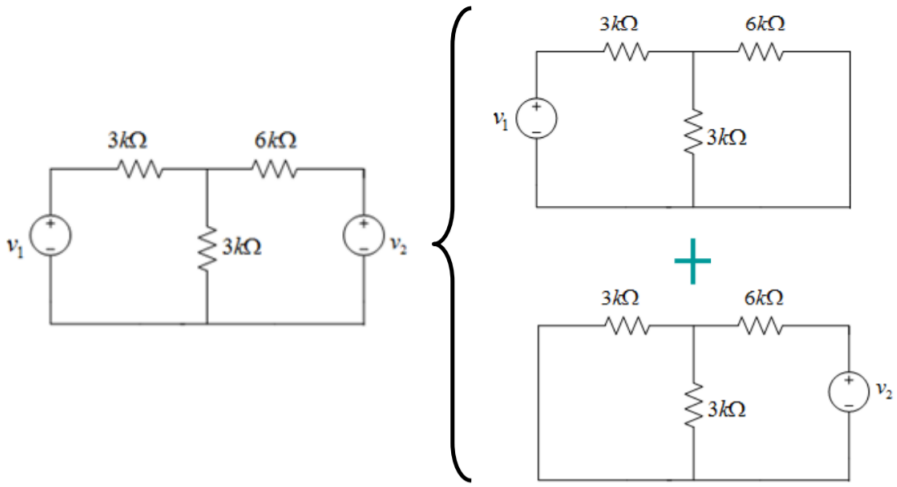


**Figura 14: Teorema da superposição.**

## 2.9 Conversão Analógico/Digital

Após os dados físicos serem aquistados pelos sensores e condicionados pelos circuitos de condicionamento os dados físicos são convertidos para níveis de tensão dentro de uma faixa controlada, essa tensão ainda é contínua, porém para serem analisadas por

um computador ou sistema digital é necessário três etapas: amostragem, quantização e codificação [18].

A amostragem é a coleta de amostras do sinal contínuo em períodos igualmente espaçados de tempo, deve-se tomar muito cuidado com o período de amostragem $T_s$ escolhido, pois se este for muito longo o sinal não será reconstruído corretamente. Para realizar a amostragem corretamente deve-se respeitar o critério de Nyquist, onde a frequência de amostragem deve ser maior ou igual a duas vezes a frequência máxima do sinal a ser amostrado como pode ser visto na equação 8.[18]

$$f_s >= f_M * 2 \tag{8}$$

Onde:

$f_s = \frac{1}{T_s}$

$f_s$ = Frequência de amostragem

$f_M$ = Frequência máxima do sinal

O sinal digital, ao contrário do sinal analógico, possuí um número finito de valores o qual ele pode assumir, por esse motivo, é necessário a etapa da quantização, nessa etapa os valores obtidos em cada medição assume o valor existente mais próximo da faixa de valores do controlador. Com a quantização sendo aplicada a cada período de amostragem, os valores infinitos podem ser representadas na escala finita digital.

O processo de amostragem e quantização pode ser verificado na figura 15:

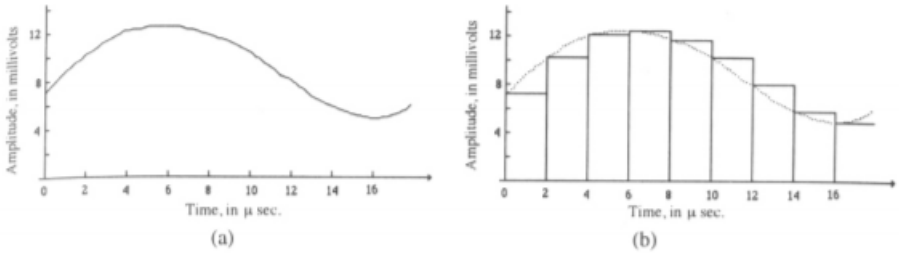


**Figura 15: Amostragem e quantização de um sinal contínuo[19]**

Por fim, a codificação tem o objetivo de atribuir um valor em bits para cada nível de tensão obtido na quantificação, a quantidade de bits máxima que o dispositivo apresenta para representar a tensão e o nível de tensão de trabalho do microcontrolador define a resolução do conversor analógico digital [20]. A resolução é o menor valor de tensão que o microcontrolador consegue converter e pode ser obtida pela equação (9):

$$Res = \frac{V_{ref}}{2^{n-1}} \tag{9}$$

Onde:

$n$ = Número de bits.

$V_{ref}$ = Tensão máxima do controlador.

## 2.10 Protocolo de comunicação

Após a conversão A/D é necessário saber como transmitir e armazenar estes dados. Para um sensoriamento remoto os dados devem ser transmitidos de alguma forma do equipamento de aquisição de dados para a máquina que irá processar, tratar e criar a visualização destes dados.

Para os parâmetros da indústria 4.0 é interessante o estudo de protocolos de rede, principalmente protocolos WiFi, dispositivos capazes de se conectar a rede WiFi são conhecidos como dispositivos IoT [14]. Muitos dispositivos possuem conexão com a internet, mas para que haja transferências de dados é necessário que exista um protocolo, para exemplificar um dos protocolos mais famosos é o HTTP, utilizado pela maioria das páginas da internet.

Porém o HTTP não é o mais otimizado para aplicações IoT, seu protocolo é baseado em uma estrutura de *Client - Server* o qual existe um trafego extenso de dados. Por isso um protocolo muito interessante para aplicações IoT foi criado, o *Message Queuing Telemetry Transport* (MQTT), este protocolo age através de uma arquitetura *Publish - Subscribe*, onde todos os dispositivos são chamados de "*Clients*", estes *clients* se conectam a um *broker* responsável por receber e distribuir as mensagens [15].

Nesta arquitetura um cliente "publica"uma mensagem, contendo por exemplo as informações de uma leitura de um sensor, este dado é publicado sob um tópico caracterizado como uma *tag* que informa qual o "assunto"da mensagem, O *broker* recebe esta mensagem e então a envia para quaisquer outros *clients* que estejam subscritos a esse tópico, assim a comunicação só ocorre quando um dado novo é enviado, e só é repassado para dispositivos que estejam interessados neste dado [14].

Essa arquitetura pode ser ilustrada a partir da Figura 16:

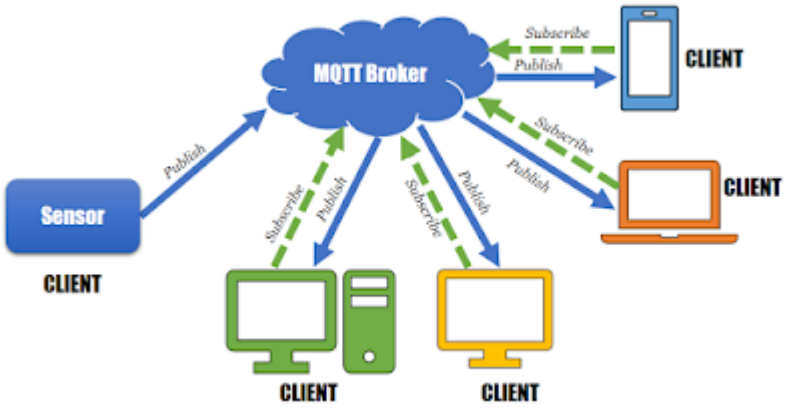


**Figura 16: Arquitetura do protocolo MQTT.**

## 2.11 Conclusão do capitulo

Foi visto neste capítulo o comportamento esperado ao se realizar a análise de Park de um motor, assim como as diversas etapas necessárias para a aquisição e condicionamento de um sinal e por fim o protocolo MQTT para transmissão dos dados via WiFi, a partir dessas bases foi possível realizar o protótipo proposto, demonstrado no capitulo 3.

# 3 MATERIAIS E MÉTODOS

Nesse capítulo foi apresentado as simulações, materiais e montagem de cada etapa do protótipo, seguindo o fluxo representado na Figura 17.

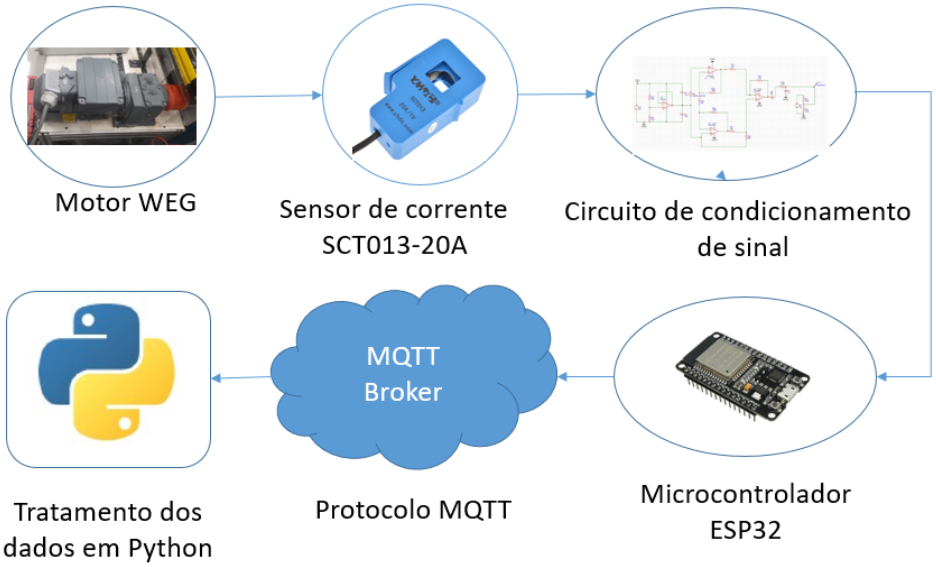


**Figura 17: Fluxograma real do projeto.**

As definições dos parâmetros, alimentação e ganhos do projeto foram definidas a partir dos parâmetros da bancada de testes e a partir da tensão que estaria disponível nos painéis os quais seria implementado o produto final, ou seja, o protótipo foi projetado para funcionar captando um sinal de corrente alternada nominal de 1,06A, e alimentado com uma única fonte de $24V_{DC}$.

## 3.1 Bancada de testes

A bancada experimental que foi utilizada para testes e validação deste protótipo é composta por um motor de indução trifásico da SEW como mostrado nas Figuras 18 e 19, e uma bancada de automação para acionamento do motor mostrado na Figura 20. Este motor foi montado na configuração de partida direta com reversão, e os sensores de corrente instalados nos cabos de alimentação do motor.


**Figura 18: Motor de indução trifásico SEW.**

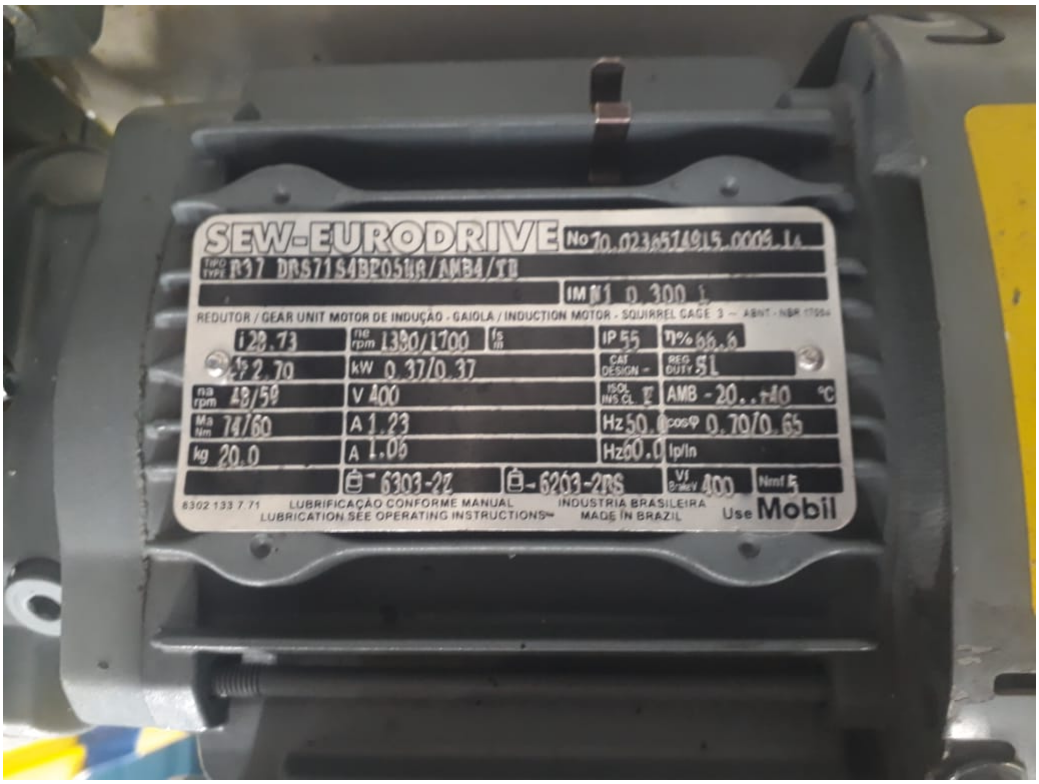


**Figura 19: Placa do motor.**

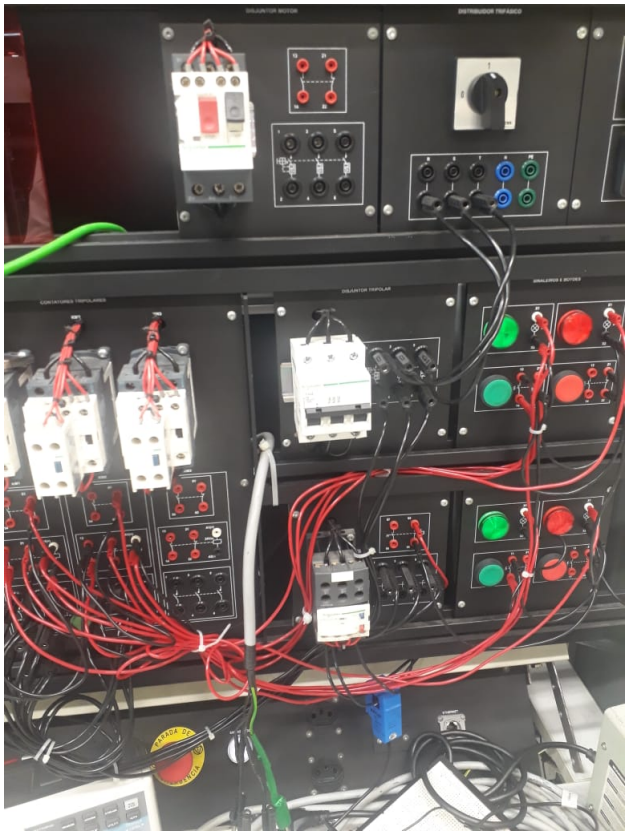

**Figura 20: Bancada de automação.**

Para substituir a fonte de 24V do painel foi utilizado uma fonte regulada de tensão como ilustrado na Figura 21, e para comparar os resultados obtidos pelo microcontrolador foi usado um osciloscópio modelo MO-2100 mostrado na Figura 22.

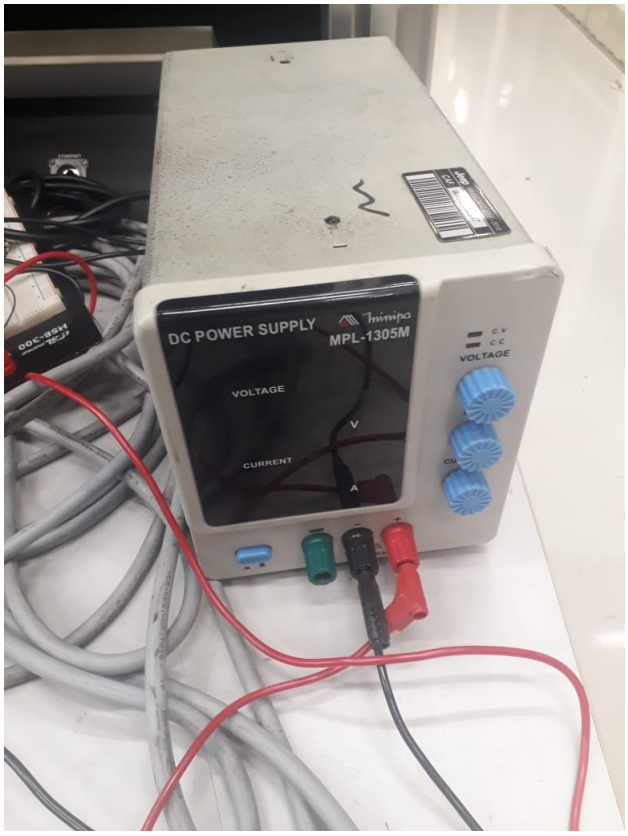


**Figura 21: Fonte regulada de tensão.**


**Figura 22: Osciloscópio.**

## 3.2 Sensor de corrente

O sensor utilizado neste protótipo foi um sensor de corrente não invasivo SCT-013-020, como ilustrado na Figura 23. Este sensor permite uma entrada de até 20A/1V, ou

seja ao se aplicar 20A na entrada deste sensor um sinal de 1V é aplicado na saída.


**Figura 23: Sensores de corrente.**

Este sensor foi escolhido por ser um sensor que funciona de maneira não invasiva, sem interferir no funcionamento do motor, e por ser um sensor muito fácil de se encontrar no mercado por conta de seu uso em microcontroladores.

Porém, para a aplicação neste protótipo, ele será utilizado para aquisitar uma corrente de pico 1.06A de corrente alternada, que é a corrente do motor trifásico da bancada, para esta corrente, a tensão na saída do sensor é muito pequena, como pode ser visto aplicando-se as equações do sensor:

$$V_{\text{saída}} = \frac{I_{entrada} * 1V}{20} = \frac{1,06A * 1V}{20A} = 0.053V$$

A partir deste sinal temos uma base do valor de entrada, permitindo assim o calculo dos ganhos necessários para adequar o sinal para o microcontrolador.

## 3.3 Definição do microcontrolador

Antes de desenvolver o circuito de condicionamento deve-se definir qual a faixa de tensão que será trabalhada, para isso foi considerado todos os requisitos do projeto para definir o microcontrolador que melhor atendia.

Os principais requisitos para o projeto são a presença de pelo menos três entradas ADC, preço, resolução, e a conectividade com WiFi.

Por este motivo, foi feito um comparativo com alguns microcontroladores para definir qual o mais indicado para esta aplicação como pode ser visto na imagem 24.

| | Arduino UNO | Arduino Mega | ESP32 | ESP8266 |
|---|---|---|---|---|
| WiFi | NÃO | NÃO | SIM | SIM |
| Portas ADC | 6 portas | 16 portas | 18 portas | 1 porta |
| Resolução | 10 bits | 10 bits | 12 bits | 10 bits |
| Freq de operação | 0-16Mhz | 0-16Mhz | 80-240Mhz | 80-160Mhz |
| Preço médio | R$ 70,00 | R$ 100,00 | R$ 40,00 | R$ 30,00 |

**Figura 24: Tabela comparativa.** [23]

Com isso pode-se verificar que o ESP32 é o que mais atende as necessidades do projeto, possuindo 18 entradas ADC com 12 bits de resolução,uma conectividade nativa WiFi e um preço médio acessível.

Para que o sinal de entrada seja discretizado de forma a possibilitar sua reconstrução por este microcontrolador, o sinal de tensão de entrada deve ser totalmente positivo, manter sua forma de onda original e estar na faixa de 0-3.3V.

## 3.4 Circuito de condicionamento de sinal

O circuito de condicionamento de sinal deve adequar o sinal para os parâmetros suportados pelo microcontrolador. Dado que neste protótipo serão monitorados 3 sinais de entrada de corrente, os seguintes itens devem ser aplicados para cada um, condicionado cada sinal para serem usados como entrada de diferentes portas do microcontrolador. Como os três circuitos são idênticos esta seção explicará como se fosse um único circuito. As etapas que o sinal será submetido são as seguintes:

- Amplificação do sinal.
- Desacoplamento do sinal CC.
- Filtragem do sinal.
- Deslocamento do zero.

O circuito completo está representado na Figura 25, em seguida cada etapa do circuito será desenvolvido para melhor entendimento de seu funcionamento e definições de valores dos componentes.

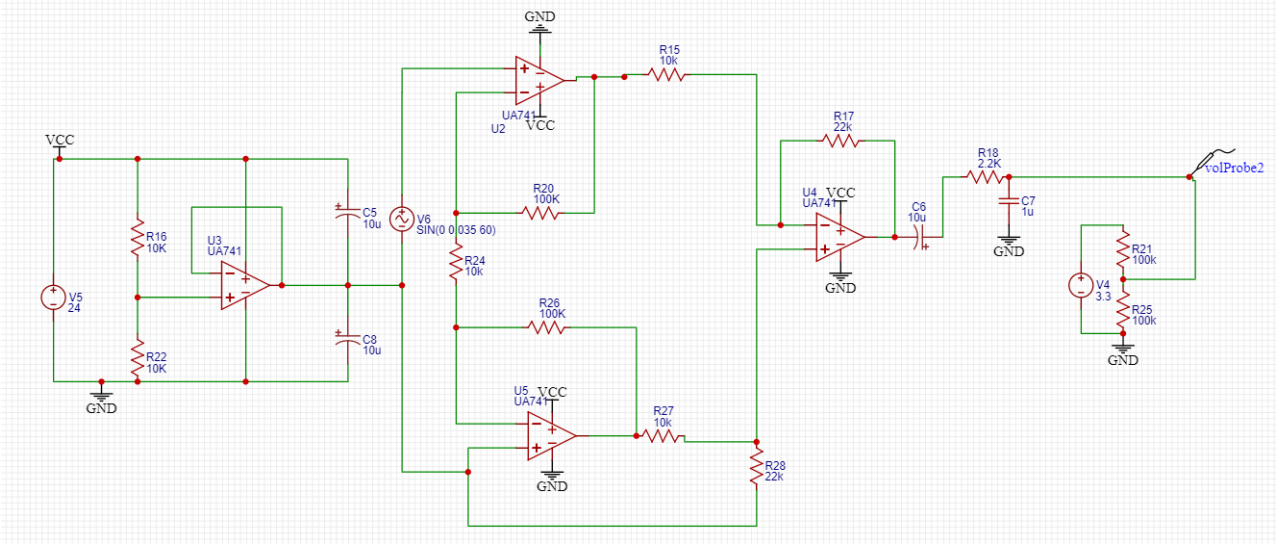


**Figura 25: Circuito de condicionamento completo.**

## 3.5 Circuito de condicionamento de alimentação

Para satisfazer a condição de alimentação de 24VDC, é necessário projetar um circuito que condicione esta tensão para Alimentar os amplificadores operacionais com uma tensão negativa e positiva. Para isso foi utilizado uma configuração de *buffer* com um divisor resistivo na entrada, gerando uma referência no ponto central da tensão de alimentação, e ao usar esta tensão como referência para o sinal, pode-se conectar o terra real na entrada negativa do amplificador operacional, e o positivo da tensão real no positivo do amplificador, desta maneira o sinal senoidal será mantido e amplificado sem a necessidade da aplicação de um sinal negativo na entrada, esse circuito pode ser visto na Figura 26.

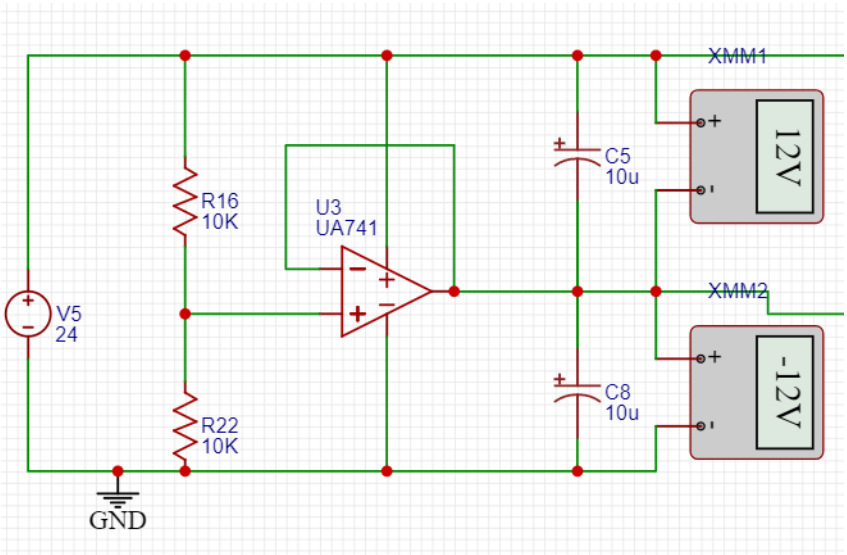


**Figura 26: Fonte simetrica a partir de fonte simples.**

## 3.6 Amplificador de instrumentação

A amplificação deste sinal é feito a partir de um amplificador de instrumentação, o sinal de entrada é esperado um sinal alternado com valor de pico de 0,037V ou 37mV, como o microcontrolador suporta no máximo 3,3V, a tensão máxima considerada ao amplificar o sinal deve ser a tensão de pico, e como o sinal será deslocado para a metade da tensão máxima, a tensão de pico após a amplificação deve ser no máximo 1,65V, assim como o ponto central deve ser neste mesmo valor.

Para isso o cálculo do ganho do amplificador operacional deve ser no máximo de:

$$G = \frac{1,65V}{0,037V} = 46,2$$

Utilizando-se da configuração do amplificador de instrumentação apresentado na parte teórica, pode-se calcular os ganhos necessários para este sinal, usando as equações (4), (5) e (6):

$$G = G1 * G2 = 46,2$$

Dividindo o ganho para as duas etapas, pode-se atribuir valores para G1 e G2 que se aproximem do ganho total desejado, escolhendo para este projeto G2 = 21 pode-se atribuir os valores de $R_3 = 100k\Omega$ e $RG = 10k\Omega$. Assim o ganho G1 deve ser igual a:

$$G1 = \frac{G}{G2} = 2,2$$

Definindo $R_3 = 10k\Omega$ se obtém o valor de $R_4$ que satisfaz a equação:

$$R_1 = 2,2 * 10k\Omega = 22k\Omega$$

Com os ganhos calculados pode se simular esta etapa do circuito, acoplada a fonte simétrica projetada, para substituir o sensor foi adicionado na entrada dos circuito uma fonte senoidal com amplitude de 0,035V e frequência de 60Hz. essa etapa pode ser vista na Figura 27

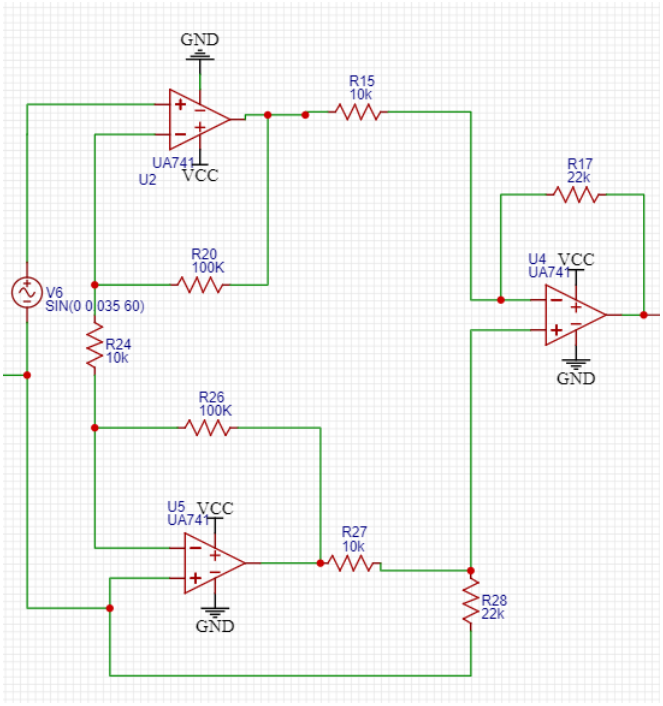


**Figura 27: Amplificador de instrumentação.**

A parte negativa do circuito de alimentação foi conectada ao terra virtual para garantir a referencia do sinal positiva em 12V, e as resistências calculadas foram implementadas para garantir o ganho desejado.

## 3.7 Desacoplagem CC

O sinal de saída do circuito anterior possuí a amplitude desejada para o sinal senoidal, porém, por conta do terra virtual, o sinal está com o seu centro ligado ao terra virtual, que por conta da fonte de 24V é em 12V, para isso foi adicionado um capacitor de desacoplamento de $10\mu F$ na saída desta etapa do circuito para permitir apenas a passagem da componente de corrente alternada mostrado na Figura 28.

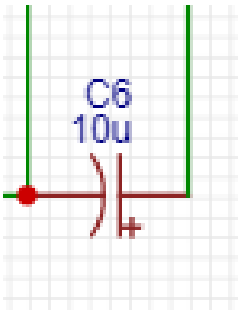


**Figura 28: Capacitor de desacoplamento.**

## 3.8 Filtragem do sinal

A aplicação final deste protótipo tem destino a um ambiente industrial, que é um ambiente muito ruidoso, por este motivo foi adicionado a esta etapa um filtro passa baixas, o sinal esperado é um sinal de 60Hz. Assim foi projetado um filtro com uma frequência de corte próxima usando a equação (7), Escolhendo o valor do capacitor como $1\mu F$ a resistência para esta obter um valor para esta frequência de corte deve ser:

$$R = \frac{1}{2 * \pi * 60 * 10^{-6}} = 2652\Omega$$

O filtro escolhido é um filtro passa-baixas passivo como visto na Figura 29 que possuí uma resposta um pouco mais lenta que os filtros ativos, por esto motivo é interessante dar uma margem na atenuação do sinal e garantir que o sinal desejado esteja dentro da faixa, além disso para facilitar o encontro do resistor no mercado, foi escolhido usar um resistor de $2,2k\Omega$, gerando uma frequência de corte de:

$$f_c = \frac{1}{2 * \pi * 2200 * 10^{-6}} = 72Hz$$

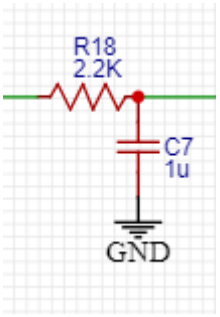


**Figura 29: filtro passa-baixas.**

## 3.9 Deslocamento do centro por superposição.

Para finalizar o circuito de condicionamento é necessário deslocar novamente o centro do sinal, o qual deve ser a metade do valor máximo suportado pelo microcontrolador que irá receber o sinal. Para aplicar este deslocamento foi utilizado o princípio da superposição, assim um sinal aplicando 3,3V do microcontrolador a um divisor resistivo conectando esta tensão ao sinal mostrado na Figura 30 obtem-se o sinal final que será monitorado pelo microcontrolador.

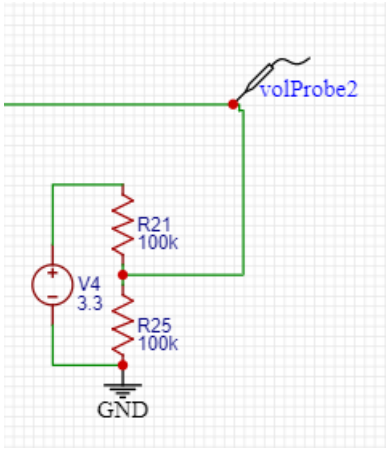


**Figura 30: Adição do sinal DC por superposição.**

## 3.10 Circuito montado

Após todas as etapas terem sido devidamente calculadas e simuladas o circuito de condicionamento foi montado em protoboard em conjunto com o microcontrolador ESP32 para se realizar os testes de discretização e envio do sinal. O circuito montado e replicado para os três sensores de corrente pode ser observado na Figura 31.

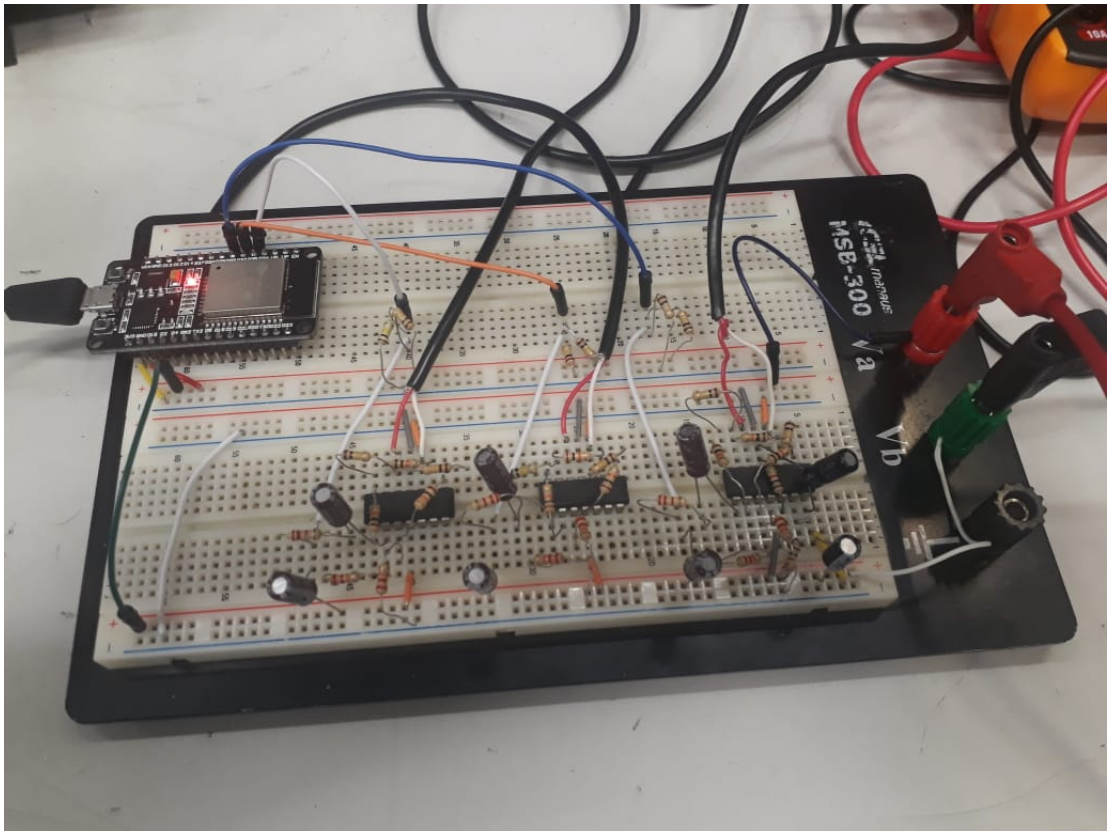


**Figura 31: Circuito real montado em protoboard**

### 3.11 Conversão Analógico/Digital

Neste ponto, o sinal de tensão é obtido pelas entradas ADC do ESP32 que fica responsável pela discretização e amostragem do sinal. A saída do circuito anterior é aplicada nos pinos 32, 33 e 35, que foram configuradas como entradas (*inputs*), e nelas aplicadas a função analogRead(), em um *loop* até obter a quantidade de amostras desejadas. O número de amostras definido foi de 256, garantindo a aquisição de 4 ciclos, para melhor visualização das correntes, com isso os 256 valores de cada entrada foram armazenados em um vetor, que em seguida foi condicionado em uma *string* para preparar o envio, essa configuração pode ser verificada na Figura 32.

```
#include <math.h>
#include <driver/adc.h>
#define ADC_INPUT1 33
#define ADC_INPUT2 32
#define ADC_INPUT3 35
#define AMOSTRAS 256
```

**Figura 32: Definição das portas ADC e número de amostras.**

A definição da quantidade de amostras anteriormente é importante, além disso é importante se definir o período de amostragem para poder executar corretamente a reconstrução do sinal. A frequência da alimentação é de 60Hz, e para realizar o calculo da FFT respeitando o Radix-2 o número de amostras deve ser uma potência de 2 para garantir a reconstrução computacional do sinal [25], dessa forma foi escolhido para satisfazer a quantidade de amostragem de $2^n$, um n=6, gerando um total de 64 leituras em um ciclo. para isso é necessário que no período de $\frac{1}{60Hz} = 16,67ms$, esteja as 64 leituras, para isso é necessário um período de $\frac{16,67ms}{64} = 260\mu s$[13], gerando uma frequência de amostragem $f_s = \frac{1}{250\mu s} = 3,85kHz$ e como o tempo de execução da leitura e armazenamento do dado leva aproximadamente $10\mu s$ para cada leitura é adicionado um atraso de $77\mu s$ entre as aquisições (Figura 33).

```
for (int i=0; i<AMOSTRAS; i++)
{
sensorValue = analogRead(ADC_INPUT1);
DADO[i] = sensorValue;
delayMicroseconds(77);
sensorValue2 = analogRead(ADC_INPUT2);
DADO2[i] = sensorValue2;
delayMicroseconds(77);
sensorValue3 = analogRead(ADC_INPUT3);
DADO3[i] = sensorValue3;
delayMicroseconds(77);
}
```

**Figura 33: *Loop* para aquisição do ADC.**

### 3.12 Protocolo de comunicação MQTT

Para utilizar o protocolo de comunicação MQTT para enviar os dados obtidos é necessário ter um *broker* configurado na rede, para estes testes foi utilizado um *broker* já previamente configurado pela empresa, como se trata de um *broker* justamente para testes não foi preciso nenhuma autenticação de login e senha, sendo necessário apenas o endereço de IP do *broker*.

Já existindo o *broker* é necessário configurar o ESP32 como um cliente, utilizando-se da biblioteca "PubSubClient.h", para usa-lo como *client* ele foi conectado ao *broker* usando o comando "MQTT.setServer(server, BROKER PORT);"onde *server* é o IP do *broker* e *BROKER PORT* a porta do *broker* . Em seguida definiu-se o tópico em que o *payload*, o qual é a mensagem a ser enviada, será publicado pela função "MQTT.publish(TOPICO PUBLISH, payload)", essa configuração pode ser observada na Figura 34.

```
//Wifi
#include "WiFi.h";
// Configuracao ID e senha da rede Wireless
#define SSID        "FCA-DEVICES"
#define PASSWORD    [redacted]

//MQTT
//Carrega a biblioteca de publish/subscription para mqtt
#include <PubSubClient.h>
#define ID_MQTT  "CORRENTE_SEW"

#define TOPICO_SUBSCRIBE "TRC/SEW/CORRENTE/"

#define TOPICO_PUBLISH   "TRC/SEW/CORRENTE/"

int BROKER_PORT = 1883; // Porta do Broker MQTT
IPAddress server(172, 29, [redacted], [redacted]);

//Variáveis e objetos globais
WiFiClient espClient; // Cria o objeto espClient
PubSubClient MQTT(espClient); // Instancia o Cliente MQTT passando o objeto espClient
```

**Figura 34: Configurações de WiFi e do MQTT.**

Para esta aplicação, o *payload* por ser um vetor contendo um número muito grande de amostragem acabou sendo excedendo o número máximo de caracteres definido como padrão para a biblioteca "PubSubClient.h"de 128 caracteres, dessa forma quando a mensagem ultrapassava esse valor ela não era enviada, para resolver este problema a biblioteca foi alterada e o valor máximo de caracteres pare envio de mensagem foi definido para 4096 para englobar com folga o tamanho do *payload* criado.

A construção do *payload* foi realizada transformando os vetores em uma *string* de dados mostrado na Figura 35, e a anexando a um termo de identificação para referenciar cada entrada, essa *string* então foi transformada em um vetor de *char*, para se encaixar no formato de *payload* do protocolo, por fim a função MQTT.publish(), publica o *payload* no tópico definido realizado na Figura 36.

```
s[1]+="\"I1\":[";
for (int i=0; i<AMOSTRAS-1; i++)
{
s[1]+=DADO[i];
s[1]+=",";
}
s[1]+=DADO[AMOSTRAS-1];
s[1]+="],";

s[2]+="\"I2\":[";
for (int i=0; i<AMOSTRAS-1; i++)
{
s[2]+=DADO2[i];
s[2]+=",";
}
s[2]+=DADO2[AMOSTRAS-1];
s[2]+="],";

s[3]+="\"I3\":[";
for (int i=0; i<AMOSTRAS-1; i++)
{
s[3]+=DADO3[i];
s[3]+=",";
}
s[3]+=DADO3[AMOSTRAS-1];
s[3] += "]}}";
s[0]=s[1] + s[2] + s[3];
```

**Figura 35: Criação da string de envio com os dados obtidos.**

```
Serial.println(s[0]);
s[0].toCharArray(outstr, 5000);
MQTT.publish(TOPICO_PUBLISH, outstr);
delay(10000);
```

**Figura 36: Envio dos dados via MQTT.**

### 3.13 Aquisição e tratamento do sinal

Após publicada, a mensagem pode ser aquistada por qualquer dispositivo configurado como cliente do *broker*, nesse caso foi utilizado um algoritmo em Python rodando em um computador, responsável pela aquisição e tratamento do sinal, assim como gerar os resultados finais deste projeto.

Para receber os dados o computador deve estar na mesma rede do *broker* e do outro cliente, assim ele pode ser configurado como um cliente deste *broker*, assim como o ESP32 só é necessário configurar o IP e porta do *broker*. A biblioteca em *python* que permite realizar este procedimento é a paho.mqtt.client (Figura 37). Como agora estamos configurando com outra biblioteca e linguagem a formatação da configuração é levemente diferente, explicitando previamente quais ações serão tomadas quando ocorrer a conexão ao *broker* MQTT usando a função "def on_connect", onde é informado se a conexão foi feita e ocorre a subscrição no tópico desejado indicado na Figura 38.

```
import base64
import paho.mqtt.client as mqtt
import serial
import csv
from numpy import arange, fft, angle
import matplotlib.pyplot as plt

dado = [ ]
dado2 = [ ]
dado3 = [ ]
I1 = [ ]
I2 = [ ]
I3 = [ ]
I22 = [ ]
I32 = [ ]
id = [ ]
iq = [ ]
idq = [ ]

AMOSTRAS = 256
```

**Figura 37: Definição das bibliotecas e do número de amostras.**

```
client = mqtt.Client()
client.connect("172.29.138.138", 1883, 60)
client.on_connect = on_connect
client.on_message = on_message


client.loop_forever()
```

**Figura 38: Definição dos parâmetros do MQTT e chamada das funções.**

Em seguida a rotina tomada pelo algoritmo é definido para ser executada quando receber alguma mensagem, na função "def on_message", assim quando uma nova mensagem chega neste tópico é realizado a quebra do *payload*, dividindo a mensagem completa

em três vetores distintos, e cada valor que foi enviado como char é transformado em inteiro (Figura 39).

```
def on_connect(client, userdata, flags, rc):
    print("Connect: " + str(rc))
    client.subscribe("TRC/SEW/CORRENTE/")

def on_message(client, userdata, msg):
    print("Topic: ", msg.topic)
    a = msg.payload.decode("utf-8")
    pload=list(a.split("["))
    i1=list(pload[1].split("]"))
    dado=list(map(int,i1[0].split(",")))
    print(dado)
    i2=list(pload[2].split("]"))
    dado2=list(map(int,i2[0].split(",")))
    print(dado2)
    i3=list(pload[3].split("]"))
    dado3=list(map(int,i3[0].split(",")))
    print(dado3)
```

**Figura 39: Funções do MQTT e conversão dos dados para vetor.**

Em seguida é definido alguns parâmetros que serão usados para cálculo das componentes de Park e da FFT na Figura 40.

```
n_ondas = 4 # escolhe o num. de ondas capturadas

n = n_ondas*64 # são 64 dados capturados para cada onda

T = n_ondas*1.0/60 # período em funçãoo do num. de ondas

dt = T/n # intervalo entre cada medida

t = dt*arange(0, n) # gera vetor com os instantes de tempo

Fk = fft.fft(id)/(n) # coeficientes de Fourier normalizados

nu = fft.fftfreq(n, dt) # frequências naturais

delta = angle(Fk) # ^angulo de fase de cada componente
```

**Figura 40: Definição de parâmetros e Cálculo da FFT.**

Após isso as correntes foram condicionadas para serem aplicadas no cálculo das componentes de Park id, iq e idq conforme a equação 1 usando as linhas de código representadas na Figura 41.

```
for i in range(0, AMOSTRAS):
    I1.append(((2/3)**(1/2))*dado[i])
    I2.append((1/((6)**(1/2)))*dado2[i])
    I3.append((1/((6)**(1/2)))*dado3[i])
    I22.append((1/((2)**(1/2)))*dado2[i])
    I32.append((1/((2)**(1/2)))*dado3[i])

for i in range(0, AMOSTRAS):
    id.append(I1[i]-I2[i]-I3[i])
    iq.append(I22[i]-I32[i])
    idq.append((id[i]**2 + iq[i]**2)**(1/2))
    idq.append((id[i]**2 + iq[i]**2)**(1/2))
```

**Figura 41: Geração dos vetores de Park.**

Por fim foi estabelecida uma função para a apresentação gráfica das correntes trifásicas no tempo (plot), das componentes id e iq no tempo, do vetor de Park e da FFT na Figura 42.

```
fig = plt.figure()
plt.subplot(2, 2, 1)
plt.ylim(0,4095)
plt.plot(t,dado,'-')
plt.plot(t,dado2,'-')
plt.plot(t,dado3,'-')
plt.xlabel('Tempo(s)')
plt.ylabel('Correntes 123')
plt.subplot(2, 2, 2)
plt.plot(id, iq, '-')
plt.ylim(-2000,2000)
plt.xlim(-2000,2000)
plt.xlabel('id')
plt.ylabel('iq')
plt.subplot(2, 2, 3)
plt.ylim(-2000,2000)
plt.plot(t,id,'-')
plt.plot(t,iq,'-')
plt.xlabel('Tempo(s)')
plt.ylabel('Correntes idq')
plt.subplot(2, 2, 4)
plt.xlim(0, 300)
plt.ylim(0, 250)
plt.bar(nu1, abs(Fk), width=5, align='center', alpha=0.4, color='b',
label='Frequencia')
plt.xlabel('freq (Hz)')
plt.ylabel('|A(freq)|')
plt.subplots_adjust(wspace=1)
plt.show()
```

**Figura 42: Plot das formas de ondas.**

# 4 APRESENTAÇÃO E ANÁLISE DOS RESULTADOS

Os resultados desse trabalho serão apresentados de maneira comparativa, objetivando demonstrar a correlação dos resultados obtidos mediante simulação e montagem, comentando o que era esperado segundo a teoria de cada etapa.

## 4.1 Simulação das harmônicas

Para o testar o funcionamento do filtro, e aproximar o sinal de entrada ao sinal real foi adicionado algumas fontes para simular algumas harmônicas em série com o sinal de entrada, como pode ser visto na Figura 43.

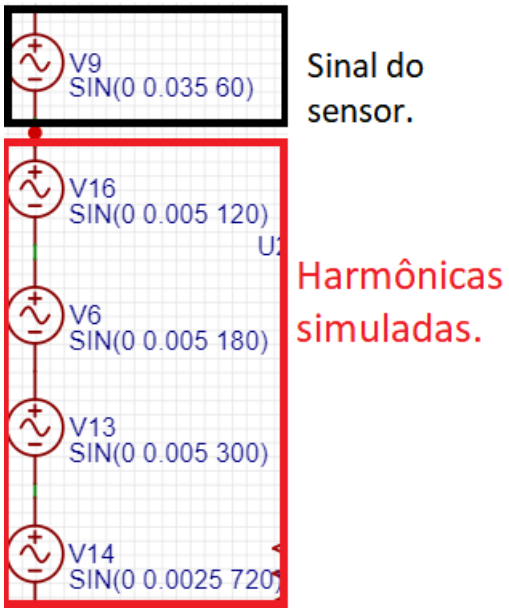


**Figura 43: Adição de harmônicas ao sinal.**

## 4.2 Sensor de corrente

A saída do sensor de corrente foi observada pelo osciloscópio (Figura 44), pode-se observar a periodicidade da onda referente a onda fundamental de 60Hz, é possível observar também a presença de outras harmônicas e ruído, que somadas a onda fundamental aumentam sua amplitude. Segundo a referência do osciloscópio a sua largura de banda é de até 100MHz, gerando uma banda de aquisição de até 50MHz, dessa forma captando muitas harmônicas e ruídos. Por conta dessas influências o valor de pico que teoricamente deveria ser de 37mV se tornou 49,6mV. Para tentar se aproximar pelo menos o formato da onda simulada foi adicionados alguns harmônicos na simulação mostrado na Figura 45, foi adicionados apenas harmônicos até 720Hz, porém não foi possível gerar todos os harmônicos e ruídos que elevaram a amplitude ao ponto obtido no osciloscópio.

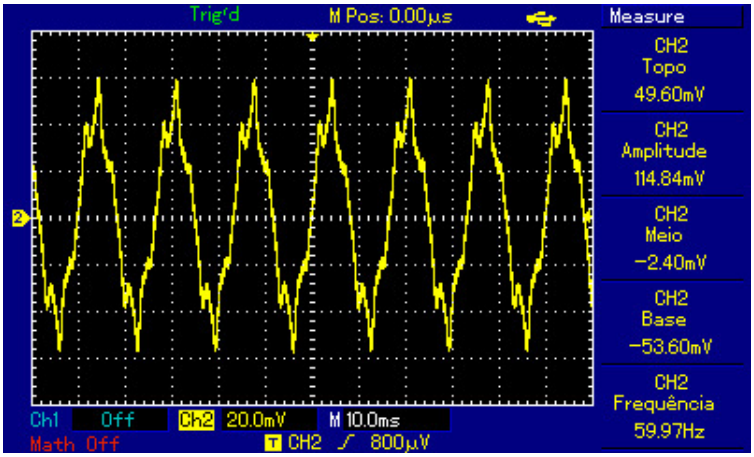


**Figura 44: Saída do sensor de corrente.**

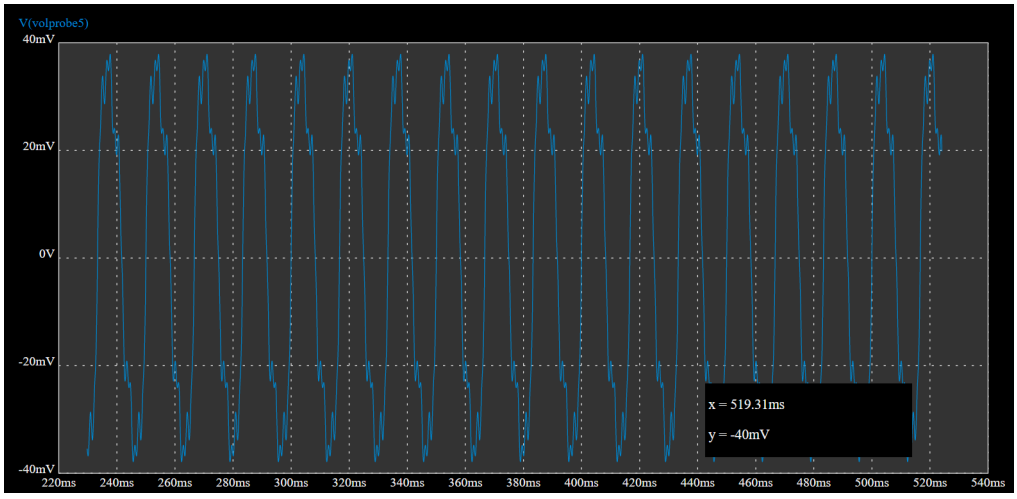


**Figura 45: Simulação da saída do sensor de corrente.**

Pode-se verificar que existe ainda uma diferença de formato e amplitude do real para o simulado, visto que o sinal real tem uma amplitude de 114,84mV, enquanto o simulado gera um sinal entre 76mV, essa diferença se da provavelmente por conta dos ruídos simulados que não foram altos o suficiente, porém como a comparação objetiva a prova do funcionamento de cada etapa o resultado está próximo o suficiente.

## 4.3 Amplificador de instrumentação

A primeira etapa o qual o sinal passa é pelo amplificador de instrumentação, como neste caso foi utilizado anteriormente a fonte simétrica com o terra virtual, o centro do sinal deve estar em 12V, com o sinal ainda apresentando componentes harmônicos e ruído aditivo, porém agora com um ganho de 46 vezes o sinal original mostrado na Figura 46 e simulado mostrado na Figura 47.

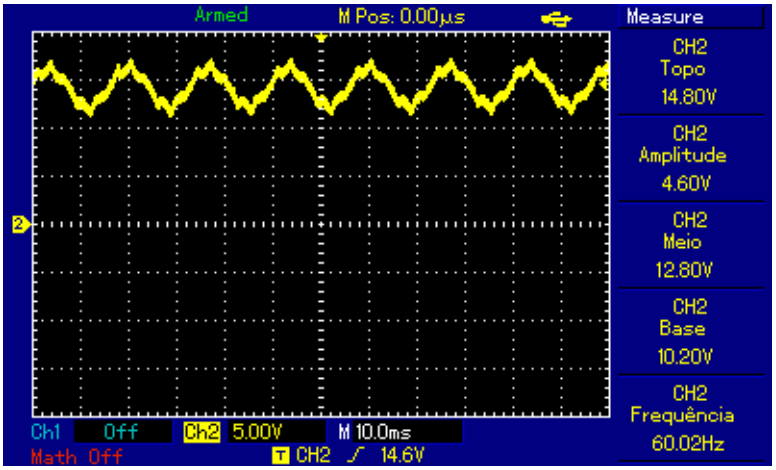


**Figura 46: Sinal amplificado.**

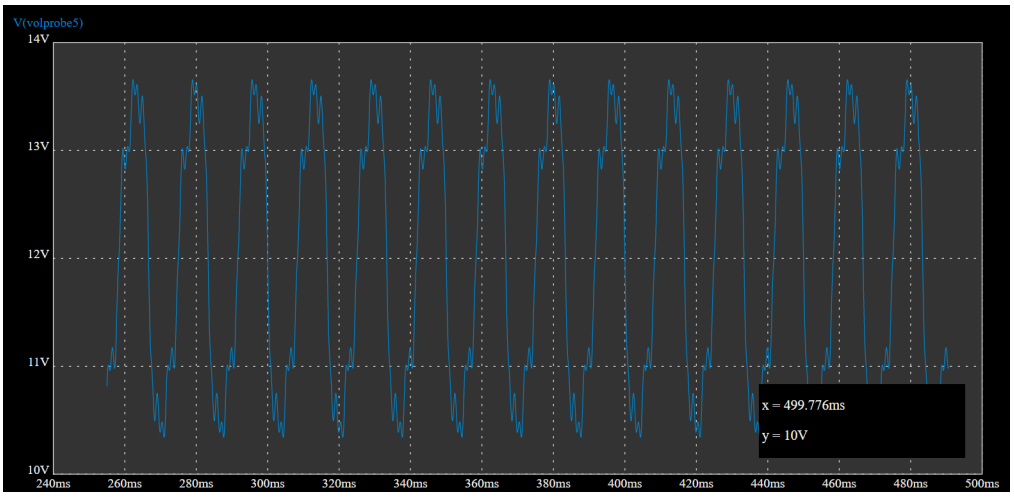


**Figura 47: Simulação do sinal amplificado.**

## 4.4 Capacitor de desacoplamento

Após a amplificação do sinal ele pode ser representado com a referencia real, já que a partir desse ponto nenhum Amp Op é mais utilizado, então é feito o desacoplamento da componente DC, tornando o sinal agora puramente AC, o levando para referencia em zero volts na Figura 48 e a simulação na Figura 49.

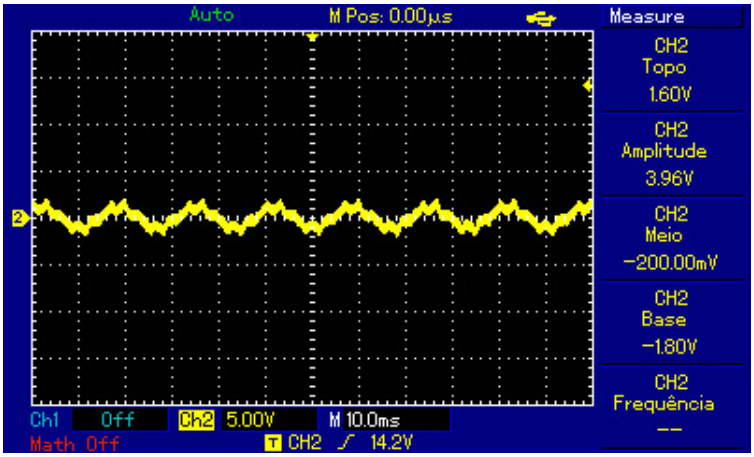


**Figura 48: Sinal desacoplado.**

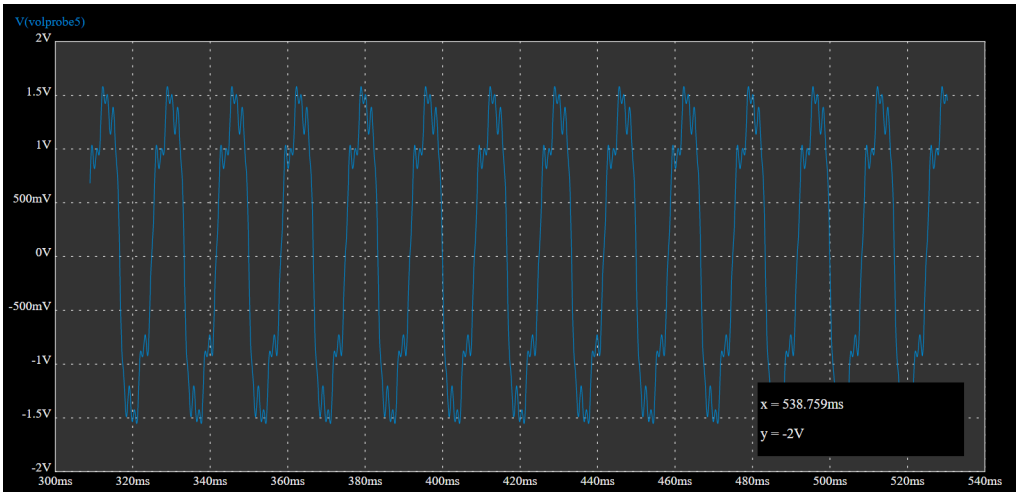


**Figura 49: Simulação do sinal desacoplado.**

## 4.5 Filtro passa baixas

Em seguida foi aplicado o filtro passa baixas com frequência de corte de 72 Hz, para remover as componentes harmônicos e os ruídos aditivos na Figura 50 e simulação na Figura 51.

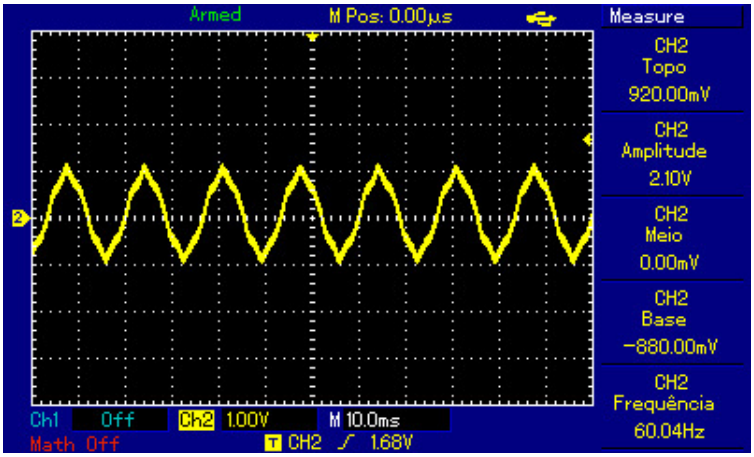


**Figura 50: Sinal filtrado.**

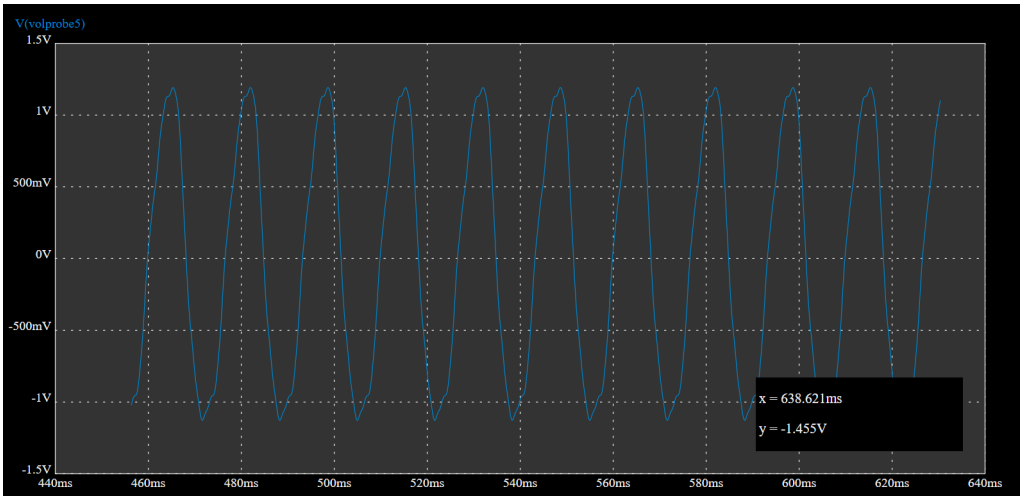


**Figura 51: Simulação do sinal filtrado.**

O procedimento de filtragem foi aplicado ao sinal simulado, na qual apresenta a influência de componentes de harmônicos com frequência não tão alta. Alguns desses sinais continuaram a modificar o sinal, isso influenciará no formato do plot dos vetores de Park, porém não impediu a mesma de ser realizada, por isso prosseguiu com estes sinais.

## 4.6 Deslocamento da referência

Para finalizar a etapa de tratamento de sinal foi deslocado a referência do sinal para metade da tensão máxima do microcontrolador ESP32 na Figura 52 e simulação na Figura 53.

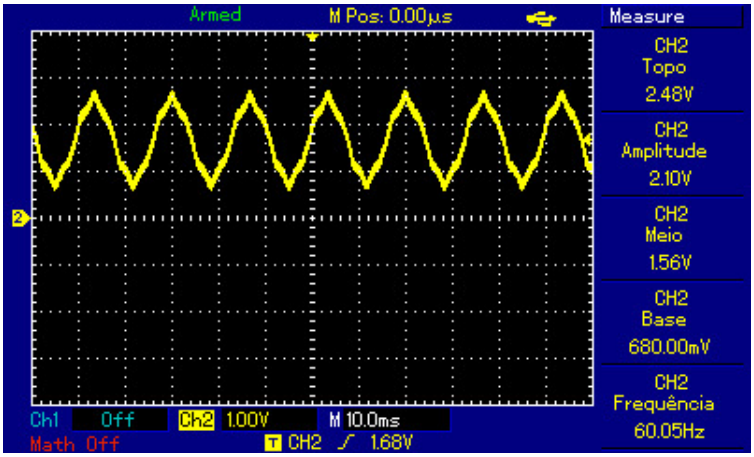


**Figura 52: Sinal condicionado.**

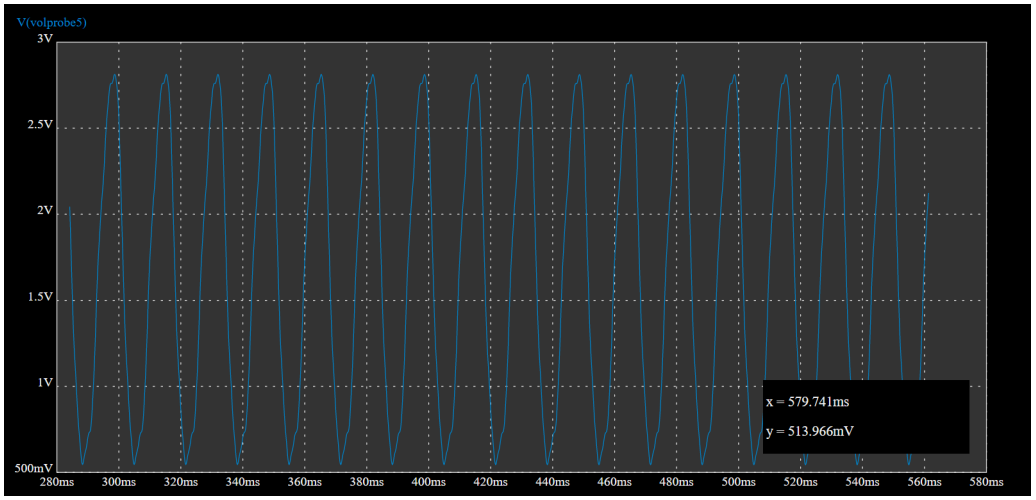


**Figura 53: Simulação do sinal condicionado.**

Observa-se que apesar de uma leve diferença de amplitude os dois sinais estão dentro da faixa de operação do microcontrolador, sendo assim podem sem problemas serem aplicadas na entrada do ESP32.

## 4.7 Conversão e envio do sinal

Uma vez realizado o condicionamento de sinais, o microcontrolador iniciou o processo de aquisição por meio da conversão analógico para digital no período e com a quantidade de amostras pré-definidos. Esses dados foram organizados em três vetores de dados que foram enviados pelo protocolo MQTT, recebidos pelo algoritmo em Python e reconstruídos no mesmo. O vetor recebido pode ser visto na Figura 54.

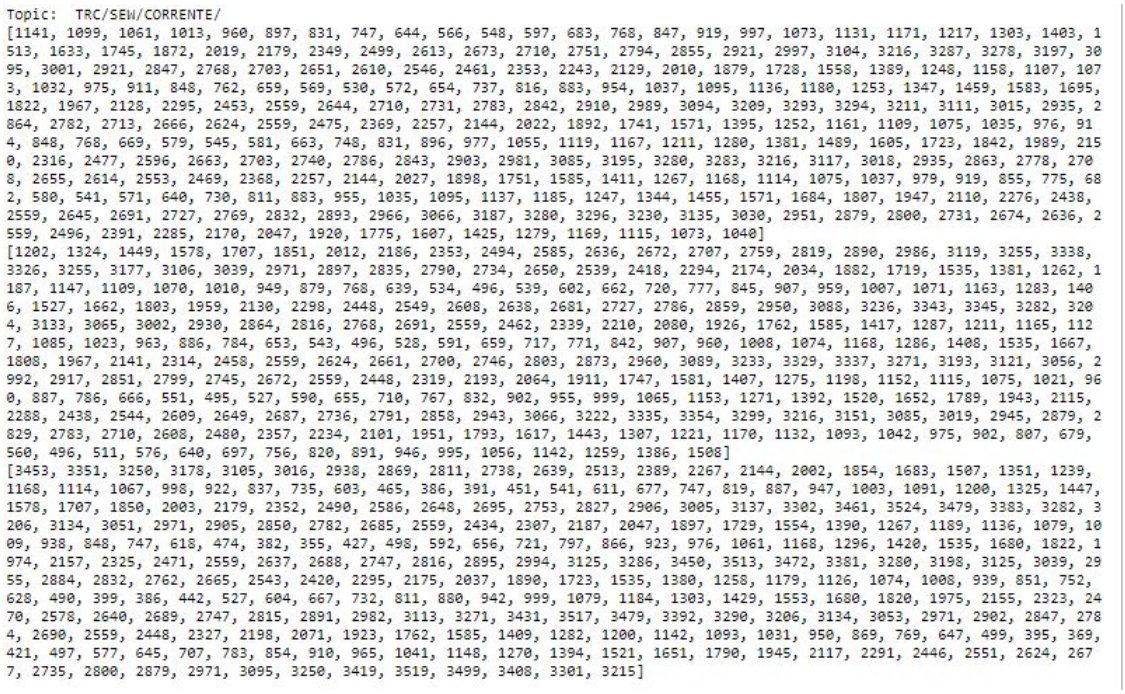


**Figura 54: Vetor com os dados obtidos pelo ESP32.**

## 4.8 Reconstrução do sinal e desenho dos vetores de Park

Com os dados do vetor recebido das correntes foram calculados os vetores de Park $i_d$ e $i_q$ através das linhas de código representadas na Figura 41, esses vetores foram representados de duas formas para análise, em função do tempo para demonstrar seu comportamento no referencial estatórico, e $i_d$ x $i_q$ para observar a excentricidade da transformada que aponta o correto funcionamento do motor. Além disso foi representado a FFT do vetor $i_d$, essa transformada apesar de ter sido executada corretamente deveria ter sido aplicada no vetor idq para permitir a análise de severidade do motor. As imagens dos resultados estão dispostos na seguinte ordem de representação:

- Correntes 123 no tempo.
- Vetores de park (id x iq).
- Correntes id e iq no tempo.
- FFT da corrente id.

O primeiro resultado a ser apresentado é com o motor funcionando normalmente. Na Figura 55 pode-se observar que existe um leve desbalanceamento da terceira fase, podendo ser causado por conta de algum defeito no motor ou por essa fase alimentar outros itens da bancada, pode-se observar também que todas as três ondas possuí um

certo pico não se comportando completamente de maneira senoidal, devido aos harmônicos que não foram completamente eliminados, esses harmônicos refletiram nas correntes $i_d$ e $i_q$, que não possibilitou o desenho perfeito do círculo, porém a imagem criada manteve a representação da excentricidade da transformada, apontando o correto funcionamento do motor. Por fim pode ser visto a transformada FFT do vetor $i_d$, nela existe um pico em 60Hz, e pode-se observar alguns elementos se destacando na terceira e na quinta harmônica, que reforçam as variações aplicadas na circunferência de Park.

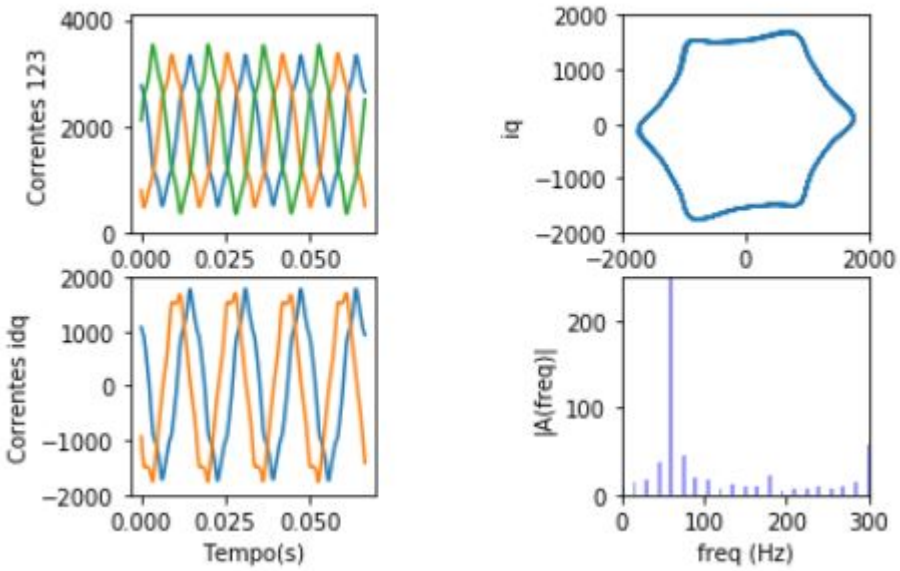


**Figura 55: Motor em funcionamento normal.**

Logo em seguida, foi executado o mesmo teste retirando-se os sensores das fases um, dois e três respectivamente, como ilustrado nas Figuras 56, 57 e 58. Aqui é possível observar que o comportamento do vetor de Park se deforma ou se inclina de maneira diferente com a ausência de cada uma dessas fases, com isso pode-se verificar que cada fase contribui de forma diferente para a composição do círculo, e a variação dessas fases de formas diferentes causam diferentes variações no campo girante que ao ser monitorado pode indicar as possíveis falhas da máquina.

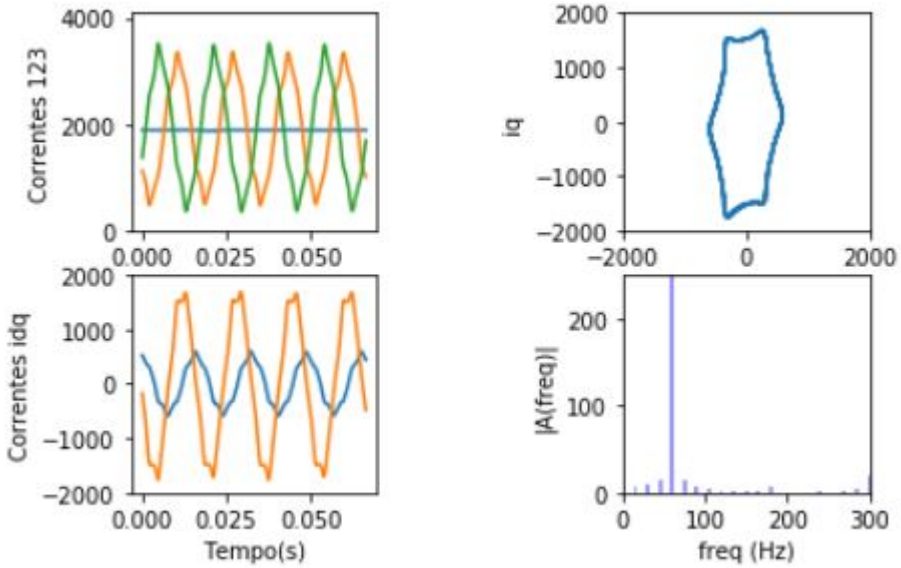


**Figura 56: Motor sem medidas da fase 1.**

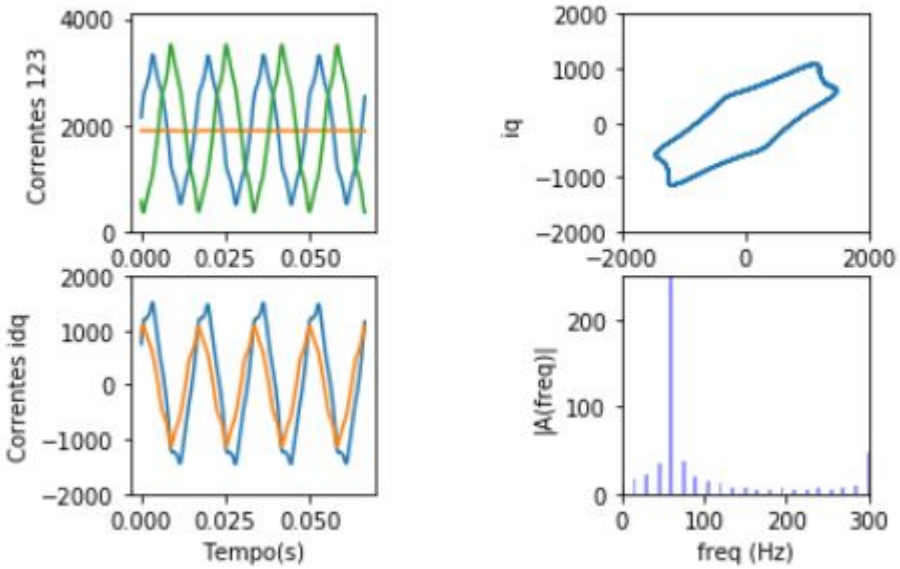


**Figura 57: Motor sem medidas da fase 2.**

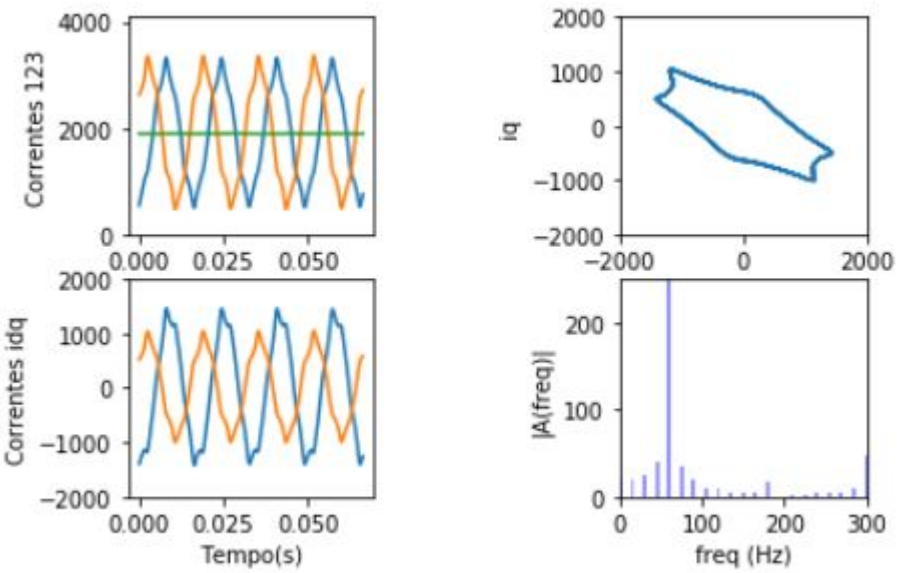


**Figura 58: Motor sem medidas da fase 3.**

Por fim, a Figura 59 apresenta como as formas de ondas são representadas quando não há alimentação, ou seja com o motor desligado.

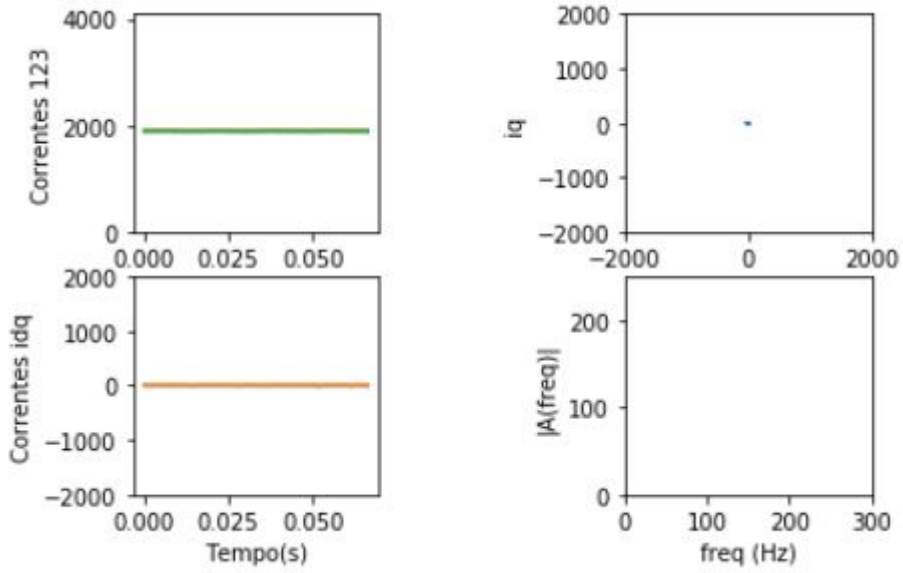


**Figura 59: Motor desligado.**

# 5 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho alcançou o objetivo de criar um protótipo de aquisição remota dos dados de corrente e de realizar a transformada de Park para análise preditiva.

O sensor realizou uma boa aquisição do sinal, apesar das harmônicas e ruídos, o ganho do amplificador de instrumentação real calculado a partir da relação de amplitudes obtidas no osciloscópio foi de 40 muito próximo do teórico de 46, provavelmente afetado apenas pela incerteza dos ruídos aleatórios e de perdas nos componentes, ou seja se comprovou bem eficiente para a aplicação. O desacoplamento ocorreu sem problemas, porém a filtragem do sinal não foi capaz de eliminar todos os ruídos e harmônicos, deixando o sinal um pouco distorcido, podendo ser feito a análise de substituir esse filtro passivo por um filtro ativo em um próximo teste para garantir uma banda de passagem menor. por fim a soma do sinal DC para realocar o centro realizada a partir do princípio da superposição foi capaz de atingir a tensão desejada, e o uso de resistores de valores altos ($100k\Omega$) no divisor resistivo permitiu que ocorresse baixas perdas.

A conversão AD teve um bom desempenho operacional, observando-se que a resolução do ADC do ESP32 de 12 bits permitiu uma quantização com resolução de 0,8mV/-bit, bem menor do que se fosse usado os outros microcontroladores comparados que garantiriam no máximo 3,22mV/bit. Por conta do processador as instruções de coleta e armazenamento dos dados consumiram apenas $10\mu s$, dessa forma foi possível controlar o atraso de $260\mu s$ dividindo esse atraso para as três aquisições do ADC considerando o tempo das instruções, atingindo a frequência de amostragem desejada, essa frequência não seria possível com um Arduíno, pois o mesmo consumiria $122\mu s$ para cada instrução de leitura e armazenamento.

O protocolo de comunicação se demonstrou muito eficaz, durante os testes 100% da informação foi enviada sem perdas de pacotes, e os vetores foram completamente reconstruídos pelo algoritmo em Python possibilitando a reconstrução do sinal para todos os enviados.

A reconstrução dos vetores de Park foi executada corretamente, porém como houve os problemas na etapa de filtragem descritos anteriormente, os vetores propagaram as harmônicas no desenho do vetor de Park, dificultando um pouco sua análise. A FFT foi calculada corretamente, apontando os picos na frequência fundamental de 60Hz e em suas harmônicas remanescentes, porém para usa-la como material de análise do condicionamento do motor deveria ter sido executada na corrente idq, e por conta de um erro de aplicação os resultados estão apresentando sua execução nos vetores id, com isso pode-se validar o correto cálculo da FFT porém não pode-se verificar a influência dos testes de falha nessa forma de onda.

Como trabalhos futuros, espera-se transformar o protótipo realizado em protoboard para uma versão realizada em PCB, dessa forma o projeto estaria menos sujeita a ruídos, assim como com menos riscos de algum componente sair durante algum teste, podendo também ser encapsulado para maior segurança.

Outra ampliação do trabalho seria o ensaio com vários modos de falha, permitindo a criação de um banco de dados, podendo usar isso para criação de uma rede neural que reconhecesse cada falha, criando alarmes e planejando ciclos de manutenção para as máquinas defeituosas, com isso seria alcançado o modelo ideal para uma manutenção preditiva.

# REFERÊNCIAS


[1] SCHWAB, Klaus. **The fourth industrial revolution.** Currency, 2017.

[2] SANTOS, Beatrice Paiva et al. Indústria 4.0 desafios e oportunidades. Revista Produção e Desenvolvimento, v. 4, n. 1, p. 111-124, 2018.

[3] BORLIDO, David José Araújo. Indústria 4.0 Aplicação a Sistemas de Manutenção. 2017.

[4] BORGES, JOSÉ WILLIAM RIBEIRO et al. INDÚSTRIA 4.0: APLICAÇÃO EXPERIMENTAL EM MOTOR DE INDUÇÃO MONOFÁSICO. 2018.

[5] THOMSON, William T.; FENGER, Mark. Current signature analysis to detect induction motor faults. IEEE Industry Applications Magazine, v. 7, n. 4, p. 26-34, 2001.

[6] BRITO, Jorge Nei et al. Desenvolvimento de um sistema inteligente híbrido para diagnóstico de falhas em motores de indução trifásicos. 2002.

[7] WEIDLICH, Felipe. Avaliação da lubrificação de rolamentos de motores elétricos por ultrassom. 2009.

[8] BALDISSARELLI, Luciano; FABRO, Elton. Manutenção Preditiva na indústria 4.0. Scientia cum Industria, v. 7, n. 2, p. 12-22, 2019.

[9] THOMAZINI, Daniel. Sensores industriais: fundamentos e aplicações. Saraiva Educação SA, 2005.

[10] FRANCISCO, António MS. MOTORES DE INDUÇÀO TRIFÊSICOS. Escola Superior de Tecnologia e Gestão de Viseu, 2006.

[11] SAITO, Minoru. Current transformer. U.S. Patent No 7,106,162, 2006.

[12] PERTENCE JR, Antonio. Amplificadores Operacionais e Filtros Ativos-8. Bookman Editora, 2015.

[13] DIONISIO, Guilherme; SPALDING, Luiz Eduardo Schardong. Visualização da forma de onda e conteúdo harmônico da corrente elétrica alternada em eletrodomésticos. Revista Brasileira de Ensino de Física, 2017, 39.1.

[14] LAMPKIN, Valerie et al. Building smarter planet solutions with mqtt and ibm websphere mq telemetry. IBM Redbooks, 2012.

[15] HUNKELER, Urs; TRUONG, Hong Linh; STANFORD-CLARK, Andy. MQTT-S—A publish/subscribe protocol for Wireless Sensor Networks. In: 2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE'08). IEEE, 2008. p. 791-798.

[16] ZHAO, Yan. Client/server two-way communication system framework under HTTP protocol. U.S. Patent Application n. 09/776,478, 8 ago. 2002.

[17] TAVARES, Marley Fagundes. Circuitos Elétricos. Londrina: Educacional S.A, 2018. 208 p.

[18] FERNANDES, Tales Gouveia; PANAZIO, Aline Neves. Do analógico ao digital: amostragem, quantização e codificação. II Simpósio de Iniciação Científica da Universidade Federal do ABC-SIC-UFABC, p. 44, 2009.

[19] DE ALBUQUERQUE, Márcio Portes; DE ALBUQUERQUE, Marcelo Portes. Processamento de imagens: métodos e análises. Rio de Janeiro, Brasil, v. 12, 2000.

[20] VERMA, Naveen; CHANDRAKASAN, Anantha P. An ultra low energy 12-bit rate-resolution scalable SAR ADC for wireless sensor nodes. IEEE Journal of Solid-State Circuits, v. 42, n. 6, p. 1196-1205, 2007.

[21] CARVALHO, Antônio Carlos Lemos. Laboratório de eletrônica analógica e digital: teoria e experimentos práticos. Teoria e experimentos práticos. São Paulo: Senai, 2009.

[22] PINHEIRO, Diego Dias. ANÁLISE E PROPOSIÇÃO DE ESTRATÉGIAS DE ESTIMAÇÃO E CONTROLE DE VELOCIDADE PARA MOTORES DE INDUÇÃO TRIFÁSICO. 2016. 161 f. Dissertação (Mestrado) - Curso de Engenharia Elétrica, Universidade Tecnológica Federal do Pará, Pato Branco, 2016.

[23] OLIVEIRA, Jailson. Arduino, ESP32 e ESP8266 – Comparação. [S. l.], 1 ago. 2019. Disponível em: http://xprojetos.net/arduino-esp32-e-esp8266-comparacao/. Acesso em: 19 mar. 2020.

[24] DUARTE, Pedro Miguel Gonçalves. Análise experimental do desempenho do motor de indução trifásico com entreferro excêntrico. 2017. Dissertação de Mestrado.

[25] SANTHOSH, Lakshmi; THOMAS, Anoop. Implementation of radix 2 and radix 2 2 FFT algorithms on Spartan6 FPGA. In: 2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT). IEEE, 2013. p. 1-4.

# ANEXO A – Código ESP32

```
static char outstr[5000];
String s[20];
#include <math.h>
#include <driver/adc.h>
#define ADC_INPUT1 33
#define ADC_INPUT2 32
#define ADC_INPUT3 35
#define AMOSTRAS 256
int DADO[AMOSTRAS];
int DADO2[AMOSTRAS];
int DADO3[AMOSTRAS];
int sensorValue;
int sensorValue2;
int sensorValue3;

//Wifi
#include "WiFi.h";
// Configuracao ID e senha da rede Wireless
#define SSID        "FCA-DEVICES"
#define PASSWORD    "dev@JEEP!2020"

//MQTT
//Carrega a biblioteca de publish/subscription para mqtt
#include <PubSubClient.h>
#define ID_MQTT  "CORRENTE_SEW"

#define TOPICO_SUBSCRIBE "TRC/SEW/CORRENTE/"

#define TOPICO_PUBLISH   "TRC/SEW/CORRENTE/"

int BROKER_PORT = 1883; // Porta do Broker MQTT
IPAddress server(172, 29, 138, 138);

//Variáveis e objetos globais
WiFiClient espClient; // Cria o objeto espClient
PubSubClient MQTT(espClient); // Instancia o Cliente MQTT passando o objeto espCli
```

```
//Prototypes
void initSerial();
void initWiFi();
void initMQTT();
void reconectWiFi();
void VerificaConexoesWiFIEMQTT(void);


/*
 *  Implementações das funções
 */
void setup()
{
    //inicializações:
    initSerial();
    initWiFi();
    initMQTT();
    pinMode(33, INPUT);
    pinMode(32, INPUT);
    pinMode(35, INPUT);
}

//Função: inicializa comunicação serial com baudrate 115200 (para fins de monitor
//        o que está acontecendo.
void initSerial()
{
    Serial.begin(9600);
}

//Função: inicializa e conecta-se na rede WI-FI desejada
void initWiFi()
{
    delay(10);
    Serial.println("------Conexao WI-FI------");
    Serial.print("Conectando-se na rede: ");
    Serial.println(SSID);
    Serial.println("Aguarde");
    reconectWiFi();
```

```
}

//Função: inicializa parâmetros de conexão MQTT(endereço do
//        broker, porta)
void initMQTT()
{
    MQTT.setServer(server, BROKER_PORT);
}

//Função: reconecta-se ao broker MQTT (caso ainda não esteja conectado ou em caso
//        em caso de sucesso na conexão ou reconexão, o subscribe dos tópicos é r
void reconnectMQTT()
{
    while (!MQTT.connected())
    {
        Serial.print("* Tentando se conectar ao Broker MQTT: ");
        Serial.println(server);
        if (MQTT.connect(ID_MQTT))
        {
            Serial.println("Conectado com sucesso ao broker MQTT!");
            MQTT.subscribe(TOPICO_SUBSCRIBE);
        }
        else
        {
            Serial.println("Falha ao reconectar no broker.");
            Serial.println("Havera nova tentatica de conexao em 2s");
            delay(2000);
        }
    }
}

//Função: reconecta-se ao WiFi
void reconectWiFi()
{
    //se já está conectado a rede WI-FI, nada é feito.
    //Caso contrário, são efetuadas tentativas de conexão
    if (WiFi.status() == WL_CONNECTED)
        return;
```

```
    WiFi.begin(SSID, PASSWORD); // Conecta na rede WI-FI

    while (WiFi.status() != WL_CONNECTED)
    {
        delay(100);
        Serial.print(".");
    }

    Serial.println();
    Serial.print("Conectado com sucesso na rede ");
    Serial.print(SSID);
    Serial.println("IP obtido: ");
    Serial.println(WiFi.localIP());
}

//Função: verifica o estado das conexões WiFI e ao broker MQTT.
//        Em caso de desconexão (qualquer uma das duas), a conexão
//        é refeita.
void VerificaConexoesWiFIEMQTT(void)
{
    if (!MQTT.connected())
     reconnectMQTT();
     reconectWiFi();
}

float lePorta(uint8_t outputpin)
{

}

void loop()
{
s[0]="";
s[1]="";
s[2]="";
s[3]="";

VerificaConexoesWiFIEMQTT();
```

```
for (int i=0; i<AMOSTRAS; i++)
{
sensorValue = analogRead(ADC_INPUT1);
DADO[i] = sensorValue;
delayMicroseconds(77);
sensorValue2 = analogRead(ADC_INPUT2);
DADO2[i] = sensorValue2;
delayMicroseconds(77);
sensorValue3 = analogRead(ADC_INPUT3);
DADO3[i] = sensorValue3;
delayMicroseconds(77);
}


s[1]+="\"I1\":[";
for (int i=0; i<AMOSTRAS-1; i++)
{
s[1]+=DADO[i];
s[1]+=",";
}
s[1]+=DADO[AMOSTRAS-1];
s[1]+="],";

s[2]+="\"I2\":[";
for (int i=0; i<AMOSTRAS-1; i++)
{
s[2]+=DADO2[i];
s[2]+=",";
}
s[2]+=DADO2[AMOSTRAS-1];
s[2]+="],";

s[3]+="\"I3\":[";
for (int i=0; i<AMOSTRAS-1; i++)
{
s[3]+=DADO3[i];
s[3]+=",";
}
s[3]+=DADO3[AMOSTRAS-1];
```

```
s[3] += "]}}";
s[0]=s[1] + s[2] + s[3];
Serial.println(s[0]);
s[0].toCharArray(outstr, 5000);
MQTT.publish(TOPICO_PUBLISH, outstr);
delay(10000);
}
```

# ANEXO B – Código Python

```
import base64
import paho.mqtt.client as mqtt
import serial
import csv
from numpy import arange, fft, angle
import matplotlib.pyplot as plt

dado = [ ]
dado2 = [ ]
dado3 = [ ]
I1 = [ ]
I2 = [ ]
I3 = [ ]
I22 = [ ]
I32 = [ ]
id = [ ]
iq = [ ]
idq = [ ]

AMOSTRAS = 256

def on_connect(client, userdata, flags, rc):
    print("Connect: " + str(rc))
    client.subscribe("TRC/SEW/CORRENTE/")

def on_message(client, userdata, msg):
    print("Topic: ", msg.topic)
    a = msg.payload.decode("utf-8")
    pload=list(a.split("["))
    i1=list(pload[1].split("]"))
    dado=list(map(int,i1[0].split(",")))
    print(dado)
    i2=list(pload[2].split("]"))
    dado2=list(map(int,i2[0].split(",")))
    print(dado2)
    i3=list(pload[3].split("]"))
    dado3=list(map(int,i3[0].split(",")))
```

```
print(dado3)

for i in range(0, AMOSTRAS):
    I1.append(((2/3)**(1/2))*dado[i])
    I2.append((1/((6)**(1/2)))*dado2[i])
    I3.append((1/((6)**(1/2)))*dado3[i])
    I22.append((1/((2)**(1/2)))*dado2[i])
    I32.append((1/((2)**(1/2)))*dado3[i])

for i in range(0, AMOSTRAS):
    id.append(I1[i]-I2[i]-I3[i])
    iq.append(I22[i]-I32[i])
    idq.append((id[i]**2 + iq[i]**2)**(1/2))
    idq.append((id[i]**2 + iq[i]**2)**(1/2))


n_ondas = 4 # escolhe o num. de ondas capturadas

n = n_ondas*64 # são 64 dados capturados para cada onda

T = n_ondas*1.0/60 # período em funçãoo do num. de ondas

dt = T/n # intervalo entre cada medida

t = dt*arange(0, n) # gera vetor com os instantes de tempo

Fk = fft.fft(id)/(n) # coeficientes de Fourier normalizados

nu = fft.fftfreq(n, dt) # frequências naturais

delta = angle(Fk) # ^angulo de fase de cada componente

fig = plt.figure()
plt.subplot(2, 2, 1)
plt.ylim(0,4095)
plt.plot(t,dado,'-')
plt.plot(t,dado2,'-')
plt.plot(t,dado3,'-')
plt.xlabel('Tempo(s)')
```

```
    plt.ylabel('Correntes 123')
    plt.subplot(2, 2, 2)
    plt.plot(id, iq, '-')
    plt.ylim(-2000,2000)
    plt.xlim(-2000,2000)
    plt.xlabel('id')
    plt.ylabel('iq')
    plt.subplot(2, 2, 3)
    plt.ylim(-2000,2000)
    plt.plot(t,id,'-')
    plt.plot(t,iq,'-')
    plt.xlabel('Tempo(s)')
    plt.ylabel('Correntes idq')
    plt.subplot(2, 2, 4)
    plt.xlim(0, 300)
    plt.ylim(0, 250)
    plt.bar(nu1, abs(Fk), width=5, align='center', alpha=0.4, color='b',
    label='Frequencia')
    plt.xlabel('freq (Hz)')
    plt.ylabel('|A(freq)|')
    plt.subplots_adjust(wspace=1)
    plt.show()

client = mqtt.Client()
client.connect("172.29.138.138", 1883, 60)
client.on_connect = on_connect
client.on_message = on_message


client.loop_forever()
```