Universidade Federal de Ouro Preto - UFOP
Escola de Minas
Colegiado do curso de Engenharia de Controle e Automação - CECAU

Isabella Ferreira de Oliveira

DESENVOLVIMENTO DE UM SISTEMA DE AUTOMAÇÃO RESIDENCIAL BASEADO EM IOT PARA CONTROLE E MONITORAMENTO DE DISPOSITIVOS ELÉTRICOS

Monografia de Graduação em Engenharia de Controle e Automação

Ouro Preto, 2019

Isabella Ferreira de Oliveira

DESENVOLVIMENTO DE UM SISTEMA DE AUTOMAÇÃO RESIDENCIAL BASEADO EM IOT PARA CONTROLE E MONITORAMENTO DE DISPOSITIVOS ELÉTRICOS

Monografia apresentada ao Curso de Engenharia de Controle e Automação da Universidade Federal de Ouro Preto como parte dos requisitos para a obtenção do Grau de Engenheiro de Controle e Automação.

Orientador: Prof. Dr. Antonio Santos Sánchez

Ouro Preto, 2019

O482d Oliveira, Isabella Ferreira.
Desenvolvimento de um sistema de automação residencial baseado em IoT para controle e monitoramento de dispositivos elétricos [manuscrito] / Isabella Ferreira Oliveira. - 2019.

70f.: il.: color; grafs; tabs.

Orientador: Prof. Dr. Antonio Santos Sánchez.

Monografia (Graduação). Universidade Federal de Ouro Preto. Escola de Minas. Departamento de Engenharia de Controle e Automação e Técnicas Fundamentais.

1. Internet of Things. 2. ThingSpeak. 3. Monitoramento de energia. 4. Automação residencial. I. Sánchez, Antonio Santos. II. Universidade Federal de Ouro Preto. III. Titulo.

CDU: 681.5

Catalogação: ficha.sisbin@ufop.edu.br

Universidade Federal de Ouro Preto
Escola de Minas
Departamento de Engenharia de Produção, Administração e Economia – DEPRO
Campus Universitário Morro do Cruzeiro
35400.000 – Ouro Preto, MG
Tel: 31-3559-1540 – Fax: 3559-1555 – E-mail: depro@depro.em.ufop.br

ATA DE DEFESA

Aos sete dias do mês de junho de dois mil e dezenove, às 08h00, no auditório da Escola de Minas do Campus Morro do Cruzeiro, foi realizada a defesa de Trabalho Final de Graduação - Monografia pela aluna **Isabella Ferreira de Oliveira**, sendo a banca examinadora constituída pelo professor Antonio Santos Sánchez (orientador), e pelos professores Agnaldo José da Rocha Reis e André Luis Silva. A aluna apresentou a monografia intitulada: "***Desenvolvimento de um sistema de automação residencial baseado em IoT para controle e monitoramento de dispositivos elétricos***". A banca examinadora deliberou, por unanimidade, pela aprovação do aluno, concedendo-lhe o prazo de quinze dias para incorporação, no texto final, das alterações sugeridas. Na forma regulamentar, foi lavrada a presente ata, que é assinada pelos membros da banca examinadora e pelo aluno.

Ouro Preto, 7 de junho de 2019.

Antonio Santos Sánchez
Orientador

Agnaldo José da Rocha Reis
Membro

André Luis Silva
Membro

Isabella Ferreira de Oliveira
Aluna

Resumo

Nos dias de hoje, as pessoas estão preocupadas no consumo eficiente da energia elétrica. O monitoramento e controle de dispositivos elétricos propõe contribuir com a gestão da energia em residências, trazendo comodidade e sustentabilidade para o usuário. Apresenta-se no presente trabalho o desenvolvimento de um protótipo para automação residencial, capaz de monitorar e controlar dispositivos elétricos por meio de uma aplicação em nuvem. Baseado no sistema de Internet of Things (IoT), esse dispositivo é capaz de realizar a medição da corrente e tensão alternada de um equipamento conectado à rede elétrica, e também realizar o cálculo aproximado da potência consumida, além de efetuar o acionamento de cargas. A interface com o usuário é realizada mediante a aplicação ThingSpeak, onde os dados do dispositivo são monitorados, armazenados e exportados para uma planilha onde é calculado a Energia consumida. Na aplicação também existem dois botões Liga e Desliga capazes de realizar o acionamento da carga. Foram realizados um teste de acionamento remoto da carga e testes dos sensores utilizando uma lâmpada halógena e, posteriormente, um *desktop*. Após os testes, constatou-se a viabilidade do sistema possibilitando o seu emprego para gestão energética, de forma remota, de dispositivos conectados à rede de energia alternada.

Palavras-chaves: *Internet of Things*, ThingSpeak, Monitoramento de Energia, Automação residencial.

Abstract

Nowadays, people are concerned about the efficient consumption of electricity. The monitoring and control of electrical devices proposes to contribute with the energy management in residences, bringing comfort and sustainability to the user. The present work presents the development of a prototype for home automation capable of monitoring and controlling electrical devices through a cloud application. Based on the Internet of Things (IoT) system, this device is capable of measuring the current and alternating voltage of an equipment connected to the electrical grid and also calculate the approximate of the power consumed, in addition to its activation. The user interface is performed using the ThingSpeak application, where the device data is monitored, stored and exported to a worksheet where the energy consumed is calculated. In the application there are also two On and Off buttons capable of triggering the load. A remote load trigger test and sensor tests were performed using a halogen lamp and then a desktop. After the tests, it was verified the viability of the system allowing its use for remote energy management of devices connected to the alternating energy grid.

Keywords: Internet of Things, ThingSpeak, Energy Measurement, Smart Home.

Lista de ilustrações

Lista de tabelas

Lista de siglas e abreviaturas

API *Application Programming Interface*

A Ampère

ADC *Analog to Digital Conversion*

CA Corrente Alternada

CC Corrente Contínua

IBGE Instituto Brasileiro de Geografia e Estatística

IDE *Integrated Development Environment*

IEEE *Institute of Electrical and Electronics Engineers*

IPCA Índice Nacional de Preços ao Consumidor Amplo

IoT *Internet of Things*

J Joule

MME Ministério de Minas e Energia

M2M Machine-to-Machine

NA Normalmente Aberto

NF Normalmente Fechado

PCI Placa de Circuito Impresso

PIB Produto Interno Bruto

Procel Programa Nacional de Conservação de Energia Elétrica

KWh *kilowatt hour*

REST *Representational State Transfer*

RMS Root Mean Square

S.I. Sistema Internacional de Unidades

W Watts

WiFi *Wireless Fidelity*

V Volt

Sumário

1 Introdução

Novas tecnologias e aplicações em automação predial vêm sendo progressivamente desenvolvidas com objetivo de prover soluções que facilitam a vida das pessoas tornando tão eficiente e proveitosa quanto possível a interação entre elas e o meio ambiente. Para tal, diversos conceitos e aparelhos que antes eram empregados somente no parque industrial, são atualmente empregados em edifícios (SOUZA; NUNES; BIANCHINI, 2016).

A popularização de computadores, *smartphones* e da internet tem contribuído para a crescente difusão de tecnologias. Entretanto, o emprego da inovação em residências tem acontecido de forma lenta. Pode-se perceber que os automóveis possuem muito mais sistemas embarcados do que a construção civil (MURATORI; BÓ, 2016).

A instalação de um sistema de automação residencial ainda é considerado como uma aplicação de alto custo ou então como um sistema difícil de ser implementado em uma habitação já construída. No entanto, o avanço recente em computação em nuvem, análise de dados e dispositivos eletrônicos de baixo custo têm contribuído na utilização de dispositivos provenientes da internet das coisas em residências.

Simultaneamente aos avanços das tecnologias, a população tem-se preocupado com o consumo eficiente e racional da energia elétrica. De acordo com Petry et al. (2010), a conscientização do consumo energético representa um papel fundamental para a cultura sustentável, propiciando a preservação ambiental e a redução do custo com energia elétrica.

O Programa Nacional de Conservação de Energia Elétrica (Procel) coordenado pelo Ministério de Minas e Energia (MME) e executado pela Eletrobras, tem o objetivo de promover a Eficiência Energética mediante ações de combate ao desperdício de energia elétrica e a redução do consumo. Este programa tem alcançado resultados significativos, sendo no ano de 2015 responsável pela economia de cerca de 2,5% do consumo de energia elétrica no País (ENERGIA, 2016).

Petersen et al. (2007) realizaram um estudo sobre como a informação do consumo de energia pode influenciar na sua redução. Para isso, dois grupos de dormitórios foram estudados, sendo que para o primeiro, foram disponibilizados dados do consumo de energia em tempo real. Em comparação, informações dos medidores das concessionárias de energia foram lidos para o outro grupo com frequência de uma vez por semana. O resultado do estudo foi que o grupo que possuía acesso ao consumo de energia em tempo real reduziu o gasto com a energia em 55% enquanto o segundo grupo apresentou uma redução de 31%, demonstrando então que informações sobre consumo de energia auxiliam na redução do

gasto energético.

Diante desse cenário, é notório que a utilização de tecnologias se tornou um grande aliado dos seres humanos, podendo ser utilizada não só para o conforto pessoal, mas também como forma de auxílio na gestão de energia elétrica. Assim sendo, será apresentado nesse trabalho o desenvolvimento de um sistema de automação sem fio capaz de gerenciar dispositivos conectados à rede elétrica. Esses instrumentos são conectados à internet, permitindo que o usuário acesse os dados e controle-os de qualquer lugar do mundo a qualquer hora.

1.1 Motivação

Nos últimos anos o consumo de energia elétrica no Brasil vem crescendo devido ao aumento de habitantes e também do consumo per capita. De acordo com Sistema-ONS (2017) a projeção de demanda da energia elétrica do setor residencial no período de 2017 a 2026 representa uma taxa média anual de 3,9%.

A energia elétrica tem pressionado o aumento da inflação devido ao aumento do seu custo. Segundo o Instituto Brasileiro de Geografia e Estatística (IBGE) (2018), em Julho de 2018 a prévia do Índice Nacional de Preços ao Consumidor Amplo (IPCA) chegou a 1,11%, puxado pela alta do custo da energia do grupo Habitação.

O uso de novas tecnologias que permitem o consumidor a ter informações individuais do consumo de energia dos equipamentos residenciais pode proporcionar um consumo consciente e sustentável, além de permitir uma redução do valor total da conta de energia.

Sistemas inteligentes de consumo de energia residencial são uma resposta interativa em tempo real entre a rede elétrica e o os usuários. Com isso, pode haver uma melhora no serviço de energia elétrica, como também auxiliar no uso inteligente e interativo da eletricidade (LI et al., 2018).

Atualmente, pesquisas envolvendo dispositivos de *Internet of Things* (IoT) estão crescentes. São dispositivos conectados a internet que são capazes de receber e enviar informações, além de atuar de acordo com algum evento.

Diante do exposto, um sistema de automação residencial baseado em IoT torna possível o monitoramento do consumo de equipamentos elétricos residenciais, propiciando ao usuário tomar medidas de acordo com as informações coletadas. Além disso, traz comodidade, pois é possível enviar comandos liga/desliga de forma remota.

1.2 Objetivo

Visa-se com este trabalho a elaboração de um protótipo do sistema de automação residencial capaz de realizar a medição da corrente e tensão de um dispositivo conectado à rede elétrica alternada. De maneira interativa, o usuário tem a possibilidade de visualizar estes dados e enviar comandos liga/desliga para o dispositivo por meio da aplicação ThingSpeak. Isto é possível utilizando um computador, *smartphone* ou *tablet* conectado à internet.

1.2.1 Objetivos específicos

Os objetivos específicos do trabalho são:

- Revisão da literatura de sistemas comerciais de monitoramento e gestão de energia;
- Revisão da literatura sobre trabalhos já realizados envolvendo IoT e automação residencial;
- Calibração do sensor de corrente;
- Calibração do sensor de tensão;
- Desenvolver um protótipo baseado em IoT para controle e monitoramento de dispositivos elétricos residenciais;
- Realizar teste de acionamento remoto com o protótipo;
- Realizar testes dos sensores para monitoramento do consumo de energia;
- Fornecer documentação para trabalhos futuros envolvendo o microcontrolador ESP32, aplicação ThingSpeak e IoT.

1.3 Estrutura do trabalho

O presente trabalho está organizado em 5 capítulos. O Capítulo 1 traz algumas vantagens do emprego de tecnologias em residências e o seu benefício na utilização para monitoramento de energia. Este capítulo também descreve o objetivo do trabalho, que é propor um sistema de automação residencial sem fio capaz de enviar comandos e receber informações de dispositivos residenciais. O Capítulo 2 apresenta uma revisão sobre os *hardwares* e *softwares* utilizados nas etapas de desenvolvimento do trabalho, como o microcontrolador ESP32, o sensor de corrente ACS712 e o módulo relé que foram os equipamentos motivadores da realização deste projeto. Dentre outros *softwares* revisados, destaca-se a aplicação ThingSpeak responsável pela interface gráfica com o usuário.

O Capítulo 3 expõe o detalhamento do desenvolvimento para a confecção do protótipo apresentado neste trabalho. São explicadas as calibrações dos sensores, apresentado a arquitetura do sistema e do *firmware*, o *layout* da Placa de Circuito Impresso e por fim a configuração do ThingSpeak. Os testes realizados são mostrados no Capítulo 4 e as considerações finais do projeto se encontram no Capítulo 5.

2 Revisão Bibliográfica

Este capítulo tem por finalidade apresentar a revisão da bibliografia relacionada aos equipamentos e softwares envolvidos no sistema, bem como expor os conceitos teóricos necessários para a compreensão do sistema proposto.

2.1 Automação Residencial

Com a abertura do mercado brasileiro na década de 90 para o mercado da informática e da telecomunicação, foi possível a popularização de diversas tecnologias de controle e serviços de automação. Nesse sentido, a automação predial se desenvolveu com o objetivo de melhorar o estilo de vida dos ocupantes de uma edificação, tornando o ambiente mais confortável, seguro e eficiente (BORGES; DORES, 2010).

Automação residencial, ou domótica, consiste em um sistema que por intermédio de um toque é possível gerenciar diversos dispositivos físicos e obter informações a respeito da residência. Esses sistemas automatizados, geralmente, possuem uma central de controle que verifica e possibilita a comunicação entre sensores e equipamentos presentes na habitação (SIMPLÍCIO; LIMA; SILVA, 2018).

A utilização desses sistemas em residências pode ser encontrada há algum tempo em câmeras de câmeras e alarmes de segurança, portões eletrônicos, sensores de presença, controle de acesso por meio de biometria, entre outros. Assim, essa tecnologia além de permitir o controle da residência de forma remota, pode proporcionar uma sensação de conforto, segurança e economia.

Para algumas pessoas, a automação predial trata-se apenas de equipamentos de iluminação automatizada. Para outras, representa novas descobertas, desafios e oportunidades (BOLZANI, 2007).

2.2 Sistemas Embarcados

Os sistemas embarcados podem ser encontrados em diversos dispositivos eletrônicos que engloba diversas áreas, como por exemplo: equipamentos de segurança, sistemas de iluminação, fechadura eletrônica, mecanismos de alarmes entre outros. De acordo com Cunha (2007), sistemas embarcados é definido como um circuito integrado com capacidade computacional que realiza apenas uma tarefa pré-determinada. A interação

homem-máquina com esses dispositivos é realizada por intermédio de interfaces projetadas para tal.

Basicamente, o sistema embarcado é composto por um microcontrolador capaz de realizar leituras de sensores externos, fazer o processamento desses dados, armazená-los, realizar tarefas e enviar comandos para os atuadores (CUNHA, 2007). Um esquemático mostrando esse sistema pode ser visto na Figura 1.

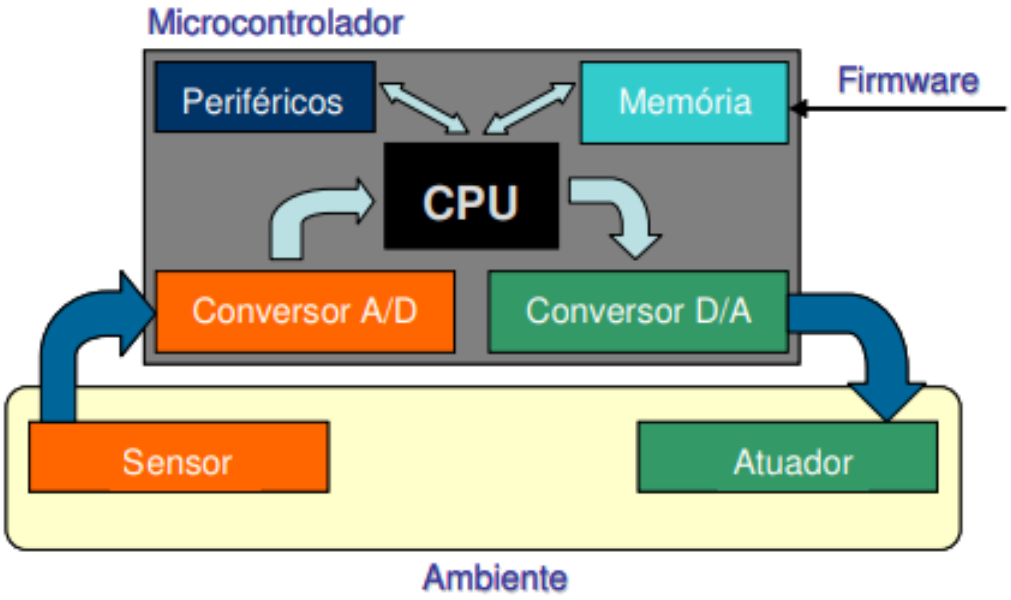


Figura 1: Diagrama básico de um sistema embarcado.

Fonte: CHASE, 2007.

O mercado de sistemas embarcados é atrativo devido ao fato de que quase todos os equipamentos conectados a eletricidade possuem algum sistema computacional embutido. Entretanto, alguns aspectos como custo, desempenho e o tempo de desenvolvimento torna esta área desafiadora, principalmente quando envolve inteligência artificial, tomada de decisões e processamento de sinais (BARROS; CAVALCANTE, 2010).

2.3 Módulo ESP32

O ESP32 é um microcontrolador fabricado pela empresa *Espressif Systems* e que possui alta performance, baixo consumo de energia, conexão *Wireless Fidelity* (WiFi) padrão 802.11 b/g/n e *Bluetooth*. Além disso, o chip possui 4 MB de memória flash, CPU *dual-core*, 448 Kbytes de memória ROM e 520 Kbytes de memória RAM, 36 portas, das quais 18 podem ser utilizadas como conversor analógico digital de 12 bits. A sua tensão de operação é de 3.3V (SYSTEMS, 2019).

O NodeMCU-32S é a plataforma de desenvolvimento que contém o microcontrolador ESP32. Esta placa é composta pelo chip ESP32, interface usb-serial, antena embutida e um regulador de tensão, podendo ser alimentada por uma tensão mínima de 4,5V e máxima de 9V.

A seguir têm-se a ilustração da plataforma de desenvolvimento e também as especificações de cada pinos (Figura 2).

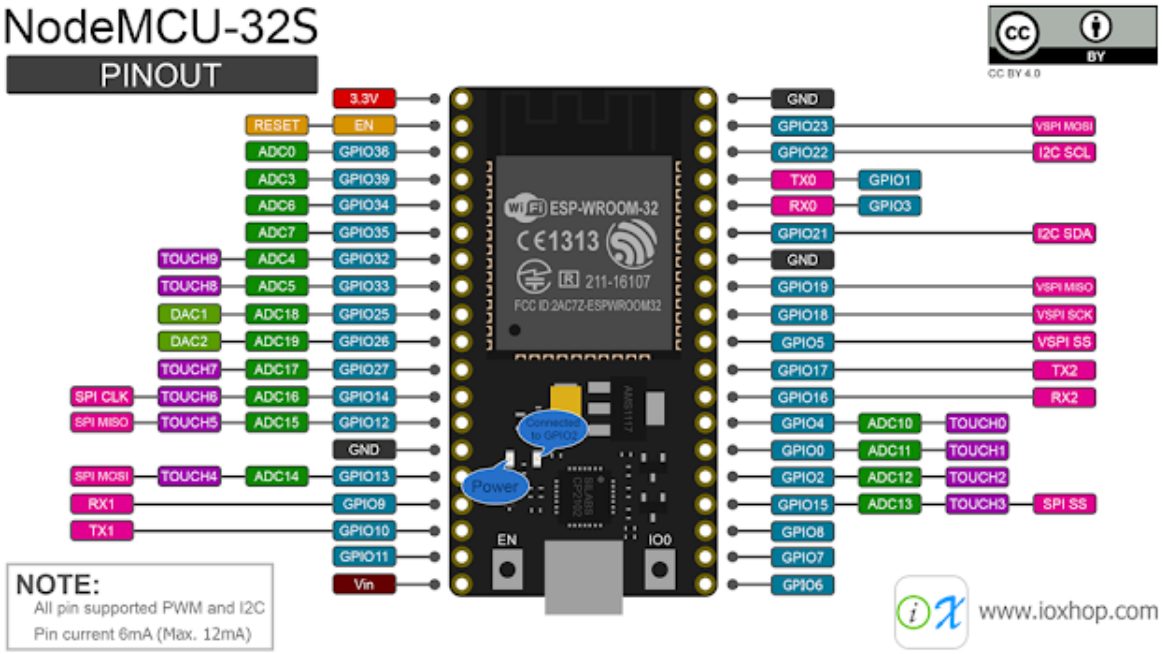


Figura 2: Plataforma de desenvolvimento NodeMCU-32s.

Fonte: KOYANAGI, 2018.

Por apresentar um *firmware* embarcado, o NodeMCU-32S facilita a programação do chip uma vez que o programa pode ser carregado no chip por meio de um cabo mini - USB. Ainda, o microcontrolador pode ser programado utilizando a *Integrated Development Environment* (IDE) do Arduino, por meio da inclusão da biblioteca do ESP.

2.3.1 Conversor Analógico Digital

O conversor analógico para digital, conhecido como *Analog to Digital Conversion* (ADC), é capaz de receber um nível de tensão e reproduzi-lo digitalmente. A resolução do ADC do ESP é de 12 bits e pode receber níveis de tensão na faixa de 0 a 3,3V representando uma escala digital de 0 a 4095 ($2^{\text{resolução}}$-1).

2.3.2 Porta Digital

A porta digital é um sistema binário definido como nível lógico que pode ser configurada como entrada ou saída. No primeiro caso, a porta retorna um dígito binário relacionado a tensão aplicada no pino. Já no segundo caso, é possível alterar a tensão de saída no pino de acordo com o dígito binário escrito.

2.4 IDE Arduino

A IDE Arduino, ilustrado na Figura 3, é um *software open source* utilizado no desenvolvimento da programação de qualquer placa Arduino, e também de alguns outros microcontroladores, tais como os da família ESP. Este ambiente possui uma gama de exemplos de códigos para leitura de dados dos sensores e botões, acionamentos de portas digitais e comunicação serial.

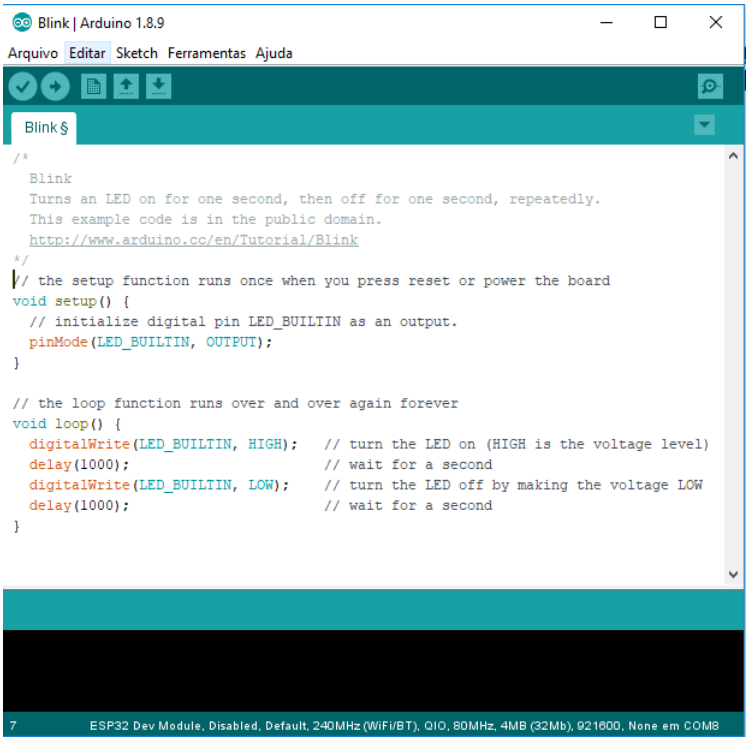


Figura 3: Arduino IDE

Este ambiente de desenvolvimento contém um editor de texto para escrita do código, uma caixa de mensagem na parte inferior onde são apresentados o status do programa, serial monitor e uma barra de ferramentas com uma série de menus e funções.

O uso do software citado é simples, primeiramente é necessário realizar a configuração do modelo da placa a ser utilizada, depois seleciona-se a porta serial virtual do dispositivo conectado ao computador e por fim transfere-se o programa para o hardware por meio do botão carregar (ARDUINO, 2015).

Esta plataforma se destaca devido a quantidade de material existentes para consulta, bibliotecas desenvolvidas que facilitam a programação e exemplos de aplicações, tornando simples a sua utilização. Ademais, possui um serial monitor que permite o monitoramento da comunicação serial possibilitando ao desenvolvedor, identificar dados de sensores ou atuações do microcontrolador. Em consequência destas variedades de benefícios, esta IDE foi escolhida para o desenvolvimento do *firmware*.

2.5 Internet das coisas

O conceito da internet das coisas, derivado do termo em inglês IoT, teve origem em uma apresentação do pesquisador Kevin Ashton em 1999. De acordo com o cientista, se os computadores pudessem coletar dados sem a ajuda de seres humanos, seria possível rastrear e coletar informações de todas as formas. Saberíamos quando algo precisa ser trocado ou consertado e assim, reduziríamos custos e desperdícios, acreditando que a *Internet of Things* tem o potencial de transformar o mundo semelhantemente da internet (ASHTON, 2009).

Atualmente, é possível observar objetos conectados a rede interagindo entre si, com uma central ou com pessoas. Esses dispositivos são capazes de coletar informações que podem ser disponibilizadas de forma gráfica para outros indivíduos analisarem. Além disso, alguns desses objetos podem ser controlados de forma remota, ou ainda realizar ações automaticamente tendo como base os dados coletados.

Em 2014, a *Institute of Electrical and Electronics Engineers* (IEEE), organização global de engenharia profissional, criou uma iniciativa em conjunto com indústrias, governos e academia com o objetivo de gerar um padrão para o IoT (CHEBUDIE; MINERVA; ROTONDI, 2014). Este padrão tem como visão os seguintes pontos:

- aceleração do crescimento do mercado de IoT, permitindo a interação, unificação e compatibilidade de sistemas IoT;
- definição de uma arquitetura de IoT;
- aumento da transparência das arquiteturas
- redução da pulverização de dispositivos industriais;
- impulsionar o desenvolvimento dos trabalhos existentes.

Atualmente o IEEE está considerando a arquitetura, demonstrada na Figura 4, como padrão para dispositivos IoT.

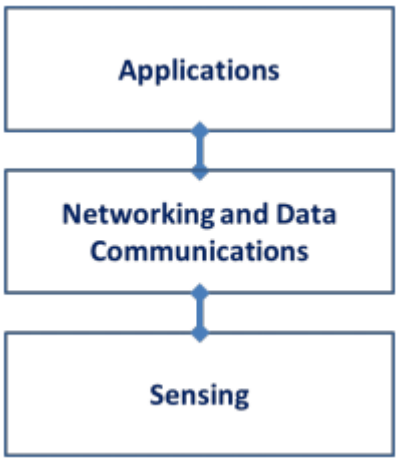


Figura 4: Arquitetura de três camadas para IoT.

Fonte: CHEBUDIE; MINERVA; ROTONDI, 2014.

A padronização da definição das tecnologias baseadas em IoT permite o seu avanço, viabilizando troca de informações entre vários dispositivos conectados à internet, além da interação com diversos usuários. Diante disso, uma gama de possibilidades surgem, como por exemplo: cidades inteligentes, saúde e casas inteligentes. Entretanto, novos desafios também surgem como: regulamentações, padronizações e segurança (SANTOS et al., 2016).

A IoT pode impactar a economia e produtividade de países proporcionando uma melhoria na qualidade de vida das pessoas, trazendo conforto, segurança e saúde. No Brasil, a IoT está em fase de crescimento. Estima-se que essa tecnologia até 2025, pode acarretar uma receita próxima de 120 bilhões de reais ao seu Produto Interno Bruto (PIB), por meio de incentivos e investimentos do governo na área (PEREIRA; CARVALHO, 2017).

2.6 Padrões de comunicação sem fio

Atualmente, a comunicação sem fio está altamente presente no ambiente, visto que a população possui diversos dispositivos conectados entre si sem a necessidade de cabos. Existem diferentes padrões de comunicação sem fio, tais como *Bluetooth*, WiFi e ZigBee. O padrão WiFi é uma das mais utilizadas para aplicações envolvendo internet das coisas.

2.6.1 Wireless Fidelity (WiFi)

A conexão WiFi é um protocolo de comunicação baseado no padrão IEEE 802.11 que possui uma diversidade de capacidade e cobertura de dispositivos além de ter um

custo baixo. Isto possibilitou a sua vasta utilização em diversos cenários que engloba redes 2G/3G e a rede residencial (RUBINSTEIN; REZENDE, 2002).

A Wi-fi *Alliance* ® é uma organização mundial que valoriza e torna a tecnologia WiFi a mais utilizada do mundo. É responsável por definir tecnologias e programas WiFi inovadores defendendo globalmente regras de uso justo. Hoje em dia encontram-se mais dispositivos WiFi em uso do que a população, além de que mais da metade do tráfego da Internet é realizado por redes WiFi (ALLIANCE, 2018).

Este padrão de comunicação sem fio é flexível, suportar diversos dispositivos conectados, possui segurança e controle de acesso. Assim, o WiFi foi escolhido para este trabalho por apresentar essas características citadas, além de estar presente na maioria das residências. Além disso, o microcontrolador ESP32 possui conectividade WiFi.

2.6.2 *Bluetooth*

Bluetooth é uma tecnologia de comunicação wireless de curto alcance projetado para substituir os cabos conectando dispositivos portáteis e/ou fixos. As características chave da tecnologia Bluetooth são a robustez, baixo consumo e baixo custo.

Basicamente existem duas topologias de conectividade que são definidas como *piconet* e o *scatternet*. Uma *piconet* é formado por um dispositivo Bluetooth servindo como mestre e um ou mais dispositivos servindo como escravos. Escravos podem se comunicar apenas com seu mestre em um ponto-a-ponto mediante o controle do mestre. Um dispositivo escravo pode estar em modo ativo ou em espera, sendo o último utilizado para reduzir o consumo de energia. Um *scatternet* é uma coleção de *piconet* que se sobrepõem no tempo e no espaço, sendo que duas *piconet* podem ser conectadas para formar uma *scatternet*. Um dispositivo Bluetooth pode participar em várias *piconets* ao mesmo tempo, e um dispositivo em um *scatternet* pode ser um escravo em vários *piconets*, mas dominar em apenas um deles (LEE et al., 2007).

2.6.3 ZigBee

O ZigBee foi desenvolvido para disponibilizar uma rede de baixo consumo, wireless, de monitoramento e controle baseado no padrão IEEE 802.15.4. Tem como principais características o baixo consumo de energia, baixo custo e vazão. Opera na frequência 2.4GHz, porém é capaz de operar em outras duas, 868MHz e 915Mhz (SANTOS et al., 2016).

Esta tecnologia é projetada para uso em aplicações embarcadas que requerem baixa taxa de dados e consumo de energia, tais como plantas industriais, automação predial e a área de saúde.

2.7 ThingSpeak

ThingSpeak é uma plataforma de serviço e *Application Programming Interface* (API) para IoT utilizada para coletar dados e enviar comandos para dispositivos que estejam conectados a internet. Os dados podem ser armazenados, analisados, visualizados em tempo real e também exportados. Para enviar e receber dados, o usuário pode escolher entre o método MQTT e o *Representational State Transfer* (REST). Além disso, possui integração com o *software* MATLAB, Twitter e Twilio. Essa solução permite a criação de protótipos e sistemas de IoT sem a necessidade de criar servidores ou desenvolver software web (THINGSPEAK, 2019). A Figura 5 ilustra o diagrama de comunicação de dispositivos IoT com o ThingSpeak.

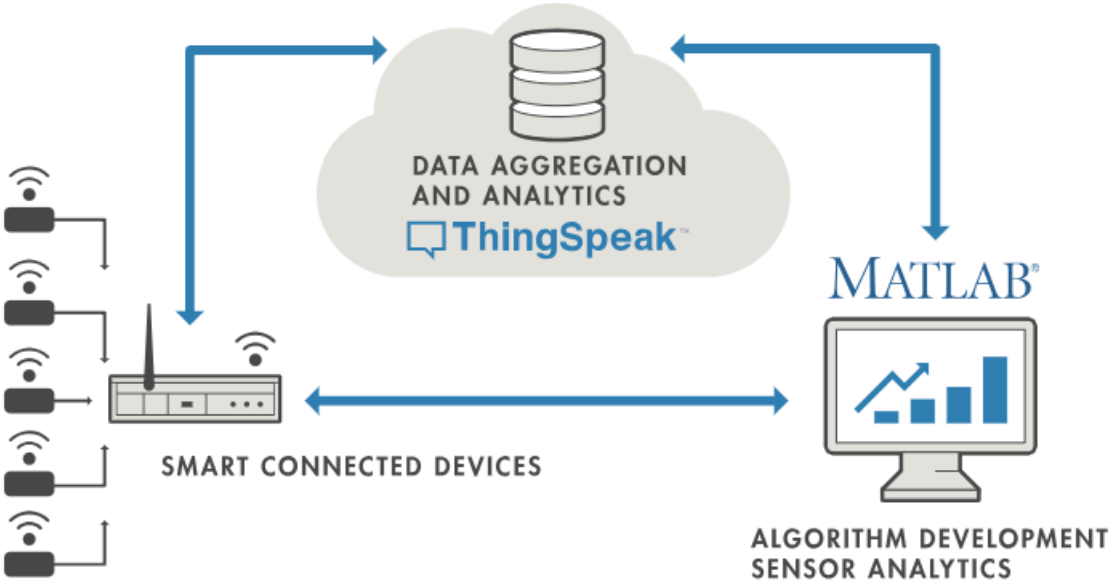


Figura 5: Arquitetura de comunicação de dispositivos com o ThingSpeak.

Fonte: THINGSPEAK, 2019.

O ThingSpeak possui licença gratuita e comercial, sendo que na primeira apresenta limitações. Os usuários podem usar o serviço gratuitamente desde que enviem no máximo três milhões de mensagens por ano e é restrito a quatro canais. Além disso, há um limite de 15 segundos de intervalo para atualizações das mensagens.

2.8 Message Queuing Telemetry Transport - MQTT

Dispositivos IoT precisam trabalhar entre si e com banco de dados, sendo então a conexão a Internet um requisito. Pensando nisso, no final da década de 90 foi desenvolvido pela IBM um protocolo de troca de mensagens entre partes de forma assíncrona, ou seja, comunicação não simultânea (YUAN, 2017).

Esse protocolo de mensagens é leve e flexível projetado para dispositivos que utilizam largura de banda baixa, com alta latência e com requisitos de hardware simples. Esses fundamentos o tornam ideal para dispositivos IoT ou Machine-to-Machine (M2M) conseguindo garantir entrega e confiabilidade (MQTT.ORG, 2019).

2.8.1 Funcionamento

O MQTT implementa um modelo de publicação e assinatura entre um agente central e infinitos clientes, sendo o primeiro um servidor responsável por receber todas as mensagens dos clientes e, posteriormente destiná-las a clientes finais. Basicamente, funciona da seguinte forma: Clientes conectam-se ao *broker* e podem publicar ou receber atualizações de um tópico. O cliente publica informações em um tópico e este as enviam para o *broker*. Seguidamente, o agente central encaminha as mensagens do tópico para todos os clientes que o assinam (YUAN, 2017). A Figura 6 ilustra um modelo de publicação e assinatura do protocolo MQTT.

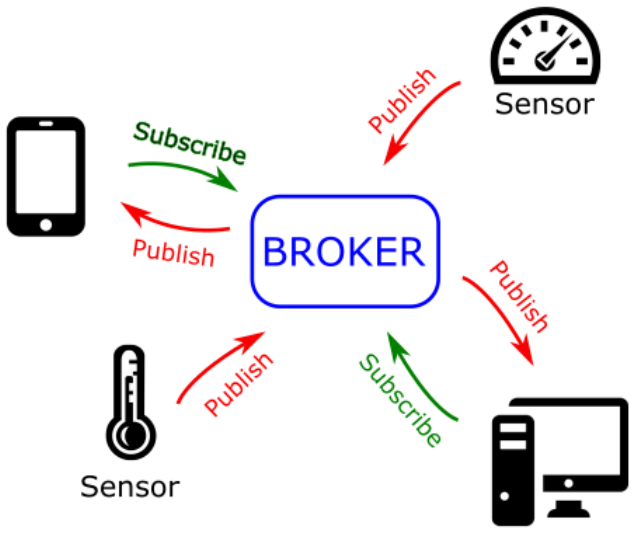


Figura 6: Modelo de publicação e assinatura do protocolo MQTT.

2.9 *Representational State Transfer* (REST)

O REST é um estilo de arquitetura que se comunica por meio de HTTP como um modelo de solicitação e envio de respostas. A implementação do cliente e do servidor podem ser feitas e alteradas de forma independente, desde que o formato de mensagens de envio de um para o outro seja conhecido por ambos. Na sua arquitetura, os clientes enviam solicitações aos servidores e estes enviam as respostas correspondentes (CODECADEMY, 2019).

Existem quatro termos HTTP básicos usados em um sistema REST para interação cliente-servidor (CODECADEMY, 2019):

- GET: resgata uma coleção de recursos ou um recurso específico;
- POST: cria um recurso;
- PUT: atualiza um recurso específico;
- DELETE: remove um recurso específico.

2.10 Valor eficaz de uma onda senoidal

A tensão e a corrente alternada são grandezas que variam ao longo do tempo e possuem uma forma de onda senoidal. Para realizar medições corretas dessas grandezas, é necessário calcular o seu Valor eficaz ou Root Mean Square (RMS) que equivale ao valor caso estas medidas fossem realizadas em Corrente Contínua (CC) (NAKASHIMA, 2007).

Pode-se calcular a potência instantânea dissipada em uma resistência pela seguinte Equação:

$$p(t) = \frac{V^2(t)}{R} = Ri^2(t) \tag{2.1}$$

Onde R é o valor da resistência. Temos que a potência média dissipada é:

$$P_{(\mathrm{A_{ve}})} = \frac{1}{T}\int_0^T p(t).dt = \frac{1}{T}\int_0^T R[i(t)]^2.dt = \frac{R}{T}\int_0^T [i(t)]^2.dt \tag{2.2}$$

Substituindo a Equação 2.3 na Equação 2.4 temos:

$$V_{(\mathrm{RMS})} = \sqrt{\frac{1}{T}\int_0^T [v(t)]^2.dt} \tag{2.3}$$

$$I_{(\mathrm{RMS})} = \sqrt{\frac{1}{T}\int_0^T [i(t)]^2.dt} \tag{2.4}$$

Para obter o valor do sinal discreto, utiliza-se a Equação 2.5

$$I_{(\mathrm{RMS})} = V_{(\mathrm{RMS})} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(X[i])^2} \tag{2.5}$$

Onde, X[i] é o valor da amostragem.

2.11 Potência Elétrica

A Potência é uma grandeza física utilizada para calcular a quantidade de energia que é consumida ou gerada em uma unidade de tempo. Logo, a razão entre energia elétrica recebida por um dispositivo e o tempo necessário para realizar algum tipo de trabalho, é denominado Potência Elétrica e sua unidade de medida é Watts (W). O seu cálculo pode ser realizado pela Equação 2.6.

$$P = VI \tag{2.6}$$

2.12 Energia Elétrica

Energia é a capacidade de realizar trabalho ou produzir uma ação capaz de ser realizada de diferentes formas tais como energia potencial, cinética, mecânica, elétrica, entre outras. A energia elétrica pode ser gerada por meio de outras fontes de energia, por exemplo mediante a energia cinética da queda d'água.

O desenvolvimento humano está fortemente associado ao uso da energia. Segundo Goldemberg e Lucon (2007), no ano de 2003, quando a população mundial era de 6,27 bilhões de habitantes, o consumo médio total de energia era de 1,69 tonelada equivalentes de petróleo per capita.

Apesar de no Sistema Internacional de Unidades (S.I.) a unidade de energia ser o Joule (J), se tratando de consumo da energia elétrica é utilizado o *kilowatt hour* (KWh) e o seu cálculo pode ser realizado mediante a Equação 2.7:

$$E = \frac{P\Delta t}{1000} \tag{2.7}$$

Substituindo a Equação 2.6 na Equação 2.7, tem-se:

$$E = \frac{VI\Delta t}{1000} \tag{2.8}$$

Onde, E é a energia elétrica em KWh, V a tensão aplicada no equipamento em Volt (V), I a corrente do equipamento em Ampère (A) e Δt a variação do tempo em horas.

2.13 Sensor de Corrente

Para a medição da corrente elétrica foi escolhido o sensor de corrente invasivo ACS712 (Figura 7). Fabricado pela Allegro MicroSystems, é um sensor preciso e econômico

capaz de medir tanto corrente alternada quanto corrente contínua. O *datasheet* do sensor informa a CC suportada, logo para encontrar qual é Corrente Alternada (CA) máxima é necessário dividir este valor pela raiz quadrada de 2 que corresponde a grandeza utilizada para encontrar o valor eficaz de uma onda senoidal (FILIPEFLOP, 2019).

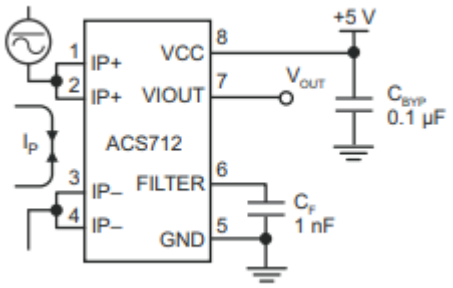


Figura 7: Sensor de Corrente ACS712.

Fonte: MICROSYSTEMS, 2007.

Existem 3 modelos deste sensor com limites de medição de corrente e sensibilidades distintas, como demonstrado na Tabela 1.

Tabela 1: Tabela das características dos modelos do sensor de corrente ACS712.

Modelo	**Corrente DC (A)**	**de Corrente AC (A)**	**Sensibilidade (mV/A)**
ACS712ELCTR-05B-T	-5 a 5	-3,54 a +3,54	185
ACS712ELCTR-20A-T	-20 a 20	-14,14 a +14,14	100
ACS712ELCTR-30A-T	-30 a 30	-21,21 a +21,21	66

Fonte: Adaptado de MICROSYSTEMS, 2007.

O funcionamento do sensor se baseia no efeito Hall que é quando a corrente que flui por um caminho de cobre (pinos 1 e 2 aos pinos 3 e 4) gera um campo magnético e este induz uma tensão proporcional a este campo (pino 7) (MICROSYSTEMS, 2007).

Os terminais do condutor de cobre apresentam um isolamento elétrico dos condutores do sensor (pinos 5 a 8), permitindo que possa ser empregado em circuitos sem o uso de um isolamento externo (MICROSYSTEMS, 2007).

2.14 Relé

Para realizar os comandos liga e desliga dos dispositivos conectados a rede elétrica, optou-se pela utilização do dispositivo eletromecânico, relé, que é capaz de operar cargas

em corrente alternada e proporcionar o isolamento entre cargas de corrente alternada e o sistema de controle em corrente contínua. A Figura 8 ilustra o circuito do dispositivo relé.

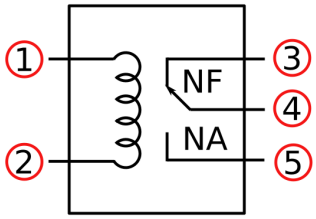


Figura 8: Circuito interno Relé.

O seu funcionamento é baseado no princípio eletromagnético e se da seguinte forma:

- A bobina interna (pinos 1 e 2) é alimentada por uma corrente criando um campo magnético que atrai a chave;
- O contato Normalmente Aberto (NA) é fechado permitindo a passagem da corrente entre o pino 5 e o terra (pino 4);
- O contato Normalmente Fechado (NF) é aberto cortando a passagem da entre o pino (3) e o terra;
- Quando a corrente na bobina é interrompida, a indução do campo magnético é extinto, consequentemente os contatos voltam para os seus estados naturais.

  A corrente fornecida pelo pino digital do ESP32 não é capaz de acionar um relé. Portanto, é necessário acrescentar um circuito com transistor, diodo e resistor para amplificar esta corrente. Atualmente, existem módulos relé (Figura 9) que já possuem esta configuração embutida em uma placa contendo o relé e bornes para ligação da carga a ser controlada, facilitando a conexão com o microcontrolador. No presente trabalho, o módulo relé empregado suporta o acionamento de uma carga com máxima tensão e corrente alternada iguais a 125V e 10A, respectivamente.

Figura 9: Circuito interno Relé.

Fonte: FILIPEFLOP, 2019.

2.15 Sistemas comerciais de monitoramento e gestão de energia

Sistemas de monitoramento e gestão de energia permitem que os usuários tenham o conhecimento e controle do seu consumo de energia elétrica. Inicialmente, esses sistemas eram desenvolvidos apenas para grandes consumidores, como industrias e comércios. Podemos ter como exemplo as soluções propostos pelas empresas Viridis Soluções em Energia S.A e Engie.

A Viridis é uma empresa dedicada a desenvolver soluções de controle e gestão de energia exclusivamente para grandes consumidores de energia. Ela disponibiliza um software com ferramentas de inteligência artificial e engenharia que propõe automatizar e auxiliar equipes de eficiência energética das empresas. Assim, é possível aumentar o controle sobre os processos em tempo real (VIRIDIS, 2019).

A Engie é uma empresa que oferece soluções e equipamentos para controle, administração e redução de custos com energia. Uma de suas soluções é uma ferramenta denominada *Follow Energy* que é capaz de gerenciar a energia elétrica e utilidades via internet. Ela permite o monitoramento remoto, notificações de irregularidades, acompanhamento gráfico e estatístico dos dados e também, gera relatórios com todas as informações coletadas. Esta ferramenta é voltada para utilização em empresas de todos os segmentos de atividade (ENGIE, 2019).

Com a popularização e redução de custos da IoT e o aumento da preocupação em relação ao consumo de energia, soluções de gestão de energia elétrica se espalharam para as aplicações domésticas. Atualmente, já é possível encontrar no mercado esses sistemas voltados para residências.

A empresa Sense propõe uma solução de gestão de energia voltada para residências. Trata-se de um dispositivo instalado no quadro de distribuição de energia elétrica da residência que é capaz de medir a corrente e a tensão. A tecnologia usa esses dados para detectar pequenas alterações na rede elétrica o que permite identificar padrões de consumo. Dessa forma, o dispositivo auxilia a família no consumo eficiente da energia elétrica (SENSE, 2019).

A empresa chinesa ITEAD Intelligent Systems Co.Ltd é repensável pelo desenvolvimento de *hardware* inteligentes para usos domésticos. Sua principal linha de produção são os produtos Sonoff. Anteriormente, a empresa desenvolveu dispositivos IoT voltados para o acionamento remoto de cargas conectadas à rede elétrica. Porém, atualmente, a empresa desenvolveu um dispositivo apto a realizar o acionamento remoto de cargas e também monitorar o seu consumo de eletricidade (CO.LTD, 2017).

2.16 Revisão da literatura de trabalhos relacionados

Os seguintes projetos se destacaram por proporem a utilização de tecnologias para automação residencial que se assemelham ou possuem características próximas à proposta por este trabalho.

Wasoontarajaroen; Pawasan e Chamnanphrai (2017), apresentaram um sistema de monitoramento de energia integrado ao conceito de IoT, para utilização em um edifício. Para isto, eles utilizaram o módulo sensor PZEM-004t, um microcontrolador Arduino Nano Mini e um microcontrolador ESP8266. O módulo sensor foi utilizado para medir a tensão, corrente, potência e energia de um sistema trifásico. O Arduino nano assumiu o papel de ler os resultados medidos pelo módulo e salvá-los a cada 15 segundos. A cada um minuto, esses resultados eram enviados para o ESP8266, por meio da comunicação serial, para então serem enviados para a nuvem ThingSpeak.

Este sistema foi testando durante uma semana e então, através dos dados obtidos, concluíram que o sistema foi satisfatório, uma vez que produziram dados confiáveis. Apesar do projeto apresentado ter apresentado resultados satisfatórios para o monitoramento energético, a utilização de apenas um microcontrolador seria mais eficiente, além de diminuir o custo total do projeto. Além disso, no trabalho, não é citado se houve calibração do módulo PZEM-004t, para a aquisição da energia, o que pode afetar na exatidão da aquisição dos dados.

Nguyen et al. (2018), propuseram um protótipo de um dispositivo que possa reconhecer o tipo de eletrodoméstico conectado a ele de acordo com a tensão, corrente e fator de potência. Eles utilizaram o módulo PZEM-Z004T, para detectar a tensão, corrente, potência ativa, energia e fator de potência do dispositivo que esteja conectado ao sistema. O microcontrolador ESP8266 foi responsável por realizar a aquisição destes dados e enviá-los para nuvem. Os autores realizaram testes para comparação com os valores encontrados com um multímetro comercial. A partir disso, encontraram medições com erros abaixo de 1%. Porém, para aparelhos de baixa corrente, o erro encontrado foi de 10%.

O protótipo apresentado pode auxiliar no consumo eficiente de energia elétrica. Porém, aparelhos que ficam operando no modo *stand-by*, normalmente, apresentam baixa corrente. Como o protótipo exposto, apresenta um alto erro para esses dispositivos, isso pode interferir na gestão do consumo elétrico desses equipamentos. Além disso, os autores não citaram se fizeram calibrações com o módulo PZEM-Z004T.

Pereira (2018),utilizou como tema de monografia para obtenção de grau de Engenheiro de Controle e Automação na Universidade Federal de Ouro Preto, o desenvolvimento de um sistema capaz de monitorar e controlar um dispositivo elétrico por meio de um aplicativo. O conjunto realiza a medição da corrente por meio do sensor ACS712ELCTR-

20A-T. O acionamento da carga é realizado por meio de um módulo relé, sendo que o envio dos comandos de forma remota é efetuado utilizando um aplicativo mobile desenvolvido. O microcontrolador ESP8266 é utilizado para aquisição dos sinais e envio dos dados para uma planilha do *Google Docs*. O autor conclui que o protótipo desenvolvido não é ideal para cargas abaixo de 2A, pois o modelo do sensor possue baixa precisão. Além disso, o dispositivo não é capaz de medir a tensão do equipamento. Ainda, o autor não realizou a calibração do sensor de corrente ACS172 e também não fez o cálculo do valor eficaz do seu sinal.

3 Metodologia

Este capítulo tem por finalidade apresentar detalhadamente os procedimentos realizados para atingir o objetivo do trabalho. Também será abordado, a arquitetura do sistema proposto, a aquisição dos sinais dos sensores, suas calibrações e o desenvolvimento do *firmware* e *hardware*. Além disso, será abordado a integração do dispositivo com a aplicação web.

3.1 Arquitetura do sistema

O sistema proposto neste trabalho é baseado em IoT. Basicamente, é composto por um sensor de corrente (ACS712), um circuito para medir a tensão e um microcontrolador capaz de processar os dados e enviar para a aplicação ThingSpeak. Os testes do sistema serão realizados com dois dispositivos, um conectado em uma Lâmpada e outro em uma tomada. A Figura 10 ilustra a arquitetura do sistema.

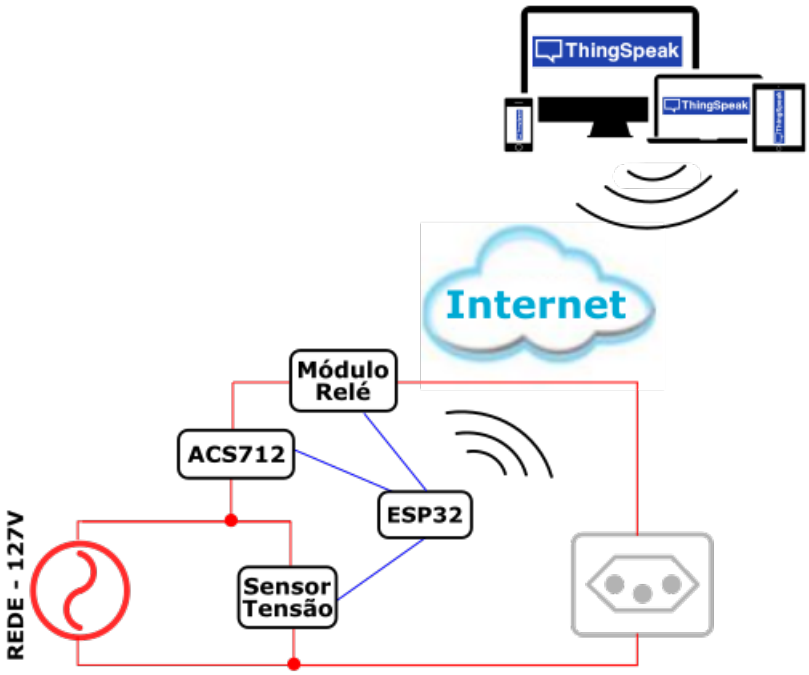


Figura 10: Arquitetura do sistema proposto.

Para medir a tensão, é necessário que o circuito responsável pelo monitoramento da tensão fique em paralelo com a rede de energia alternada. Em relação ao sensoriamento da

corrente, por se tratar de um sensor de corrente invasivo, o ACS712 é conectado em série com a carga para que a medição ocorra. Depois, é conectado em série o módulo relé que é responsável por realizar o controle Liga/Desliga do dispositivo. O microcontrolador EPS32 é ligado a todos esses periféricos sendo responsável por receber comandos, tratar os dados dos sensores e enviá-los para a nuvem. Na aplicação ThingSpeak, os dados armazenados na nuvem, podem ser visualizados e exportados. Também, por meio dessa aplicação, é realizada o acionamento remoto.

3.2 Sensor de Tensão

A tensão do dispositivo é medida por meio da utilização do circuito ilustrado na Figura 11.

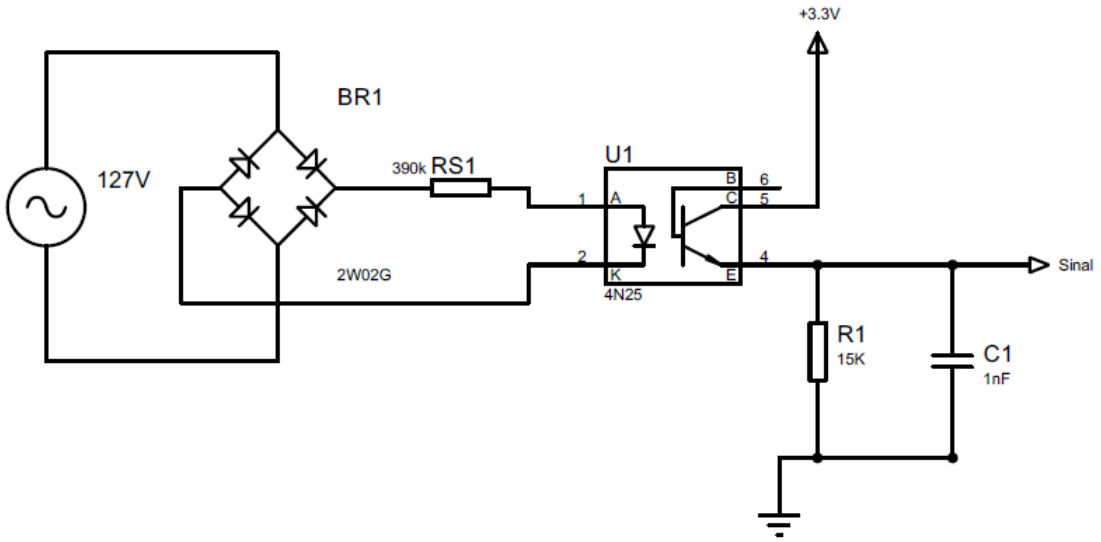


Figura 11: Circuito para medir a Tensão AC.

Primeiramente uma ponte retificadora de onda completa é utilizada para converter o meio ciclo negativo da tensão CA em positivo. Depois é utilizado um resistor para limitar a corrente que segue para o optoacoplador 4N25. Este último, é responsável por isolar o circuito de alta tensão para o de baixa tensão. Por fim, é utilizado um resistor para que o ADC do microcontrolador consiga medir o sinal e também um capacitor para filtrá-lo. Os valores lidos pelo microcontrolador são calculados por um algoritmo que retorna o valor RMS da tensão.

Este circuito para medição de tensão foi escolhido, pois o sinal correspondente a tensão aplicada é dada em forma de onda senoidal, tornando possível o cálculo do valor eficaz. Além disso, proporciona o isolamento do circuito de alta tensão com o circuito de

baixa tensão, propiciando a segurança do microcontrolador contra eventuais surtos da rede CA.

3.2.1 Calibração do Sensor de Tensão

O circuito para a medir a tensão CA possui na saída uma variação de tensão de acordo com a tensão aplicada no circuito. Para saber qual a curva que relaciona a tensão CA e a saída do sensor, foi necessário submeter o sistema a uma variação da tensão com o auxílio de um circuito *dimmer* que varia a tensão Root Mean Square (RMS). Assim, com o auxílio de um multímetro, anotou-se o valor da tensão aplicada no sensor e a saída recebida pelo microcontrolador. Na Figura 12 pode-se visualizar a configuração utilizada.

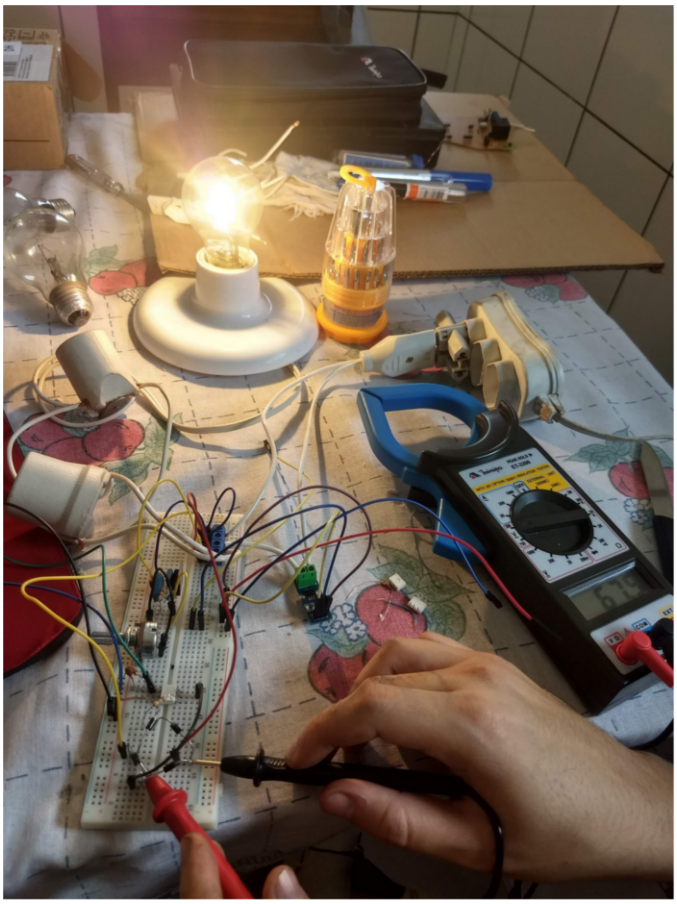


Figura 12: Configuração utilizada para calibração do sensor de Tensão.

De acordo com o teorema de Nyquist, a taxa de amostragem do sinal precisa ser maior que o dobro da frequência deste sinal (BALCH, 2003). A partir disso, o *firmware* utilizado para a calibração foi desenvolvido da seguinte forma: a frequência da rede elétrica no Brasil é igual a 60Hz, que equivale a 16,67ms. Optou-se então, a capturar 1920 amostras durante 1 segundo, correspondendo a uma frequência de 1920Hz, respeitando o teorema de Nyquist. Estes valores são calculados de acordo com a Equação 2.5. Como o ADC do

ESP32 é de 12 bits, o valor máximo que o ADC lê é de 4095 correspondendo a um sinal de 3,3V. A seguir tem-se a parte principal do código utilizado.

```
for (int i = 0; i <= 1920; i++){
      leitura=analogRead(PinoTensao); //aquisição da leitura do ADC
      soma+=leitura*leitura; //somatório do quadrado da leitura
      delayMicroseconds(390); //aguarda 390ms para próxima leitura
   }
   valorADC = sqrt(soma/1920); //faz o calculo do valor eficaz
   valorADC = (3.30*valor)/4096; //Calcula a tensão da saída do sensor
```

Após a aquisição dos valores, foi construído um gráfico (Figura 13) e extraído a linha tendência e a sua função.

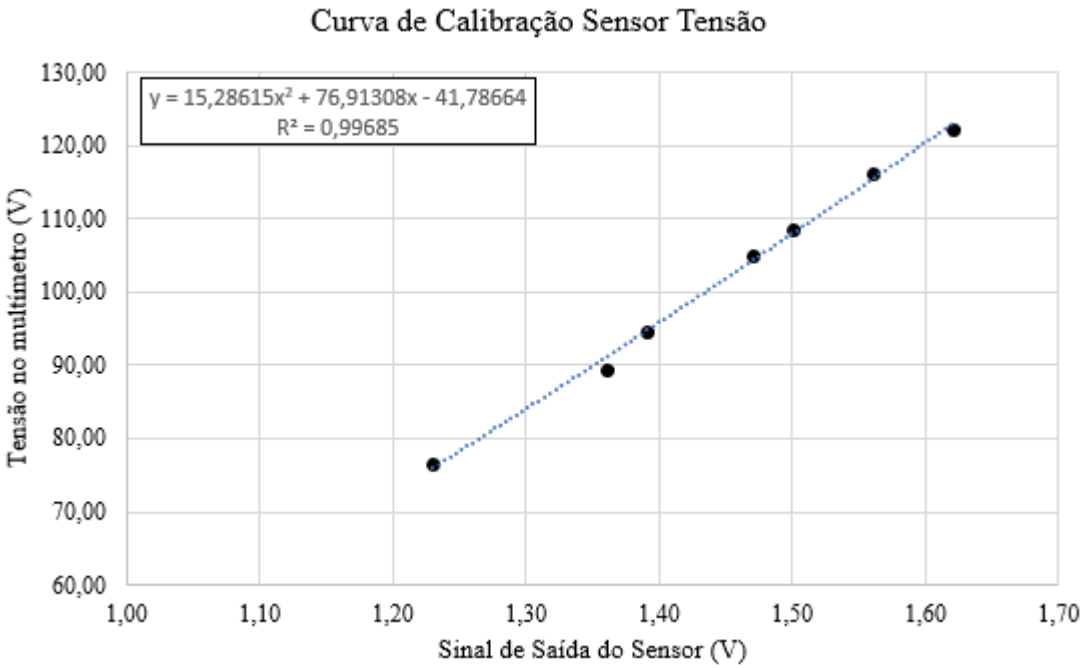


Figura 13: Curva da calibração do sensor de tensão com a linha de tendência.

3.3 Sensor de Corrente

Neste trabalho foi utilizado o módulo do sensor ACS712 com a faixa de leitura ±5A (Figura 14) que simplificadamente representa uma placa de circuito impresso contendo o sensor (1), capacitores necessários para o seu funcionamento (2), bornes para a ligação da corrente a ser medida (3) e por fim, um LED e um resistor que indica que está alimentado por um tensão 5V (4).

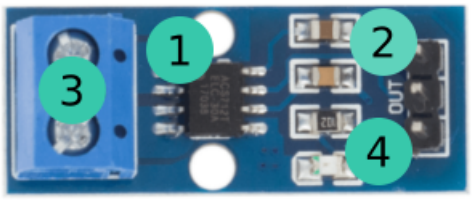


Figura 14: Módulo do Sensor de Corrente ACS712.

Fonte: Adaptado de FILIPEFLOP, 2019.

Por se tratar de um sensor de corrente invasivo, é necessário colocá-lo em série com a carga. A Figura a seguir ilustra a configuração do circuito.

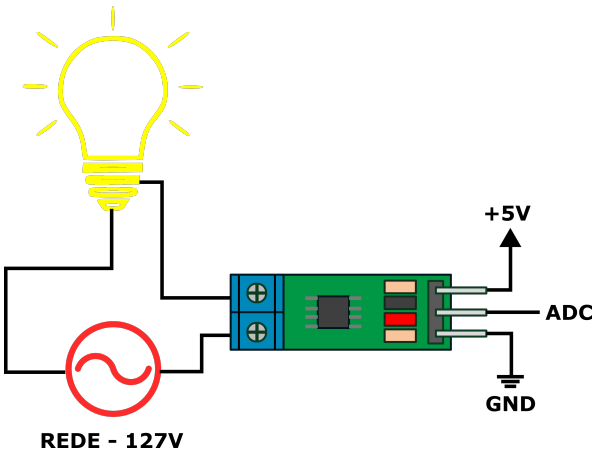


Figura 15: Circuito do sensor de corrente ACS712.

Similarmente ao que foi realizado na calibração do sensor de tensão, é capturado 1920 amostras durante 1 segundo e então calcula-se o valor eficaz da tensão da saída do sensor. Porém, de acordo com o seu *datasheet* a tensão de saída é proporcional à corrente de entrada e possui um *offset* correspondente ao zero do sinal senoidal da corrente. Esta medida é igual a metade da alimentação. Diante disso, para encontrar o valor correspondendo ao *offset*, foi medido o valor ADC sem carga conectada ao sensor. O valor médio encontrado foi igual a 3000. Então, este valor é extraído da leitura ADC.

Por fim, a medida é divida por 0,185 que representa a sensibilidade do sensor (185mv/A). A parte principal do código utilizado pode ser observado a seguir:

```
1 for (int i = 0; i <= 1920; i++){
2       leitura=analogRead(PinoCorrente); //aquisição da leitura do ADC
3       soma+=leitura*leitura - 3000; //somatório do quadrado da leitura
4       delayMicroseconds(390); //aguarda 390ms para próxima leitura
```

```
5     }
6     valorADC = sqrt(soma/1920); //faz o calculo do valor eficaz
7     valorADC = (3.30*valorADC)/4096; //Calcula a tensão da saída do sensor
8     corrente = valorADC/0.185; //Converte a tensão da saída em saída
```

Entretanto, estes valores não corresponderam ao valor medido no alicate amperímetro. Então, com auxílio de 3 lâmpadas, a corrente foi variada e os valores anotados. A configuração utilizada está representada na Figura 16.

Figura 16: Configuração utilizada para calibração do sensor de corrente.

Finalmente, construiu-se a curva de calibração (Figura A) do sensor e obteve-se a equação da linha de tendência com sua respectiva função.

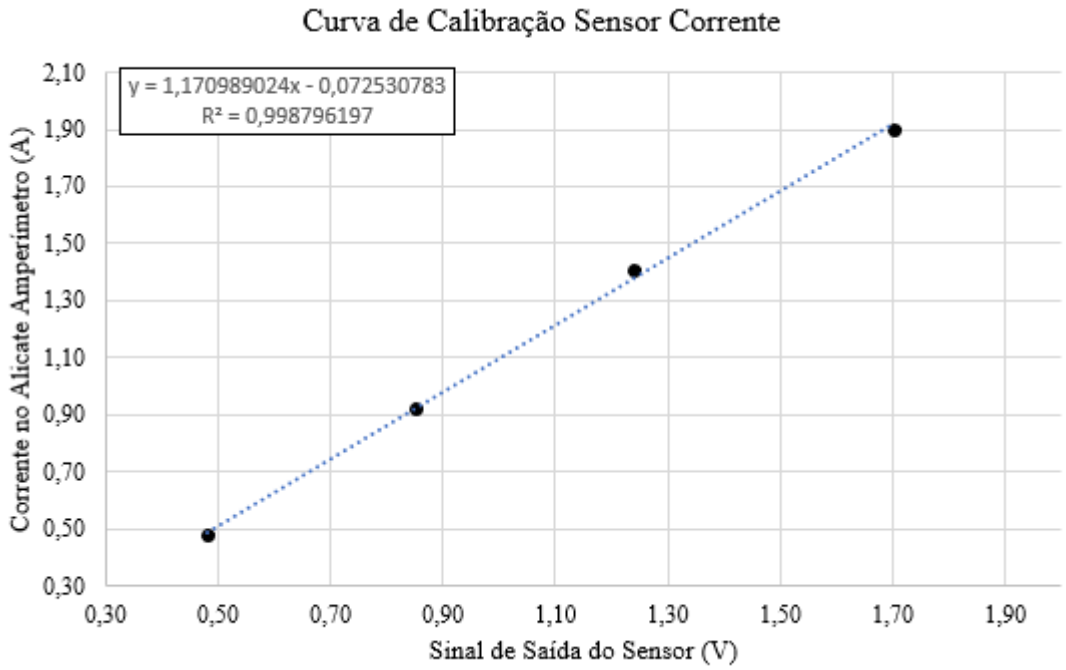


Figura 17: Curva de calibração do sensor de corrente ACS712 - 05B.

3.4 Placa de Circuito Impresso

A placa de circuito impresso foi desenhada utilizando o *software* Proteus, que permite exportar o desenho do circuito para impressão (Figura 18). Nesta placa, encontram-se todas as conexões e componentes necessários para o funcionamento do protótipo. Além disso, incluiu-se um circuito para inserção de um botão e um divisor de tensão para conectar outros modelos do sensor de corrente ACS712, aumentando a gama de aplicações do sistema. O diagrama esquemático dessas conexões se encontra no Apêndice A.

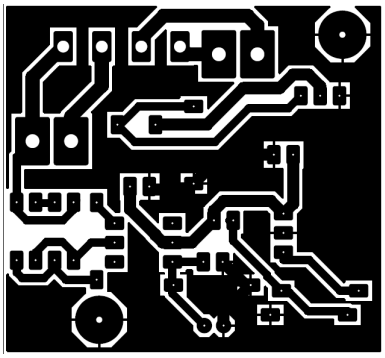


Figura 18: Desenho da PCI exportado do Proteus.

Após a confecção da Placa de Circuito Impresso (PCI), os componentes foram soldados e os módulos conectados como pode ser visto na Figura 19.

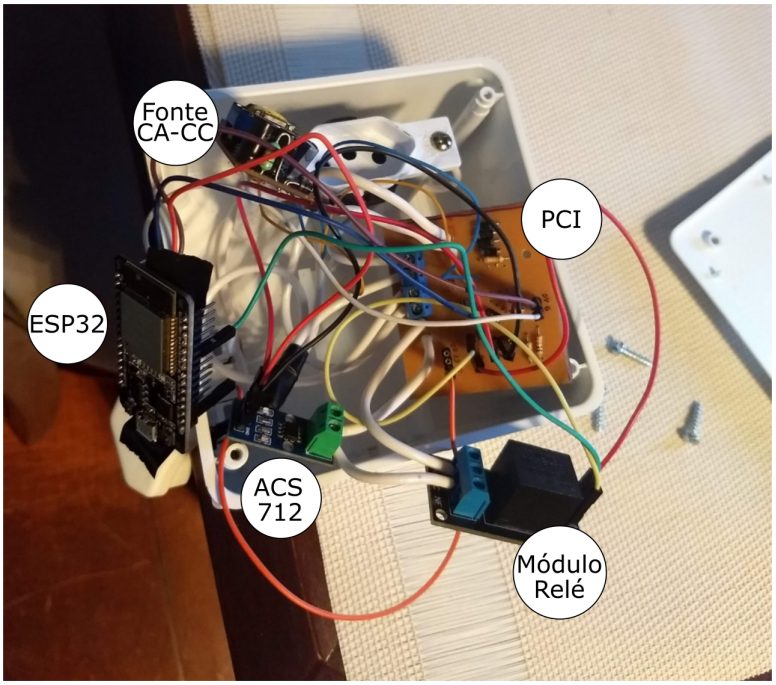


Figura 19: Montagem do protótipo.

Finalmente, a PCI e os módulos foram organizados em uma caixa com tampa tendo como resultado o protótipo ilustrado na Figura 20.

Figura 20: Protótipo final.

3.5 Configuração do ThingSpeak

Para utilizar o ThingSpeak inicialmente é necessário realizar um cadastro no site da plataforma. Após isto, automaticamente o usuário já pode utilizar o sistema no modo gratuito.

Primeiramente, na aba *settings* e em *new channel* cria-se o canal utilizado pelo dispositivo (Figura 21).

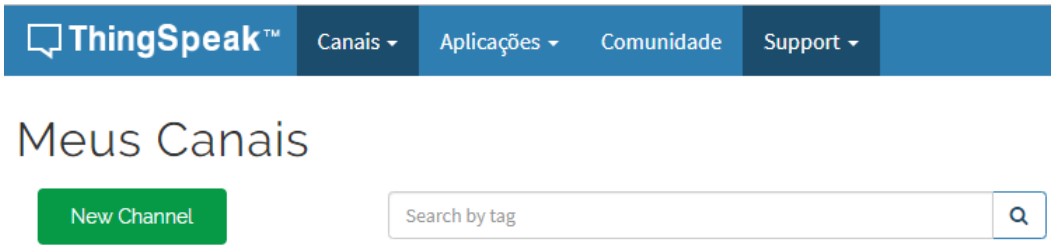


Figura 21: Criação do canal no ThingSpeak.

A seguir, o canal é configurado com cinco campos (Figura 22) que representa a Tensão, Corrente, Potência e Status da carga conectada ao protótipo. Os campos 1, 2 e 3 são publicados pelo ESP32 e o campo 4 é subscrito pelo microcontrolador.

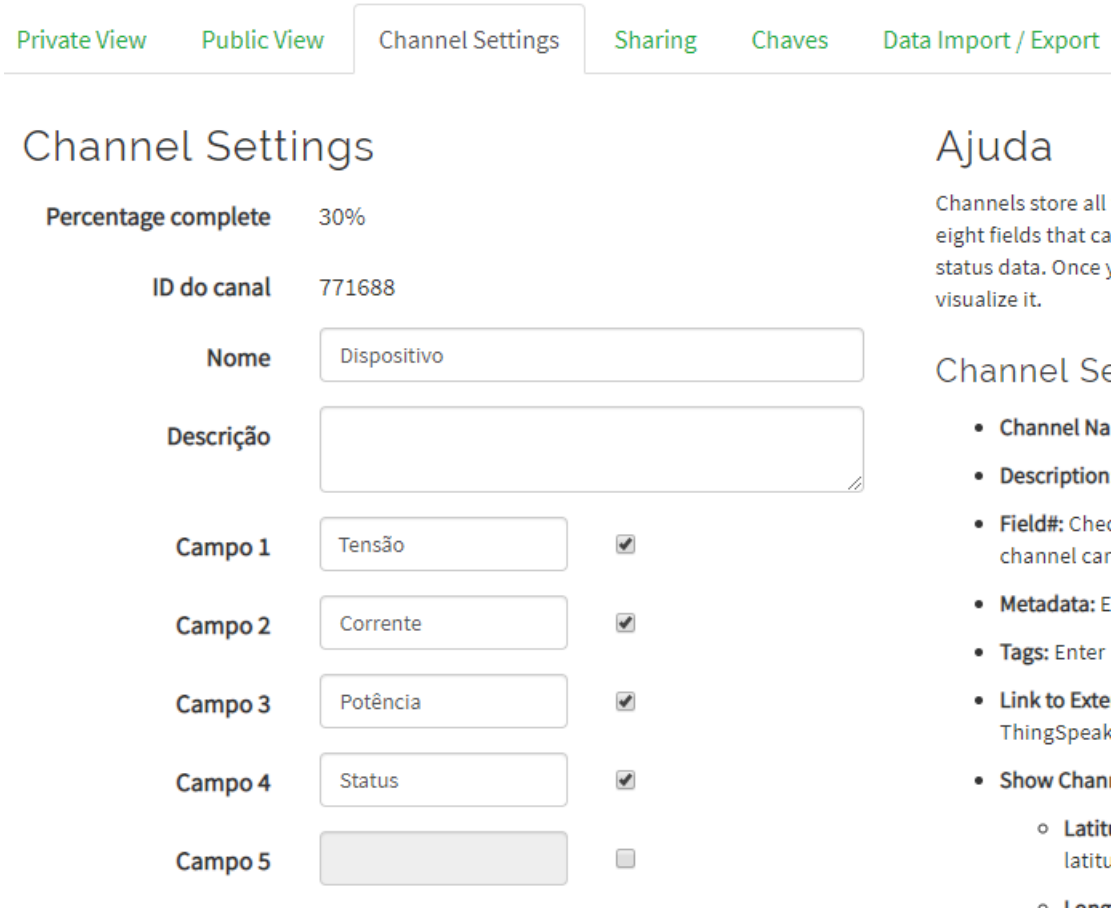


Figura 22: Configuração do canal no ThingSpeak.

As chaves necessárias para ler e escrever dados nos campos do canal são encontradas na aba Chaves. Para enviar comandos de liga e desliga ao campo cinco, é criado um *plugin* que utiliza as linguagens de programação HTML, CSS e Javascript. Para isso, clica-se em aplicações, *plugin* e depois em novo. Obtém-se uma tela com três espaços para os códigos. No espaço para o HTML adiciona-se um código que contém dois botões, Ligar e Desligar, que quando recebem um *click* chamam uma função em Javascript (Figura 23).

HTML

```
<html>
<head>

  <script type="text/javascript" src="//ajax.googleapis.com/ajax/libs/jquery/1.

  %%PLUGIN_CSS%%
  %%PLUGIN_JAVASCRIPT%%

</head>
<body>

  <button onclick="Ligar()">Ligar</button>
  <button onclick="Desligar()">Desligar</button>

</body>
</html>

```

Figura 23: Código em HTLM para o plugin.

O código em CSS é utilizado para dar estilo ao plugin. Porém, neste projeto este recurso não é utilizado. No código em Javascript (Figura 24), são definidas as funções dos botões, Ligar e Desligar, que são responsáveis por enviar 1 (Ligar) ou 0 (Desligar) ao campo cinco do ThingSpeak. Esses dados são enviados utilizando o método GET do REST API.

JavaScript

```
<script type='text/javascript' src='https://ajax.googleapis.com/ajax/libs/jquery/
<script type='text/javascript' src='https://www.google.com/jsapi'></script>
<script type='text/javascript'>
  function Ligar(){
    var url = "https://api.thingspeak.com/update.json?api_key=LVN75X7UEZZNL8CW&fi
    $.getJSON(url, function(data){
      console.log(data);
    });
  }
  function Desligar(){
    var url = "https://api.thingspeak.com/update.json?api_key=LVN75X7UEZZNL8CW&fi
    $.getJSON(url, function(data){
      console.log(data);
    });
  }
</script>
```

Figura 24: Código em Javascript para o *plugin*.

Os códigos completos utilizados para a criação do *plugin* podem ser vistos no Apêndice B. Por fim, são adicionados na aba de visualização privada, o *widget Lamp Indicador* e o *plugin* criado. O resultado desta configuração é visto na Figura 25.

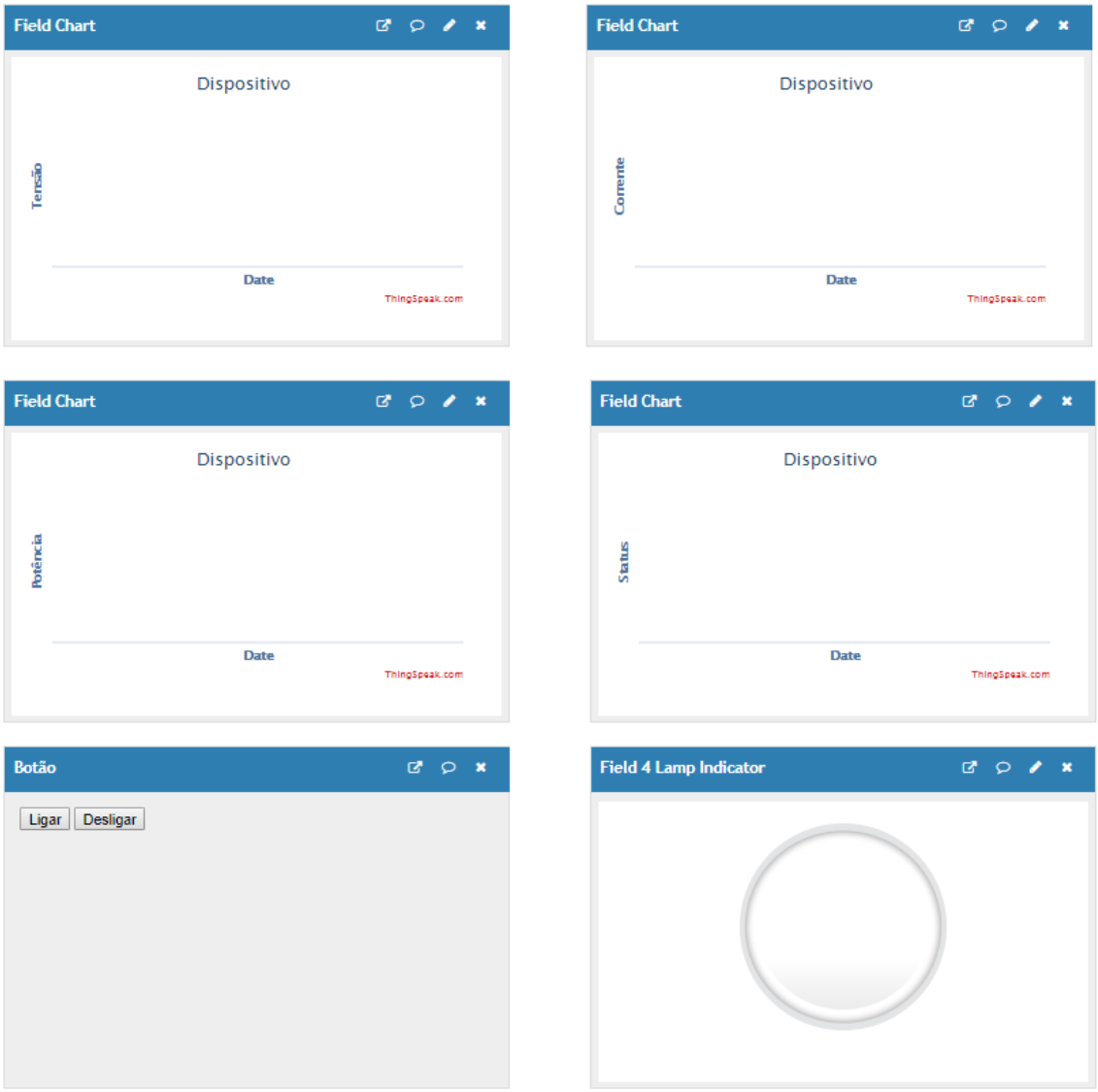


Figura 25: Visualização final da configuração do ThingSpeak.

3.6 Desenvolvimento do *Firmware*

O desenvolvimento do *firmware* para o ESP32 foi realizado na IDE do Arduino previamente descrito no Capítulo 2, seguindo um tutorial da MathWorks, desenvolvedora da plataforma ThingSpeak. O código completo utilizado é demonstrado no Apêndice C. Basicamente o código pode ser divido em três partes:

- 1° parte: processa todas as configurações iniciais necessárias para o funcionamento do *firmware*;
- 2° parte: realiza a conexão a rede WiFi e ao *broker* MQTT;
- 3° parte: faz a publicação e subscrição dos tópicos MQTT.

A Figura 26 expõe um fluxograma do código implementado.

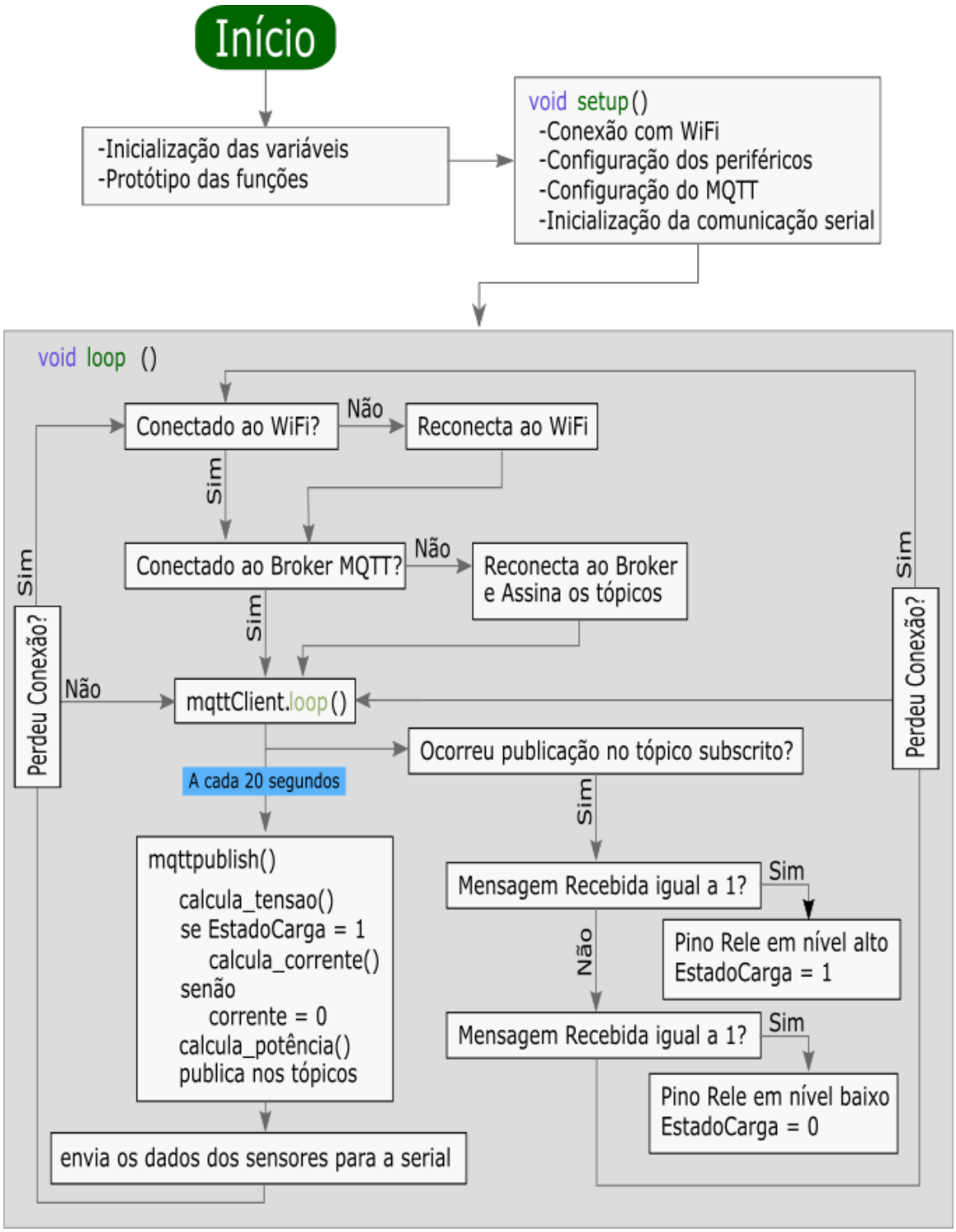


Figura 26: Fluxograma do funcionamento do *firmware*.

4 Testes e Resultados

Neste capítulo serão apresentados e discutidos os principais resultados obtidos com os testes realizados utilizando o protótipo desenvolvido. Os testes foram divididos em duas partes e em ambas o protótipo foi conectado à rede elétrica e a carga conectada ao protótipo.

4.1 Parte 1 - Teste de acionamento

Como teste para os comandos de Liga e Desliga, os botões do *plugin* foram acionados enquanto o microcontrolador também publicava dados nos campos 1, 2 e 3 e subscrevia o campo 4. Percebeu-se que, quando o ESP32 está publicando nos tópicos, existe algum conflito para que os botões do *plugin* escrevam dados, e portanto não eram escritos.

Então, testou-se o acionamento dos botões quando o ESP32 não publicava nos tópicos. O funcionamento ocorreu como esperado, a cada 15 segundos os dados 0 ou 1 são escritos no campo. O intervalo de 15 segundos é necessário, pois corresponde a limitação de tempo para escrita no plano gratuito do ThingSpeak.

Com o auxílio da ferramenta do DevTools do Google Chrome (Figura 27), percebeu-se que quando o botão é acionado o dado não é escrito e o *plugin* recebe 0 como resposta. Mas, quando o comando é escrito, o *plugin* recebe dados no formato JSON.

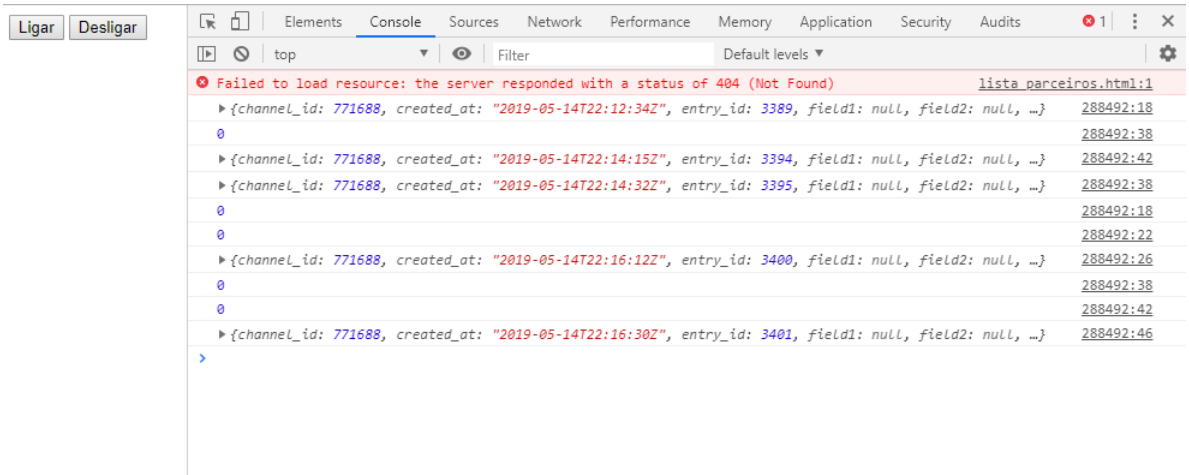


Figura 27: *plugin* com botões e a ferramenta DevTools.

Portanto, o código de Javascript do *plugin* foi modificado para contornar este

problema. Ao acionar o botão, o método POST é ativado aguarda-se uma resposta da página. Caso a resposta recebida seja igual a zero, espera-se 5 segundos para enviar um novo pedido ao servidor.

4.2 Parte 2 - Teste dos sensores

Para testar os sensores, foi conectado ao protótipo uma lâmpada halógena com potência igual a 120W. Os dados foram enviados ao ThingSpeak e os gráficos da Tensão, Corrente e Potência estão ilustrados na Figura 28.

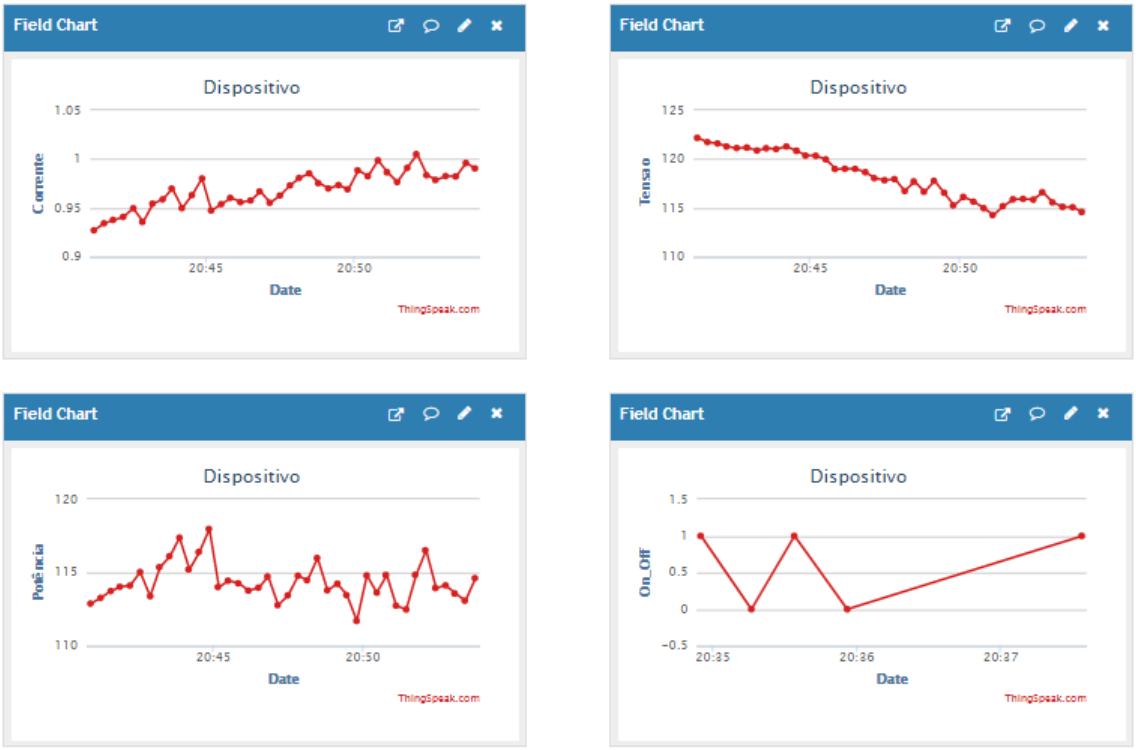


Figura 28: Resultados do teste realizado com uma lâmpada.

Para encontrar a Energia Consumida pela lâmpada, os dados do ThingSpeak foram exportados e organizados em um planilha no Excel, onde foi possível calcular o tempo do teste realizado, tensão média, corrente média, potência média e Energia. Além disso, foi construído um gráfico contendo a soma de 16 períodos. O resultado é mostrado na Figura 29.

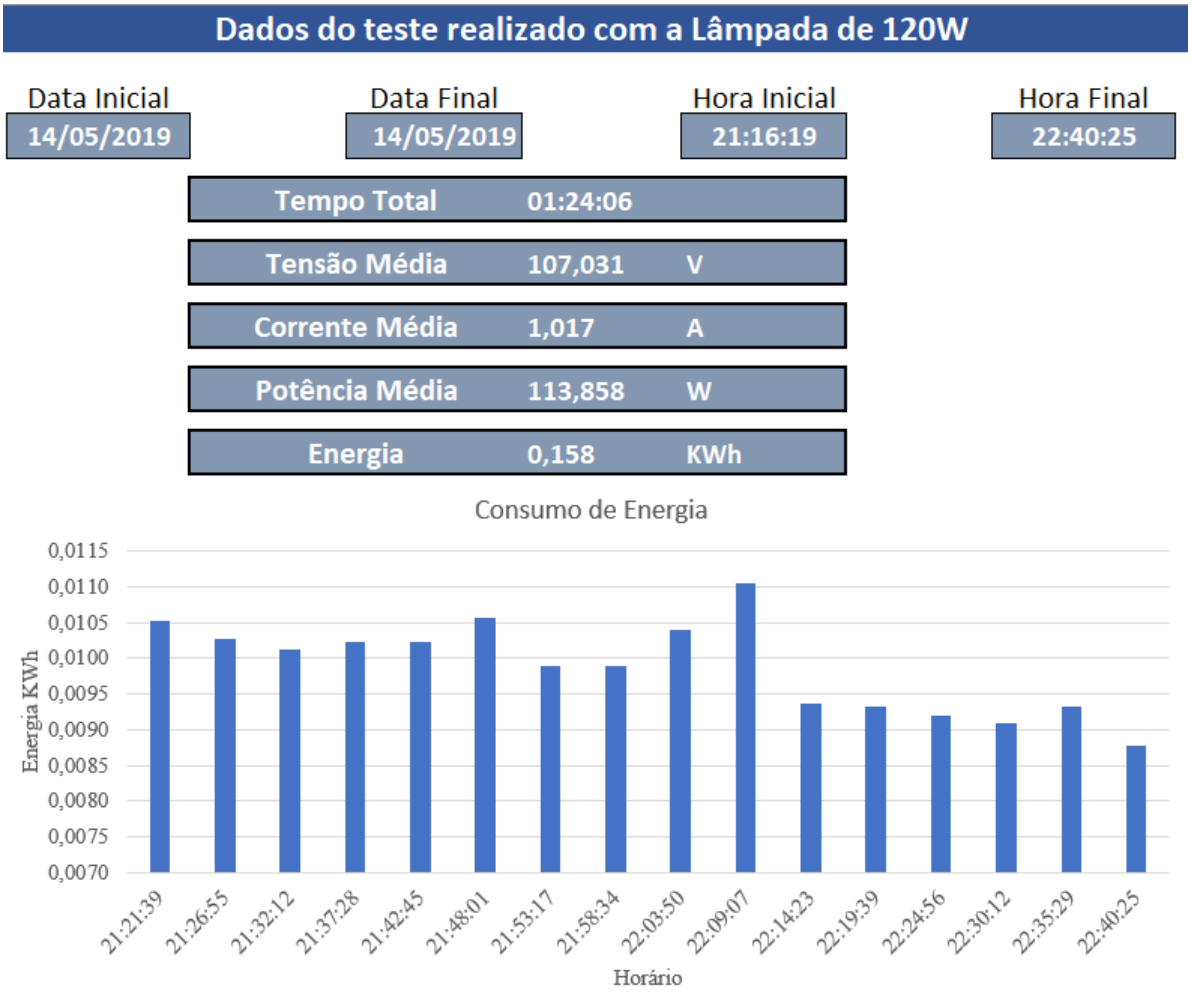


Figura 29: Resultados da organização dos dados obtidos com o teste realizado utilizando a lâmpada.

De acordo com os dados retornados pelo protótipo, a energia consumida pela lâmpada durante o período de uma hora, vinte e quatro minutos e seis segundos foi de 0,158KWh. Sabe-se que a potência da lâmpada é igual a 120W. Portanto, de acordo com a Equação 2.8, temos que a energia consumida durante o período do teste é igual a 0,168KWh. Assim sendo, os valores encontrados são satisfatórios, pois se aproximam da realidade.

Depois do teste com a Lâmpada, um *desktop* conectado ao estabilizador foi plugado no protótipo. Durante um certo período, o computador ficou em modo *standby* e depois foi ligado. A partir dos dados coletados (Figura 30), é possível perceber o momento no qual o *desktop* é inicializado, pois notou-se um aumento na corrente.

Figura 30: Resultados do teste realizado com um *desktop*.

O cálculo da energia consumida pelo *desktop*, foi realizado da mesma forma que o da lâmpada, com a diferença que o gráfico foi construído com a soma de 49 períodos. Ao analisar o gráfico, é possível perceber o momento no qual o *desktop* foi ligado, pois exite um aumento considerável de energia. Os resultados obtidos podem ser observados na Figura 31.

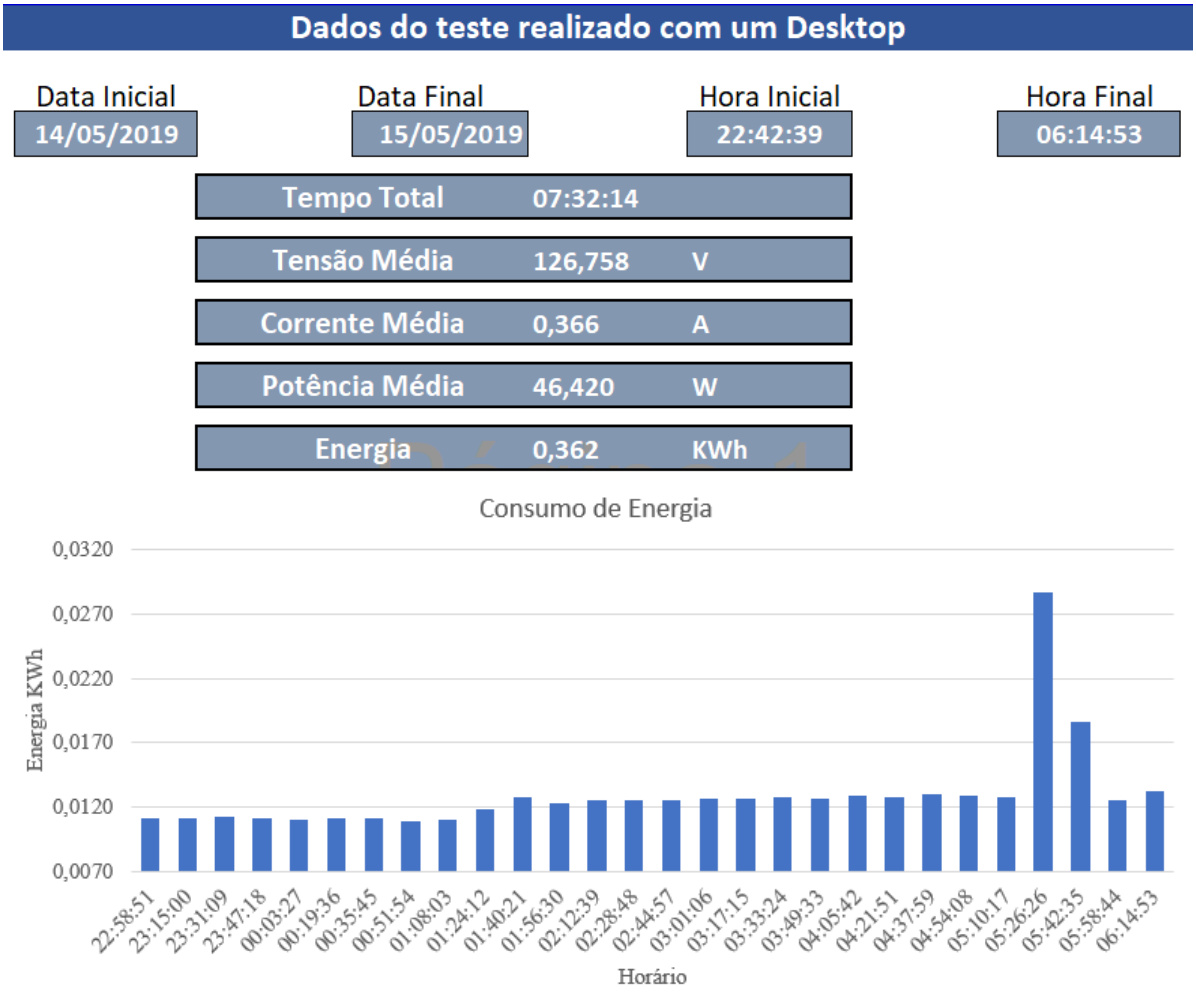


Figura 31: Resultado da organização dos dados obtidos com o teste realizado utilizando o *desktop*

4.3 Resumo dos resultados

Foi possível observar que o sensor de corrente ACS712 05B utilizado para a aquisição de corrente alternada apresenta grande ruído quando não há corrente, sendo necessário a não medição quando o dispositivo está desligado. Entretanto, quando há corrente, apesar de apresentar ruídos, a medição é satisfatória.

Em relação ao sensor de tensão construído, a medição apresentou resultados satisfatórios. No entanto, o optoacoplador 4n25 não apresenta linearidade, o que pode interferir na precisão da medida dependendo da tensão aplicada. Outro ponto a ressaltar, é a necessidade de realizar a calibração do circuito para diferentes 4n25.

Apesar da potência do dispositivo calculada ter sido realizada com os valores RMS, não se levou em consideração o fator de potência da rede que corresponde a defasagem da corrente em relação a tensão. Ademais, o multímetro utilizado para calibração dos sensores, também não retorna os valores *True* RMS da corrente e tensão.

No que se refere a aplicação em nuvem, o ThingSpeak se mostrou uma ótima ferramenta gratuita para analisar os dados de soluções IoT, além de proporcionar segurança em seu acesso. Porém, a sua limitação de intervalo de 15 segundos para escrita dos dados gera um conflito para enviar comandos Liga e Desliga.

5 Conclusões

O objetivo do trabalho foi alcançado visto que apresentou o desenvolvimento de um sistema de automação residencial capaz de medir a corrente e tensão de um dispositivo conectado a uma rede elétrica alternada, além de possibilitar o envio remoto de comandos liga/desliga ao dispositivo. A aplicação ThingSpeak foi configurada para visualização dos dados coletados e para a possibilidade do acionamento remoto.

Diante do teste realizado para o envio de comandos liga/desliga, pode se concluir que o protótipo apresentou limitações devido ao atraso no envio do controle. Esse problema se deve ao fato da utilização da versão gratuita do ThingSpeak que contém algumas restrições. Em relação aos testes realizados para medição dos sensores, os resultados obtidos foram satisfatórios, obtendo-se valores aproximados dos reais. No teste realizado com a lâmpada, a energia consumida encontrada apresentou um erro de aproximadamente +5,95%. Já no teste com o *desktop*, pôde-se perceber que a energia consumida pelo dispositivo no modo *stand-by* representa metade da energia consumida quando o dispositivo está operando.

O sistema foi projetado para aparelhos monofásicos e que possuem corrente alternada máxima de 3,54A, devido ao fato do módulo do sensor de corrente adotado suportar no máximo este valor. Além disso, o módulo relé suporta uma tensão máxima equivalente a 125V e corrente máxima de 10A.

O microcontrolador ESP32 teve um funcionamento conforme esperado e se demonstrou como uma ótima opção em aplicações IoT. O ThingSpeak também se mostrou satisfatório por possibilitar uma boa visualização das grandezas medidas além de viabilizar a exportação dos dados para que assim, possam ser analisados.

Ao final do desenvolvimento do trabalho e com base nos resultados obtidos, pode-se concluir que a instalação do sistema é uma ferramenta que auxilia o usuário na gestão de energia elétrica de dispositivos, podendo ser um grande aliado na redução do desperdício de energia. Embora que, inicialmente, o protótipo foi desenvolvido para soluções residenciais, pode ser empregado em prédios para comandar várias luminárias criando um sistema de iluminação automático.

Um sistema similar pode ser desenvolvido utilizando outros modelos do sensor ACS712 e um módulo relé que suporte uma maior corrente. No entanto, é importante salientar que os outros modelos do sensor ACS712 (ACS712ELCTR-20A-T e ACS712ELCTR-30A-T) conseguem medir no mínimo 1A, em razão do seu sinal apresentar grande ruído.

O desenvolvimento de uma página web contendo um painel visual, apresentando de

forma centralizada e organizada, as informações armazenadas na aplicação é uma sugestão de trabalho futuro. Além disso, a inclusão de um botão para comando local pode sanar o problema de atraso do acionamento remoto com a versão gratuita do ThingSpeak. Por fim, proponho o acréscimo de um circuito apto a medir o fator de potência da rede, aumentando a precisão do cálculo da energia consumida.

Referências

ALLIANCE, W.-F. **Who We Are**. 2018. Acessado em 17/02/2019. Disponível em: <https://www.wi-fi.org>.

ARDUINO. 2015. Acessado em 18/01/2019. Disponível em: <http://arduino.cc>.

ASHTON, K. That 'internet of things' thing. **RFID journal**, 2009. Jun, v. 22, n. 7, p. 97–114, 2009.

BALCH, M. **Complete digital design: a comprehensive guide to digital electronics and computer system architecture**. [S.l.]: McGraw-Hill, 2003.

BARROS, E.; CAVALCANTE, S. Introdução aos sistemas embarcados. **Artigo apresentado na Universidade Federal de Pernambuco-UFPE**, 2010. p. 36, 2010.

BOLZANI, C. Desmistificando a domótica. **Escola Politécnica da Universidade de São Paulo**, 2007. 2007.

BORGES, L. P.; DORES, R. Automação predial sem fio utilizando bacnet/zigbee com foco em economia de energia. **Trabalho de Graduação em Engenharia de Controle e Automação, Publicação FT. TG**, 2010. n. 06, 2010.

CHASE, O. Sistemas embarcados. **Mídia Eletrônica. Página na internet:< www. sbajovem. org/chase>, capturado em**, 2007. v. 10, n. 11, p. 13, 2007.

CHEBUDIE, A. B.; MINERVA, R.; ROTONDI, D. **Towards a definition of the Internet of Things (IoT)**. Tese (Doutorado), 08 2014.

CODECADEMY. **What is REST?** 2019. Acessado em 09/05/2019. Disponível em: <https://www.codecademy.com>.

CO.LTD, I. I. S. **Sonoff Pow R2**. 2017. Acessado em 03/06/2019. Disponível em: <https://www.itead.cc/>.

CUNHA, A. F. O que são sistemas embarcados. **Saber Eletrônica**, 2007. v. 43, n. 414, p. 6, 2007.

ENERGIA, M. de Minas e. **Lei que fortalece o Procel e ações de Eficiência Energética é sancionada**. 2016. Acessado em 16/02/2019. Disponível em: <http://www.mme.gov.br>.

ENGIE. **Gerenciamento de Energia**. 2019. Acessado em 13/05/2019. Disponível em: <https://www.engie.com.br/para-sua-empresa/gerenciamento-de-energia/>.

FILIPEFLOP. **Componentes Eletrônicos**. 2019. Acessado em 21/02/2019. Disponível em: <http://www.filipeflop.com/>.

GOLDEMBERG, J.; LUCON, O. Energia e meio ambiente no brasil. **Estudos avançados**, 2007. SciELO Brasil, v. 21, n. 59, p. 7–20, 2007.

KOYANAGI, F. **ESP32: Detalhes internos e pinagem**. 2018. Acessado em 15/04/2019. Disponível em: <https://www.fernandok.com/>.

LEE, J.-S.; SU, Y.-W.; SHEN, C.-C. et al. A comparative study of wireless protocols: Bluetooth, uwb, zigbee, and wi-fi. **Industrial electronics society**, 2007. v. 5, p. 46–51, 2007.

LI, M.; GU, W.; CHEN, W.; HE, Y.; WU, Y.; ZHANG, Y. Smart home: Architecture, technologies and systems. **Procedia computer science**, 2018. Elsevier, v. 131, p. 393–400, 2018.

MICROSYSTEMS, A. **ACS712 datasheet**. 2007. Acessado em 21/02/2019. Disponível em: <https://www.sparkfun.com>.

MQTT.ORG. **FAQ**. 2019. Acessado em 19/02/2019. Disponível em: <http://mqtt.org/>.

MURATORI, J. R.; BÓ, P. **Automação residencial: Histórico, definições e conceitos**. 2016.

NAKASHIMA, K. Valor médio e eficaz. **Universidade Federal de Itajubá**, 2007. 2007.

NGUYEN, T. D.; TRAN, V. K.; NGUYEN, T. D.; LE, N. T.; LE, M. H. Iot-based smart plug-in device for home energy management system. In: IEEE. **2018 4th International Conference on Green Technology and Sustainable Development (GTSD)**. [S.l.], 2018. p. 734–738.

PEREIRA, C.; CARVALHO, F. A internet das coisas (iot): Cenário e perspectivas no brasil e aplicações práticas. **VII SRST – Seminário de Redes e Sistemas de Telecomunicações**, 2017. Instituto Nacional de Telecomunicações - INATEL, p. 7, 2017.

PEREIRA, L. H. J. Monitoramento do consumo de energia elétrica e controle de equipamentos via aplicativo. 2018. 2018.

PETERSEN, J. E.; SHUNTUROV, V.; JANDA, K.; PLATT, G.; WEINBERGER, K. Dormitory residents reduce electricity consumption when exposed to real-time visual feedback and incentives. **International Journal of Sustainability in Higher Education**, 2007. Emerald Group Publishing Limited, v. 8, n. 1, p. 16–33, 2007.

PETRY, B. M.; SILVA, A. K.; MOREIRA, D. R.; RAMON, G. Projeto use-uso sustentável da energia na pucrs. **Anais do XI**, 2010. 2010.

RUBINSTEIN, M. G.; REZENDE, J. F. Qualidade de serviço em redes 802.11. **XX Simpósio Brasileiro de Redes de Computadores (SBRC2002)**, 2002. p. 26, 2002.

SANTOS, B. P.; SILVA, L.; CELES, C.; BORGES, J. B.; NETO, B. S. P.; VIEIRA, M. A. M.; VIEIRA, L. F. M.; GOUSSEVSKAIA, O. N.; LOUREIRO, A. Internet das coisas: da teoria à prática. **Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuıdos**, 2016. 2016.

SARAIVA, A. **Alimentos, energia elétrica e combustíveis pressionam prévia da inflação em junho**. Agência IBGE, 2018. Acessado em 16/02/2019. Disponível em: <https://agenciadenoticias.ibge.gov.br/>.

SENSE. **Technology**. 2019. Acessado em 13/05/2019. Disponível em: <https://sense-.com/technology>.

SIMPLÍCIO, P. V. G.; LIMA, B. R.; SILVA, G. S. Automação residencial: Uma solução social e econômica. **Caderno de Graduação-Ciências Exatas e Tecnológicas-UNIT-ALAGOAS**, 2018. v. 4, n. 3, p. 17, 2018.

SISTEMA-ONS, E. E. **Projeção da Demanda de Energia Elétrica: para os próximos 10 anos (2017-2026)**. 2017. Acessado em 15/02/2019. Disponível em: <http://www.epe.gov.br>.

SOUZA, A. F.; NUNES, G. M. O.; BIANCHINI, T. H. **Sistema supervisório para monitoramento de consumo de água.** 2016. 86 p. Trabalho de Conclusçã de Curso, Universidade Tecnológica Federal do Paraná, Curitiba.

SYSTEMS, E. **ESP32 Series datasheet**. 2019.

THINGSPEAK. **Learn More About ThingSpeak**. 2019. Acessado em 17/03/2019. Disponível em: <https://thingspeak.com/>.

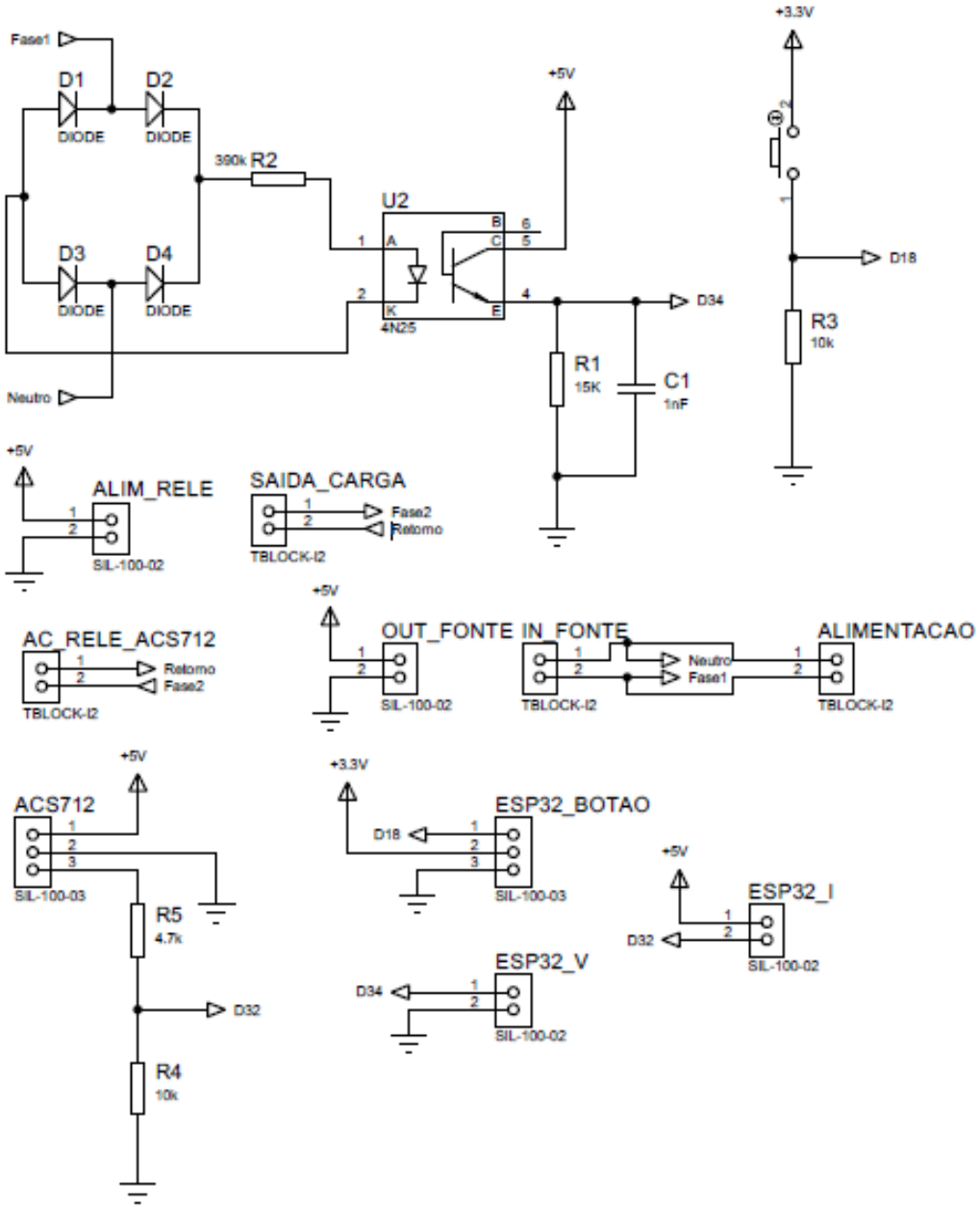
VIRIDIS. **Gestão de energia de ponta a ponta**. 2019. Acessado em 13/05/2019. Disponível em: <https://viridis.energy/pt/solucoes>.

WASOONTARAJAROEN, S.; PAWASAN, K.; CHAMNANPHRAI, V. Development of an iot device for monitoring electrical energy consumption. In: IEEE. **2017 9th International Conference on Information Technology and Electrical Engineering (ICITEE)**. [S.l.], 2017. p. 1–4.

YUAN, M. **Conhecendo o MQTT**. 2017. Acessado em 19/02/2019. Disponível em: <https://www.ibm.com>.

Apêndices

APÊNDICE A – Circuito do *Hardware*



APÊNDICE B – Código do *plugin*

```
1  <html>
2  <head>
3
4    %%PLUGIN_CSS%%
5    %%PLUGIN_JAVASCRIPT%%
6
7  </head>
8  <body>
9    <button onclick="Ligar()">Ligar</button>
10   <button onclick="Desligar()">Desligar</button>
11
12   <p id="texto"></p>
13
14 </body>
15 </html>
```

```
1  <script type='text/javascript' src='https://ajax.googleapis.com/ajax
     /libs/jquery/1.4.4/jquery.min.js'></script>
2  <script type='text/javascript' src='https://www.google.com/jsapi'>
     </script>
3  <script type='text/javascript'>
4    function Ligar(){
5      var url = "https://api.thingspeak.com/update.json?
         api_key=<API_READ_KEY>&field4=1"
6      $.getJSON(url, function(data){
7        console.log(data);
8        if (data == 0){
9        document.getElementById("texto").innerHTML = "Comando nao
           enviado, nova tentativa sera feita dentro de 5 segundos.";
10       setTimeout(function(){
11         $.getJSON(url, function(data){
12             console.log(data);
13             if (data == 0){
14               document.getElementById("texto").innerHTML = "Comando
                   nao enviado, nova tentativa sera feita dentro de 5
                   segundos.";
15               setTimeout(function(){
16               $.getJSON(url, function(data){
```

```
17                  console.log(data);
18                  if (data==0){
19                    document.getElementById("texto").innerHTML = "
                          Comando nao enviado. Por favor, tente
                          novamente.";
20                  }
21                  else {
22                   document.getElementById("texto").innerHTML = "
                        Comando enviado com sucesso. Status = 1.";
23                  }
24                });
25              }, 5000);
26              }
27            else {
28              document.getElementById("texto").innerHTML = "Comando
                    enviado com sucesso. Status = 1.";
29            }
30            });
31        }, 5000);
32        }
33        else {
34        document.getElementById("texto").innerHTML = "Comando enviado
              com sucesso. Status = 1.";
35        }
36      });
37    }
38    function Desligar(){
39      var url = "https://api.thingspeak.com/update.json?
            api_key=<API_READ_KEY>&field4=0"
40      $.getJSON(url, function(data){
41        console.log(data);
42        if (data == 0){
43          document.getElementById("texto").innerHTML = "Comando nao
              enviado, nova tentativa sera feita dentro de 5 segundos."
              ;
44          setTimeout(function(){
45          $.getJSON(url, function(data){
46              console.log(data);
47              if (data == 0){
48                document.getElementById("texto").innerHTML = "Comando
                    nao enviado, nova tentativa sera feita dentro de 5
                    segundos.";
49                setTimeout(function(){
50                $.getJSON(url, function(data){
51                  console.log(data);
52                  if (data==0){
53                    document.getElementById("texto").innerHTML = "
```

```
                    Comando nao enviado. Por favor, tente
                    novamente.";
54                }
55                else {
56                 document.getElementById("texto").innerHTML = "
                      Comando enviado com sucesso. Status = 0.";
57                }
58              });
59            }, 5000);
60            }
61            else {
62              document.getElementById("texto").innerHTML = "Comando
                    enviado com sucesso. Status = 0.";
63            }
64          });
65      }, 5000);
66      }
67      else {
68        document.getElementById("texto").innerHTML = "Comando
            enviado com sucesso. Status = 0.";
69      }
70    });
71  }
72 </script>
```

APÊNDICE C – Código do Firmware

```
#include <WiFi.h> // biblioteca que possibilita a conexão do ESP32 à internet
#include <PubSubClient.h> // biblioteca responsável pela comunicação MQTT

// Definição dos pinos
#define PinoTensao 34
#define PinoCorrente 32
#define PinoRele 19

char ssid[] = "SSID"; // Nome da rede
char pass[] = "PASSWORD"; // Senha da rede

// Parâmentos da conexão com o ThingSpeak
const char* server = "mqtt.thingspeak.com"; //Servidor do broker MQTT
char mqttUserName[] = "ESP32_client"; // Nome do usuário
char mqttPass[] = "<MQTT_API_KEY>"; // Chave do MQTT API
long ChannelID = CHANNELID; // ID do canal
char writeAPIKey[] = "<READ_KEY>"; // Chave de escrita do canal
char readAPIKey[]="<WRITE_KEY>"; // Chave de leitura do canal

// Inicialização das bibliotecas WiFiClient e PubSubClient
WiFiClient esp32client;
PubSubClient mqttClient( esp32client );

// Variáveis globais
float Vrms;
float Irms;
float P;
boolean EstadoSaida; //indica ao loop que a carga está ativada
unsigned long lastConnectionTime = 0;
```

```
30 const unsigned long postingInterval = 18L * 1000L; // Publica dados a
   ↪ cada 20 segundos
31
32 // Protótipos das funções
33 void calcula_tensao();   // Calcula a tensão
34 void calcula_corrente(); // Calcula a corrente
35 int mqttSubscriptionCallback(char* topic, byte* payload, unsigned int
   ↪ length); // Recebe a mensagem do topic MQTT subscrito
36 void mqConnect(); // Gera um Client ID único e conecta ao MQTT broker
37 int mqttSubscribe(); // Subscreve um campo de um canal do ThingSpeak
38 int mqttUnSubscribe(); // Para de subscrever um campo de um canal do
   ↪ ThingSpeak
39 void mqttPublish(); // Publica mensagens no canal
40 int connectWifi(); // Conecta na rede WiFi
41 void getID(char clientID[], int idLength); // Gera um client ID
   ↪ aleatório para a conexão MQTT
42
43 //Função de configurações
44 void setup() {
45   Serial.begin( 115200 );
46   Serial.println( "Inicio" );
47   connectWifi();  // conecta ao WiFi da rede
48   mqttClient.setServer( server, 1883 ); // Conecta a porta do MQTT
49   mqttClient.setCallback( mqttSubscriptionCallback );   // Configura a
     ↪ função que recebe a mensagem do MQTT
50
51   //Configura os pinos do ESP32
52   pinMode(PinoTensao,INPUT); //Configura o PinoTensao como entrada
53   pinMode(PinoCorrente,INPUT); //Configura o PinoCorrente como entrada
54   pinMode(PinoRele,OUTPUT); //Configura o PinoRele como saída
55   digitalWrite(PinoRele,LOW); //Inicializa o pino do Relé com nível
     ↪ baixo
56 }
57
58 //Função principal
59 void loop() {
60
61     if (WiFi.status() != WL_CONNECTED) {
62         connectWifi();
```

```
63      }
64
65      if (!mqttClient.connected())
66      {
67         mqConnect(); // Conecta ao client MQTT se não estiver conectado
68         if(mqttSubscribe()==1 ){
69                  Serial.println( " Subscrito " );
70              }
71      }
72
73      mqttClient.loop(); // Chame o loop para manter a conexão com o
        ↪ servidor
74
75                  // Se o intervalo de tempo passou desde a última conexão,
                    ↪ publica dados ao ThingSpeak
76                  if (millis() - lastConnectionTime > postingInterval)
77                  {
78                          mqttpublish();
79                          //Escreve os dados publicados na porta Serial
80                          Serial.print(Vrms);
81                          Serial.print(" - ");
82                          Serial.print(Irms);
83                          Serial.print(" - ");
84                          Serial.println(EstadoSaida);
85                  }
86  }
87
88  /**
89   * Publica os dados no topico correpondente ao canal do ThingsPeak
90   */
91
92  void mqttpublish() {
93
94          calcula_tensao();
95          if(EstadoSaida == 1){ //Testa se o dispositivo está ligado para
            ↪ então realizar a medição da corrente
96                  calcula_corrente();
97          }
98          else Irms = 0;
```

```
99          P = Vrms*Irms; // Calcula a potência
100
101   // Cria uma string com os dados para enviar ao ThingSpeak
102   String data = String("field1=" + String(Vrms, DEC) + "&field2=" +
      ↪  String(Irms, DEC) + "&field3=" + String(P, DEC));
103   int length = data.length();
104   char msgBuffer[length];
105   data.toCharArray(msgBuffer,length+1);
106   Serial.println(msgBuffer);
107
108   // Cria uma string com o topic e publica dados para os campos do canal
      ↪  do ThingSpeak
109   String topicString ="channels/" + String( ChannelID ) +
      ↪  "/publish/"+String(writeAPIKey);
110   length=topicString.length();
111   char topicBuffer[length];
112   topicString.toCharArray(topicBuffer,length+1);
113
114   mqttClient.publish( topicBuffer, msgBuffer );
115
116   lastConnectionTime = millis(); //Reseta o temporizador para publicar
      ↪  as mensagens
117 }
118
119 /**
120  * Faz a conexão ao broker MQTT
121  */
122
123 void mqConnect()
124 {
125     char clientID[ 9 ];
126
127     // Loop until connected.
128     while ( !mqttClient.connected() )
129     {
130
131         getID(clientID,8);
132
133         // Connect to the MQTT broker.
```

```
134         Serial.print( "Tentando conexão MQTT..." );
135         if ( mqttClient.connect( clientID, mqttUserName, mqttPass ) )
136         {
137             Serial.println( "Conectado com o Client ID:  " + String(
                ↪  clientID ) + " Usuário "+ String( mqttUserName ) + " Pwd
                ↪  "+String( mqttPass ) );
138
139         } else
140         {
141             Serial.print( "Falha, rc = " );
142             Serial.print( mqttClient.state() );
143             Serial.println( " Nova tentativa será feita em 5 segundos" );
144             delay( 5000 );
145         }
146     }
147 }
148
149 /**
150  * Cria um cliente aleatório client ID
151  *   clientID - char matriz para saída
152  *   idLength - tamanho do clientID
153  */
154
155 void getID(char clientID[], int idLength){
156 static const char alphanum[] ="0123456789"
157 "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
158 "abcdefghijklmnopqrstuvwxyz"; // Utilizado para geração aleatória do
    ↪  Client ID
159
160     // Gera o ClientID
161     for (int i = 0; i < idLength ; i++) {
162         clientID[ i ] = alphanum[ random( 51 ) ];
163     }
164     clientID[ idLength ] = '\0';
165 }
166
167 /**
168  *        Subscreve o campo 4 do canal
169  */
```

```
170
171 int mqttSubscribe(){
172     String myTopic;
173
174     myTopic="channels/"+String( ChannelID
        ↪ )+"/subscribe/fields/field4/"+String( readAPIKey );
175
176     Serial.println( "Subscrito de " +myTopic );
177     Serial.println( "Estado= " + String( mqttClient.state() ) );
178     char charBuf[ myTopic.length()+1 ];
179     myTopic.toCharArray( charBuf,myTopic.length()+1 );
180
181     return mqttClient.subscribe(charBuf);
182 }
183
184 /**
185  *        Deixa de subscrever o campo 4 do canal
186  */
187
188 int mqttUnSubscribe(){
189     String myTopic;
190
191     myTopic="channels/"+String( ChannelID
        ↪ )+"/subscribe/fields/field4/"+String( readAPIKey );
192
193     char charBuf[ myTopic.length()+1 ];
194     myTopic.toCharArray( charBuf, myTopic.length()+1 );
195     return mqttClient.unsubscribe( charBuf );
196
197 }
198
199 /**
200  *        Faz a conexão com o WiFi
201  */
202
203 int connectWifi()
204 {
205
206     while ( WiFi.status() != WL_CONNECTED ) {
```

```
207         WiFi.begin( ssid, pass );
208         delay( 2500 );
209         Serial.println( "Conectando ao WiFi" );
210     }
211     Serial.println( "Conectado" );
212 }
213
214 /**
215  *        Faz o cálculo da tensão
216  */
217
218 void calcula_tensao(){
219   int leitura_v;
220   float soma_v = 0;
221   float valor_v;
222
223     for (int i = 0; i <= 1920; i++){
224         leitura_v=analogRead(PinoTensao); //aquisição da leitura do ADC
225         soma_v+=leitura_v*leitura_v; //somatório do quadrado da leitura
226         delayMicroseconds(510); //aguarda 390ms para próxima leitura
227      }
228      //Calcula a tensão da saída do sensor
229      valor_v = (float) sqrt(soma_v/1920)*(3.300/4096.00);
230
231      if (valor_v == 0) {
232       Vrms=0;
233      }
234      else{
235       Vrms = (float) 15.28615*valor_v*valor_v + 76.91308*valor_v -
        ↪  41.78664;
236      }
237 }
238
239 /**
240  *        Faz o cálculo da corrente
241  */
242
243 void calcula_corrente(){
244   int leitura_i;
```

```
245   float soma_i = 0;
246   float valor_i;
247
248   soma_i = 0;
249   for (int i = 0; i <= 1920; i++){
250       leitura_i=analogRead(32); //aquisição da leitura do ADC
251       leitura_i = leitura_i - 3000;
252
253
254       soma_i+=leitura_i*leitura_i; //somatório do quadrado da leitura
255       delayMicroseconds(390); //aguarda 390ms para próxima leitura
256    }
257    //Calcula a tensão da saída do sensor e divide pela sensibilidade
258    valor_i = (float) sqrt(soma_i/1920)*(3.300/4096.00)/0.185;
259    Irms = (float) 1.170989024*(valor_i)-0.072530783;
260 }
261
262 /**
263  *        Função de callback que recebe a mensagem do tópico subscrito
264  */
265
266 int mqttSubscriptionCallback( char* topic, byte* payload,unsigned int
    ↪ mesLength )
267 {
268     String msg;
269     //obtem a string do payload recebido
270     for(int i = 0; i < mesLength; i++)
271     {
272        char c = (char)payload[i];
273        msg += c;
274     }
275     //Executa uma ação dependendo da string recebida:
276
277     //verifica se deve colocar nivel alto na saída do PinoRele:
278     if (msg.equals("1"))
279     {
280         digitalWrite(PinoRele, HIGH);
281         EstadoSaida = 1;
282     }
```

```
283
284     //verifica se deve colocar nivel baixo na saída do PinoRele:
285     if (msg.equals("0"))
286     {
287         digitalWrite(PinoRele, LOW);
288         EstadoSaida = 0;
289     }
290 }
```

Certifico que a aluna **Isabella Ferreira de Oliveira,** autora do trabalho de conclusão de curso intitulado "**Desenvolvimento de um sistema de automação residencial baseado em IoT para controle e monitoramento de dispositivos elétricos**", efetuou as correções sugeridas pela banca examinadora e que estou de acordo com a versão final do trabalho.

Prof. Dr. Antonio Santos Sánchez
Orientador