

CRIAÇÃO DE UMA REDE MESH DE SENSORES SEM FIO E DE UM SOFTWARE WEB PARA VISUALIZAR OS DADOS COLETADOS

Daniel da Veiga¹; Fábio Weber Albiero²

Resumo

As redes *mesh* de sensores sem fio são amplamente empregadas para monitorar grandezas físicas nos mais variados ambientes. Há inúmeros estudos que fazem uso dessa tecnologia para monitorar áreas de cultivo de hortaliças, analisar condições climáticas e do solo, além de aplicativos médicos e veterinários. Porém, a maioria desses trabalhos concentra seus esforços na coleta e na transmissão de dados, sem dar a devida atenção à forma como as informações coletadas pelos sensores serão apresentadas ao usuário final. Nesse contexto, o presente artigo apresenta a criação de uma rede *mesh* de sensores sem fio, de pequena proporção, com o objetivo de realizar testes de coleta, de transmissão e, em especial, de visualização dos dados. Os dados são coletados e transmitidos pelos nós clientes da rede *mesh*, os microcontroladores ESP32 e os sensores *hall*. A visualização dos dados é realizada por meio de uma interface web intuitiva em tempo real, no formato de árvore, em que o nó raiz representa o nó principal da rede e os nós filhos, os clientes ou sensores. O projeto em questão se posiciona de forma intermediária entre a utilização genérica de redes *mesh* de sensores sem fio e uma aplicação específica e otimizada para a visualização dos dados coletados pelos sensores distribuídos no ambiente.

Palavras-chave: interface web; rede *mesh*; sensores.

Abstract

Wireless sensor mesh networks widely monitor physical quantities in the most varied environments. Numerous studies make use of this technology to monitor areas of vegetable cultivation, and analyze climate and soil conditions, in addition to medical and veterinary applications. However, most of these works focus on data collection and transmission, without giving due attention to how the information collected by the sensors will be presented to the end user. In this context, this article presents the creation of a mesh network of wireless sensors, of small proportion, to carry out collection, transmission, and, in particular, data visualization tests. Data are collected and transmitted by the mesh network client nodes, ESP32 microcontrollers, and hall sensors. Data visualization is performed through an intuitive web interface in real-time, in a tree format, in which the root node represents the main node of the network and the child nodes, clients, or sensors. The project in question is positioned immediately between the generic use of wireless sensor mesh networks and a specific and optimized application for the visualization of data collected by sensors distributed in the environment.

Keywords: Web interface; mesh network; Sensors.

¹ Discente do Curso de Pós-Graduação Lato Sensu Especialização em Computação Aplicada ao Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia Farroupilha (IFFar), campus Santo Ângelo/RS. E-mail: danieldaveiga@gmail.com.

² Mestre em Ciência da Computação pela Universidade Federal de Santa Maria – UFSM, professor do Instituto Federal de Educação, Ciência e Tecnologia Farroupilha (IFFar), campus Santo Ângelo/RS. E-mail: fabio.albiero@iffarroupilha.edu.br.

1 Introdução

Nas últimas décadas, a presença de dispositivos eletrônicos tem se tornado cada vez mais marcante na vida cotidiana. Dentre estes dispositivos, merecem destaque os assistentes de voz, os *smartphones*, as *smart TVs* etc. Concomitantemente, observou-se o advento da IoT (*Internet of Things*), a qual possibilitou que os equipamentos eletrônicos estabelecessem conexão com a Internet e trocassem informações entre si. Em virtude do grande número de equipamentos que passaram a possuir tal conexão, tornou-se possível coletar dados de um ambiente, tais como: temperatura, pressão e umidade; por meio de sensores acoplados nesses dispositivos e transmiti-los a outras máquinas para fins de análise, estudos de caso e demais propósitos pertinentes. Uma vez que os dispositivos eletrônicos são dotados de sensores, é possível configurar uma rede de sensores sem fio (RSSF). Esses sensores, denominados nós ou nodos, são dispositivos autônomos e encontram-se distribuídos espacialmente no ambiente a ser monitorado. Os nodos podem ser posicionados a diversas distâncias, o que não é viável com redes de sensores convencionais. De modo geral, os nós se comunicam entre si por meio de uma rede *ad-hoc* sem fio e topologia de rede mais utilizada é a *mesh* (malha).

O monitoramento de características físicas de um ambiente em tempo real é essencial em diversas áreas do conhecimento: Ciências Agrárias; Biológicas; da Saúde; Engenharias; Exatas e da Terra. Sensores são amplamente utilizados em aplicações como medições climáticas, supervisões de desastres, monitoramento de animais, cidades inteligentes etc. No entanto, na maioria das vezes, os dados coletados pelos sensores não estão disponíveis para o usuário final de modo claro e nem em tempo real, sendo esses armazenados em bancos de dados ou em planilhas eletrônicas de difícil compreensão.

Diante deste contexto, este trabalho propõe uma solução para este problema: criar uma interface web onde o usuário possa visualizar os dados coletados pelos sensores dispersos no ambiente de forma simples e em tempo real. Para que tal objetivo seja alcançado, é necessário, primeiramente, projetar e construir uma rede *mesh* de sensores sem fio para a execução dos testes. Portanto, constituem os objetivos específicos deste trabalho: 1) criar uma rede *mesh* sem fio através da utilização de microcontroladores ESP32 e sensores de baixo custo, fazendo a interação de ambos os componentes de hardware; 2) explorar a rede *mesh* para criar um sistema de coleta, tratamento, transmissão e disponibilidade dos dados e 3) desenvolver uma interface web intuitiva e de fácil interpretação para o usuário final visualizar os dados coletados pelos sensores. A interface, baseada em HTML (*HyperText Markup Language*) e Javascript, deve ser

dinâmica, atualizar-se em tempo real e não deve exigir recursos adicionais de hardware (processador, memória RAM ou espaço em disco) e software.

Este artigo está estruturado da seguinte forma: a seção 2 apresenta o referencial teórico, a seção 3 descreve os trabalhos correlatos, a seção 4 aborda o desenvolvimento do trabalho e, por fim, a seção 5 discorre sobre as considerações finais.

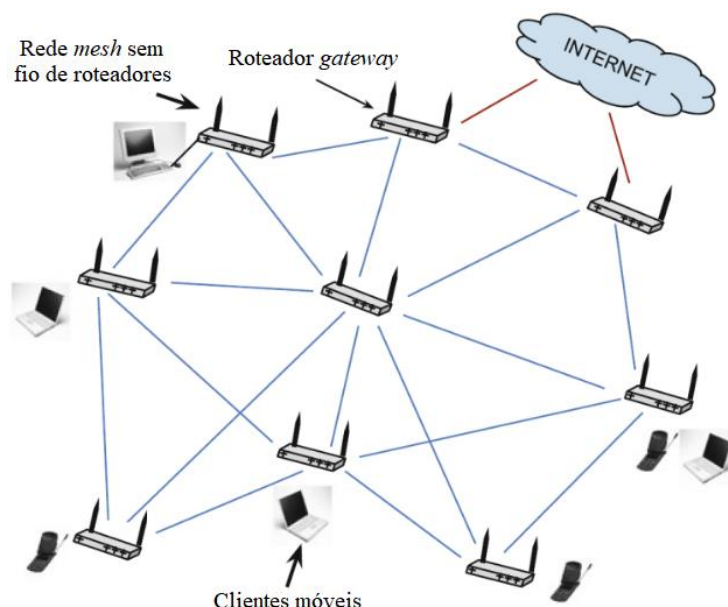
2 Referencial teórico

Esta seção apresenta o referencial teórico do trabalho, que consiste em uma revisão bibliográfica detalhada sobre os principais conceitos e teorias relacionados ao tema abordado. Serão discutidos tópicos como redes *mesh*, microcontroladores ESP e sensores.

2.1 Redes *mesh*

Redes *mesh* são compostas por várias células (também chamadas de nós ou nodos), as quais se comunicam entre si. Uma característica que define uma rede *mesh* é que cada nó da rede retransmite dados de/e para os clientes com múltiplos saltos, assim formando um rede com finalidade específica (*ad-hoc*) (Sichitiu, 2005). A figura 1 apresenta a arquitetura de uma rede *mesh*, composta por *gateways*, roteadores e dispositivos móveis. Cada *gateway* e roteador representa um nó, assim como os dispositivos móveis conectados nela representam os nós clientes.

Figura 1 - Arquitetura de uma rede mesh.



Fonte: Adaptada de Pathak e Dutta (2013).

A rede *mesh* apresenta um conjunto de vantagens que merecem destaque. Em primeiro lugar, os clientes podem desfrutar de mobilidade de forma transparente, sem interrupções ou

quedas de sinal. Ademais, cada nó da rede *mesh* atua como um repetidor de sinal, o que amplia significativamente a área de cobertura das redes durante as transmissões de dados. A existência de múltiplos *gateways* proporciona redundância e confiabilidade à rede (Maia, 2008). Além disso, há otimizações voltadas para o baixo consumo de energia elétrica no hardware, o que se torna importante em ambientes móveis, cujos nós podem ser alimentados por meio de baterias (Sanni *et al.*, 2013; Secco, 2019).

Uma rede *mesh* pode ser composta por componentes de hardware com conexões sem fio (*wireless*) e, a partir disso, pode-se criar uma rede *mesh* sem fio que disponibiliza acesso à Internet para nodos móveis ou estacionários. Posteriormente, é possível acrescentar sensores nos nodos da rede para monitorar grandezas físicas de algum ambiente, o que a transforma em uma rede de sensores sem fio - RSSF.

Esse tipo de rede é abrangido na IoT, um conceito recente que define dispositivos interligados dentro de uma aplicação específica, em um ambiente (por exemplo, doméstico) onde lâmpadas, alto falantes, geladeiras, televisões e celulares estão todos interconectados e trocando informações entre si. Basicamente, é a interconexão digital de objetos cotidianos com à Internet (Rosa, 2020). Este tipo conexão exige um baixo consumo de energia elétrica e oferece alta disponibilidade, resiliência e atualização constante dos dados, o que é ideal para a aplicação neste trabalho.

De acordo com Akyildiz *et al.* (2005), as redes *mesh* também podem ser híbridas, o que significa que combinam elementos de infraestrutura da rede (os roteadores, por exemplo) com os clientes (dispositivos móveis). Os nós clientes da rede *mesh* podem acessar a própria rede através de roteadores ou diretamente através de outros clientes. Enquanto isso, a infraestrutura fornece acesso a outras redes, tais como: Internet, Wi-Fi, WiMax, celulares e sensores. A capacidade de roteamento dos nós clientes aumentam a conectividade e a cobertura dentro de uma rede *mesh* sem fio.

2.2 Microcontroladores ESP

Segundo Silva (2021), a evolução da eletrônica digital permitiu a criação de dispositivos com capacidade de memorização, como as válvulas termiônicas e posteriormente os circuitos transistorizados integrados. Os microcontroladores são pequenos computadores em um único circuito integrado, composto por hardware e software, que inclui o processador, as memórias e as entradas e saídas (Oliveira Júnior; Duarte, 2010). Essas entradas e saídas podem ser programadas de acordo com as necessidades de cada desenvolvedor, utilizando várias

linguagens de programação, como C e Python. Os microcontroladores mais utilizados são o Arduino, o Raspberry, o ESP32 e o PIC16F877A (FRANZIN LEITE *et al.*, 2020).

Todos os microcontroladores foram pensados de tal forma que pudessem ser adquiridos a baixo custo, permitindo que qualquer pessoa pudesse desenvolver um projeto, seja para fins didáticos ou profissionais. [...] Os módulos de conectividade Wi-Fi e Bluetooth já adicionados às placas ESP32 fazem com que ele seja melhor empregado em projetos de IoT, por exemplo. É muito mais simples e adequado utilizar um ESP quando há a necessidade de conectar o projeto à Internet, pois o dispositivo tende a ficar mais integrado. De fato, algumas vezes um módulo ESP é utilizado em conjunto com outros microcontroladores para facilitar a conectividade desse com outro dispositivo. O ESP32 também é um dispositivo open-source, o que permite uma fácil programação. Seu processador é capaz de ler várias linguagens e de várias plataformas: utilizando a própria IDE do Arduino, com o núcleo específico; programando em JavaScript ou Python, por exemplo. (Imejunior, 2019).

O uso de microcontroladores permite a troca de informações em tempo real entre si ou com um controlador central ou uma nuvem para armazenamento e atualização síncrona dos dados. Além disso, eles permitem a programação por meio de diferentes interfaces.

2.3 Sensores

Por definição, uma grandeza física é toda manifestação física que pode ser mensurável, ou seja, tudo que pode ser atribuído uma unidade de medida (Young, 2016). Neste contexto, os sensores são dispositivos eletrônicos que transformam grandezas físicas em sinais elétricos.

Para que os sensores sejam utilizados nos sistemas digitais, sua saída deve ser digital. Para os sensores analógicos, é necessário o uso de um conversor analógico digital (Urbanetz Júnior; Maia, 2010; Da Silva Júnior, 2013). O conversor analógico-digital (ou ADC - *analog-to-digital converter*) recebe como sinal analógico uma tensão variável e produz como saída um conjunto de bits. Existem vários métodos de conversão que os ADCs utilizam, e o mais usado é o método de aproximações sucessivas. (Tocci; Widmer; Moss, 2010).

É importante destacar que o microcontrolador ESP32 possui vários sensores embutidos, entre eles o sensor *hall*, que é um sensor magnético e tem como funções reconhecer campos eletromagnéticos e identificar a quantidade de frequência eletromagnética gerada. Tal sensor foi utilizado neste trabalho, em função deste já estar integrado no microcontrolador ESP32. O sensor *hall* permite a detecção de ímãs e a medição de correntes elétricas. Quando combinado a um ímã pode ser utilizado para detectar a abertura e fechamento de portas, assim como determinar o RPM (rotações por minuto) de um motor sem contato físico com a mesmo. Ele é capaz de fornecer uma saída analógica ou digital, dependendo da configuração utilizada. A saída analógica é uma tensão proporcional ao campo eletromagnético medido, enquanto a saída digital é um valor binário que representa a polaridade do campo eletromagnético medido.

3 Trabalhos correlatos

Redes de sensores sem fio (RSSFs) diferem de redes de computadores tradicionais em vários aspectos. Normalmente essas redes possuem um grande número de nodos distribuídos, têm restrições de energia e devem possuir mecanismos para autoconfiguração e adaptação devido a problemas como falhas de comunicação e perda de nodos. Uma RSSF tende a ser autônoma e requer um elevado nível de cooperação para executar as tarefas definidas para a rede. “Os desafios e considerações de projeto de RSSFs vão muito além das redes tradicionais” (Loureiro *et al.*, 2003).

Loureiro *et al.* (2003) ainda citam que nessas redes, cada nó pode ter uma variedade de sensores, tais como acústico, sísmico, infravermelho, câmeras, calor, temperatura e pressão. Sensores podem ser embutidos em peças industriais para testes. Pode-se monitorar remotamente ambientes como prédios, residências, florestas ou perímetros e também o tráfego de veículos em rodovias, na malha urbana ou em estacionamentos. Na Medicina e na pecuária, pode-se monitorar o funcionamento de órgãos como o coração, detectar a presença de substâncias que podem comprometer o funcionamento do organismo, seja em humanos ou animais. Pode ter até mesmo aplicações militares, na detecção de movimento inimigo ou utilizado como forma de comunicação resiliente entre as tropas. De forma genérica, as RSSFs podem ser utilizadas na segurança, no controle, na atuação e na manutenção de sistemas complexos e no monitoramento de ambientes internos e externos.

A importância de utilizar uma rede resiliente, aliada à flexibilidade de usar qualquer sensor compatível com a plataforma, provê os meios necessários para acessar, tratar e transmitir os dados. Essa transmissão pode ser local, sem acesso à Internet, usando um servidor para concentrar os dados e servir como nó principal da rede. É possível ainda utilizar um ou mais *gateways* (portas de saída) redundantes para a Internet e armazenar esses dados diretamente na nuvem. A abordagem de armazenamento na nuvem tem a vantagem de manter os dados acessíveis de qualquer lugar do mundo, mas exige uma conexão com a Internet, e toda a segurança necessária que decorre desta conexão. Armazenar os dados localmente em um nó principal tem a vantagem de ser mais veloz, porém exige que o acesso aos dados seja feito localmente. É possível ainda criar um ambiente híbrido, em que apenas o nó principal esteja conectado à Internet. Nesse ambiente, os outros nós coletam e transmitem os dados para o nó principal, que os disponibiliza diretamente na Internet ou na nuvem. Esta infraestrutura é mais fácil de se manter e possui menos problemas de segurança, pois os nós sensores são isolados do restante da rede, comunicando-se por um canal próprio.

Existem inúmeras aplicações possíveis para o monitoramento de ambientes através do uso de redes *mesh* de sensores sem fio. No que tange o monitoramento ambiental com o uso de redes *mesh* de sensores, pode-se destacar:

[...] Os nós constituintes das redes incorporam módulos de detecção (sensores), um módulo de processamento, um módulo de comunicação e uma fonte de alimentação que é, geralmente, uma bateria de baixo custo e dimensão reduzida. Os nós da rede cooperam entre si, de modo a que os dados recolhidos alcancem o nó com funções de coordenação e que esteja ligado a um computador, que processa a informação através de uma aplicação criada para o efeito. A colocação dos nós no meio ambiente poderá ser feita de forma aleatória. Para zonas de difícil acesso poderá ser necessário recorrer a meios aéreos, tais como uma avioneta ou um helicóptero para lançar os nós na região a ser monitorizada. Na maior parte das aplicações práticas estes são colocados manualmente na zona a monitorizar. (Braga, 2010).

A revisão bibliográfica mais completa encontrada foi publicada por Nurlan *et al.* (2021). Neste trabalho foram analisados diversos estudos que integravam os conceitos de redes *mesh* e redes de sensores, explorando diversas configurações.

Dispositivos sensores IoT podem ser implementados como uma rede de sensores sem fio (WSN) em uma topologia *mesh*. Tanto WSNs quanto redes *wireless mesh* (WMNs) são fáceis de organizar e implementar. Nesse caso, há várias razões para combinar esses diferentes tipos de redes. Em particular, a detalhada capacidade sensorial de redes de sensores pode ser melhorada aumentando a banda, a confiabilidade e o consumo de energia elétrica numa topologia *mesh*. No entanto, atualmente há poucos estudos integrando estes diferentes tipos de rede. Ainda, não há revisões sistemáticas de métodos de conexão existentes. (Nurlan *et al.*, 2021).

Por sua vez, Martillano *et al.* (2022) criou uma rede *mesh* para testar sua capacidade e utilidade. Esse trabalho foi de extrema relevância, pois teve como objetivos testar e reportar as capacidades intrínsecas da rede *mesh*. O trabalho de Martillano e outros modificou a biblioteca *painlessMesh* (Martin, 2022), a qual proporciona o microcontrolador ESP8266 identificar cada dispositivo, conectar uns aos outros e compor a rede *mesh*. Cada nó da rede pode servir tanto como um ponto de acesso para outros nós conectarem, assim como uma estação, enquanto ainda são capazes de encontrar a melhor conexão alternativa existente quando um nó perde a conexão com a rede. No entanto, aqueles dispositivos que fazem uso da biblioteca *painlessMesh* para formar uma rede *mesh* não podem ter conexões simultâneas com a própria rede e a Internet, e por este motivo, tal estudo fez uso de dois dispositivos do tipo ponte (*bridge*).

É importante destacar que Martillano e outros utilizaram microcontroladores menos potentes (o ESP8266) do que os utilizados na implementação deste trabalho (o ESP32). Além disso, como a mesma biblioteca foi utilizada e não foi encontrado o problema relatado (o nó raiz da rede *mesh* se conecta a um ponto de acesso diretamente com a Internet), pode-se inferir que há diferenças entre os microcontroladores que impediram o seu uso na configuração do trabalho.

Esses são apenas alguns exemplos encontrados na pesquisa sobre a utilização de RSSF para o monitoramento de ambientes. Vários outros trabalhos utilizam as mesmas definições para o monitoramento de áreas de cultivo de hortaliças, análise do clima e do solo. Há ainda, conforme já citado anteriormente, aplicações médico-hospitalares e veterinárias. Seguindo os mesmos princípios, qualquer conjunto de dados é passível de coleta automática e posterior análise, fazendo uso desse tipo de infraestrutura de hardware.

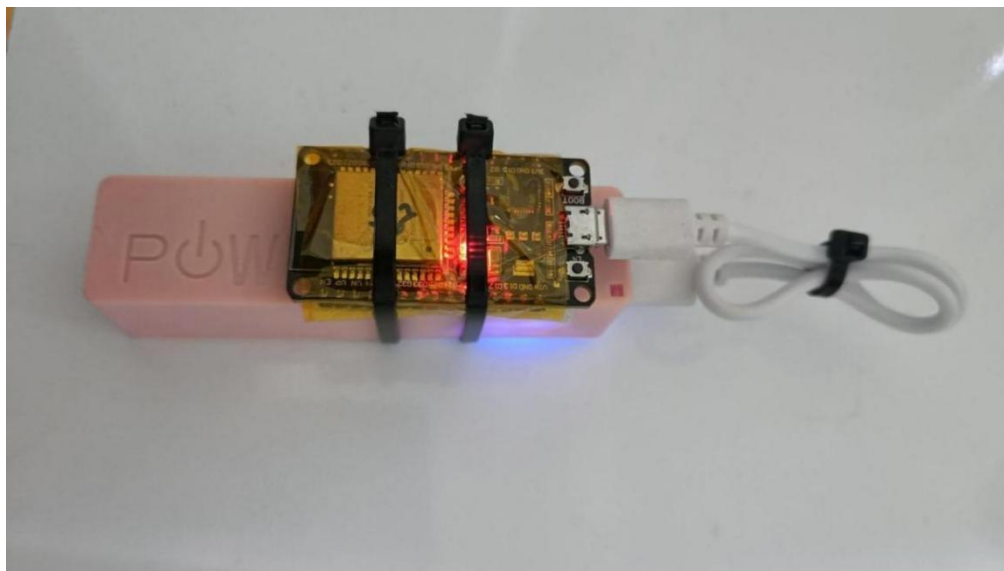
Após a coleta e tratamento dos dados pelos microcontroladores ou nó principal, é necessário criar uma interface para apresentar as informações de maneira clara ao usuário. No entanto, a maioria dos trabalhos científicos foca apenas na coleta de dados, sem abordar a visualização em tempo real. Um trabalho que se destacou nesse sentido foi realizado por Teles (2019), que estudou mecanismos de ambientes 3D para representar dados da qualidade do ar de forma imersiva.

[...] ambientes 3D e a representação de dados, podendo considerar *big data* devido à quantidade de dados que se irão recolher com os sensores. Estas duas áreas são apresentadas juntas, uma vez que a intenção desta dissertação é a de conseguir representar dados da qualidade do ar já existentes de uma forma imersiva. Para tal, ir-se-a estudar mecanismos de ambientes 3D de forma a proporcionar ao usuário final uma experiência de imersão enquanto visualiza os dados. (Teles, 2019).

A maioria dos trabalhos não se concentrou em interfaces móveis simples e com alta taxa de atualização, provavelmente porque uma taxa de atualização mais rápida pode levar a um uso maior dos microcontroladores e, conseqüentemente, um consumo maior da energia elétrica armazenada nas baterias. No entanto, uma interface simples é vantajosa por tornar o sistema mais portátil, permitindo a visualização em qualquer navegador. O trabalho de Pfeifer *et al.* (2018) foi o que mais se aproximou dos objetivos desta pesquisa. Eles implementaram uma rede *mesh* de sensores para medir o consumo de energia elétrica em edifícios da Universidade Federal de Santa Catarina - Campus Araranguá/SC. O projeto se diferencia dos demais por utilizar arquiteturas voltadas para a IoT, como os protocolos MQTT (*Messaging Queue Telemetry Transport*) e CoAP (*Constrained Application Protocol*), para coletar e transferir os dados para um sistema de visualização já finalizado, incluindo uma visualização na web e um aplicativo móvel. Ao avaliar o trabalho, foi possível concluir que ele não utilizou o conceito de nó raiz da biblioteca *painlessMesh*, o que pode acarretar problemas na formação da rede. Além disso, a utilização de dois microcontroladores para coletar, processar e enviar os dados tornou o nó raiz mais complexo, assim como a utilização de um servidor MQTT e outras interfaces. Nesse sentido, o trabalho proposto nesta pesquisa busca simplificar a coleta, o tratamento e a visualização dos dados, garantindo uma interface móvel simples e com uma alta taxa de atualização (em tempo real).

módulo carregador e conversor de energia com entrada e saída padronizadas. Embora o kit de desenvolvimento seja alimentado internamente por 3.3V, a voltagem fornecida pela interface USB é de 5V e a voltagem fornecida pela bateria 18650 varia entre 3V e 4.2V, resultando em duas conversões sendo realizadas nesse circuito.

Figura 3 – Um nó completo, com o microcontrolador ESP32 e um módulo de energia ligado.



Fonte: Os próprios autores.

O ESP32 é um microcontrolador que possui módulos de conectividade Wi-Fi e Bluetooth, tornando-o ideal para projetos de IoT, especialmente quando há a necessidade de conectar o sistema à Internet. Seu código aberto permite fácil programação, com suporte a diversas linguagens, incluindo a própria IDE do Arduino, JavaScript e Python.

A escolha deste módulo de bateria foi baseada na facilidade de uso e no controle de carga e descarga para preservar a saúde da bateria. Embora outros módulos carregadores com conversores de saída para 3.3V possam ser utilizados, não foram encontrados no mercado opções com proteção contra descarga excessiva. Destaca-se ainda que o consumo energético deste sistema foi medido utilizando um testador USB que registra a voltagem e a carga utilizada. Em 5 horas de uso contínuo monitorado, foi medido um total de 250mA de consumo no módulo raiz. Nos testes dos nodos clientes foi medido um consumo energético médio de 175mA no período de 10 horas, indicando uma autonomia com uma carga variando entre 40 e 110 horas de uso, dependendo da bateria utilizada.

Embora nenhuma otimização no consumo de energia elétrica tenha sido utilizada no desenvolvimento deste trabalho, é importante mencionar que o ESP32 possui diversas formas de economia de energia que podem ser utilizadas para reduzir o consumo e aumentar a sua autonomia. No entanto, algumas dessas formas preveem o desligamento da comunicação *wireless*, o que pode romper a comunicação dos nós com o nó raiz na rede *mesh*. De modo

geral, o hardware utilizado no projeto ofereceu um bom desempenho e autonomia, atendendo às necessidades do projeto.

4.2 Software

O ESP32 pode ser programado através de diversos ambientes integrados de desenvolvimento. Neste projeto, escolhe-se utilizar o Visual Studio Code, um editor de código simplificado com suporte para operações de desenvolvimento como depuração, tarefas e controle de versão. Utilizou-se a linguagem C++, baseada em C, que aceita múltiplos paradigmas de programação e permite um desenvolvimento rápido para plataformas embarcadas, tendo requisitos mínimos de memória e processamento. Utilizou-se o *framework* do Arduino, disponibilizado pela fabricante para interação com o ESP32.

Para a compilação e o gerenciamento da interação entre o ambiente de desenvolvimento e o microcontrolador, utilizou-se o *plugin Platform.IO* (Platformio, 2022). Desenvolvido em Python, o *Platform.IO* é integrado ao Visual Studio Code e oferece ferramentas para compilação cruzada, além de interagir com o kit de desenvolvimento para criar e atualizar arquivos internos, apagar e reescrever a memória e monitorar o programa através da interface serial.

As opções de desenvolvimento são definidas em um único arquivo com extensão *INI*, onde são armazenados parâmetros como plataforma alvo, bibliotecas utilizadas e parâmetros de comunicação. A estrutura de arquivos segue um padrão, onde o nome dos diretórios define suas funções. O diretório *data* contém os arquivos destinados ao armazenamento interno e o diretório *src* contém os códigos-fontes. Para o desenvolvimento deste trabalho foram criados dois projetos diferentes, um para o nó raiz e outro para os nós clientes. O *plugin Platform.IO* também é responsável por fazer o download, instalar e atualizar as bibliotecas, tornando desnecessária a instalação manual de quaisquer bibliotecas utilizadas no projeto, bastando o nome da biblioteca (se ela fizer parte do repositório do *plugin*) ou seu endereço GIT - GitHub (Github, 2023).

4.3 Criação e testes da rede *mesh*

A biblioteca escolhida para criar e manter a rede *mesh* foi a *painlessMesh*, uma biblioteca de código aberto que utiliza a implementação de controle de rede sem fio da fabricante do microcontrolador ESP32, garantindo velocidade e estabilidade na rede. Para utilizar a biblioteca, foi necessário definir um nome e um canal para a rede. É importante ressaltar que o canal da rede pode limitar o uso de *gateways* para a Internet, já que o ESP32 não consegue

trabalhar com dois canais diferentes. Dessa forma, a rede que dá acesso à Internet deve estar trabalhando no mesmo canal da rede *mesh*.

Durante a implementação foi observado que o ESP32 utilizado (versão *WROOM-32*) tem limitações quanto ao uso de segurança no ponto de acesso de saída. Qualquer criptografia imposta à rede impedia o ESP32 de conectar ao ponto de acesso. Além disso, o acesso à rede anunciado pelo nó raiz exige que os dispositivos clientes trabalhem com IP (*Internet Protocol*) fixo, visto que a rede opera sob o protocolo TCP (*Transmission Control Protocol*), não tendo o protocolo IP. Em outras palavras, os nós não possuem endereços IPs, sendo endereçados através de seus respectivos números seriais, os quais são únicos.

Para contornar este problema, foi utilizado um pequeno ponto de acesso com restrições, permitindo apenas o acesso do ESP32 raiz através de uma lista branca de endereços físicos de rede. Este ponto de acesso foi configurado para fornecer um endereço de IP fixo ao ESP32, facilitando os testes da rede *mesh*. Em uma implementação definitiva, pode-se utilizar o servidor DHCP (*Dynamic Host Configuration Protocol*) da rede para atribuir um endereço estático ao nó raiz e configurar o ponto de acesso para encaminhar os pacotes DHCP ao roteador principal. Este ponto de acesso pode ser acessado pelos clientes para conectar a interface de visualização. Na figura 4 é possível visualizar o nó raiz (conectado via USB ao computador), os nós clientes e o ponto de acesso em funcionamento.

Figura 4 – Nó raiz, nós clientes (sensores), bateria e o AP (*Access Point*) utilizado nos testes.



Fonte: Os próprios autores.

O envio de mensagens foi testado usando um dos botões do hardware presentes no microcontrolador ESP32, permitindo testar o envio e o recebimento de mensagens de cada nó, processá-las e exibi-las no monitor serial do nó raiz. As mensagens eram enviadas via *broadcast*

para todos os nós da rede, contendo informações como o emissor, sendo estas apenas conjuntos de caracteres em um pacote JSON (*JavaScript Object Notation*).

Para testes mais elaborados, foi utilizado um aplicativo para Android publicado pelo criador da biblioteca *painlessMesh* na loja de aplicativos Play Store. O aplicativo permitiu ao *smartphone* fazer parte da rede *mesh*, enviando e recebendo mensagens, e foi muito útil nos primeiros testes. A figura 5 exibe o terminal serial do nó raiz durante a conexão com o ponto de acesso e quando um nó cliente novo é conectado na rede *mesh*.

4.4 Coleta dos dados através dos sensores

Tanto os nós clientes quanto o nó raiz da rede *mesh* são responsáveis por capturar os dados dos sensores. No caso deste trabalho, o sensor escolhido para os testes foi o sensor de efeito *hall* integrado ao ESP32, que mede o campo eletromagnético e retorna um valor inteiro, indicando a polaridade e a intensidade do campo. Durante os testes, foi possível verificar a variação do valor em tempo real na interface web de visualização ao aproximar ou afastar o ímã do sensor *hall*. Para obter o valor do sensor, utilizou-se a função *hallRead* disponível no *framework* Arduino para o ESP32.

Qualquer outro sensor pode ser facilmente interligado aos pinos do ESP32 e seu valor acessado e enviado à rede *mesh*, bastando para isso inserir a biblioteca correspondente e adaptar a chamada da função. Também foram realizados testes com um sensor BMP180, utilizando a biblioteca *Adafruit BMP085.h*. A temperatura foi acessada e enviada pelo nó cliente em substituição à medição do sensor de efeito *hall*. O nó raiz, por sua vez, recebeu e exibiu o dado normalmente na interface web desenvolvida.

Figura 5 - Captura do terminal serial com as informações do nó raiz durante a conexão.

```
CONNECTION: connectToAP(): Best AP is 276482496<---
CONNECTION: connectToAP(): Trying to connect, scan rate set to 4*normal
Meu IP de estação é 192.168.2.100
CONNECTION: eventSTAGotIPHandler: ARDUINO_EVENT_WIFI_STA_GOT_IP
CONNECTION: New AP connection incoming
CONNECTION: painlessmesh::Connection: New connection established.
CONNECTION: newConnectionTask():
CONNECTION: newConnectionTask(): adding 3878256257 now= 89303811
Conexões mudaram...
Quantidade de nós: 2
Lista de nós: 2991085985 3878256257
Ajustando relógios 1547120 Diferença = -87792515
Ajustando relógios 1809889 Diferença = 12440
Ajustando relógios 2313789 Diferença = 64581
Ajustando relógios 2682754 Diferença = -3189
CONNECTION: stationScan(): BridgeTCPMesh
CONNECTION: eventScanDoneHandler: ARDUINO_EVENT_WIFI_SCAN_DONE
CONNECTION: scanComplete(): Scan finished
CONNECTION: scanComplete():-- > Cleared old APs.
```

Fonte: Os próprios autores.

4.5 Envio dos dados coletados para o nó raiz

Nos nodos clientes, o valor do sensor *hall* é coletado e enviado para o nodo raiz da rede. Inicialmente, o valor era lido e enviado em *broadcast* para todos os nós a cada 1 segundo, o que poderia causar colisões e perda de dados, já que não havia confirmação de recebimento no fluxo TCP. Para resolver esse problema, optou-se por enviar os dados diretamente para o nó raiz, que foi definido como constante global nos nós clientes através de seu identificador serial. Já para evitar colisões durante o envio, foi adotado um intervalo pseudo-aleatório de tempo entre 1000 e 3000 milissegundos para cada transmissão. Essa abordagem aumentou a eficiência na transmissão dos dados.

A recepção dos dados foi testada e aprovada através da atribuição de índices aos envios e comparação destes índices no nodo raiz. Este algoritmo de depuração foi removido após constatar a recepção bem-sucedida de todos os pacotes enviados em um intervalo de algumas horas, em uma rede *mesh* formada por 3 nodos clientes e o nodo raiz.

4.6 Tratamento e visualização dos dados

A coleta, processamento e visualização dos dados do sistema ocorrem de forma eficiente e escalável. Os valores dos sensores são armazenados em uma estrutura de dados global que contém o identificador serial do emissor. Diversas funções acessam essa estrutura, como a que atualiza seus valores, a que promove a busca de índices e a que realiza a leitura dos valores para construir a sequência de caracteres JSON. O nó raiz lê seu sensor da mesma forma que os nós clientes, porém seu valor é diretamente escrito na estrutura de dados global.

A rede *mesh* responde a uma chamada de função que retorna a lista de nós hierarquizada em formato JSON. A sequência de caracteres JSON retornada pela rede é manipulada para inserir os valores de sensores. A manipulação do tipo de dado *string* é mais rápida e consome menos memória dos microcontroladores do que armazenar os dados em estruturas JSON próprias do Arduino, pois estas precisariam ser convertidas em diversos tipos de dados para serem manipuladas posteriormente. Além disso, utilizar a *string* diretamente torna as funções menos complexas e mais fáceis de serem tratadas.

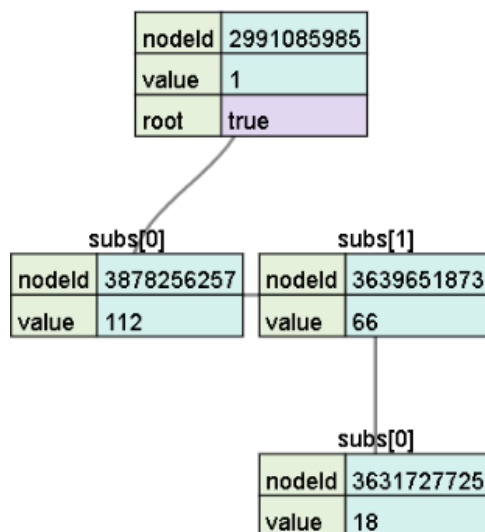
Os dados dos sensores são simplesmente concatenados logo após o identificador serial dos nós, mantendo a hierarquia do arquivo e evitando a necessidade de recriar objetos JSON a cada modificação na rede. A sequência de caracteres é armazenada em uma variável global, atualizada apenas ao final de todo o processamento para evitar que uma *string* parcialmente preenchida ou com valores faltando seja exibida na interface web.

O microcontrolador ESP32 possui um espaço reservado para o armazenamento em memória *flash*, compartilhado com o tamanho da aplicação, mas esse espaço é utilizado apenas para escrever os arquivos HTML, Javascript e CSS da aplicação web de visualização. O armazenamento em memória *flash* é lento e tem número limitado de escritas, por isso não foi utilizá-lo para gravar os dados dos sensores.

O servidor web usado é a biblioteca *ESPAsyncWebServer* e que responde às requisições HTTP com os códigos corretos e serve dados em caracteres vindos de leitura de variáveis disponíveis no escopo da função. A variável que contém a sequência de caracteres JSON formatada com os dados é global para que possa ser acessada por esta função, tornando a resposta mais rápida.

Para criar uma visualização hierárquica dos dados em um navegador web, foi utilizada a biblioteca *vtree.js* (IVANOV, 2019), a qual faz uso da biblioteca *D3.js* para criar uma árvore de dados. A *vtree.js* aceitou o JSON formatado e o apresentou em forma de um mapa de vários níveis, facilitando a interpretação dos dados e atualizando automaticamente o gráfico quando houve mudanças na hierarquia da rede. A figura 6 mostra o resultado da renderização da árvore hierárquica, com os identificadores serias de cada um dos nós e os dados coletados dos sensores. Destaca-se a relevância da atualização automática da página web em intervalos regulares de 1 segundo, permitindo assim a apresentação das alterações que eventualmente se manifestem na rede *mesh*. Tal funcionalidade possibilita ao usuário a visualização em tempo real dos dados coletados pelos sensores, sem a necessidade de acessar uma base de dados ou planilhas eletrônicas para tal propósito.

Figura 6 – Interface web exibindo os dados coletados pelos sensores (variável *value*).



Fonte: Os próprios autores.

Na figura 6, pode-se observar que as arestas representam as conexões diretas entre os nós. O nó raiz da árvore (*nodeId*: 2991085985) representa o nó principal da rede *mesh* de sensores sem fio. Neste nó estão conectados dois nós clientes com os identificadores seriais 3878256257 e 3639651873, respectivamente. O outro nó cliente (3631727725) está conectado ao nó com *id* 3639651873. De modo geral, o sistema de coleta, processamento e visualização de dados é eficiente e escalável, tornando a interface do usuário amigável e fácil de usar.

Ademais, a fim de avaliar a funcionalidade da interface web de exibir mais nodos e a capacidade de transmissão de mais dados na rede *mesh*, foram incorporados a ela, além dos 4 nodos preexistentes, 3 *smartphones*. A interface web passou a exibir os novos equipamentos conectados à rede, apresentando unicamente seus identificadores seriais (*nodeId*), visto que os referidos equipamentos não dispõem de sensores para mensurar o campo eletromagnético.

5 Considerações finais

Observa-se um expressivo interesse na utilização de microcontroladores como nodos de redes *mesh* sem fio, com uma vasta gama de recursos disponíveis para a integração desses dispositivos. Tal interesse é evidenciado por meio da existência de bibliotecas como a *painlessMesh*, fóruns e listas de discussão digitais. A coleta de dados do ambiente por meio de sensores também é objeto de ampla documentação, inclusive pelos próprios fabricantes dos componentes de hardware. Não obstante, estudos prévios têm se dedicado na otimização do consumo energético desses dispositivos de hardware, no desenvolvimento de protocolos IoT e concentradores para a coleta de dados em redes *mesh* de sensores sem fio.

Nesse sentido, o objetivo principal deste trabalho foi monitorar, em tempo real, os dados provenientes dos múltiplos sensores dispersos em um determinado ambiente, mantendo uma visualização simples e atualizada desses dados. Para alcançar tais objetivos, optou-se por concentrar o processamento e o serviço de visualização dos dados no nodo raiz, sem que houvesse perda da funcionalidade de coleta de dados nesse mesmo nodo, o que proporcionou uma economia de energia e uma diminuição na complexidade do sistema. Ademais, realizou-se a integração do nó raiz com uma rede TCP/IP já existente, mantendo, contudo, a rede *mesh* isolada em sua topologia TCP, o que possibilitou um acesso mais simples aos dados, sem a necessidade de outras plataformas, aplicativos ou dispositivos específicos.

Os objetivos deste trabalho foram todos contemplados uma vez que foi criada uma rede *mesh* de sensores sem fio para a realização dos testes, sendo esta estável e escalável, assim como uma interface web para a visualização das informações coletadas pelos sensores para o usuário final. A rede *mesh* foi concebida a partir da utilização do ESP32 e do sensor *hall*, já

incluso no próprio microcontrolador. Por sua vez, a interface web permitiu exibir os dados de forma clara e em tempo real, no intervalo de tempo de 1 segundo; e é acessível por meio de um simples navegador web, não havendo a necessidade de instalação de hardware ou software adicionais nos dispositivos.

Aprimoramentos podem ser adicionados neste projeto, como por exemplo, a aquisição de kits de desenvolvimento mais modernos, contendo novas versões do ESP32, com maior desempenho e menor consumo energético, assim como a implementação de novos protocolos para a transmissão de dados sem fio e com enfoque nas questões de segurança (criptografia de dados). Ademais, a otimização do circuito de alimentação pode aumentar a autonomia dos nós, enquanto outras formas de alimentação, como o uso de painéis solares, por exemplo, podem ser consideradas. É possível encontrar no mercado kits de desenvolvimento que contam com alimentação integrada por meio de baterias do tipo 18650. No que concerne a coleta de dados dos sensores, esta pode ser realizada através de outros dispositivos que possam ser conectados ao microcontrolador ESP32, ou mesmo ser introduzida de forma manual, por meio de uma interface em cada nó. Isso torna o projeto versátil e adaptável a diversas aplicações, voltadas para outras áreas do conhecimento.

Em suma, este trabalho se coloca como um projeto intermediário entre uma utilização genérica de redes *mesh* de sensores e uma aplicação específica e otimizada para a visualização dos dados coletados pelos sensores distribuídos no ambiente. Os objetivos foram definidos com base no tempo hábil para a implementação e a realização dos testes. No entanto, acredita-se que este trabalho possa servir como base para a criação de soluções mais específicas e otimizadas em diferentes ambientes e aplicações.

Referências

AKYILDIZ, I. F.; WANG, X.; WANG, W. **Wireless mesh networks: a survey**. Computer networks, 47(4), pp. 445–487, 2005.

BRAGA, T. C. Monitorização ambiental em espaços florestais com rede de sensores sem fios. PhD thesis, Universidade da Madeira (Portugal), 2010.

BLOG CURTO CIRCUITO. **Conhecendo o ESP32**, Guarulho/SP, 31 out. 2018. Disponível em: <https://curtocircuito.com.br/blog/Categoria%20IoT/conhecendo-esp32>. Acesso em: 17 abr. 2023.

DA SILVA JÚNIOR, H. S. Sensores digitais e analógicos: pesquisa, avaliação e documentação. In: Seminário de Iniciação Científica. 6., 2012. Goiânia. **Anais...** Goiânia: IFG, 2012. Disponível em:

<https://conferencias.ifg.edu.br/seminarioict/SICT/paper/view/329/48>. Acesso em: 19 dez. 2023.

DOTTA, F. P.; SEVERO, L. C. Estudos e aplicações iniciais do processador esp32 em sistemas de internet das coisas-Iot. In: Salão Internacional de Ensino, Pesquisa e Extensão. 12., 2020. **Anais...** Unipampa, 2020. Disponível em: https://ei.unipampa.edu.br/uploads/evt/arq_trabalhos/22464/etp2_resumo_expandido_22464.pdf. Acesso em: 19 dez. 2023.

FRANZIN LEITE, M. A. *et al.* Proposta de automação de baixo custo para aviários utilizando o microcontrolador esp32. In: Congresso de AgroInformática-CAI. 12., 2020. **Anais...**La Plata: UNLP, 2020. Disponível em: http://sedici.unlp.edu.ar/bitstream/handle/10915/115670/Documento_completo.pdf?sequence=1. Acesso em: 19 dez. 2023.

GITHUB, Inc. Disponível em: <https://github.com>. Acesso em: 18 abr. 2023.

IMEJUNIOR. **Microcontroladores ESP**. Rio de Janeiro: Instituto Militar de Engenharia (IME), Disponível em: <https://imejunior.com.br/2019/10/17/conheca-a-familia-de-microcontroladores-esp/>. Acesso em: 19 dez. 2023.

IVANOV, I. **Ivan111/vtree: Json tree visualizer**, 20 abr. 2019. Disponível em: <https://github.com/ivan111/vtree>. Acesso: 21 set. 2022.

LOUREIRO, A. A. *et al.* Redes de sensores sem fio. In Simpósio Brasileiro de Redes de Computadores (SBRC). 21, 2003. **Anais...** Natal: SBRC, 2003, pp. 179–226, 2003. Disponível em: <https://www.dcc.ufmg.br/~loureiro/cm/docs/sbrc03.pdf>. Acesso em: 19 dez. 2023.

MAIA, E. W. **Redes mesh**. Router & Wireless Systems, 2008.

MARTILLANO, D. A. *et al.* Development of a simulated portable mesh network via esp8266-based devices with utility application. **Journal of Communications**, v. 17, n. 5, 2022. Disponível em: <http://www.jocm.us/uploadfile/2022/0420/20220420030107917.pdf>. Acesso em: 19 dez. 2023.

MARTIN, G. **gmag11/painlessmesh**, 05 ago. 2019. Github. Disponível em: <<https://github.com/gmag11/painlessMesh>>. Acesso: 21 set. 2022.

NURLAN, Z. *et al.* Wireless sensor network as a mesh: vision and challenges. **IEEE Access**, v. 10, n. 46, p. 46-67, 2021. Disponível em: <https://ieeexplore.ieee.org/iel7/6287639/6514899/09656902.pdf>. Acesso em: 19 dez. 2023.

OLIVEIRA JÚNIOR, M.; DUARTE, R. O. **Apostila sobre introdução ao projeto com microcontroladores e programação de periféricos**. Belo Horizonte: UFMG, Departamento de Engenharia Eletrônica, 2010. Disponível em: <https://www.academia.edu/download/34567061/Apostila-MCU.pdf>. Acesso em: 19 dez. 2023.

PATHAK, P. H.; DUTTA, R. *Designing for Network and Service Continuity in Wireless Mesh Networks*. Nova Iorque: Springer, 2013.

PFEIFER, E. A. Uma rede de sensores sem fio em malha para monitoramento on-line do consumo de energia nos edifícios da Universidade Federal de Santa Catarina: um estudo de caso. 2018. 127 f. Trabalho de Conclusão de Curso (Graduação) - Departamento de Engenharia de Computação, Araranguá, 2018. Disponível em: https://repositorio.ufsc.br/bitstream/handle/123456789/192373/TCC_Eduardo_Pfeifer_ed.pdf?sequence=3. Acesso em: 19 dez. 2023.

PLATFORMIO. **Platformio is a professional collaborative platform for embedded development**. Disponível em: <https://platformio.org/>. Acesso: 21 set. 2022.

ROSA, A. **Internet of things (iot) aplicações, desafios e tendências (palestra)**. PMI3817 – Empreendedorismo e Inovação em Engenharia Universidade de São Paulo, e-Disciplinas, 2020. Disponível em: https://edisciplinas.usp.br/pluginfile.php/5199901/mod_resource/content/1/IoT%20-%20Alexandre%20Rosa%20-%20Angoera.pdf. Acesso em: 19 dez. 2023.

SANNI, M. *et al.* **How to model wireless mesh networks topology**. In: IOP Conference Series: **Materials Science and Engineering**, v. 53, n.1, 2013. Disponível em: <https://iopscience.iop.org/article/10.1088/1757-899X/53/1/012037/pdf>. Acesso em: 19 dez. 2023.

SECCO, R. A. M. **Desenvolvimento de uma rede de sensores sem fio em topologia mesh utilizando dispositivos Iot**. 2019. 66 f. Trabalho de Conclusão de Curso (Graduação) – Ciência da Computação, Universidade Federal Tecnológica do Paraná-UFTPR, Medianeira, 2019. Disponível em: <https://repositorio.utfpr.edu.br/jspui/bitstream/1/12516/2/redessensoresetopologiamesh.pdf>. Acesso em: 19 dez. 2023.

SICHITIU, M. L. Wireless mesh networks: opportunities and challenges. In **Proceedings of World Wireless Congress**, 2005. Disponível em: https://www.researchgate.net/profile/Mihail-Sichitiu/publication/228525550_Wireless_mesh_networks_opportunities_and_challenges/link/s/0fcfd50f41892962ee000000/Wireless-mesh-networks-opportunities-and-challenges.pdf. Acesso em 19 dez. 2023.

SILVA, S. R. C. **Sistema de monitoramento e controle de energia utilizando o microcontrolador esp32**. 2021. 62 f. Trabalho de Conclusão de Curso (Graduação) – Engenharia Elétrica, Universidade Federal Tecnológica do Paraná-UFTPR, Ponta Grossa, 2021. Disponível em: <http://repositorio.utfpr.edu.br/jspui/bitstream/1/29084/4/monitoramentocontroleenergiamicrocontrolador.pdf>. Acesso em: 19 dez. 2023.

TELES, B. M. N. **City on stats: visualização imersiva de dados recolhidos por rede de sensores móveis**. 2019. 116 f. Dissertação (Mestrado) - Departamento de Ciências e Tecnologias da Informação, Instituto Universitário de Lisboa, Lisboa, 2019. Disponível em: https://repositorio.iscte-iul.pt/bitstream/10071/20129/1/Master_Bruno_Nunes_Teles.pdf. Acesso em: 19 dez. 2023.

TOCCI, R. J.; WIDMER, N. S.; MOSS, G. L. **Sistemas digitais: princípios e aplicações**. 12. ed. Pearson Universidades, 2019.

URBANETZ JÚNIOR, J.; MAIA, J. S. **Eletrônica Aplicada**. Curitiba: Base, 2010.

YOUNG, H. D.; FREEDMAN, R. A. **Física I: mecânica**. 14. ed. Pearson, 2016.