UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA E ELETRÔNICA
CURSO DE ENGENHARIA ELÉTRICA

George Antonio Paes de Barros

Desenvolvimento de Dispositivo IoT para Monitoramento Autônomo de Temperatura

Florianópolis
2023

George Antonio Paes de Barros

Desenvolvimento de Dispositivo IoT para Monitoramento Autônomo de Temperatura

Trabalho de Conclusão de Curso submetido ao curso de graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal de Santa Catarina como requisito parcial para a obtenção do título de Bacharel em engenharia elétrica.

Orientador: Prof. Richard Demo Souza, Dr.

Florianópolis

2023

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Paes de Barros, George Antonio
Desenvolvimento de Dispositivo IoT para Monitoramento Autônomo de Temperatura / George Antonio Paes de Barros ; orientador, Richard Demo Souza, 2023.
104 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro Tecnológico, Graduação em Engenharia Elétrica, Florianópolis, 2023.

Inclui referências.

1. Engenharia Elétrica. 2. Monitoramento ambiental. 3. Temperatura. 4. IoT. 5. Internet das Coisas. I. Souza, Richard Demo. II. Universidade Federal de Santa Catarina. Graduação em Engenharia Elétrica. III. Título.

George Antonio Paes de Barros

Desenvolvimento de Dispositivo IoT para Monitoramento Autônomo de Temperatura

Este Trabalho Conclusão de Curso foi julgado adequado para obtenção do Título de "Bacharel em Engenharia Elétrica" e aceito, em sua forma final, pelo Curso de Graduação em Engenharia Elétrica.

Florianópolis, 06 de julho de 2023.

Documento assinado digitalmente
Miguel Moreto
Data: 06/07/2023 18:15:47-0300
CPF: ***.850.100-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Miguel Moreto, Dr.
Coordenador do Curso de Graduação em Engenharia Elétrica

Banca Examinadora:

Documento assinado digitalmente
Richard Demo Souza
Data: 06/07/2023 19:01:29-0300
CPF: ***.267.379-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Richard Demo Souza, Dr.
Orientador
Universidade Federal de Santa Catarina

Documento assinado digitalmente
BARTOLOMEU FERREIRA UCHOA FILHO
Data: 10/07/2023 10:53:14-0300
CPF: ***.362.114-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Bartolomeu Ferreira Uchôa Filho, Dr.
Universidade Federal de Santa Catarina

Documento assinado digitalmente
GUILHERME LUIZ MORITZ
Data: 06/07/2023 19:04:17-0300
CPF: ***.463.239-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Guilherme Luiz Moritz, Dr.
Universidade Tecnológica Federal do Paraná

AGRADECIMENTOS

Agradeço à Sensorweb Tecnologia de Sistemas de Informação S/A, por ter apoiado conceitual, material e financeiramente o desenvolvimento deste projeto, bem como o meu próprio desenvolvimento profissional.

Agradeço aos meus pais por todo o apoio que me prestaram e pela compreensão das minhas limitações durante esta longa jornada.

Agradeço à minha namorada, Gabrielle, pelo companheirismo, pela paciência, pelas ocasiões em que “levamos o TCC para passear” realizando ensaios, e pela ajuda valiosa com a formatação deste documento.

Agradeço aos amigos e colegas que, direta ou indiretamente, colaboraram comigo durante o curso e me motivaram a chegar até aqui.

RESUMO

Este Trabalho de Conclusão de Curso descreve o desenvolvimento de um dispositivo da Internet das Coisas para monitoramento autônomo de temperatura, para aplicações móveis bem como estacionárias, comunicando-se através de Wi-Fi e/ou rede celular LTE com uma plataforma comercial já existente, assim como a necessidade de tal desenvolvimento, que contemplou a seleção de componentes e dispositivos, com base em especificações definidas *a priori*, prototipagem com módulos de desenvolvimento, passando pelo desenvolvimento de *firmware* e *layout* de placa de circuito impresso, e por fim alguns ensaios verificando o adequado funcionamento do protótipo resultante desse desenvolvimento. Os resultados obtidos através dos ensaios são analisados, discutidos e pontos de melhoria e desenvolvimento futuros são apontados.

Palavras-chave: monitoramento; temperatura; Internet das Coisas.

ABSTRACT

This Bachelor's Thesis describes the development of an Internet of Things device for the autonomous monitoring of temperature, for mobile applications as well as stationary, communicating with a pre-existing commercial platform through Wi-Fi and/or LTE cellular networks, and also the rationale for such development, which included component and device selection, based on *a priori* specifications, prototyping using development modules, firmware development and printed circuit board layout, and finally a few experiments verifying whether the prototype resulting from such development functions as intended. The results obtained through these experiments are analyzed, discussed, and points for improvement and further development are pointed out.

Keywords: temperature; monitoring; Internet of Things.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

3G/4G/5G	3ª/4ª/5ª Geração
3GPP	3rd Generation Partnership Project
A/D	Analógico/Digital
APN	Access Point Name
ASCII	American Standard Code for Information Interchange
AVC	Advanced Very-Low-Voltage CMOS
BMS	Battery Management System
BSD	Berkeley Software Distribution
CMOS	Complementary Metal Oxide Semiconductor
DC	Direct Current
EDA	Electronic Design Automation
FRAM	Ferroelectric Random Access Memory
FTP	File Transfer Protocol
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile Communications
HTTP	HyperText Transfer Protocol
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
IP	Internet Protocol
JTAG	Joint Test Action Group
LCO	Lithium Cobalt Oxide
LDO	Low Drop-out
LFP	Lithium Iron Phosphate
LGPL	Lesser General Public License
LPM	Low-Power Mode
LPWAN	Low-Power Wide-Area Network
LTE	Long-Term Evolution
M2M	Machine-to-Machine
MQTT	Message Queuing Telemetry Transport

NB-IoT	Narrowband-IoT
NMC	[Óxido de Lítio,] Níquel, Manganês e Cobalto
NTP	Network Time Protocol
PTC	Positive Temperature Coefficient
RF	Radiofreqüência
RTC	Real-Time Clock
SIM	Subscriber Identity Module
SRAM	Static Random Access Memory
TCP	Transmission Control Protocol
UART	Universal Asynchronous Receiver
UDP	User Datagram Protocol
URL	Uniform Resource Locators
USB	Universal Serial Bus
UTC	Coordinated Universal Time

SUMÁRIO

1 INTRODUÇÃO

Este Trabalho de Conclusão de Curso tem como objetivo o desenvolvimento de um protótipo de dispositivo IoT autônomo de monitoramento de temperatura, de acordo com especificações predeterminadas. Este dispositivo deve se comunicar com um serviço *cloud-based* (baseado na nuvem), através de conexão à Internet proporcionada por interfaces de telefonia celular e/ou Wi-Fi, enviando periodicamente ao serviço as medições de temperatura, que devem ser realizadas através de uma sonda em barramento 1-Wire. Os envios deverão usar o protocolo HTTP. Além disso, como medida de contingência contra interrupções da comunicação, o dispositivo deverá armazenar localmente as últimas medições realizadas, para que estas possam ser enviadas quando a comunicação for restabelecida.

O dispositivo resultante desse desenvolvimento tem aplicação estacionária, para monitoramento de instalações com poucos pontos de medição, bem como móvel, com uso pretendido especialmente no setor logístico, em veículos como caminhões e furgões.

O escopo deste Trabalho abrange desde a especificação de componentes, passando pela concepção e prototipagem do circuito eletrônico, desenvolvimento de *firmware*, até o projeto de uma placa de circuito impresso passível de ser fabricada em escala comercial, bem como uma estimativa dos custos do dispositivo.

1.1 MOTIVAÇÃO

Este tema de Trabalho de Conclusão de Curso surgiu a partir de uma demanda interna da Sensorweb Tecnologia de Sistemas de Informação S/A.

A Sensorweb é uma empresa de cerca de 60 funcionários, especializada em monitoramento ambiental - temperatura e umidade relativa do ar - da cadeia do frio, através de dispositivos IoT, com foco nas áreas de saúde e logística.

O monitoramento de condições ambientais, particularmente a temperatura, é de grande importância para o setor da saúde, já que este setor depende largamente de insumos sensíveis à temperatura, como produtos hemoderivados ou medicamentos termolábeis, isto é, que se decompõem ou perdem suas propriedades em resposta ao calor.

Recentemente, com a introdução da Resolução da Diretoria Colegiada n° 430 da Agência Nacional de Vigilância Sanitária, estipulou-se a obrigatoriedade de monitoramento e controle de temperatura para o transporte desses insumos (ANVISA, 2020), o que prestou importância legal a um fato de importância física, e ressaltou a necessidade de dispositivos capazes de realizar esse tipo de monitoramento de maneira móvel e autônoma.

Atualmente, os dispositivos usados pela Sensorweb para esta finalidade são produzidos por terceiros. Com o intuito de reduzir sua dependência desses fornecedores, foi tomada a decisão estratégica, por parte da empresa, de projetar e construir um protótipo de dispositivo de monitoramento autônomo de temperatura, a ser integrado a sua plataforma *cloud-based*, o desenvolvimento do qual constitui este Trabalho de Conclusão de Curso.

1.2 ESPECIFICAÇÕES

O projeto deve ter dupla finalidade: uso estacionário e uso móvel. Para ambos os casos, foram definidos os seguintes requisitos comuns:

- Compatibilidade com as sondas de temperatura 1-Wire usadas nos equipamentos da empresa, baseadas no sensor Dallas/Maxim DS18B20;
- Transmissão periódica das leituras obtidas para o servidor da empresa, com períodos típicos da ordem de 5 a 15 minutos, através de rede celular, seja como meio primário ou redundância;
- Funcionalidade de armazenagem dos últimos dados lidos (*datalogger*), para posterior transmissão em caso de falha de comunicação.

Para uso estacionário, foram definidos os seguintes requisitos adicionais:

- Alimentação por fonte DC externa;
- Comunicação primária via Wi-Fi.

Para uso móvel, foram definidos os seguintes requisitos adicionais:

- Alimentação através de bateria automotiva;

- Bateria interna como alimentação redundante.

Pretende-se atender a tais requisitos com apenas um projeto, desenvolvido de maneira suficientemente modular para que se possa escolher entre os conjuntos de características, omitindo ou incluindo partes/componentes de uma mesma placa de circuito impresso conforme o caso.

2 REVISÃO DO ESTADO DA ARTE

2.1 PRODUTOS COMERCIALMENTE DISPONÍVEIS

Alguns produtos comercialmente disponíveis foram pesquisados a título de contextualização e embasamento prévio.

2.1.1 Smetro WE e MB

A Smetro é uma empresa nacional, sediada em Florianópolis, que fabrica alguns aparelhos de monitoramento IoT. Entre estes equipamentos, encontra-se o Smetro WE, que tem a capacidade de medição de temperatura e umidade relativa do ar através de até duas sondas, e se comunica através de Wi-Fi, Ethernet e/ou Bluetooth. É alimentado por fonte externa e bateria interna de lítio, recarregável, de 700 mAh. Pode acumular até 8000 pontos de dados de medição em caso de falta de comunicação.

O Smetro MB, por sua vez, é um equipamento similar, mas que se comunica exclusivamente via rede celular. Serve também como rastreador GPS; fora isso, tem as mesmas características de medição e alimentação do Smetro WE, mas seu *datalogger* é limitado a 1024 pontos.

Ambos os equipamentos obrigatoriamente enviam os dados para a plataforma da Smetro, que pode redirecionar os dados para uma plataforma de terceiros (SMETRO, 2022).

2.1.2 Nextcon DSBWIFI e DTHWIFI

Também uma empresa nacional, a Nextcon oferece produtos que se comunicam via Wi-Fi e medem temperatura e umidade: seu produto DSBWIFI mede apenas temperatura, através de uma sonda externa, e o DTHWIFI mede temperatura e umidade, através de um sensor interno.

É alimentado exclusivamente por fonte externa, não tem funcionalidade de *datalogger*, e o intervalo de aquisição de dados é fixo em 2.5 minutos. Os dados são

disponibilizados localmente, e podem ser acessados via *browser* ou aplicativo de celular (NEXTCON, 2022).

2.1.3 DCMTech SW-10

Outra empresa nacional, a DCMTech possui em seu portfólio o DCM-10, que também se comunica via Wi-Fi e é capaz de medir temperatura e umidade através de uma sonda externa. É alimentado exclusivamente por fonte externa, e serve como *datalogger* com capacidade para 350 pontos de dados. Os dados coletados podem ser enviados através dos protocolos HTTP, MQTT ou SNMP para uma aplicação remota (DCMTECH, 2022).

2.1.4 Elitech RCW360

A Elitech é uma empresa chinesa com filial no Brasil. Seu equipamento RCW360 tem submodelos Wi-Fi e 4G, implementando comunicação em cada um destes protocolos, separadamente. O modelo 4G dispõe ainda de rastreamento GPS e *datalogger* com 100 mil pontos de dados, enquanto que o modelo Wi-Fi pode armazenar até 32 mil pontos.

Ambos medem temperatura e umidade através de sonda externa, e enviam os dados para um serviço *cloud-based.* Ainda, ambos os modelos podem ser alimentados por fonte externa ou bateria interna, com autonomia de até 3 meses com intervalo de aquisição de dados de uma hora (ELITECH, 2022).

2.1.5 Novus Logbox WiFi

O Logbox WiFi, da Novus, uma empresa originalmente fundada no Brasil, mas que hoje tem sua sede nos EUA, é um equipamento que pode medir diversas grandezas físicas, pois conta com interfaces de entrada analógica genéricas de 0-50 mV, 0-5 V, 4-20 mA etc., além de poder ser conectado a termopares de diversos tipos, sondas Pt100 etc.

Como o nome sugere, se comunica, via Wi-Fi, através dos protocolos MQTT ou Modbus TCP a uma aplicação remota, e implementa *datalogger* com 140 mil

pontos. Opera com fonte de alimentação externa ou com baterias "AA", mas não se comunica enquanto alimentado por baterias, ficando restrito à função de acúmulo local de dados (NOVUS, 2022).

2.1.6 King Pigeon S266

Este modelo, fabricado na China, se comunica exclusivamente via rede celular, aplicando os protocolos TCP/IP ou Modbus TCP. Tem 8 canais de sondas, que podem medir apenas temperatura ou temperatura e umidade, e armazena localmente até 200 mil pontos de medição.

É alimentado por fonte externa, ou bateria interna de 900 mAh (KING PIGEON, 2022).

2.1.7 Efento NB-IoT Temperature and Humidity Sensor

A Efento, empresa da Polônia, oferece diversos dispositivos de monitoramento de condições ambientais IoT, e o que mais se aproxima dos requerimentos deste projeto é seu NB-IoT Temperature and Humidity Sensor, que implementa comunicações através de Bluetooth e NB-IoT, uma tecnologia de comunicação de baixa potência e largura de banda estreita, que usa a infraestrutura de rede celular.

Mede temperatura e umidade através de um sensor interno, conta com *datalogger* de até 40 mil pontos, e é alimentado exclusivamente através de baterias de lítio primárias tipo "AA". Se integra ao serviço *cloud-based* do fabricante (EFENTO, 2022).

2.1.8 Kizy K2

A Kizy, empresa suíça, oferece este produto, primariamente concebido como rastreador de posição geográfica, mas com a funcionalidade opcional de medição de temperatura, implementando armazenamento local de até 35 mil pontos de medição e comunicação via celular e Wi-Fi.

É alimentado exclusivamente por baterias, com autonomia de, no mínimo, 10 dias. O acesso aos dados se dá através do serviço *cloud-based* do fabricante, mas dispõe de uma API aberta para comunicação com outras plataformas (KIZY, 2022).

Nota-se que a maioria das alternativas encontradas não implementa, num mesmo produto, comunicações WiFi e celular; assim, não dispondo de redundância de comunicação, em caso de interrupção do único meio de conexão à internet, perderia o monitoramento em tempo real durante a interrupção, enquanto que o projeto desenvolvido neste Trabalho prevê o uso de ambas as modalidades de comunicação, de forma redundante.

Além disso, muitas das alternativas encontradas são oferecidas como parte de um modelo de negócios efetivamente *Hardware-as-a-Service*, obrigatoriamente implicando em taxas periódicas pelo uso do *hardware*, já que este é programado para se integrar exclusivamente à plataforma do fabricante. Este projeto utiliza o protocolo HTTP para envios, por questão de especificação, e, portanto, será possível utilizá-lo em conjunto com qualquer plataforma que suporte esse tipo de envio de dados, embora a intenção primária seja a integração ao sistema da Sensorweb.

2.2 A INTERNET DAS COISAS

A Internet das Coisas, ou IoT, é definida por Reyes et al. (2019, p. 2, tradução nossa) como "a rede de coisas, com identificação clara de elementos, imbuída de inteligência de *software*, sensores e conectividade ubíqüa com a Internet [...] com a premissa básica de monitorar e controlar coisas de qualquer lugar do mundo".

Similarmente, Whitmore et al. (2014, p. 262, tradução nossa) dizem que "o conceito fundamental [da Internet das Coisas] é que objetos do dia-a-dia possam ser equipados com capacidades de identificação, sensoriamento, conexão em rede e capacidades de processamento que os permitam se comunicarem uns com os outros e com outros dispositivos e serviços através da Internet para atingir algum objetivo útil".

De acordo com Shin (2014), estas "coisas" e "objetos" podem ser desde dispositivos criados pelo ser humano, incorporando sensores de grandezas

relevantes ao seu funcionamento, até mesmo o próprio ser humano, com implantes de monitoramento médico, bastando ser uma "coisa" à qual possam ser atribuídos um endereço IP e a capacidade de transferir dados através de uma rede.

Ainda de acordo com Reyes et al. (2019), a Internet das Coisas é algo que apenas se tornou possível devido a uma confluência de fatores. Dentre os fatores citados, especialmente dignos de nota são a explosão da Internet móvel, que se desenvolveu sinergisticamente com os *smartphones* e originou a infraestrutura de comunicação ubíqüa necessária à emergência da Internet das Coisas, a explosão da tecnologia associada (sensores, unidades de computação com suficiente desempenho) a baixos custos, a digitalização de indústrias, serviços e processos de modo geral, e a computação baseada na nuvem (*cloud computing*).

A digitalização de indústrias, serviços e processos começou com o intuito de operar "sem papel" e que acabou por produzir grandes quantidades de dados passíveis de serem processados e analisados de maneira muito mais eficiente, e que criou uma demanda por maneiras mais automatizadas e inteligentes de adquirir e processar estes dados.

Por sua vez, a computação baseada na nuvem, definida por Foster et al. (2008, tradução nossa) como "um paradigma de computação distribuída em larga escala que é impulsionado por economias de escala, em que agrupamentos abstraídos, virtualizados, dinamicamente escaláveis e gerenciados de poder de computação, armazenamento, plataformas e serviços são entregues sob demanda para clientes externos através da Internet", e que é responsável por agregar dados provenientes das "coisas" e processá-los, de maneira facilmente escalável.

Embora não tenha sido inicialmente projetada para isso, a Internet intrinsecamente possibilita a integração das "coisas" de maneira útil, uma vez que os elementos conectados à Internet são unicamente identificados, como característica fundamental do protocolo sob o qual ela opera. Além disso, ainda que, originalmente, os elementos conectados através da Internet fossem majoritariamente computadores, o progresso da densidade de integração de circuitos, observado empiricamente pela Lei de Moore, possibilitou dotar dispositivos simples de considerável capacidade de processamento a baixos custos.

Desse modo, mesmo os dispositivos simples dos dias de hoje correspondem ou excedem a capacidade de processamento de computadores propriamente ditos da época em que a Internet foi concebida; sob vários aspectos, estes dispositivos

simples hoje são ou incluem computadores, sendo assim capazes de incorporar elementos de inteligência de *software*, em grau suficiente para que possam coletar dados de qualquer local do mundo, usufruindo de armazenamento destes dados em uma nuvem remota, onde inteligências de *software* adicionais podem, ainda, realizar processamentos sobre estes e grandes volumes de outros dados relevantes ao monitoramento e controle de algo, coletados de outros dispositivos.

Aliados a sensores e atuadores, faz sentido, portanto, que esses dispositivos possam tirar proveito da infraestrutura de comunicação proporcionada pela Internet de modo a aumentar o escopo e abrangência dessas inteligências, para que possam monitorar, analisar e controlar "coisas" e processos do mundo real, constituindo, assim, a Internet das Coisas.

2.3 CONECTIVIDADE VIA REDE CELULAR PARA IOT

Como mencionado anteriormente, o surgimento da Internet móvel foi um fator decisivo para a viabilização da Internet das Coisas. Da mesma forma que a própria Internet inicialmente não foi concebida para isto, também a rede celular foi originalmente concebida apenas para comunicação por voz: a primeira geração de telefonia celular, demonstrada em 1970, se baseava em uma rede analógica (NADEEM et al., 2021). A transição para um sistema digital - o chamado 2G, de *segunda geração* - se deu a partir do início da década de 90, e abriu as portas para a comunicação não apenas por voz, que passou a ser digitalmente encriptada, provendo maior segurança, mas também de outros dados, em particular através do GPRS, que oferece, entre outros serviços, suporte ao protocolo de comunicação da Internet, permitindo que dispositivos móveis façam proveito da ubiqüidade das redes celulares para acessar a Internet de qualquer lugar com cobertura de telefonia celular. Esta modalidade de comunicação, entre dispositivos de dados, também é conhecida como M2M.

Tal acesso, no âmbito de dispositivos IoT, geralmente é realizado através de *modems* ou de módulos de comunicação, estes implementando ao menos parcialmente a funcionalidade de *modems:* funcionam como interface entre o equipamento e a rede celular, mas não são dispositivos autocontidos, necessitando de alimentação dedicada - por vezes em mais de um nível de tensão - e conexões externas ao módulo, como à antena e ao cartão SIM, necessário à identificação e

autenticação do aparelho junto à rede, enquanto que um *modem* dedicado necessitaria apenas de uma conexão de dados ao equipamento que se pretende conectar à rede, além de uma fonte de alimentação comum.

A opção entre uma ou outra solução depende de fatores como a escala de produção e a necessidade de portabilidade, não sendo uma intrinsecamente superior à outra. Uma terceira opção seria o uso de um *chipset* dedicado, mas isto envolveria o projeto de um *frontend* RF, entre outras complexidades, e é uma solução que encontra maior aplicação quando há grande necessidade de integração e miniaturização, como em *smartphones* modernos. Dada a complexidade de tal solução, esta não foi considerada no escopo deste Trabalho.

Tanto *modems* quanto módulos muitas vezes se comunicam através de uma interface baseada no padrão RS-232, que pode, ainda, ser encapsulada dentro de uma interface USB. Devido ao maior nível de integração, que elimina a necessidade ou mesmo a desejabilidade de níveis lógicos elevados, os módulos se comunicam através de um protocolo serial assíncrono, baseado em RS-232, mas com níveis lógicos reduzidos (5, 3.3 ou 1.8 V). Tais módulos podem, ainda, contar com interfaces I2C ou PCIe, além de uma interface de áudio digital. Estas últimas funcionalidades não são, contudo, relevantes para o escopo deste Trabalho, já que não há necessidade de comunicações em alta taxa, dada a simplicidade e o longo período entre envios dos dados a serem transmitidos, nem tampouco necessidade de comunicação por voz.

Para comunicação baseada em RS-232, esses dispositivos utilizam o conjunto de comandos AT, que foi desenvolvido na década de 80 para coordenar a comunicação de *modems* de linha telefônica discada, e hoje é padronizado para a comunicação em telefonia celular pelo 3GPP (3GPP, 2022), o consórcio responsável por desenvolver protocolos para comunicações móveis. Este conjunto de comandos padronizado pode fazer parte de um superconjunto não-padronizado, que varia de fabricante para fabricante, adicionando mais funcionalidades. Por exemplo, um módulo pode implementar um *stack* TCP/IP e torná-lo acessível através de comandos AT, simplificando o desenvolvimento de um projeto que inclua esse módulo.

Naturalmente, esses dispositivos acompanharam a evolução da comunicação celular, e estão disponíveis nas principais tecnologias usadas hoje: 3G, 4G/LTE e 5G, com os equipamentos 3G caindo em desuso e os equipamentos

5G começando a ser largamente aplicados. Além disso, o padrão LTE também define tecnologias de comunicação LPWAN, mais adequadas a aplicações IoT, como o LTE Cat M e LTE Cat NB, este último também conhecido como NB-IoT. Um sumário das características dessas tecnologias é apresentado na Tabela 1, abaixo.

Tabela 1 - Comparação entre as tecnologias Cat M1 e Cat NB1

Tecnologia	LTE Cat M1	LTE Ca-NB1
Download	1 Mbps	30 kbps
Upload	1 Mbps	60 kbps
Largura de banda	1.4 MHz	200 kHz
Latência	Milissegundos	Segundos
Consumo	Maior	Menor
Custo	Maior	Menor
Handover	Sim	Não

Fonte: Adaptada de UBLOX (2023).

De um ponto de vista de velocidade de transferência de dados e latência, ambas as soluções atenderiam aos requisitos do projeto: os volumes de dados são pequenos, da ordem de centenas de bytes, com periodicidade típica de 5 minutos. Dada esta periodicidade, uma latência de alguns segundos seria aceitável. De um ponto de vista de consumo e custo, a tecnologia Cat NB seria vantajosa. Porém, esta não suporta o *handover*, que é a entrega ou transferência de uma conexão entre o dispositivo celular e a célula para uma outra célula sem que haja interrupção da conexão. Esta funcionalidade é importante para aplicações móveis, já que o dispositivo, ao se deslocar fisicamente, transita entre as áreas de cobertura de células diferentes.

Como as especificações do projeto prevêem o uso tanto estacionário quanto móvel, para que o projeto use uma das tecnologias celulares orientadas a IoT, faz-se necessário o uso de Cat M1; a tecnologia Cat NB1 pode opcionalmente ser suportada, quando a aplicação for sabidamente estática, em um regime caso-a-caso.

2.4 CONECTIVIDADE VIA WI-FI PARA IOT

Enquanto que a comunicação por rede celular, explicada anteriormente no âmbito de dispositivos IoT é, de fato, um modo de conexão sem fios, não é um modo local de realizar tal conexão, depende da infraestrutura de terceiros, e necessita pagamento de um pacote de franquia de dados para cada dispositivo: é um recurso quando não há alternativa viável, especialmente em aplicações móveis que dependam de conectividade em qualquer lugar. Quando se trata de uma aplicação de movimento restrito a um local delimitado ou mesmo uma aplicação estacionária, especialmente se forem instalados vários dispositivos, uma rede local pode se prestar melhor.

O padrão de rede local de propósito geral mais difundido na atualidade, com mais de 18 bilhões de dispositivos em operação (WI-FI ALLIANCE, 2022) é o chamado Wi-Fi, normatizado pelo padrão IEEE 802.11 e suas variantes. Originalmente introduzido em 1997, foi continuamente expandido em suas capacidades de largura de banda e técnicas de autenticação. É capaz de operar em diversas topologias de rede, diferentes freqüências e larguras de banda, e é concebido com o objetivo primário de prover conectividade à Internet, como um análogo sem fio às redes Ethernet, e como tal suporta nativamente o protocolo TCP/IP.

A larga difusão do Wi-Fi o torna interessante para aplicações IoT, que podem aproveitar infraestrutura pré-existente a um custo insignificante, e a revisão mais recente do padrão, de acordo com a Wi-Fi Alliance (2022), traz diversos recursos que podem ser úteis em aplicações IoT, como localização espacial, recursos de economia de energia, e modos de operação otimizados para baixa largura de banda. Por estes motivos, suporte a este padrão de comunicação sem fios faz parte das especificações do projeto.

Conectividade via Wi-Fi pode ser conseguida através do uso de *chipsets* dedicados, ou através de um microcontrolador que integre esta funcionalidade. Como o uso de *chipsets* implica, tal como no caso dos *modems* celulares, em um projeto complexo, esta alternativa não foi considerada no âmbito deste Trabalho; ao invés disso, foi usado um microcontrolador com Wi-Fi embutido.

Como exemplos de microcontroladores com essa funcionalidade, se pode citar alguns membros da série CC3200, da Texas Instruments, o RTL8195AM, da

Realtek, o CYW43340 da Infineon (antes Cypress) e os microcontroladores ESP8266 e ESP32, da Espressif. Todos estes suportam os padrões 802.11b/g/n, porém a Espressif é a empresa líder mundial, por fatia de mercado, de microcontroladores Wi-Fi (TSR *apud* ESPRESSIF, 2021), e seus microcontroladores são amplamente disponíveis a baixos custos em módulos pré-fabricados, inclusive com certificação Anatel. Além disso, há bibliotecas disponíveis que implementam os *stacks* TCP/IP e Wi-Fi, agilizando o desenvolvimento de soluções e assim proporcionando menor *time-to-market.*

2.5 PROTOCOLO HTTP

O HTTP é o principal protocolo para transmissão de páginas web, desenvolvido inicialmente para transferência de hipertexto, e mais tarde estendido para outros formatos de dados (FIELDING et al., 2022). É baseado num modelo de requisição e resposta sem armazenamento de estados: um cliente faz uma requisição a um servidor, que contém e provê os dados requisitados.

O padrão HTTP define alguns *métodos* a serem usados pelos clientes para que sejam realizadas as requisições, cada um indicando um propósito para estas requisições. Este projeto primariamente envia dados para um servidor remoto, e para essa finalidade são usados, em geral, um de dois métodos: GET ou POST.

O método GET, como o nome sugere, é usado principalmente para requisitar um envio de dados do servidor ao cliente. Isto é realizado através da própria URL da requisição, acrescentando parâmetros e seus valores. Este método seria adequado, por exemplo, para um mecanismo de busca, onde as palavras-chave a serem buscadas pelo servidor seriam enviadas como valores de algum parâmetro, e o servidor retornaria o conteúdo referente à busca.

Embora seja um método concebido para receber dados, nota-se que há o envio de dados – no exemplo dado, as palavras-chave que se deseja buscar – por parte do cliente; desta forma, o método GET também pode ser usado para enviar dados simples, ainda que em pequenas quantidades: requisições GET são limitadas a um comprimento de 2048 bytes. Além disso, não é um método seguro para envio de dados, uma vez que estes são claramente visíveis na URL, exclusivamente em codificação ASCII.

O método POST é usado explicitamente para enviar dados do cliente ao servidor. Ao contrário do método GET, os dados são armazenados no corpo da requisição, e não na URL; assim, não há limite de quantidade de dados, os quais podem ainda ser codificados e encriptados de maneiras arbitrárias.

2.6 ENDRIXX – PLATAFORMA SENSORWEB

Para melhor compreender a aplicação pretendida, faz-se necessário apresentar a plataforma *cloud-based* com que o projeto interage.

A plataforma desenvolvida pela Sensorweb para monitoramento, predominantemente, porém não exclusivamente, de temperatura e umidade, chamada “Endrixx”, é responsável por receber dados de medição através da Internet, usando os protocolos HTTP ou MQTT, registrar estes dados de maneira auditável, apresentá-los aos usuários e produzir alertas com base em critérios predeterminados de faixas de valores, tempo sem atualizações, etc.

Cada cliente da Sensorweb tem acesso a uma instância da plataforma Endrixx, para a qual os aparelhos de medição instalados nas dependências do cliente enviam os dados, devidamente identificados. A principal tela de interação com o usuário é a chamada *dashboard* (painel), mostrada na Figura 1, que apresenta as medições mais recentes recebidas pelo sistema, bem como eventuais alertas e acessos a outras partes da plataforma, como a Seção de emissão de relatórios de eventos e valores históricos.

Figura 1 - Dashboard da plataforma Sensorweb.

Fonte: o autor.

A partir do *dashboard*, é possível acessar todos os dados coletados de cada ponto individual, bem como o histórico de alertas, valores mínimo e máximo, e traçar os dados em função do tempo, dentro de intervalos configuráveis.

O Endrixx dispõe do que internamente são chamados *data sources* (fontes de dados): receptores HTTP e MQTT para coleta de dados, assim como fontes virtuais, entre outros tipos. Decidiu-se usar o receptor HTTP para obter os dados, por questão de simplicidade: além do HTTP ser um protocolo comum, e por isso ser fácil encontrar bibliotecas e dispositivos que o suportem, o Endrixx é capaz de adicionar novos dispositivos HTTP automaticamente, funcionalidade que ainda não foi implementada sob o protocolo MQTT.

O receptor de dados HTTP do Endrixx pode lidar com requisições do tipo GET ou do tipo POST. Uma vez que o volume de dados a ser transmitido é baixo, em formato de pares de parâmetros e valores, decidiu-se usar requisições pelo método GET, que já é usado por outros dispositivos que interagem com a plataforma. Para este método, o Endrixx espera dados no seguinte formato:

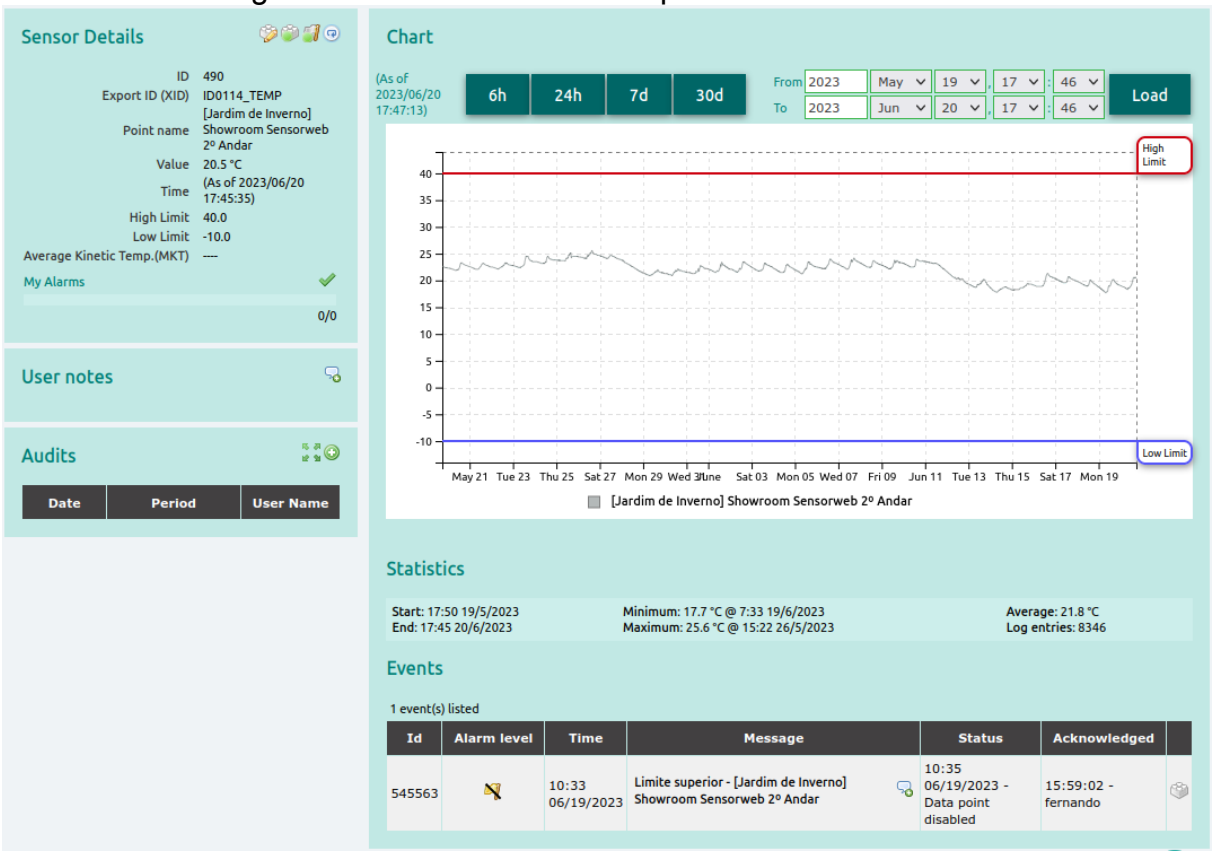
```
http://endpoint.sensorweb.com/nome_do_cliente/httpds?
__device=usuário:senha&variável1=valor1&variável2=valor2
```

Neste caso, dentro do ambiente de aplicação do cliente, seriam criados objetos chamados *datapoints* (pontos de dados) com os nomes “variável1” e “variável2”, caso até então não existissem, e, marcados com o instante em que

foram recebidos pelo servidor, seriam registrados os dados “valor1” e “valor2”, para as respectivas variáveis. Um novo envio em um instante posterior atribuiria valores novos a estas variáveis, com *timestamps* (marcações de horário) novas, mantendo os valores e *timestamps* anteriores; desta forma, os objetos *datapoint* contêm variáveis que são, na realidade, séries temporais.

É alocado, nos servidores que contêm as aplicações, tanto espaço quanto necessário para manter estas séries temporais por até 5 anos, período durante o qual permanecem disponíveis ao cliente para quaisquer consultas ou análises. Um exemplo de como os dados históricos podem ser exibidos, contemplando um período de um mês, é mostrado na Figura 2.

Figura 2 - Dados históricos na plataforma Sensorweb.



Fonte: o autor.

Alternativamente, é possível atribuir uma *timestamp* ao envio de dados. Isto é útil caso o dispositivo que envia os dados necessite enviá-los de modo retroativo – caso tenha momentaneamente perdido a conexão ao servidor, por exemplo, e tenha de realizar uma nova tentativa de envio posteriormente; neste caso, os dados que se pretende transmitir não correspondem ao instante de transmissão, mas a um

instante anterior. Para indicar este fato, adiciona-se à requisição GET a *timestamp* correspondente, no seguinte formato:

```
variável=valor@YYYY-MM-DDThh:mm:ssZ
```

onde “YYYY” denota o ano, com 4 dígitos, “MM” denota o mês em formato numérico, “DD” denota o dia do mês, “hh” denota a hora, em formato 24 horas, “mm” denota o minuto, e “ss” denota o segundo. Os horários são dados no fuso horário UTC, para evitar ter-se de lidar, localmente aos aparelhos, com questões de horário de verão, e a *timestamp* apensada da forma mostrada é atribuída pela plataforma apenas à variável que a precede, e não a eventuais outras em uma mesma requisição.

2.7 SENSOR DE TEMPERATURA DS18B20

Dentre as especificações para o projeto determinadas pela Sensorweb, uma das principais é a compatibilidade com as sondas de temperatura usadas pela empresa, baseadas no circuito integrado Dallas/Maxim DS18B20, que incorpora um sensor de temperatura, conversor analógico/digital com resolução configurável entre 9 e 12 bits, e toda a lógica de configuração e operação através de protocolo 1-Wire, com suporte a alimentação parasita (derivada da linha de comunicação), e que opera, nominalmente, de -55 a +125 °C, com acurácia de ±0,5 °C ao longo da faixa de -10 a +85 °C. Está disponível em encapsulamento TO-92, o que permite sua montagem dentro de sondas herméticas compactas, como a mostrada na Figura 3.

O DS18B20 opera numa faixa de tensão de 3,0 a 5,5 V e consome um máximo de 1,5 mA (MAXIM INTEGRATED, 2019).

Figura 3 - Sonda de temperatura usando o DS18B20.

Fonte: o autor.

2.8 PROTOCOLO 1-WIRE

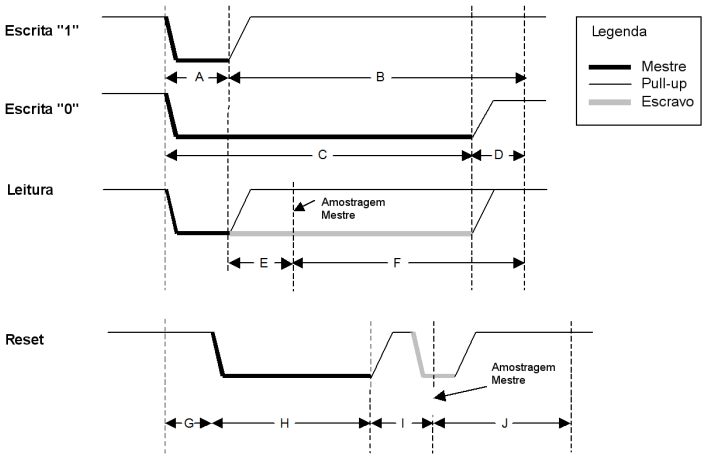
O protocolo 1-Wire é um protocolo serial desenvolvido pela Dallas Semiconductor (hoje Maxim Integrated) que usa, além do condutor de referência (terra), apenas um condutor para dados, em comunicação bidirecional *half-duplex,* isto é, com um dispositivo se comunicando por vez. Estas comunicações são iniciadas e controladas por um dispositivo mestre, e um barramento 1-Wire pode ter um ou mais dispositivos escravos. Cada dispositivo é identificado por um número de série único, armazenado na memória do dispositivo e acessível pelo dispositivo mestre.

Opcionalmente, pode haver mais um condutor, de alimentação, mas geralmente os dispositivos 1-Wire são capazes da chamada *alimentação parasita*, derivando a própria alimentação do sinal de comunicação. A linha de dados é mantida em nível lógico alto por um resistor de *pull-up* junto ao dispositivo mestre do barramento, o que permite que os dispositivos escravos permaneçam alimentados enquanto o barramento se encontra ocioso. Os dispositivos transmitem dados chaveando a linha em pulsos ao nível lógico baixo; os dispositivos escravos se mantêm alimentados, nestes instantes, quando no modo de alimentação parasita, por um capacitor interno, o que impõe um limite mínimo de taxa de transferência de dados (MAXIM, 2002).

A distinção entre os diferentes símbolos e comandos é feita através de pulsos com intervalos definidos: um valor "1" é escrito do mestre para o escravo através de um pulso curto. Um valor "0" é escrito na mesma direção através de um pulso longo. Além disto, o mestre pode ler um bit do dispositivo escravo com um pulso curto, após o qual há uma janela de tempo em que o dispositivo escravo pode deixar a linha de dados alta ou forçá-la baixa: o dispositivo mestre amostrará a linha dentro dessa janela de tempo, e a linha de dados alta será interpretada como um valor "1", ou "0" se esta se encontrar baixa. Além disso, um comando de *reset* pode ser atribuído ao(s) dispositivo(s) escravo(s) pelo mestre, a fim de detectar a

presença de dispositivos escravos no barramento e prepará-los para um comando, mantendo a linha em nível baixo por um determinado período de tempo. Caso haja algum dispositivo escravo no barramento, este deve, por sua vez, sinalizar sua presença ao mestre, mantendo a linha em nível baixo um certo período após o pulso de *reset* finalizar. Este sistema de comunicação é representado mais claramente pela Figura 4, a seguir.

Figura 4 - Diagramas de forma de onda do barramento 1-Wire.



Fonte: Adaptada de Application Note 126: 1-Wire Communication through Software (Maxim, 2002).

Cada intervalo, de A a J, tem durações ideais, dadas pela Tabela 2. Estes valores são os considerados padrão; há, ainda, um modo *overdrive*, com intervalos mais curtos, que proporcionam maiores taxas de transferência, mas que não será abordado aqui, pois os requerimentos do projeto não implicam necessidade dessas taxas de transferência mais elevadas.

Tabela 2 - Temporização do protocolo 1-Wire.

Intervalo	Duração (µs)
A	6
B	64
C	60
D	10
E	9
F	55
G	0
H	480
I	70
J	410

Fonte: Adaptada de Application Note 126: 1-Wire Communication through Software (Maxim, 2002)

Com as transferências elementares definidas acima, é possível implementar operações mais complexas, como o envio de palavras inteiras de vários bits, ou realizar a varredura do barramento em busca de dispositivos conectados.

2.9 BATERIAS

As especificações do projeto prevêem uma bateria como fonte de energia redundante para aplicações móveis. Diante disto, pode-se optar por uma bateria primária ou secundária (recarregável).

Não pareceu adequado, para este projeto, o uso de uma bateria primária: supondo-se que, por exemplo, um baú de caminhão, cuja temperatura fosse monitorada através deste projeto, fosse deixado num pátio por alguns dias, ao invés de a bateria ser simplesmente recarregada quando o baú tornasse a ser conectado à bateria/alternador do caminhão, esta teria de ser substituída, o que constituiria uma dificuldade logística e operacional. Optou-se, portanto, pelo uso de uma bateria secundária.

Baterias secundárias estão disponíveis em diversas formulações, com características e requerimentos variados. Entre as mais comuns, destacam-se as

baterias de chumbo-ácido (Pb), níquel-cádmio (NiCd), níquel-hidreto metálico (NiMH) e íons de lítio (Li-ion), estas subdivididas em tipos diferentes, de acordo com os materiais usados. Os tipos mais comuns são os baseados em óxido de lítio e cobalto (LCO ou $LiCoO_2$), níquel, manganês e cobalto (NMC), e lítio-ferro-fosfato (LFP ou $LiFePO_4$). A Tabela 3 reúne figuras de mérito encontradas em Linden e Reddy (2001), relativas a baterias portáteis, além de informações a respeito das baterias $LiFePO_4$ de acordo com o *datasheet* fornecido por Lithiumwerks (2021) e de acordo com Preger et al. (2020), para fins comparativos.

Tabela 3 - Características de baterias secundárias.

Tipo	Pb	NiCd	NiMH	Li-ion (LCO)	Li-ion (LFP)
Densidade energética	90 Wh/L	100 Wh/L	240 Wh/L	600 Wh/L	415 Wh/L
Tensão máxima por célula	2.4 V	1.35 V	1.35 V	4.2 V	3.6 V
Tensão nominal por célula	2.0 – 2.1 V	1.2 V	1.2 V	3.7 V	3.3 V
Tensão de fim de descarga	1.75 V	1.0 V	1.0 V	3.0 V	2.5 V
Taxa de auto-descarga	2-8%/mês	15-20%/mês	2-5%/mês	2%/mês	3%/mês
Vida útil	250-500 ciclos	300-700 ciclos	300-600 ciclos	>1000 ciclos	> 4000 ciclos

Fonte: adaptada de Linden e Reddy (2001), Lithiumwerks (2021), Preger et al. (2020)

Com base nas figuras de mérito expostas pela Tabela 3, e considerando que, para a aplicação pretendida, é desejável que o equipamento seja tão inconspícuo quanto possível, tal que opere de modo a não afetar as operações normais de carga, descarga e transporte de veículos em que venha a ser instalado, pode-se eliminar as baterias Pb e NiCd com base na baixa densidade energética de ambas. Ademais, ambas as tecnologias utilizam metais pesados e, como explicado em Linden e Reddy (2001), as baterias de NiCd sofrem do chamado "efeito memória", em que a curva de tensão em função da capacidade disponível desenvolve uma depressão em função do ciclo de Trabalho usual da bateria.

Embora essa depressão seja reversível, é um problema que requer tratamento específico e periódico; as demais tecnologias não sofrem desse efeito.

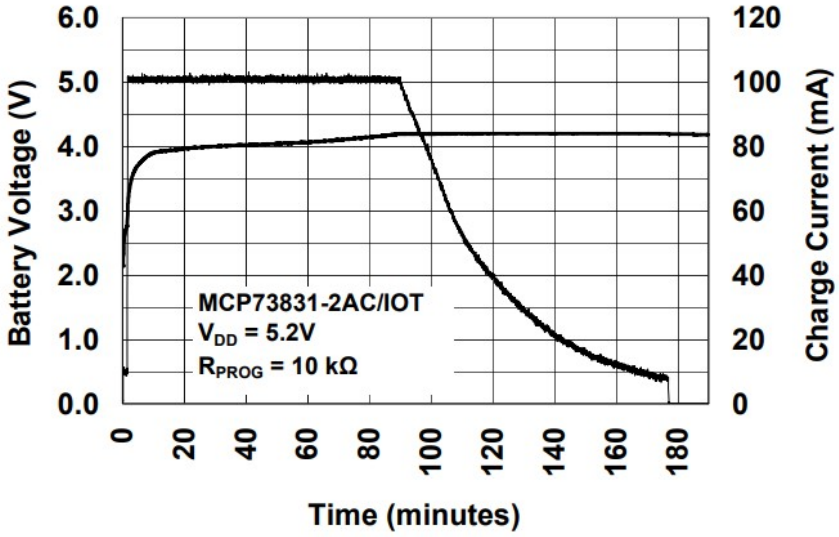
Com base no mesmo critério de densidade energética, parece justo eleger o uso de uma bateria com química baseada em lítio; além deste critério, a vida útil em ciclos de carga e descarga é um valor importante, pois tem impacto direto sobre o tempo entre manutenções do equipamento como um todo, e é consideravelmente superior no caso das baterias de lítio.

Considerando-se o fato de que muitos dos componentes eletrônicos modernos operam com alimentação na faixa entre 1,8 e 3,3 V, é interessante, ainda, o fato de a tensão nominal por célula deste tipo de bateria ser maior que a dos demais, já que isto elimina a necessidade de associar mais células (o que em si diminui a densidade energética, já que são somados os volumes dos invólucros de cada célula), ou de implementar um conversor DC-DC *step-up*, a fim de aumentar a tensão disponível.

Linden e Reddy (2001) alertam, no entanto, que as baterias Li-ion requerem circuitos de controle, de carga bem como de descarga, para que sejam operadas com segurança, eficiência e durabilidade. Estes circuitos são conhecidos como BMS. Isto é necessário porque essas baterias podem ser permanentemente danificadas tanto por sobrecarga quanto por sobredescarga, bem como por correntes superiores a determinados valores *C*, que representam valores de corrente numericamente iguais a alguma fração da capacidade de carga da bateria: por exemplo, uma bateria com capacidade de 2 Ah provê uma corrente de *C*/10 quando esta corrente tem um valor de 0,2 A.

Ainda de acordo com esse autor, um modo aceitável de recarregar uma bateria Li-ion é através de um regime de corrente constante até que a tensão da bateria atinja o patamar de 4,1 ou 4,2 V, no caso de baterias LCO, seguido de um regime em tensão constante a este valor, durante o qual a corrente de carga gradualmente se reduz, e deve ser cortada ao atingir determinado valor *C* (EHRLICH, 2001); os valores exatos variam, e os valores recomendados são informados nos *datasheets* de cada bateria. Uma ilustração da operação deste algoritmo é dada pela Figura 5.

Figura 5 - Curvas típicas de carga.



Fonte: *MCP73831/2 Datasheet* (Microchip Technology, 2020).

Como citado anteriormente, o BMS deve controlar também a descarga, limitando a tensão mínima a que a bateria chega durante a descarga, bem como a corrente máxima de descarga: não apenas correntes demasiado elevadas podem danificar a bateria, como também estas baterias são capazes de fornecer correntes muito elevadas em caso, por exemplo, de curto-circuito.

Um BMS pode ser implementado de diversas formas: desde um circuito totalmente analógico, à base de comparadores, até um circuito microcontrolado que monitore a bateria e tome decisões com base nesse monitoramento, ou ainda através do uso de um circuito integrado dedicado. Estes circuitos integrados implementam um BMS com variados graus de complexidade: há, por exemplo, o MCP73831, da Microchip, que implementa o algoritmo de carga explicado e mostrado na Figura 5, acima, a uma única célula, com correntes de carga programáveis através de resistores externos, em encapsulamentos tão pequenos quanto DFN de 8 pinos, com medidas 2 x 3 mm, mas que, no entanto, não supervisiona a descarga; ou, ainda, a série BQ769x0, da Texas Instruments, que não apenas monitora tensão e corrente de carga de associações em série de até 15 células, mas também implementa proteção térmica das células através de medição de temperatura via múltiplos termistores, proteções de sobrecorrente e curto-circuito, sobre- e subtensão, além de realizar o balanceamento da carga das células e monitorar a capacidade armazenada da associação, com interface de comunicação digital.

Alguns BMS são oferecidos por diversos fabricantes como módulos pré-fabricados, que se integram às células que se pretende gerenciar: módulos para

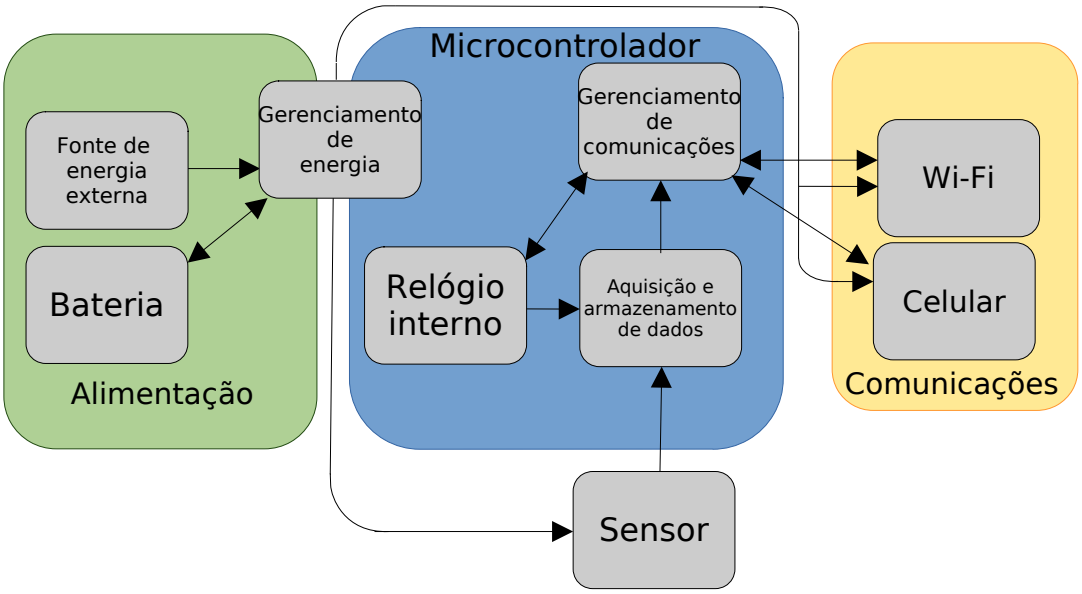
células cilíndricas podem ser confeccionados em placas de circuito impresso redondas, com o mesmo perfil da secção transversal da célula, resultando em uma montagem muito compacta.

3 DESENVOLVIMENTO

3.1 VISÃO GERAL DO PROJETO

Com base nas especificações e requerimentos estabelecidos, é possível traçar uma visão geral do projeto, que é esquematizada na forma de diagrama em blocos na Figura 6, abaixo.

Figura 6 - Diagrama em blocos do projeto.



Fonte: o autor.

Como especificado anteriormente, o circuito será alimentado por uma fonte externa, ou, como contingência, por uma bateria interna. O chaveamento entre a alimentação por uma fonte de energia ou outra é feito automaticamente por um subcircuito dedicado.

O bloco de gerenciamento de energia controla a recarga da bateria, através de outro subcircuito dedicado, bem como quais partes do bloco de comunicações devem ser energizadas e que tarefas devem ser realizadas, de acordo com o nível da bateria. Este segundo conjunto de funções é assumido pelo microcontrolador, e por isso este bloco é dividido entre os superblocos do microcontrolador e de alimentação.

Além de realizar parte do gerenciamento de energia, o microcontrolador gerencia outras funções vitais do projeto: ele adquire periodicamente os dados de um sensor digital externo e os armazena, por padrão, em uma porção de memória não-volátil, onde são armazenados com carimbo de data e hora reais de aquisição (*timestamp*). Os dados são adquiridos diretamente do sensor, sem quaisquer interfaces adicionais, já que o sensor especificado já incorpora um conversor analógico/digital.

A um período não necessariamente coincidente com o processo de aquisição, os dados armazenados são transmitidos para a nuvem, através da interface de comunicação mais conveniente no momento, a ser decidida pelo bloco de gerenciamento de comunicação. O dado a ser transmitido é removido da pilha de armazenamento interno não-volátil se, e somente se, a transmissão for bem-sucedida; caso contrário, uma nova tentativa de transmissão será realizada. Deste modo, assegura-se a integridade e completude dos dados a serem adquiridos.

O bloco de gerenciamento de comunicações também informa ao relógio interno do microcontrolador a data e hora reais, de modo a haver uma referência temporal comum entre o dispositivo e a plataforma na nuvem, que irá receber os dados. Esta informação é obtida através de comunicação, via Internet, com um servidor NTP.

O bloco do relógio interno do microcontrolador incorpora, portanto, um relógio de tempo real, bem como um contador de tempo local, que armazena o tempo desde a inicialização do microcontrolador, com a finalidade de controlar temporizações do fluxo do programa, como rotinas de *delay* (atraso); como são operações sem necessidade de referência de tempo externa, é mais simples e rápido realizá-las através de uma referência local, ao invés de constantemente realizar conversões e requisições de dados ao relógio de tempo real.

Finalmente, os blocos de comunicação Wi-Fi e Celular são as interfaces do microcontrolador com o mundo externo. Incorporam todo o *stack* (pilha) TCP/IP necessário à comunicação com a nuvem, bem como as camadas de mais baixo nível da comunicação, radiofreqüência etc., de modo que o microcontrolador não precisa implementar estas funcionalidades, mas apenas realizar requisições de comunicação de mais alto nível, como consultas a um servidor NTP, ou envio de dados a um servidor HTTP.

3.2 SELEÇÃO DE COMPONENTES

Uma vez estabelecidas quais as funções a serem realizadas, pode-se detalhar os componentes mais específicos selecionados para a realização dessas funções, como circuitos integrados. Componentes passivos, tais como resistores e capacitores, foram selecionados de acordo com a necessidade, e serão apresentados de maneira mais específica, onde for relevante, na Seção a respeito do projeto do circuito propriamente dito.

3.2.1 Microcontroladores

Decidiu-se, para executar as funções principais de aquisição e armazenamento intermediário de dados, gerenciamento de energia e gerenciamento de conectividade, usar o microcontrolador MSP430FR5969, da Texas Instruments, um microcontrolador de 16 bits que opera a até 16 MHz.

Esta decisão deveu-se, em primeiro lugar, ao fato de que este microcontrolador incorpora 64 kB de memória não-volátil ferroelétrica (FRAM), ao invés da tradicional memória Flash. A memória FRAM apresenta uma série de vantagens sobre a memória Flash, tal como um número máximo de ciclos de escrita muito maior, os quais podem ser executados com menor gasto de energia e com retenção de dados por mais tempo. Algumas figuras de mérito comparativas entre as duas tecnologias são mostradas na Tabela 4:

Tabela 4 - Características de memórias não-voláteis.

Tecnologia	Flash	FRAM
Tempo de escrita	500 ns	50 ns
Tempo de retenção	10-100 anos	100 anos a 25 °C 40 anos a 70 °C
Ciclos de escrita	10^5	10^{15}
Energia para escrita	1108 µJ/kB	52 µJ/kB

Fonte: Texas Instruments, 2017.

Um aspecto a se levar em conta é que, ao contrário da memória Flash, as leituras da memória FRAM são destrutivas: é necessário reescrever o dado a cada vez que este é lido, o que representa um fator de desgaste não só na escrita mas também na leitura. Contudo, dada a resiliência grandemente superior da FRAM, este fator se torna um problema menor: a um período de leitura/escrita de 5 minutos, valor típico para esta aplicação, 10^{15} ciclos representam 9.5 bilhões de anos. Em contraste, a resiliência de 10^5 ciclos da memória Flash representaria pouco menos de 1 ano, o que obrigaria, ao contrário do que foi adotado neste projeto, uma abordagem que evitasse escrever dados na memória não-volátil por padrão.

Esses 64 kB de memória FRAM são compartilhados entre programa e dados. Embora o microcontrolador conte, ainda, com 2 kB de memória SRAM para registradores e variáveis comuns, é possível definir, em tempo de compilação e *linking*, variáveis a serem armazenadas no espaço de FRAM, de modo que os acessos à memória não-volátil se dêem como a uma variável qualquer; como esse espaço de memória faz parte do microcontrolador, se torna desnecessário o uso de um componente externo para armazenamento, o que simplifica a programação, o *layout* físico do projeto, e reduz a quantidade total de componentes.

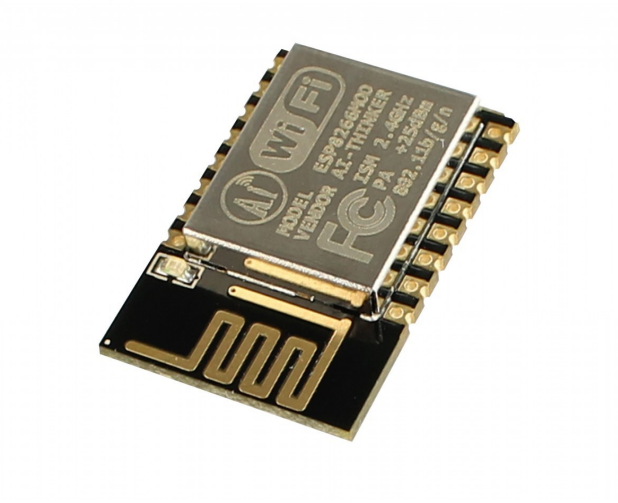
O MSP430FR5969 incorpora, ainda, um relógio de tempo real que opera com baixo consumo de energia, o que também simplifica o projeto. Além disso, como citado anteriormente, é um microcontrolador de 16 bits, que conta com instruções dedicadas para multiplicação, características úteis para a manipulação de dados, o que é o caso: é necessário formatar *strings* com valores numéricos de data, hora e temperatura, esta em ponto flutuante, processos que se beneficiam dessas características.

O MSP430FR5969 se caracteriza também por ser um microcontrolador com modos de operação em muito baixa energia, com consumo de corrente que chega a valores tão baixos quanto 0,3 μA. Mesmo em modo ativo, a 16 MHz, consome, no pior caso, 2,6 mA. É capaz de operar a tensões entre 1,8 e 3,6 V (TEXAS INSTRUMENTS, 2018).

Para prototipagem, foi adquirida uma placa de desenvolvimento EXP-MSP430FR5969 LaunchPad, produzida pela própria Texas Instruments. Esta placa incorpora funcionalidades completas de *debugging* (depuração) e análise, que podem ser usadas através do Code Composer Studio, também da Texas Instruments, ao ser conectada a um computador através de uma porta USB.

Para prover conectividade via redes Wi-Fi, optou-se por usar o ESP8266, da Espressif, que integra em um único pacote um núcleo Tensilica/Cadence Extensa a 80 MHz e conectividade IEEE 802.11 b/g/n. Este microcontrolador está prontamente disponível em módulos que incorporam tudo o que é necessário para sua operação: memória Flash, oscilador e antena. Por simplicidade, decidiu-se usar um desses módulos, mostrado na Figura 7, abaixo.

Figura 7 - Módulo ESP8266.



Fonte: https://www.usinainfo.com.br/1021486-thickbox_default/modulo-esp8266-esp12e-wifi-serial.jpg

A escolha do ESP8266 se deu, primariamente, devido ao ecossistema estabelecido ao redor do SDK Arduino, que inclui diversas bibliotecas que já implementam um *stack* TCP/IP, bem como diversas funções relacionadas à Internet necessárias ao desenvolvimento deste projeto, como envio de dados através do protocolo HTTP, sem que essas funções tenham que ser desenvolvidas a partir do zero.

Neste projeto, o ESP8266 é usado como um periférico Wi-Fi, sem grande autonomia, controlado diretamente pelo microcontrolador principal. Durante transmissões, o consumo de corrente típico do ESP8266 em pior caso é 170 mA. É capaz de operar a tensões entre 2,5 e 3,6 V (ESPRESSIF, 2023).

Para prototipagem, foi adquirida uma placa de desenvolvimento do tipo NodeMCU, que inclui um módulo ESP8266 como o mostrado na Figura 7. Esta placa, por ser *open-source*, é oferecida por diversos fornecedores chineses, e pode

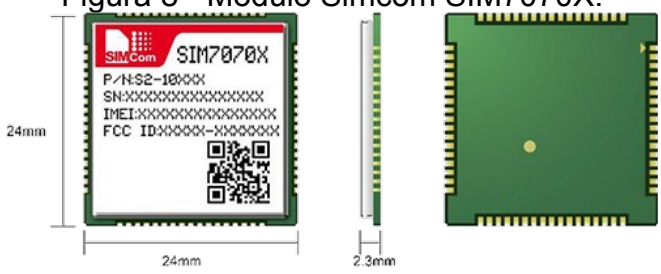
ser prontamente encontrada no mercado local. Placas NodeMCU integram algumas funcionalidades mínimas, como interface de programação via USB, e são suportadas diretamente pelo ambiente de desenvolvimento Arduino.

3.2.2 Modem Celular

Levando-se em consideração as necessidades de tráfego de dados deste projeto, expostas na Seção 3 – pequenos volumes de dados, a serem transmitidos periodicamente, com períodos na ordem de minutos, de posições fixas ou móveis – fez-se necessário o uso de um modem celular que suportasse, no mínimo, o padrão LTE Cat M1. Seria desejável que esse modem também suportasse comunicações em padrões mais antigos, para o caso de uma aplicação móvel sair do alcance da cobertura LTE. Como bônus, seria também desejável que suportasse o padrão LTE Cat NB1, o que poderia ser vantajoso para aplicações exclusivamente estacionárias, embora o padrão LTE Cat M1 seja perfeitamente aplicável a estas aplicações também.

Um dispositivo que reúne essas características, e que foi escolhido para uso neste projeto, é o módulo Simcom SIM7070G. Este módulo foi escolhido, entre outros com as mesmas capacidades de comunicação, por ser relativamente simples: outros módulos oferecem características como comunicação através de interface PCI Express e/ou encapsulamentos que exigem processos de fabricação e subseqüente controle de qualidade mais compexos, como LGA ou BGA. O SIM7070G usa um encapsulamento LCC, mostrado na Figura 8, o que facilita o processo de manufatura em relação a encapsulamentos LGA ou BGA, uma vez que não há necessidade de realização e inspeção de soldas por baixo do componente. O número de pinos comparativamente menor também impõe menos restrições ao *layout* da placa de circuito impresso, passível de ser projetado em duas camadas.

Figura 8 - Módulo Simcom SIM7070X.



Fonte: https://www.simcom.com/product/SIM7070G.html

Como mencionado, o SIM7070G pode se comunicar com a rede usando os padrões LTE Cat M e Cat NB, além de GSM/GPRS, o que faz dele uma boa solução para aplicações móveis, que nem sempre estarão sob cobertura da infraestrutura 4G/5G, a qual ainda não é ubíqüa em toda a extensão do território nacional. Com o microcontrolador, a comunicação é serial assíncrona (baseada em RS-232, mas com níveis lógicos baixos), baseada em comandos AT.

Este módulo implementa, ainda, funções para comunicação através dos protocolos TCP, UDP, HTTP, HTTPS, FTP, NTP, MQTT, entre outros, através de comandos AT, como mencionado acima. Assim, o desenvolvimento se torna simplificado, já que a implementação desses protocolos já está embutida no módulo.

A Tabela 5, elaborada com base em informações constantes no manual de projeto de *hardware* do SIM7070G (SIMCOM, 2020) traz, de forma sintetizada, os requerimentos típicos de corrente deste componente em algumas situações diferentes.

Tabela 5 - Corrente típica do SIM7070G em diferentes condições.

Modo	Condições	Corrente
Inativo	Plena funcionalidade	12-16 mA
Dormindo	Plena funcionalidade	2.1 mA
Dormindo	Mínima funcionalidade	0.45 mA
Economia de energia	-	3.5μA
Transmissão	GPRS	274-500 mA
Transmissão	EDGE	172-311 mA
Transmissão	LTE Cat M, 0 dBm	90 mA
Transmissão	LTE Cat M, 10 dBm	100 mA
Transmissão	LTE Cat M, 21 dBm	119 mA
Transmissão	LTE Cat M, 27 dBm	348 mA
Transmissão	LTE Cat NB2, 0 dBm	55 mA
Transmissão	LTE Cat NB2, 10 dBm	87 mA
Transmissão	LTE Cat NB2, 21 dBm	153 mA
Transmissão	LTE Cat NB2, 26 dBm	302 mA

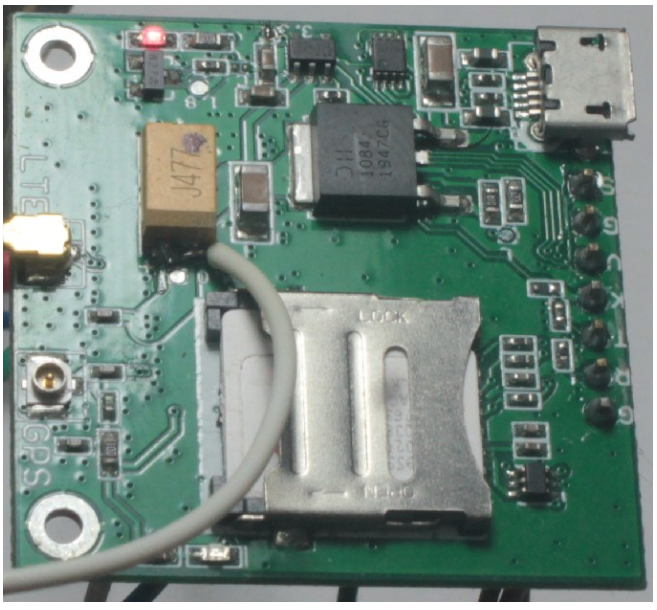
Fonte: Adaptada de SIMCOM (2020).

Para prototipagem, foi adquirido um módulo BK-SIM7070, do fabricante chinês ANDTech, que expõe o mínimo necessário ao uso do modem – interface

UART, alimentação e outros pinos de controle – sem que seja necessário ocupar-se de outros detalhes igualmente necessários ao uso, como reguladores de tensão, conversores de nível, cartão SIM etc., já que estes foram integrados ao módulo. O módulo também inclui uma antena para comunicação celular, ligada à placa através de um conector U.FL, e que foi usada para testes.

Foi necessário, para realizar testes de autonomia, realizar uma modificação no módulo de desenvolvimento do modem: este é projetado para ser alimentado a partir de 5 V, e conta com dois reguladores de tensão cascateados: um para 4 V, que alimenta o modem propriamente dito, e cascateado a este se encontra outro regulador, para 3,3 V, que alimenta o conversor de níveis lógicos para a comunicação. Foi necessário realizar um *bypass* da alimentação da bateria para a linha de 4 V do módulo; do contrário, o regulador de tensão para 4 V não operaria corretamente e faria com que o módulo, subalimentado, não funcionasse. Esta modificação é mostrada abaixo, na Figura 9.

Figura 9 - Modificação para injeção da alimentação após o regulador de 4 V do modem

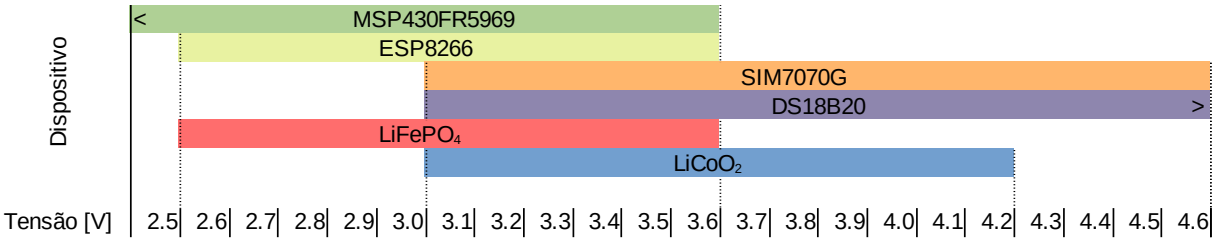


Fonte: o autor.

3.2.3 Bateria

Na Seção 4.9 foi estabelecido que a bateria deveria ser secundária, de íons de lítio, devido a fatores como tensão de operação e densidade de energia. Entretanto, nem todas as baterias de íons de lítio são iguais: decidiu-se usar uma bateria LFP, pois além de ser mais durável e segura, tem níveis de tensão mais compatíveis com a aplicação. A Figura 10, abaixo, mostra as faixas de tensão em que cada um dos componentes considerados opera.

Figura 10 - Faixas de tensão de operação dos componentes escolhidos.



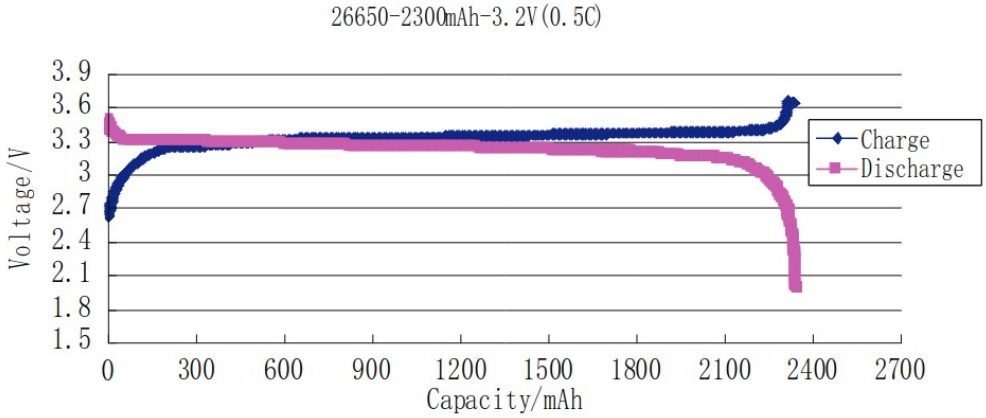
Fonte: o autor.

Da Figura 10, pode-se verificar que a intersecção das tensões de operação dos componentes escolhidos ou especificados resulta numa faixa que vai de 3,0 a 3,6 V, sendo 3,0 V o limite mínimo de operação tanto do modem SIM7070G quanto do sensor DS18B20, e 3,6 V sendo o limite máximo de operação dos microcontroladores escolhidos. Tanto baterias LFP como LCO seriam capazes de alimentar esses dispositivos, mas no caso da bateria LCO seria necessário usar um regulador LDO para alimentar os microcontroladores, já que a tensão máxima desta bateria ultrapassa os 3,6 V máximos dos microcontroladores. Pareceu mais simples usar uma bateria LFP, eliminando a necessidade do LDO e podendo fazer uso do canal interno de medição da tensão de alimentação de que dispõe o microcontrolador MSP430FR5969 para monitoramento da bateria, novamente simplificando o projeto, além de aproveitar as vantagens já mencionadas das baterias LFP sobre as LCO. Além disso, as baterias LFP são mais amigáveis de um ponto de vista ambiental: produzem menos resíduos, já que duram por muito mais ciclos que as LCO.

Embora o limite inferior de tensão da bateria LFP esteja especificado e mostrado na Figura 10 como 2,5 V, estabelecendo-se uma tensão de corte de 3,0 V,

a uma taxa de descarga modesta (0,5*C*), ainda se consegue usufruir de mais de 90% da capacidade nominal da bateria, já que esta tem uma curva de descarga bastante plana, como mostra a Figura 11, vinda da folha de dados de uma célula LFP de 2,3 Ah do fabricante Enix Energies. Adotar esta estratégia também diminui o desgaste sobre a bateria, uma vez que a profundidade máxima de descarga se torna menor.

Figura 11 - Curvas de carga e descarga a 0.5C de uma bateria LFP.



Fonte: ENIX ENERGIES (2014).

Desta forma, considerou-se que se pode prescindir de regulação de tensão da bateria, seja para um nível acima ou abaixo da tensão de operação desta, que é admissível, portanto, para o projeto.

Para finalidades de prototipagem, escolheu-se uma bateria LFP disponibilizada no mercado nacional através da marca Rontek, em envólucro 18650 e capacidade nominal de 1500 mAh, da qual, infelizmente, não foi possível localizar folha de dados.

3.2.4 Gerenciamento de Bateria

Para gerenciar a recarga da bateria $LiFePO_4$, foi escolhido um circuito integrado dedicado para este propósito: o CN3058E, da fabricante chinesa Consonance. Trata-se de um dispositivo que reúne, em um encapsulamento SOP8, toda a lógica de controle de recarga, através de um algoritmo de recarga por corrente constante seguida de tensão constante, com corrente de recarga

configurável através de resistor externo até 500 mA e proteção de temperatura (CONSONANCE, 2022). Opera com tensão de entrada de 3,8 a 6 V, de modo que é possível alimentar o dispositivo com uma fonte comum de 5 volts. Esta corrente máxima de 500 mA representa um valor *C*/3, que é bastante razoável para a bateria escolhida.

Este dispositivo foi escolhido por ser específico para uma única célula $LiFePO_4$, por ser simples – requer apenas alguns resistores e capacitores externos - e por estar disponível a um preço bastante inferior ao de outros dispositivos semelhantes, como o BQ25070, da Texas Instruments.

Optou-se por gerenciar a descarga através do próprio microcontrolador principal do projeto: ao atingir a tensão mínima de 3,0 V, as funções de comunicação e aquisição de temperatura são suspensas, e apenas o relógio de tempo real continua efetivamente operando, de modo a conservar a bateria e as informações de relógio por tanto tempo quanto possível, operando sob a presunção de que o aparelho será colocado em recarga antes que a bateria se esgote totalmente. Como todas as funções requerem uma corrente relativamente baixa, optou-se, ainda, por implementar uma proteção de descarga simples, através de fusível, para desconectar a bateria caso a corrente se torne elevada por algum motivo. A Tabela 6 mostra uma estimativa de corrente do projeto em pior caso, com todos os componentes ativos e com consumo máximo, situação esta que não deve ocorrer jamais em operação normal, já que o aparelho deve operar em Wi-Fi ou rede celular, mas nunca ambos ao mesmo tempo; porém, é um valor orientativo quanto ao dimensionamento de proteções.

Tabela 6 - Estimativa de consumo de corrente do projeto.

Componente	Corrente máxima
MSP430FR5969	2.6 mA
ESP8266	170 mA
SIM7070G	500 mA
DS18B20	1.5 mA
Total	674.1 mA

Fonte: o autor.

Uma corrente de 674,1 mA representa, para a bateria escolhida, de 1500 mAh, uma taxa de descarga 0,45*C*, aceitável para baterias LFP e que não deve causar quaisquer prejuízos à longevidade da bateria, reiterando que, a princípio, este valor jamais deve ocorrer na prática, já que o dispositivo deve operar ou através de Wi-Fi, ou através da rede celular, mas nunca em ambas as modalidades simultaneamente.

Uma bateria submetida a descargas intermitentes pode recuperar um pouco da tensão em seus terminais uma vez que a descarga seja interrompida (FUHS, 2009). Se houver um único limiar de tensão, aquém do qual a operação normal é interrompida e além do qual a operação é permitida, a tensão da bateria pode flutuar além e aquém desse limiar, causando um comportamento indesejado do dispositivo. Para evitar tal situação, decidiu-se incorporar um comportamento histerético ao gerenciamento de energia: uma vez que o limiar mínimo de 3,0 V seja atingido, a operação do dispositivo é interrompida até que a tensão suba além de 3,1 V, quando então a operação normal é retomada.

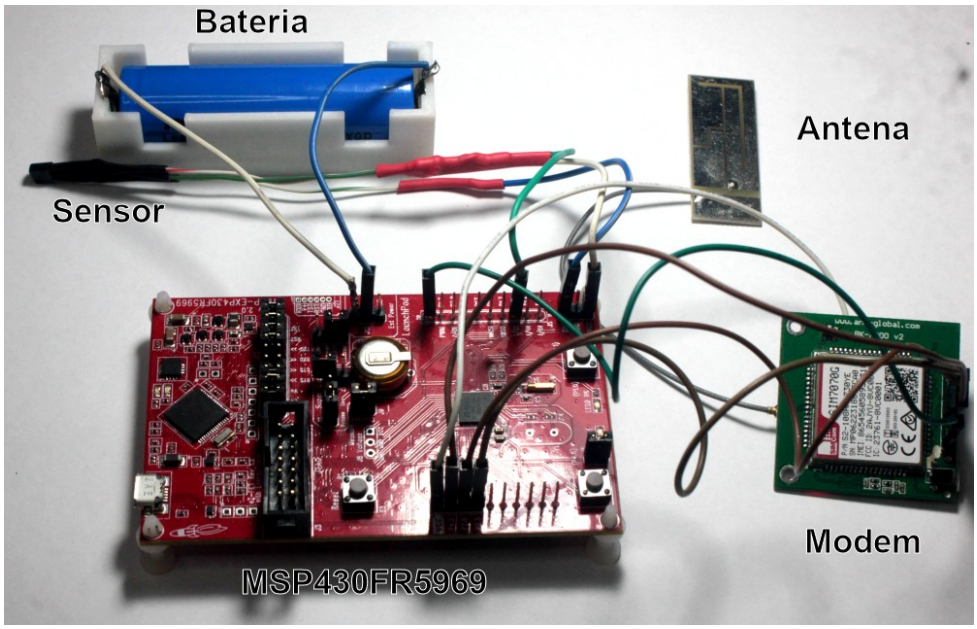
Para realizar a seleção de alimentação, entre bateria ou alimentação externa, foi implementado um circuito baseado em diodo ideal, que é apresentado em mais detalhes na Seção 5.3.4.

3.3 CIRCUITO

O circuito do projeto foi inicialmente prototipado com módulos e placas de desenvolvimento, conforme mostra a Figura 12. A partir disto, foi desenhado um diagrama esquemático, e a partir desse diagrama esquemático foi desenhada uma placa de circuito impresso.

O diagrama foi alterado em relação ao protótipo quando conveniente, por questões de disposição física de componentes na placa de circuito impresso.

Figura 12 - Protótipo com base em placas de desenvolvimento.

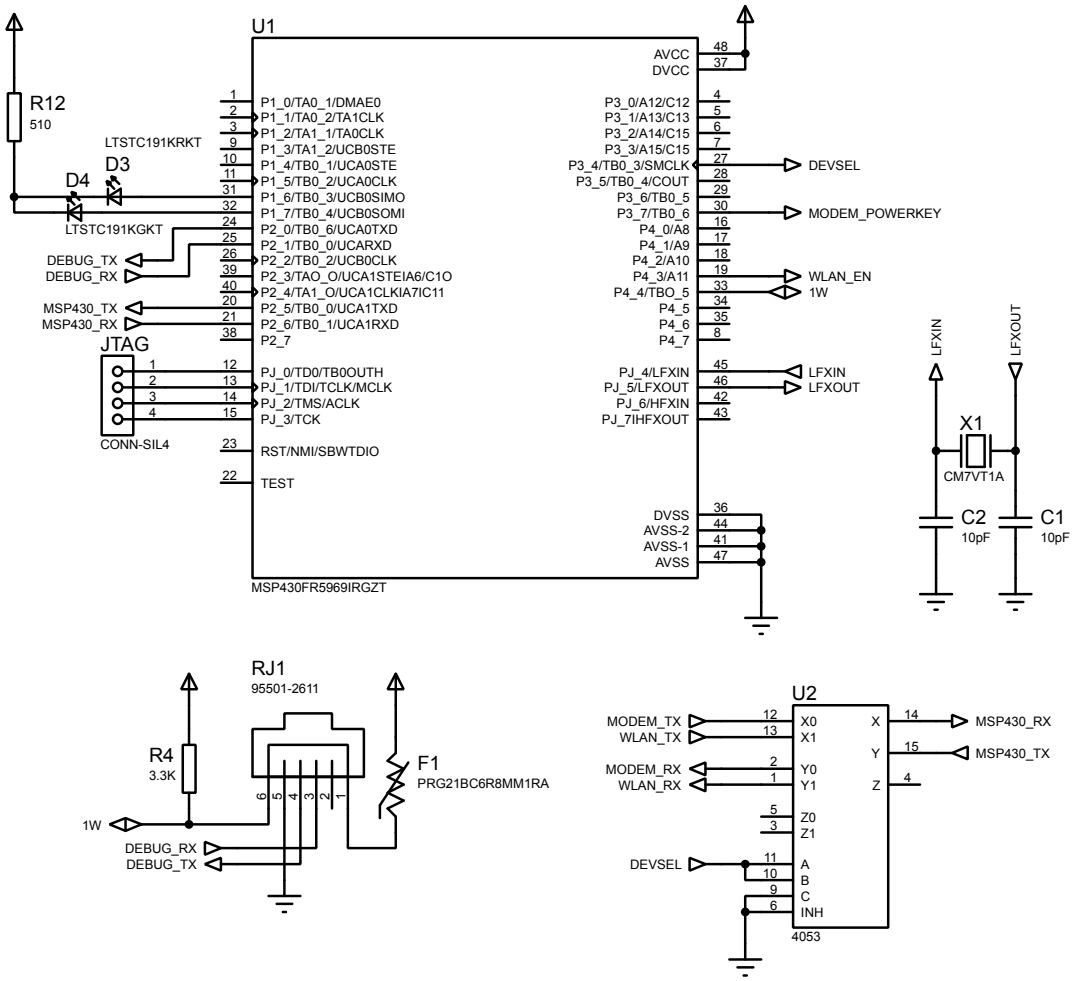


Fonte: o autor.

3.3.1 Microcontrolador

O diagrama esquemático será apresentado por partes. A Figura 13 traz o diagrama correspondente ao microcontrolador principal e alguns componentes periféricos.

Figura 13 - Diagrama esquemático parcial (microcontrolador).



Fonte: o autor.

U1, no diagrama, representa o microcontrolador MSP430FR5969. Este tem seu sinal de *clock* principal gerado internamente, por um oscilador RC a aproximadamente 8 MHz, já que o processamento realizado por este microcontrolador não necessita de grande precisão temporal. O RTC interno, porém, tem como fonte de *clock* o sinal proveniente de X1, um cristal oscilador do tipo comumente usado em relógios, de 32768 kHz, com precisão de 20 ppm, que confere ao relógio de tempo real do sistema o seguinte erro máximo diário: $\epsilon = 86400 \cdot 20 \cdot 10^{-6} = 1.728\,\mathrm{s}$

Embora este valor não faça parte da especificação do projeto, presume-se que seja adequado, pois, para a aplicação, é mais importante a periodicidade da medição do que a precisão temporal absoluta. De maneira análoga ao cálculo do erro diário, admitindo-se um período entre leituras de 5 minutos, uma precisão de 20 ppm resulta em um erro máximo de 6 ms por período, que é ao menos uma ordem de grandeza inferior ao tempo de latência inerente aos meios de comunicação adotados. Ademais, é possível sincronizar periodicamente o relógio de tempo real do sistema a um servidor NTP, de modo a limitar o *drift* desse relógio.

O microcontrolador U1 é alimentado pelo barramento principal de alimentação do sistema, que por sua vez pode ser alimentado a partir da bateria ou de uma fonte externa, e está restrito a uma faixa de 3,0 a 3,6 V. Detalhes a respeito da alimentação serão abordados mais adiante, após a apresentação dos subcircuitos pertinentes.

Do microcontrolador U1 partem alguns sinais de controle e comunicação:

- DEVSEL: seleciona o dispositivo de comunicação (modem ou módulo Wi-Fi);
- MODEM_POWERKEY: ao ser mantido em nível lógico baixo por 1 segundo, liga ou desliga o modem;
- WLAN_EN: controla a linha de *reset* do módulo Wi-Fi;
- MSP430_TX: transmissão de dados da interface serial do microcontrolador aos dispositivos de comunicação.

Além disso, chega ao microcontrolador o sinal MSP430_RX, de recepção de dados provenientes dos dispositivos de comunicação pela interface serial do microcontrolador, e também há a linha bidirecional 1W, do barramento 1-Wire, para comunicação com a sonda de temperatura através de um *jack* modular 6P6C (J1). Esse barramento conta com um resistor de *pull-up* de 3,3 kΩ (R4), valor que, considerando-se que o sensor será alimentado a 3,3 V, é consistente, de um ponto de vista de corrente de *pull-up*, com a recomendação do *datasheet* do componente, de 4,7 kΩ a 5 V, de modo a não comprometer a integridade de sinal em casos de cabeamentos mais longos.

A linha de alimentação que chega ao *jack* J1 é protegida por um fusível resetável PTC (F1), para 120 mA, contra curto-circuitos que, sabe-se, a partir da

vivência prática das atividades da empresa, ocorrem ocasionalmente, devido a eventuais danos ao cabeamento da sonda. Embora o sensor DS18B20 especificado consuma, em modo ativo, apenas 1,5 mA, optou-se por escolher um valor de corrente mais elevado do que isso para F1 que, por um lado, protege efetivamente o circuito de energia, enquanto que, por outro lado, não restringe a eventual aplicação futura de outras sondas que possam vir a consumir correntes mais elevadas.

O conector JTAG expõe as linhas necessárias à programação de U1 *in situ,* por programador/jiga externos, após ser populada a placa de circuito impresso. U2 é um multiplexador analógico 4053. Por ter largura de banda de 30 MHz, suficiente para as comunicações a 9600 bps entre os dispositivos no circuito, por ser compatível com os níveis lógicos e de alimentação utilizados, e por ser bidirecional, optou-se por usar este componente para multiplexar a UART do microcontrolador U1.

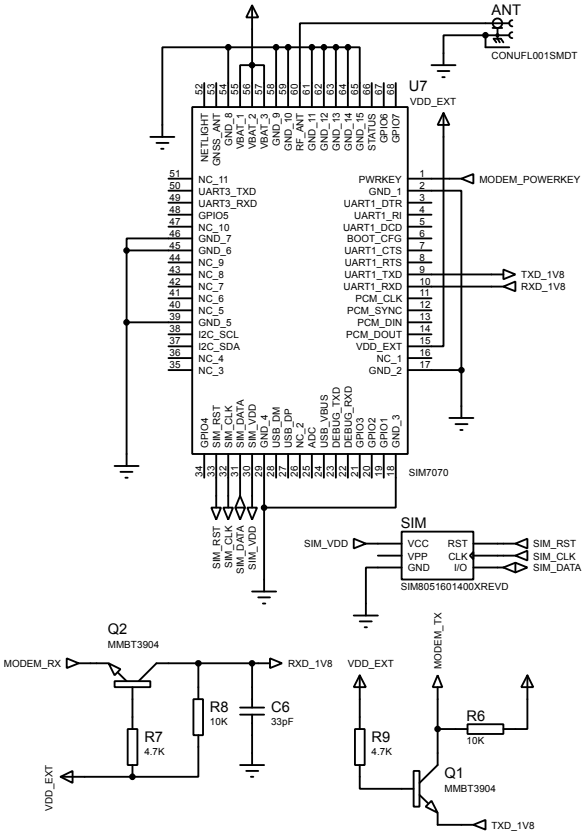
O uso de multiplexação da comunicação serial se fez necessário, pois, uma vez que as comunicações através da UART são realizadas de acordo com o padrão RS-232 (a níveis lógicos reduzidos, porém), não há suporte a múltiplos dispositivos em um mesmo barramento; no entanto, deseja-se que o microcontrolador se comunique tanto com o módulo Wi-Fi quanto com o modem; muito embora o microcontrolador disponha de duas UARTs, uma delas é usada para *debugging*, e não se desejou abrir mão dessa funcionalidade durante o desenvolvimento. Para contornar esta limitação, a linha MSP430_TX é chaveada, através de U2, entre as linhas MODEM_RX e WLAN_RX, de recepção de dados do modem e do módulo Wi-Fi, respectivamente, e a linha MSP430_RX é chaveada entre as linhas MODEM_TX e WLAN_TX, de transmissão de dados do modem e do módulo Wi-Fi. A linha DEVSEL seleciona o chaveamento dessas linhas.

De um ponto de vista de custo, considerou-se esta uma solução satisfatória, já que U2 é um componente disponível a preços muito baixos, e cujo uso não implicou em um aumento da área do *layout* da placa de circuto impresso.

3.3.2 Modem

A Figura 14, a seguir, mostra o diagrama esquemático parcial referente ao modem.

Figura 14 - Diagrama esquemático parcial (modem).



Fonte: o autor.

Para operar o modem SIM7070G, designado no circuito como U7, é obrigatório que seja ligado a ele um cartão SIM, que se comunica com o modem através de um barramento serial e contém as informações de identificação e autenticação da assinatura de telefonia móvel. Para realizar essa ligação, adotou-se um receptáculo do tipo “nano”, a fim de economizar espaço na placa de circuito impresso. Outras ligações importantes ao modem são a alimentação, proveniente do barramento de alimentação geral do sistema, a antena celular, ligada através de um conector padrão U.FL, e as linhas de comunicação com o microcontrolador U1.

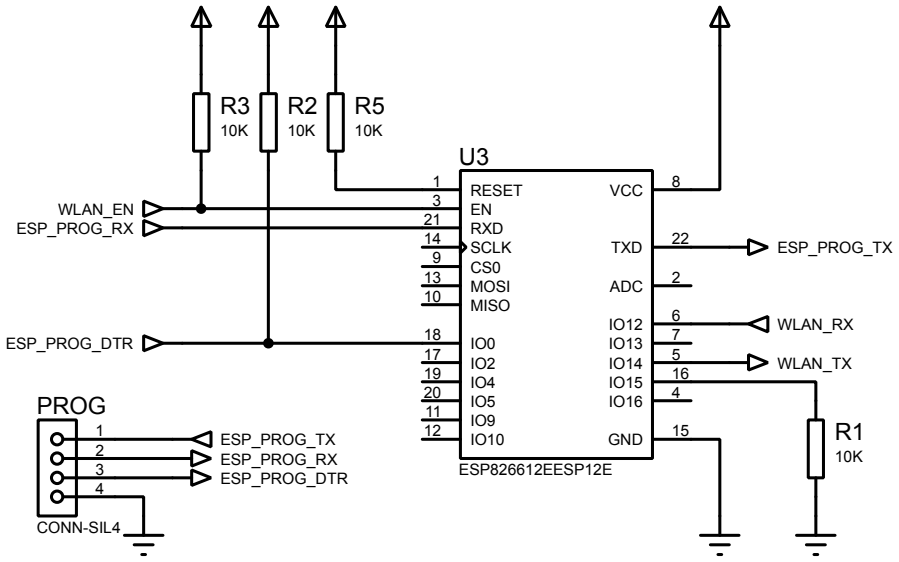
É importante notar que, embora o modem possa ser alimentado de 3,0 a 4,6 V, suas linhas de comunicação serial trabalham a níveis AVC, de 0 a 1,8 V. O microcontrolador U1, alimentado a 3,3 V, trabalha com níveis lógicos de 0 a 3,3 V (nominais); portanto, fez-se necessário efetuar uma conversão de níveis lógicos, para que o microcontrolador e o modem possam se comunicar de forma que o microcontrolador interprete adequadamente o nível lógico “alto” do modem, e o

modem aceite o nível lógico “alto” do microcontrolador sem ser danificado. Para tanto, adotou-se o esquema de conversão sugerido pela folha de dados do modem. Este esquema faz uso da saída de tensão de 1,8 V (VDD_EXT), fornecida pelo modem, e usa transistores NPN como chaves que elevam o nível lógico de 1,8 a 3,3 V (Q1) e reduzem de 3,3 a 1,8 V (Q2), para as linhas de transmissão e recepção de dados, respectivamente. Ainda seguindo a sugestão da folha de dados do modem, foram usados dois transistores MMBT3904 para esta função.

3.3.3 Módulo Wi-Fi

A Figura 15, a seguir, mostra o diagrama esquemático parcial do módulo Wi-Fi.

Figura 15 - Diagrama Esquemático Parcial (Wi-Fi).



Fonte: o autor.

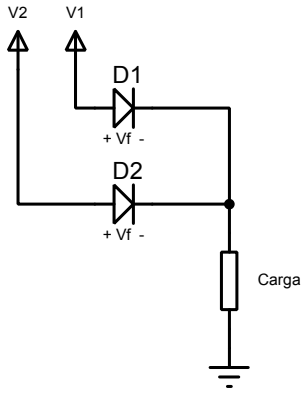
Este diagrama é mais sucinto, uma vez que, como exposto anteriormente, o módulo Wi-Fi, U3, é constituído por um módulo (U3), que integra todos os componentes necessários ao funcionamento de um microcontrolador ESP8266, inclusive antena. A ele são ligados apenas os sinais de controle e comunicação, alimentação, e um conector para programação *in situ.* Ao contrário do modem, não é necessário realizar conversões de nível lógico para a comunicação, uma vez que esta é realizada ao mesmo nível lógico por parte do ESP8266 e do MSP430FR5969.

Tanto o ESP8266 quanto o MSP430FR5969 necessitam ser programados com um *firmware*, que nada mais é que o programa executado por esses microcontroladores. O desenvolvimento desses programas é detalhado mais adiante, na Seção 5.4.

3.3.4 Alimentação

O projeto necessita ser alimentado por fonte externa ou por bateria, mas não por ambos simultaneamente: quando houver uma fonte externa, ele deve ser alimentado por esta fonte; do contrário, deve ser alimentado pela bateria. No entanto, enquanto alimentado pela fonte externa, esta deve também carregar a bateria, sem que o restante do circuito seja, todavia, energizado pela bateria. Uma solução para este problema é um circuito de lógica “ou” a diodos, como mostrado na Figura 16:

Figura 16 - Lógica “ou” a diodos.



Fonte: o autor.

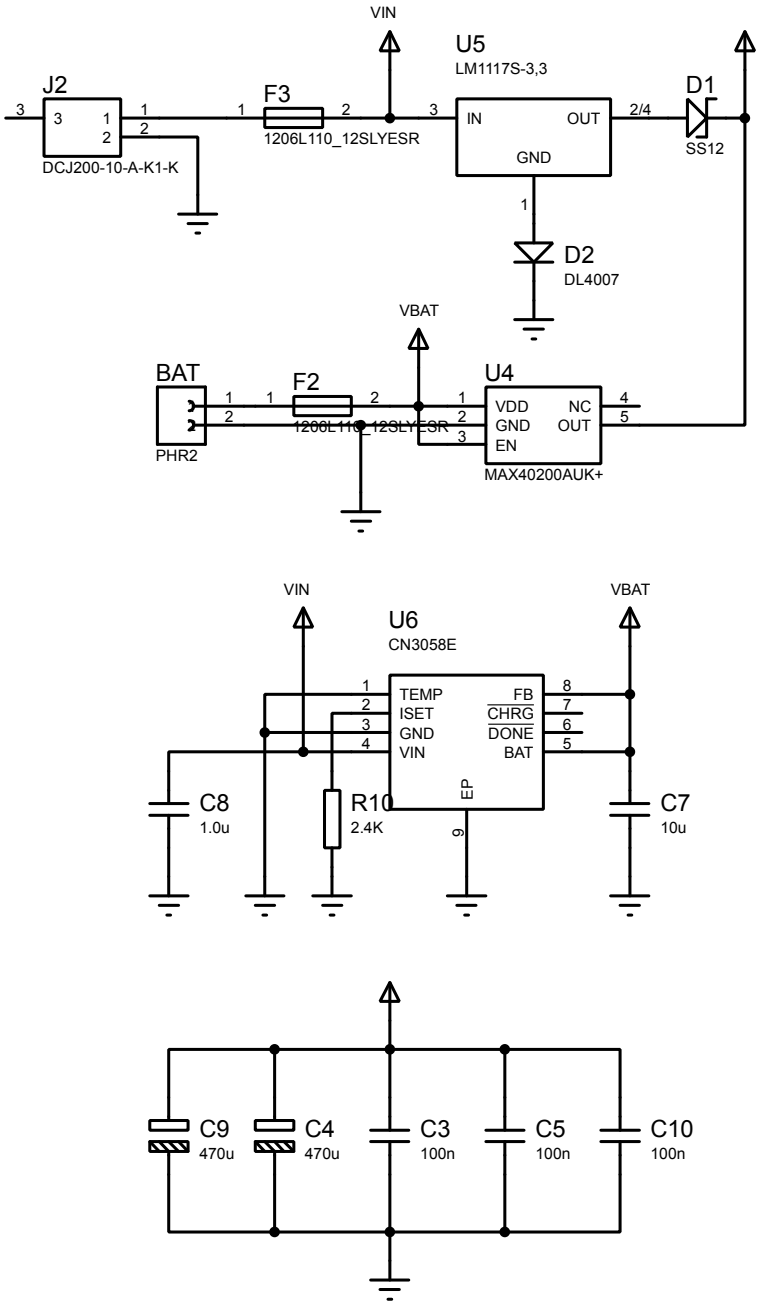
Sejam D1 e D2 diodos não-ideais idênticos, com tensão de polarização direta *Vf*, e com V_1 e V_2 > *Vf*. No caso em que $V_1 > V_2$, D1 é polarizado diretamente, D2 é polarizado reversamente e a tensão sobre a carga é V_1 - *Vf*; conversamente, se $V_2 > V_1$, D2 é polarizado diretamente, D1 é polarizado reversamente, e a tensão sobre a carga é V_2 – *Vf*. Portanto, o circuito de lógica “ou” a diodos seleciona V_1 **ou** V_2, conforme a tensão que for maior, e a tensão selecionada chega à carga decrescida de *Vf.*

Na aplicação para este projeto, o decréscimo de *Vf* da tensão sobre a carga não é um grande problema, no caso da alimentação proveniente da fonte externa: é possível dimensionar a tensão de alimentação dessa fonte de modo a compensar pelo decréscimo. Esta flexibilidade, contudo, não existe para a alimentação proveniente da bateria, que é dotada de uma tensão muito mais rigidamente determinada por suas características eletroquímicas e estado de carga. De fato, mesmo considerando-se o uso de um diodo Schottky, com tensão de polarização direta tipicamente inferior à de diodos de silício, na ordem de 300 mV, este decréscimo sobre a tensão da bateria seria suficiente para que uma bateria quase totalmente carregada não fosse capaz de atender aos requerimentos de alimentação do circuito. Faz-se necessária uma solução que traga uma tensão de polarização direta muito inferior a isso.

Tal solução é encontrada na forma de circuitos integrados que implementam a funcionalidade de “diodos ideais”. Não são, na realidade, ideais de fato, mas possuem tensão de polarização direta de tal modo inferior – quase uma ordem de grandeza – à dos diodos Schottky, que são chamados de “ideais”.

Um circuito integrado que implementa essa funcionalidade é o MAX40200, citado anteriormente, e mostrado sob o designador U4 na Figura 17:

Figura 17 - Diagrama esquemático parcial (alimentação).



Fonte: o autor.

O MAX40200 (U4) implementa um diodo chamado ideal, com corrente máxima de 1 A, sob a qual a tensão de polarização direta é especificada a 85 mV; sob uma corrente de 0,5 A, essa tensão cai para 43 mV. Desta forma, a tensão de polarização direta desse componente pode ser desconsiderada. É um componente compacto, em encapsulamento SOT23-5, econômico (corrente quiescente de, no

máximo, 18 μA) e que incorpora, ainda, proteção térmica (ANALOG DEVICES, 2023). No circuito, está protegido, adicionalmente, por um fusível resetável (F2) de 1 A, assim como o regulador de tensão LM1117-3.3 (U5), protegido por um outro fusível idêntico (F1).

U5 é um regulador de tensão LDO (*Low Drop-Out,* de baixa queda de tensão mínima), alimentado por uma fonte externa de 5 V, através de um *jack* DC de 5,5 mm (J2), com saída de 3,3 V. Essa tensão é desreferenciada do terra através de D2, um diodo DL4007, de silício, para 1 A, com tensão de polarização direta de cerca de 0,65 V, o que faz com que a saída do regulador, em relação ao terra, não mais seja de 3,3 V, mas de aproximadamente 3,95 V. Isto é feito para garantir o correto funcionamento do circuito "ou" a diodo, implementado por D1 e U4: após D1, um diodo Schottky de 1A, a tensão proveniente de U5 cai para aproximadamente 3,65 V, valor superior à tensão da bateria, assegurando que U4 bloqueie a bateria quando o circuito for alimentado pela fonte externa.

U6 é o circuito integrado responsável pela recarga da bateria, conforme exposto na Seção 5.2.4. Por simplicidade, optou-se por não usar a proteção térmica da bateria, aterrando-se o pino TEMP desse circuito integrado. R10 é o resistor que determina a corrente de carga da bateria; arbitrou-se uma corrente de 500 mA, a partir da qual se pode determinar o valor de R10 através da fórmula fornecida na folha de dados do CN3058E: $I_{CH} = 0.5 = \frac{1218}{R_{ISET}} \therefore R_{ISET} = 2436\,\Omega.$

Os valores de C7 e C10, capacitores cerâmicos, seguiram as sugestões da folha de dados do CN3058E. Pode-se observar, na Figura 17, que este circuito integrado encontra-se isolado do barramento principal de energia do circuito por U4, já que é alimentado por V_{IN} e sua saída é em V_{BAT}, de modo que, como resultado da operação do circuito de lógica "ou" a diodos, U6 nunca alimentará diretamente o circuito, mas apenas a bateria, de modo a não atrapalhar a operação de recarga.

Os capacitores C3, C4, C5, C9 e C10 são de desacoplamento da alimentação. O módulo de desenvolvimento do modem usa um capacitor de 470 μF, valor que se observou ser adequado e foi replicado aqui para o capacitor que desacopla a alimentação do modem. Por uma questão de simplicidade e reaproveitamento de lista de materiais, adotou-se o mesmo valor para o desacoplamento da alimentação do módulo de Wi-Fi, decisão esta justificada pelo fato de que a corrente média em pior caso do ESP8266 não alcança a corrente em

pior caso do modem, portanto, um capacitor de 470 µF é suficiente. Estes capacitores – C4 e C9 – são de tântalo; os demais, de 100 nF, são cerâmicos. Foram posicionados, na placa de circuito impresso, próximos aos circuitos integrados e módulos do circuito, de modo a minimizar a impedância entre os capacitores e os dispositivos por eles desacoplados.

3.4 FIRMWARE

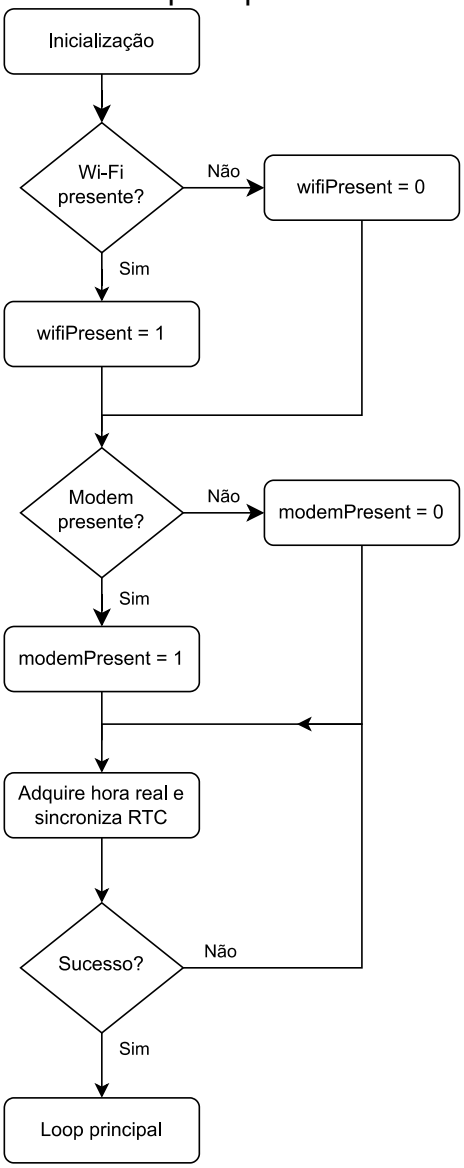
Por utilizar dois microcontroladores, foram desenvolvidos dois *firmwares*: o principal, para o MSP430FR5969, e o do módulo Wi-Fi, para o ESP8266. O *firmware* principal foi desenvolvido em linguagem C no Code Composer Studio 12.2, da Texas Instruments; o *firmware* do módulo Wi-Fi foi desenvolvido em linguagem C++ na Arduino IDE, usando as bibliotecas para ESP8266 fornecidas pelo fabricante sob licença LGPL. O *firmware* principal usa, ainda, uma adaptação do exemplo fornecido pela Texas Instruments para a leitura de sensores DS18B20 sob licença BSD.

Por brevidade, os códigos-fonte propriamente ditos foram omitidos. Esta Seção trata de uma descrição diagramática dos algoritmos desenvolvidos.

3.4.1 Firmware Principal

O *firmware* principal inicia sua execução por uma etapa de inicialização e verificação da configuração do dispositivo. Uma vez que o projeto pode ser dotado de qualquer combinação de conectividade Wi-Fi e/ou celular, uma vez configurados os periféricos internos e as portas de entrada e saída, o microcontrolador interroga os módulos de comunicação: caso algum não responda, este é considerado ausente e não serão feitas outras tentativas de comunicação ao longo da execução do código. Conversamente, aquele módulo que responder será considerado presente; caso ambos os módulos estejam presentes, a comunicação Wi-Fi terá precedência sobre a comunicação celular. Esta etapa de inicialização é mostrada, em forma de fluxograma, na Figura 18:

Figura 18 - Fluxograma de inicialização do firmware principal.



Fonte: o autor.

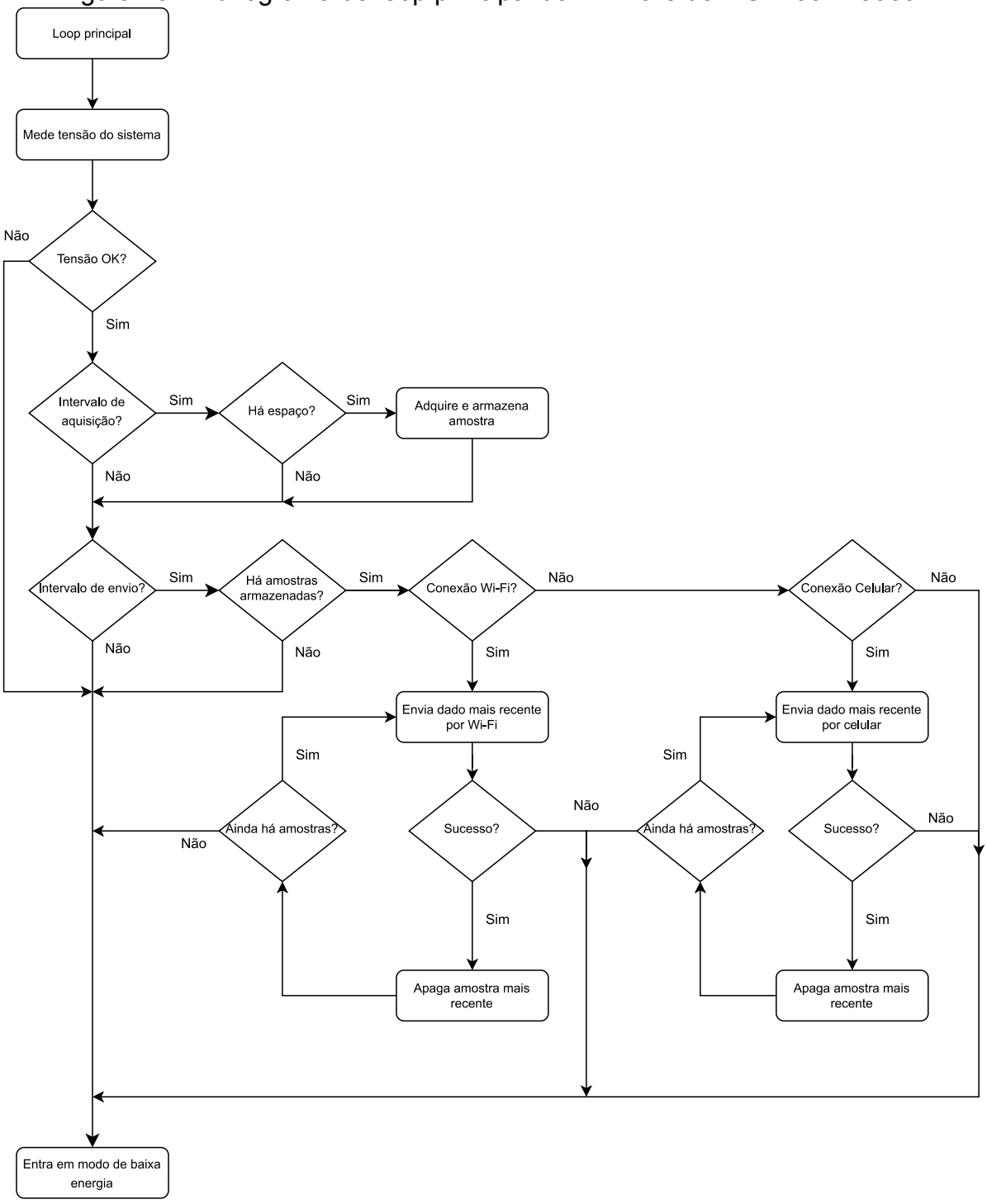
Independente de quais dispositivos de comunicação sejam detectados, também faz parte da etapa de inicialização, após essa detecção, a aquisição da data e hora reais e sincronização do relógio de tempo real interno a essa data e hora. Isto é realizado interrogando um servidor NTP remoto. Esta ação é crítica ao funcionamento do sistema, pois se ele está sendo inicializado, é porque esteve desenergizado, e desta forma o relógio interno foi perdido. Sem uma referência de tempo real, é impossível atribuir data e hora às leituras de temperatura; portanto, o

programa não continua para seu *loop* principal até que valores de data e hora válidos sejam adquiridos de um servidor NTP.

Verificou-se ocorrer, por exemplo, que o modem retorne um valor inválido de data e hora, por ainda não ter conseguido se conectar ao servidor NTP; por isso, é feita uma verificação de sanidade: se o ano for menor que 2023 e maior que 2050, então o valor retornado é espúrio e uma nova tentativa de aquisição de data e hora é realizada.

Realizada a etapa de inicialização, o programa entra em seu *loop* principal, mostrado na Figura 19.

Figura 19 - Fluxograma do loop principal do firmware do MSP430FR5969.



Fonte: o autor.

A primeira ação realizada no *loop* principal é a leitura do valor de tensão do sistema, através do periférico conversor analógico/digital do MSP430FR5969, que dispõe de um canal analógico interno, ligado à alimentação através de um divisor

resistivo também interno, facilitando o monitoramento da tensão de alimentação do microcontrolador.

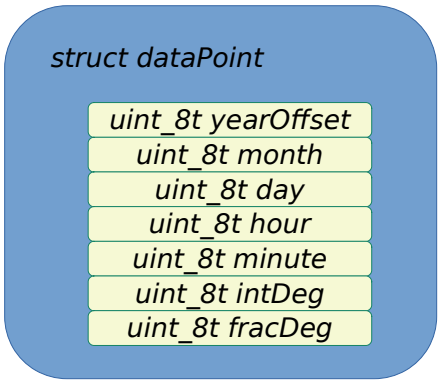
Caso esta tensão seja menor ou igual a 3,0 V, nenhuma outra ação é tomada, uma *flag* é marcada e o microcontrolador entra em modo de baixa energia (LPM3: consumo típico de 0,4 µA).

Caso a tensão do sistema tenha estado menor ou igual a 3,0 V, fato constatável pelo estado da *flag* supramencionada, mas esteja ainda abaixo de 3,1 V, a *flag* não é livrada, e o microcontrolador continua retornando diretamente ao LPM3. Caso contrário, se a tensão do sistema houver estado abaixo de ou igual a 3,0 V, mas, no momento, se encontrar acima de 3,1 V, isto é indicativo de que a bateria foi posta sob recarga, e é seguro retornar ao modo normal de operação. Para isto, é livrada a *flag* que indica que a bateria está esgotada. Toda esta tomada de decisão está contida, por simplicidade, no bloco “Tensão OK?” do fluxograma.

Encontrando-se o sistema em condições aceitáveis de alimentação, o programa verifica se o tempo, em minutos, durante o qual o sistema esteve ligado, é múltiplo de uma variável que determina o intervalo de aquisição de dados. O tempo em minutos de operação do sistema é guardado em uma variável do tipo *long unsigned int*, de 32 bits (TEXAS INSTRUMENTS, 2021) e que portanto armazena valores inteiros de 0 a 4294967295 minutos, ou mais de 8170 anos, e portanto não se preocupou com a possibilidade de *overflow* desta variável.

Caso o tempo de operação do sistema seja múltiplo do intervalo de aquisição configurado, verifica-se a disponibilidade de espaço de memória para uma nova aquisição. Do modo como se encontra configurado, o microcontrolador dispõe de espaço para 640 entradas, representadas em memória de modo estruturado, através de um *array* de *structs*, as quais são ilustradas pela Figura 20.

Figura 20 - Estrutura de dados armazenados.



Fonte: o autor.

Cada entrada de dados deve ter um *timestamp* associado, contendo ano, mês, dia, hora e minuto da aquisição, bem como a leitura de temperatura, armazenada em partes inteira e fracionária separadamente. Cada um destes valores é armazenado em uma variável do tipo *uint_8t*, de 8 bits, portanto capaz de armazenar valores de 0 a 255. Como o ano tem, pelos próximos 7976 anos, 4 dígitos, decidiu-se armazená-lo como a quantidade de anos desde 2020: assim, o ano de 2023 seria armazenado na variável *yearOffset* como o valor 3, de modo que é possível armazenar, em uma variável de 8 bits, valores de ano até 2275.

Cada ponto de dados, então, ocupa 7 bytes, e as 640 posições de memória totalizam 4480 bytes. Embora o espaço de memória não-volátil do microcontrolador seja compartilhado com a memória de programa, há amplo espaço para ampliação dessa capacidade; em tempo de desenvolvimento, porém, para evitar constantes reconfigurações do mapa de memória, optou-se por manter a capacidade de armazenamento nesta quantia, que ainda assim representa, a uma taxa de aquisição típica, a cada 5 minutos, uma autonomia de armazenamento de 53 horas e 20 minutos.

Caso haja espaço disponível, é feita uma leitura de temperatura, que é registrada dentro de um novo elemento no *array* de *structs* do tipo *dataPoint*, junto com as informações de data e hora de aquisição, para posterior envio. Cabe reforçar que, portanto, o comportamento padrão do sistema é de armazenar os dados em memória não-volátil, de modo a assegurar a integridade dos dados. A cada nova entrada, é incrementado o valor de uma variável de índice, que indica a posição, dentro do *array,* da última leitura realizada. Naturalmente, a leitura e armazenamento

não são realizados caso não haja espaço em memória, e de todo modo o programa segue seu fluxo principal em seguida.

A próxima verificação, de maneira análoga ao caso para aquisição de dados, é se o tempo de operação do sistema é múltiplo do intervalo de transmissão de dados. Caso seja, é iniciado um ciclo de transmissão de dados para o servidor.

Primeiramente, o microcontrolador verifica se há mensagens de dados armazenadas no *datalogger*. Se não houver, o ciclo de transmissão é interrompido, e o microcontrolador retorna ao modo de baixa energia.

Havendo dados armazenados, o microcontrolador interroga o módulo Wi-Fi, caso tenha sido determinado, durante a inicialização, que este se encontra presente. Em caso positivo, o programa verifica se há conexão Wi-Fi ativa. Caso qualquer das verificações até aqui, relativas à conexão Wi-Fi, falhe, verificações análogas são feitas em relação à conexão via celular, se houver sido detectado, durante a inicialização, que há um modem celular presente.

Seja qual for o meio de comunicação a ser usado, determinado pelas verificações do programa, uma vez estabelecido que há uma conexão ativa, este procura realizar um envio um ponto de dados, através do método HTTP GET, como detalhado na Seção 4.6. Em caso de sucesso no envio, constatado pelo código de status HTTP 200 (OK) retornado pelo servidor, o índice do *array* de dados é decrementado, o que tem o efeito de marcar aquela posição de memória, ocupada pelo elemento do *array*, como livre para ser sobrescrita – em outras palavras, efetivamente apagando o dado enviado da memória, já que dados fora do índice jamais são acessados novamente.

A seguir, se o decremento do índice de pontos não fizer com que este índice se torne zero – ou seja, se ainda houver elementos no *array* de dados – é feita nova tentativa de envio, até que não restem elementos no *array*, ou até que ocorra uma condição de erro no envio, que pode ser causada pelo retorno, por parte do servidor, de um código de status diferente de 200, ou por transcorrer um tempo-limite da requisição.

Caso ocorra algum erro no envio, naturalmente, o índice do *array* de dados não é decrementado, e os dados remanescentes no *array* são mantidos. Uma vez que ocorra um erro no envio, não são feitas novas tentativas imediatamente; ao invés disso, uma nova tentativa é realizada no ciclo de transmissão de dados seguinte, caso este logre êxito em seus envios. Optou-se por fazer com que o

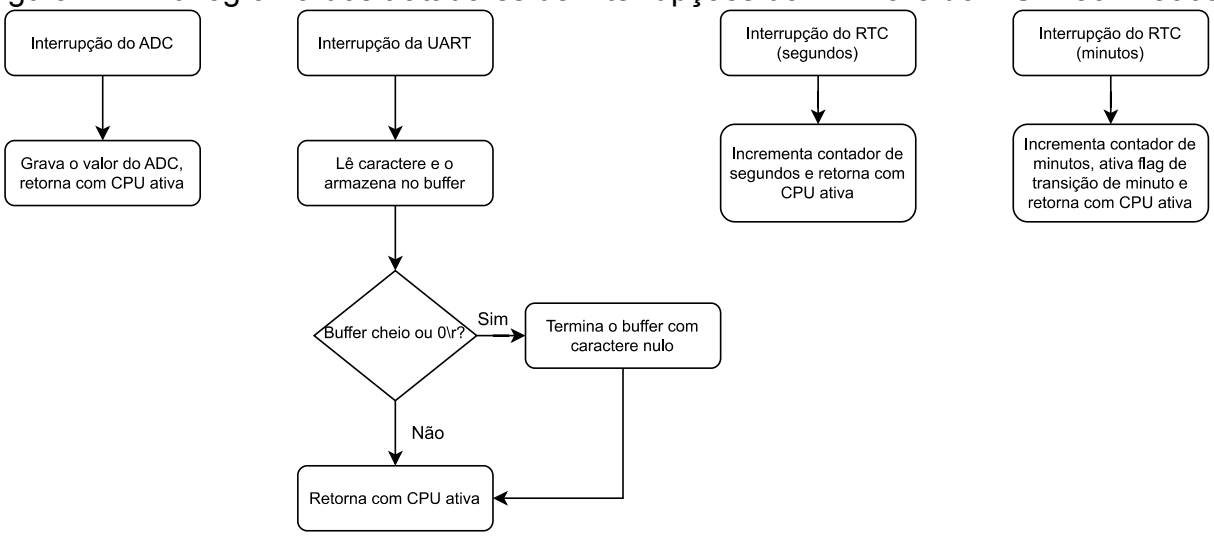
sistema trabalhe desta forma, por considerar-se que seria mais benéfico aguardar alguns minutos, até que a situação faltosa possivelmente se resolvesse, do que insistir imediatamente na transmissão, consumindo bateria com algo que possivelmente não se resolveria imediatamente, já que a criticidade da aplicação é mais em relação à integridade e regularidade de dados, e menos em relação ao caráter de monitoramento em tempo real.

Após o fim do ciclo de transmissão, seja por todos os dados terem sido transmitidos ou por uma falha ter interrompido o ciclo, o microcontrolador desliga os módulos de comunicação e volta a dormir, em modo de economia de energia. O mesmo acontece caso o microcontrolador seja acordado por alguma interrupção sem que o intervalo de algum dos ciclos tenha transcorrido ainda.

É importante notar que os ciclos de leitura e transmissão de dados são desacoplados, ou seja, ocorrem a intervalos independentes um do outro: é possível, por exemplo, adquirir dados a cada 3 minutos e transmití-los a cada 10, caso isto seja desejável. Imaginou-se que, por implicar em apenas uma etapa de configuração do modem para múltiplos envios, haveria alguma economia de energia, mas isto não se verificou nos testes realizados, expostos mais adiante, na Seção 7.

Em paralelo ao *loop* principal, ocorrem as interrupções do conversor A/D, do RTC e da UART, mostradas em forma de fluxograma na Figura 21.

Figura 21 - Fluxograma dos tratadores de interrupções do firmware do MSP430FR5969.



Fonte: o autor.

A interrupção do ADC é disparada quando uma conversão A/D é concluída. Desta forma, é possível entrar em modo de baixa energia enquanto se espera por uma conversão, e a interrupção faz com que o microcontrolador volte ao modo de execução normal.

A interrupção da UART é disparada quando há recepção de um caractere. Quando isto ocorre, o caractere recebido é armazenado em um *buffer* – um *array* de comprimento fixo do tipo *char* – e, caso o buffer esteja cheio, ou for recebida a seqüência “0\r”, onde \r denota o caractere de avanço de linha (ASCII 10), o *buffer* é terminado com o caractere nulo (ASCII 0).

Isto se dá porque, recebendo *strings* de texto, caractere por caractere, é necessário haver um critério de parada, que defina o fim da transmissão; adotou-se a seqüência “0\r” por ser esta a seqüência enviada pelo SIM7070G como sinal de “OK”, que é usado para indicar que o comando foi recebido e/ou executado com sucesso.

As rotinas de comunicação, fora da interrupção, aguardam pelo final do envio, detectado pelo tratador de interrupção pela seqüência “0\r”, por um determinado período, e caso o final do envio não ocorra dentro deste período, seja pelo SIM7070G ter retornado um resultado que não “OK”, ou se porventura não responder nada, considera-se que houve uma falha na comunicação ou no último comando enviado.

A terminação do *buffer* com um caractere nulo é feita para que o *buffer* seja considerado uma *string* em linguagem C, e desta forma possa ser processado por funções que manipulam *strings*, que usam o caractere nulo para determinar até onde vai o conteúdo relevante dentro de um *array* do tipo *char.* Estas funções são usadas para extrair as informações relevantes das mensagens enviadas pelo modem.

O relógio de tempo real dispara periodicamente duas interrupções: uma a cada segundo, que é usada para temporizações mais curtas dentro do programa, como a espera por uma resposta do dispositivo de comunicação, e uma a cada minuto, para controle da aquisição e transmissão de dados.

A interrupção que ocorre a cada segundo apenas incrementa um contador de segundos transcorridos desde a energização do circuito. As rotinas de temporização fora da interrupção, quando acionadas, guardam o valor do contador no momento do acionamento e aguardam até que a diferença entre o valor do

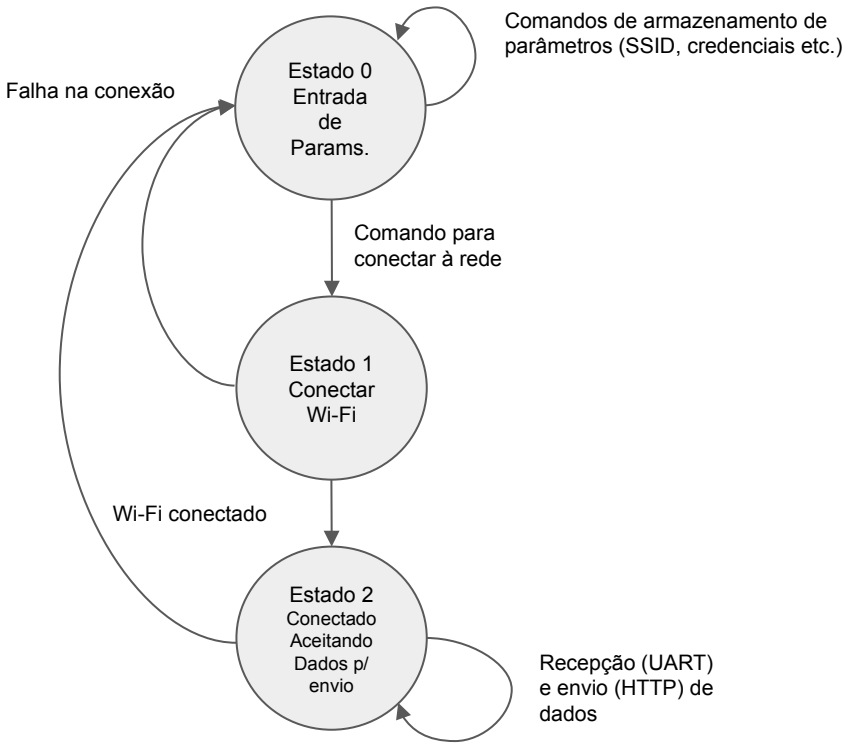
contador, que é incrementado pela interrupção, e o valor guardado atinja um determinado valor de intervalo de espera da rotina.

A interrupção que ocorre a cada minuto, além de incrementar um contador de minutos, de forma análoga à interrupção de segundos, marca uma *flag* que indica que houve uma transição de valor: isto é feito para evitar a contínua repetição de um ciclo de aquisição ou envio enquanto o relógio estiver naquele minuto. Ao fim de um ciclo, o *loop* principal livra a *flag* de transição do relógio, que só volta a ser marcada no próximo minuto, pelo tratador de interrupção.

3.4.2 Firmware do módulo Wi-Fi

A programação do módulo Wi-Fi, implementado por um ESP8266, tem por objetivo servir às demandas de comunicação do microcontrolador principal. Seu funcionamento é mostrado em forma de diagrama de estados na Figura 22:

Figura 22 - Diagrama de estados do firmware do módulo Wi-Fi.



Fonte: o autor.

Inicialmente, o programa se encontra no estado de entrada de parâmetros. Neste estado, o dispositivo encontra-se desconectado, e aceita parâmetros de conexão, tais como identificação da rede Wi-Fi, informações de autenticação na rede, além de parâmetros de envio de dados, como URL do servidor remoto, informações de autenticação no servidor remoto e número de série do dispositivo.

Mediante recepção de um comando de solicitação de conexão, o programa transiciona para o estado de conexão à rede Wi-Fi. Este é um estado intermediário, em que o programa passa alguns segundos tentando se conectar à rede informada no estado anterior, com as informações de autenticação fornecidas neste estado. Caso uma conexão não seja estabelecida no intervalo arbitrado, o programa retorna, pela UART do ESP8266, um código de erro, e retorna ao estado de entrada de parâmetros. Caso obtenha sucesso na conexão, o programa retorna um código indicando este resultado através da UART, e transiciona para o estado final, em que se encontra conectado e aceitando dados para envio.

No estado de aceitação de dados para envio ao servidor remoto, é possível que a conexão Wi-Fi seja, por algum motivo, desfeita, situação em que o programa retorna ao estado inicial. Não ocorrendo isso, o programa se encontra apto a receber dados através da UART para envio ao servidor remoto. Esses dados são enviados pré-formatados pelo microcontrolador principal, e nenhum processamento ou verificação é feita sobre eles por parte do módulo Wi-Fi.

Embora uma requisição HTTP GET seja realizada a cada envio de dados, optou-se por não enviar, do microcontrolador principal ao módulo Wi-Fi, a URL completa da requisição a cada envio pois, apesar de que isso simplificaria a programação e operação do módulo Wi-Fi, implicaria em envios de dados mais longos, contendo informações redundantes. A fim de minimizar o tempo em que o dispositivo Wi-Fi se encontra ligado e conectado, consumindo energia, com operações de comunicação, optou-se por enviar as informações que, do contrário, seriam redundantes, durante o estado inicial do programa, para que então sejam reaproveitadas para múltiplos envios.

Nesse estado, o módulo Wi-Fi pode, ainda, ser interrogado a respeito do horário, para sincronização do relógio interno do microcontrolador principal, como discutido anteriormente.

3.4.3 Saída de *debug* do *firmware*

A saída de *debug*, através de uma segunda interface UART, é bastante útil para acompanhar e validar o funcionamento do programa. Alguns exemplos de saída são mostrados a seguir, mostrando também os comandos e respostas do dispositivo de comunicação.

Em sua inicialização, o código pode ter a seguinte saída:

```
1W Init:
28FEF1B3060000DC
lsb = 86
msb = 01
24.37
```

Este trecho indica que foi detectada uma sonda, com o número de série 28FEF1B3060000, *checksum* DC, e que a temperatura medida por ela é 24,37 °C. O programa prossegue:

```
WLAN enable...
RTS SSID...
WLAN not detected.
WLAN disabled.
Modem detected.
```

Neste caso, o programa detectou a presença do modem, mas não detectou um módulo Wi-Fi. O programa prossegue, inicializando o modem em modo LTE Cat M1, configurando a APN e ativando a conexão:

```
AT+CNMP=38
AT+CGDCONT=1,"IP","inlog.claro.com.br"
AT+CNACT=0,1
+APP PDP: 0,ACTIVE
OK
AT+CNACT?
+CNACT: 0,1,"10.64.26.145"
+CNACT: 1,0,"0.0.0.0"
+CNACT: 2,0,"0.0.0.0"
```

```
+CNACT: 3,0,"0.0.0.0"
Connected
```

Após isto, o microcontrolador faz consultas a um servidor NTP, a fim de sincronizar o relógio interno. Por duas tentativas as consultas retornaram valores espúrios, ao que o código reagiu, como apontado na Seção 5.4.1, repetindo as requisições até adquirir data e hora aceitáveis:

```
+CCLK: "80/01/06,00:01:08+00"
6/1/80 0:1:8
+CCLK: "80/01/06,00:01:09+00"
0
+CNTP: 1,"2023/06/26,00:01:20"
26/6/2023 0:1:20
+CCLK: "23/06/26,00:01:20+00"
26/6/23 0:1:20
```

Por fim, o microcontrolador desliga o modem, e entra em modo de economia de energia:

```
Modem cycling...
NORMAL POWER DOWN
+APP PDP: 0,DEACTIVE
Modem cycled.
```

Após algum tempo, a interrupção acorda o microcontrolador, e é hora de um ciclo de aquisição. Nota-se que haviam 116 entradas anteriores armazenadas, de antes da última inicialização, comprovando o adequado funcionamento da memória não-volátil. O microcontrolador mede a tensão da bateria, realiza uma leitura de temperatura da sonda, e armazena este novo ponto de dados, com *timestamp*, sob o índice 117:

```
mVBat = 3269
lsb = 86
msb = 01
FRAM index 117: 24,37@2023-6-26T0:3:00Z
```

Do modo como estava configurado, o microcontrolador faz outras duas aquisições antes de um ciclo de envio, o qual, no entanto, após ligar e conectar o modem à rede celular, falha, possivelmente por *timeout*:

```
Is modem alive?
No answer.
Modem cycling...
RDY
+CFUN: 1
+CPIN: READY
SMS Ready
Modem cycled.
Modem is alive, non-verbose mode
AT+CNACT=0,1
+APP PDP: 0,ACTIVE
AT+CNACT?
+CNACT: 0,1,"10.64.26.145"
+CNACT: 1,0,"0.0.0.0"
+CNACT: 2,0,"0.0.0.0"
+CNACT: 3,0,"0.0.0.0"
Modem connected.
Sending data...
24.18@2023-6-26T0:9:00Z
AT+SHREQ="/app/httpds?__device=user:pass&SN1010
1010_BATMV=3260&SN10101010_OW=24.18@2023-6-26T0:9:00Z",1
Error sending data through modem.
Disconnecting...
AT+CNACT=0,0
+APP PDP: 0,DEACTIVE
Shutting modem down...
Modem cycling...
NORMAL POWER DOWN
Modem cycled.
573
```

O ciclo é encerrado no instante em que o dispositivo estava ligado havia 573 segundos. Novos ciclos de aquisição são realizados, após os quais um novo ciclo de envio também é iniciado, desta vez conseguindo enviar várias mensagens:

```
        23.18@2023-6-26T0:18:00Z
        AT+SHREQ="/app/httpds?
__device=user:pass&SN10101010_BATMV=3253&SN10101010_OW=23.18@2023-6-
26T0:18:00Z",1
        +SHREQ: "GET",200,0
        Data sent successfully.
        23.62@2023-6-26T0:15:00Z
        AT+SHREQ="/app/httpds?
__device=user:pass&SN10101010_BATMV=3253&SN10101010_OW=23.62@2023-6-
26T0:15:00Z",1
        +SHREQ: "GET",200,0
        Data sent successfully.
        ...
        19.75@2023-6-22T4:58:00Z
        AT+SHREQ="/app/httpds?
__device=user:pass&SN10101010_BATMV=3253&SN10101010_OW=19.75@2023-6-
22T4:58:00Z",1
        +SHREQ: "GET",200,0
        Data sent successfully.
        ...
```

Nota-se que, após os envios do dia 26/6, o dispositivo passou a realizar envios do dia 22/6, o último dia em que ele estivera ligado anteriormente. Vários outros envios são realizados neste ciclo, o que se reflete no índice da próxima aquisição:

```
        mVBat = 3248
        lsb = 71
        msb = 01
        FRAM index 99: 23,6@2023-6-26T0:24:00Z
```

E assim sucessivamente. O processo se dá de forma semelhante para comunicação via Wi-Fi:

```
        1W Init:
        28FEF1B3060000DC
        lsb = 93
        msb = 01
        25.18
```

A detecção da sonda, como visto, se dá de maneira idêntica. A seguir, o microcontrolador tenta detectar os dispositivos de comunicação:

```
WLAN enable...
RTS SSID...
RTS WLAN Pass...
RTS App...
RTS App Pass...
RTS App User...
RTS Device S/N
WLAN Init
WLAN detected.
1687794347
WLAN disabled.
```

Após ativar o módulo Wi-Fi, o microcontrolador principal envia a ele os dados necessários ao estabelecimento de uma conexão. Uma vez estabelecida esta, é feita uma requisição a um servidor NTP, o que retorna o valor 1687794347, mostrado na saída do programa, que é a representação em *Unix Time* – a quantidade de segundos desde 1/1/1970 00:00:00 UTC – do horário no momento da requisição.

Em seguida, o programa busca um modem, que não se encontra disponível:

```
Modem cycling...
Modem cycled.
Waiting for Modem to come up...
Is modem up?
Modem not detected.
NOK
Modem cycling...
Modem cycled.
```

Os ciclos de aquisição, por serem independentes dos ciclos de envio, se dão da mesma forma, seja qual for o modo de comunicação preferencial, e por brevidade a saída deles será omitida. Basta dizer que foram efetuadas três aquisições, após as quais uma tentativa de envio foi realizada e falhou:

```
WLAN enable...
RTS SSID...
WLAN not connected.
615
```

O programa retornou ao modo de baixa energia ao fim da tentativa, no instante em que a execução completava 615 segundos. Mais três aquisições foram realizadas, totalizando 6, após o que foi feita nova tentativa de envio, que resultou em sucesso. Primeiramente, o módulo Wi-Fi foi reinicializado, da mesma forma quando o dispositivo foi ligado:

```
WLAN enable...
RTS SSID...
RTS WLAN Pass...
RTS App...
RTS App Pass...
RTS App User...
RTS Device S/N
WLAN Init
WLAN connected.
```

Em seguida, foram efetuados os envios de dados. Nota-se que foram enviados os dados de todas as 6 aquisições, inclusive aquelas executadas antes da falha no envio:

```
Sending data...
24.50@2023-6-26T16:4:00Z
Data sent successfully.
24.37@2023-6-26T16:1:00Z
Data sent successfully.
24.50@2023-6-26T15:58:00Z
Data sent successfully.
24.56@2023-6-26T15:55:00Z
Data sent successfully.
24.62@2023-6-26T15:52:00Z
Data sent successfully.
24.68@2023-6-26T15:49:00Z
Data sent successfully.
```

Por fim, o módulo Wi-Fi é desligado, e o microcontrolador retorna ao modo de baixa energia, aos 1231 segundos de execução, aguardando novos ciclos de aquisição e envio:

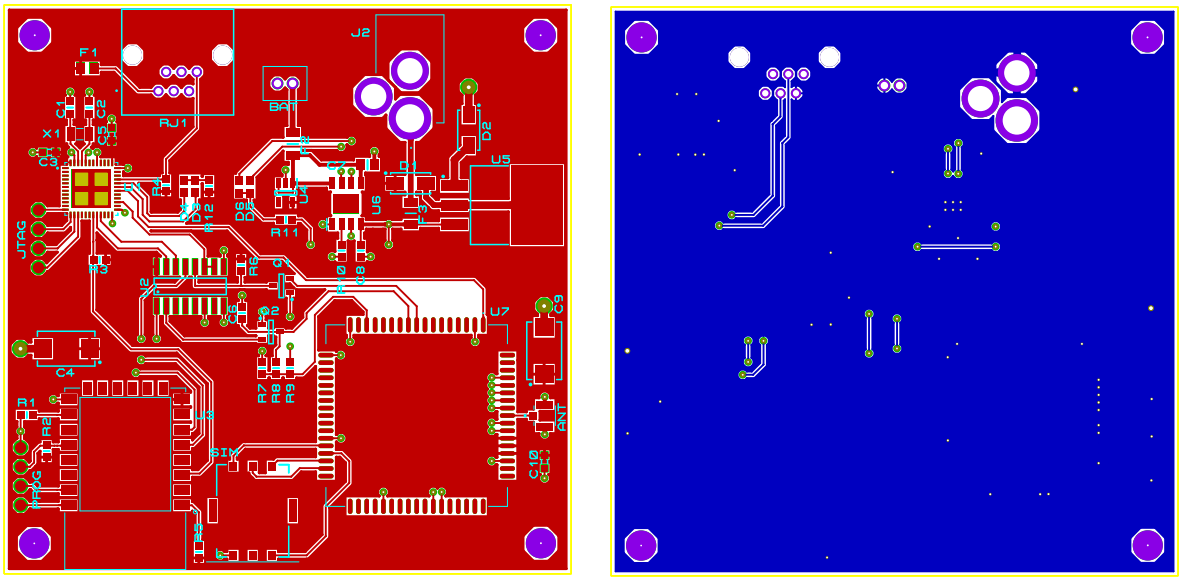
```
WLAN disabled.
1231
```

3.5 PLACA DE CIRCUITO IMPRESSO

3.5.1 Layout

Foi desenvolvido, usando a suíte EDA Labcenter Proteus 8.13, um *layout* de circuito impresso, que foi usado para fabricação pela RB Circuitos Impressos, de Pinhais/PR, que é mostrado, em tamanho real, na Figura 23.

Figura 23 - Layout da placa de circuito impresso. Camadas superior (esquerda) e inferior (direita).



Fonte: o autor.

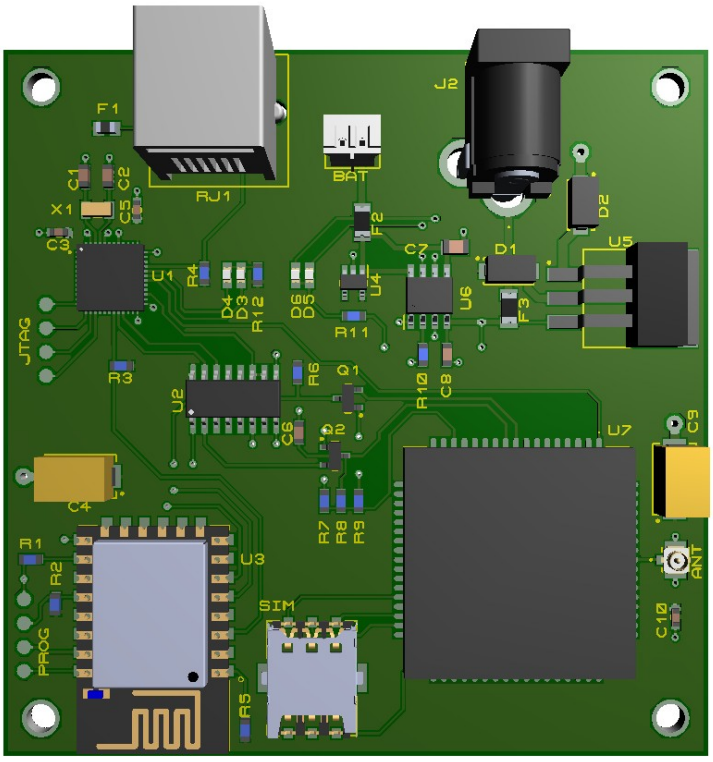
O *layout* mede 75 x 75 mm, ocupa duas camadas, e o *stackup* destas consiste em um plano terra na camada inferior, e sinais e alimentação positiva na

camada superior. Foi fabricada com substrato em fibra de vidro, na espessura de 1,6 mm, e camadas de cobre com densidade de 0,5 oz/ft².

Procurou-se, dentro das restrições de uma placa de face dupla, usar de boas práticas de *layout* de circuito impresso, evitando interromper o plano terra na medida do possível. De fato, apenas 9 trilhas de sinal, de um total de 203, passam pela camada inferior. A ligação entre o conector U.FL da antena de telefonia celular e o modem foi realizada através de uma trilha tão curta quanto possível, de modo a minimizar a irradiação espúria. O *pad* térmico do CN3058E (U6) é ligado à camada de baixo através de múltiplas vias, de modo a maximizar a transferência térmica para a ampla camada de cobre que constitui o plano terra.

A disposição dos componentes foi, tanto quanto possível, setorizada em quadrantes: o quadrante superior esquerdo é ocupado pelo microprocessador principal; o quadrante superior direito é ocupado pelo subcircuito de alimentação; o quadrante inferior direito é ocupado pelo modem celular, e o quadrante inferior esquerdo é ocupado pelo módulo Wi-Fi. Esta ocupação fica mais clara em um modelo tridimensional da placa, mostrado na Figura 24:

Figura 24 - Visão tridimensional da placa de circuito impresso.



Fonte: o autor.

Por compatibilidade com as sondas atualmente usadas pela Sensorweb, adotou-se, para conexão destas, um receptáculo modular 6P6C, por onde também passam os sinais da porta UART de *debug* do microcontrolador principal. Existem ainda *pads* expostos, designados coletivamente por “JTAG” e “PROG”, que serão usados com uma jiga externa para programação do MSP430FR5969 e do ESP8266, respectivamente.

O custo de produção da placa de circuito impresso, em escala de protótipo (4 peças) e prazo de entrega emergencial (3 dias úteis), foi de R$110/peça, conforme valores praticados em 15 de junho de 2023. Em escalas de 10 e 100 peças, com prazo de entrega menos restrito e conforme orçamento de 21 de junho de 2023, o custo de produção unitário passaria a ser de R$33,90 e R$14,80, respectivamente; em comparação, produzindo-se na China, pelo fabricante PCBWay, este custo seria de $0.89/peça (100 peças), com prazo de produção de 4 dias, conforme orçamento realizado em 21 de junho de 2023.

3.5.2 Lista de Materiais

A Tabela 7, abaixo, mostra a lista de materiais, com preços em dólares americanos, consultados nas lojas *online* Digikey e LCSC em 15 de junho de 2023.

Tabela 7 - Lista de materiais.

Capacitores					
Quantidade	Designadores	Valor	Tipo	Preço unitário	
				Digikey	LCSC
2	C1, C2	10 pF 16 V	Cerâmico 0603	$0.20	$0.0037
3	C3, C5, C10	100 nF 16 V	Cerâmico 0603	$0.10	$0.0168
2	C4, C9	470 µF 10 V	Tântalo SMD	$1.73	$0.6126
1	C6	33 pF 50 V	Cerâmico 0603	$0.16	$0.0035
1	C7	10 µF 6.3 V	Cerâmico 0805	$0.33	$0.0115
1	C8	1 µF 10 V	Cerâmico 0603	$0.20	$0.0039
Subtotal				$4.85	$1.30
Resistores					
	Designadores	Valor	Encapsulamento	Preço unitário	

Quantidade				Digikey	LCSC
6	R1-R3, R5-R6, R8	10 kΩ	0603	$0.10	$0.0008
1	R4	3.3 kΩ	0603	$0.18	$0.0011
2	R7, R9	4.7 kΩ	0603	$0.10	$0.0008
1	R10	2.4 kΩ	0603	$0.11	$0.0009
2	R11, R12	510 Ω	0603	$0.13	$0.0009
Subtotal				$1.35	$0.01
Circuitos Integrados					
	Designação	Valor	Encapsulamento	Preço unitário	
Quantidade				Digikey	LCSC
1	U1	MSP430FR5969	QFN48	$6.01	$13.599
1	U2	CD4053	SOP16	$0.57	$0.1512
1	U3	ESP12E	SMD	$7.30	$2.1128
1	U4	MAX40200	SOT23-5	$1.00	$0.3324
1	U5	LM1117S-3.3	SOT223	$0.42	$0.1445
1	U6	CN3058E	SOP8	-	$0.4503
1	U7	SIM7070		$25.74	-
Subtotal				$41.04	$16.79
Transistores					
	Designação	Valor	Encapsulamento	Preço unitário	
Quantidade				Digikey	LCSC
2	Q1, Q2	MMBT3904	SOT-23	$0.10	$0.0057
Subtotal				$0.20	$0.01
Diodos					
	Designação	Valor	Encapsulamento	Preço unitário	
Quantidade				Digikey	LCSC
1	D1	SS12	SMA (DO-214)	$0.49	$0.0554
1	D2	DL4007	SMA (DO-214)	$0.32	$0.0054
2	D3, D5	LTSTC191KRKT	0603	$0.26	$0.0309
2	D4, D6	LTSTC191KGKT	0603	$0.26	$0.0083

Subtotal				$1.85	$0.14
Outros					
	Designação	Valor	Encapsulamento	Preço unitário	
Quantidade				Digikey	LCSC
1	ANT	U.FL SMD	-	$1.32	$1.14
1	BAT	PHR2	-	$0.17	$0.01
1	F1	PTC 120 mA 60 V	0805	$0.14	$0.06
2	F2-F3	PTC 1 A 30 V	1812	$0.16	$0.06
1	RJ1	6P6C fêmea	-	$0.45	$0.10
1	SIM	SIM8051	-	$1.05	$0.60
1	X1	32768 Hz	3215-2	$0.78	$0.17
1	J1	DCJ200	-	$1.09	-
Subtotal				$4.23	$2.20
Total				$53.52	$20.45

Fonte: o autor.

Uma observação a ser feita é que, devido as conjunturas globais da indústria eletrônica, que provocam volatilidade de estoques (FORBES, 2023), alguns dos componentes se encontravam, no momento da consulta, disponíveis em uma das lojas, mas não na outra. Se os componentes faltantes de uma loja fossem comprados na outra, os preços totais de componentes, por placa, somariam $53.97 através da Digikey, com componentes faltantes comprados na LCSC, e $47.28 através da LCSC, com componentes faltantes comprados na Digikey.

Outra observação é que os valores apresentados são para compras de poucas (1-10) unidades. Em quantidades maiores, os valores seriam consideravelmente mais baixos.

4 ENSAIOS

Os ensaios foram realizados com o protótipo baseado em módulos de desenvolvimento, por questões de tempo hábil.

Entendeu-se ser mais importante o modo de operação móvel, com conexão celular, uma vez que é neste modo em que o dispositivo operará quando em trânsito, atravessando condições de baixa cobertura de telefonia celular, e portanto mais provavelmente fazendo uso de sua funcionalidade de *datalogger*, tendo de armazenar dados na ausência de cobertura de sinal e transmití-los posteriormente, na presença de cobertura. Além disso, é em trânsito, longe de uma infraestrutura fixa de Wi-Fi e possivelmente de alimentação, que a autonomia de bateria se torna um fator importante; em modo Wi-Fi, apenas validou-se o funcionamento de forma preliminar, e não foram realizados ensaios específicos.

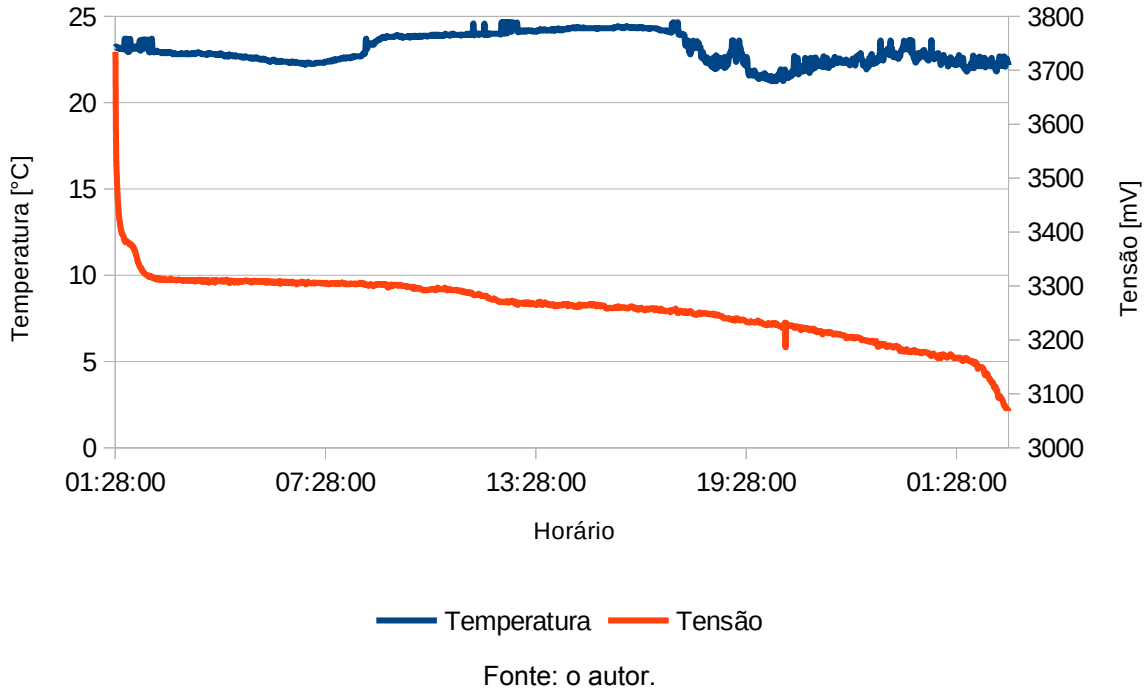
Os ensaios nesse modo de operação consistiram em carregar a bateria completamente, através de um carregador externo (Hobbyking HKC6) e energizar o circuito, de modo que realizasse aquisição e transmissão normal de dados para a instância da plataforma Endrixx designada para testes internos da empresa, até que findasse o ensaio ou a carga da bateria. Para os ensaios, o modem, configurado para operar apenas em modo LTE Cat M1, foi equipado com um cartão SIM da operadora Claro, habilitado para operação neste modo.

Estes testes se deram em áreas internas, simulando operação em modo de contingência em caso de falha da comunicação Wi-Fi, com diferentes intervalos de aquisição e envio de dados, situações em que foi também aferido o consumo de bateria, bem como no interior de um veículo em movimento, circulando pela região metropolitana de Florianópolis assim como áreas menos urbanas.

4.1 ENSAIOS ESTACIONÁRIOS

A Figura 25 mostra os dados de temperatura e tensão da bateria, transmitidos pelo protótipo em um ensaio realizado com intervalo de aquisição e envio de dados de 1 minuto, em área interna de um prédio residencial situado na cidade de Florianópolis, no entorno da Universidade Federal de Santa Catarina.

Figura 25 - Ensaio estacionário 1.



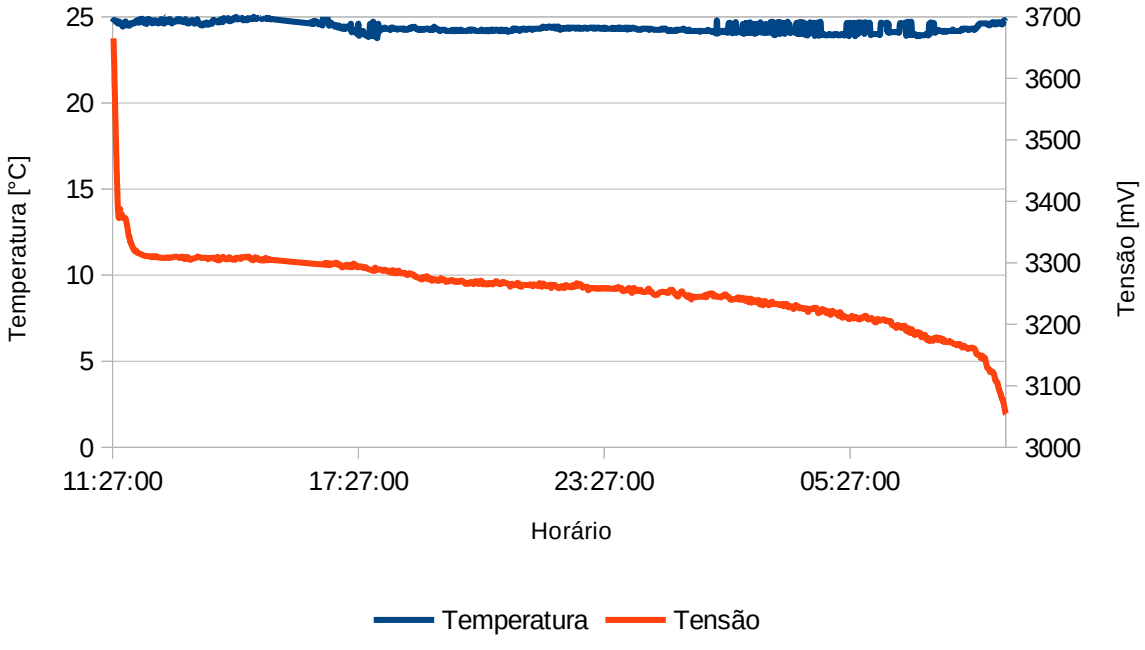
Fonte: o autor.

O ensaio foi iniciado às 1:28 do dia 24 de maio e encerrado às 2:56 do dia 25 subseqüente, de modo que, com o intervalo de aquisição e envio de dados mencionado, produziu 1752 pontos de dados em um período de 1528 minutos, ou uma média de 1,146 transmissões por minuto. A última tensão reportada pelo protótipo foi de 3068 mV, patamar após o qual as transmissões cessaram.

Admitindo-se que a bateria usada tenha de fato sua capacidade nominal (1500 mAh), e sendo 1528 minutos equivalentes a 25,467 horas, o consumo médio do protótipo no período desse ensaio foi $I_{méd\,1} = \frac{1500\,mAh}{25.467h} = 58.9\,mA$.

A Figura 26 mostra os dados recebidos pela plataforma ao longo de um segundo ensaio, nas mesmas condições, após recarga completa da bateria.

Figura 26 - Ensaio estacionário 2.



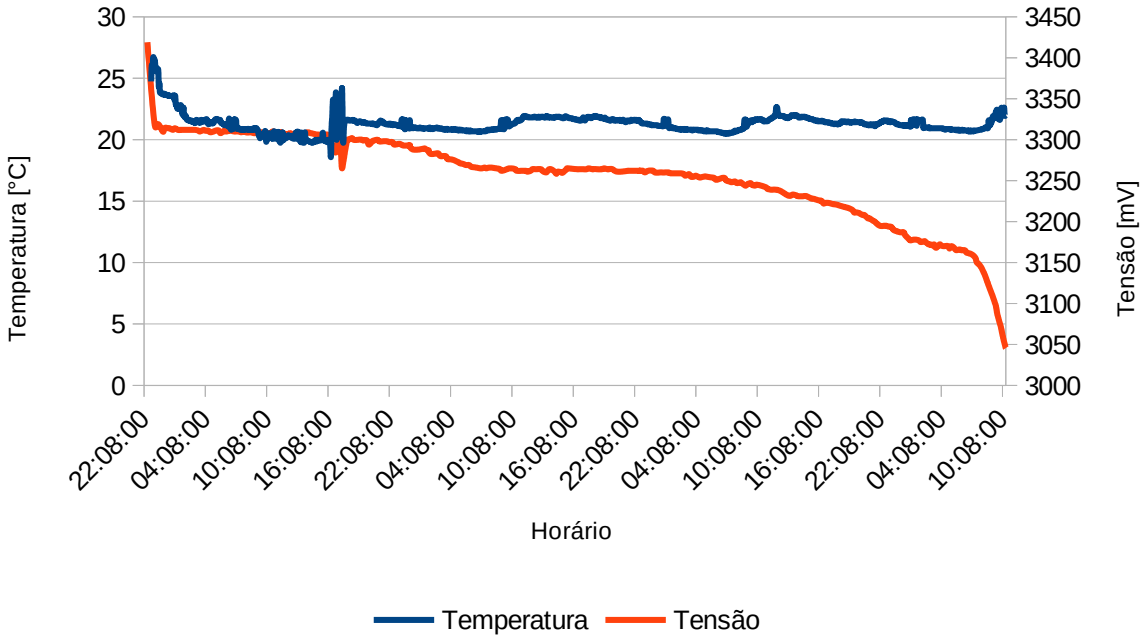
Fonte: o autor.

Este segundo ensaio foi iniciado às 11:27 do dia 25 de maio, e encerrado às 9:15 do dia 26 de maio, perfazendo um total de 1308 minutos, durante os quais foram transmitidos 1665 pontos de dados, ou uma média de 1,273 transmissões por minuto. As transmissões cessaram após um patamar de 3051 mV.

Admitindo-se, novamente, que a capacidade real da bateria seja de 1500 mAh e sendo 1308 minutos equivalentes a 21,8 horas, $I_{méd\,2} = \frac{1500\,mAh}{21.8\,h} = 68.8 mA$.

Após esses ensaios, o protótipo foi reconfigurado para que realizasse aquisições de temperatura a cada 3 minutos e envios de dados a cada 10 minutos, com o objetivo de verificar se, deste modo, haveria alguma economia em enviar blocos de dados de maneira combinada, uma vez que o processo de estabelecimento de uma conexão seria realizado uma única vez para mais que um ponto de dados. Os dados coletados durante o terceiro ensaio estacionário, sob estas novas condições de aquisição e envio de dados, são mostrados na Figura 27.

Figura 27 - Ensaio estacionário 3.



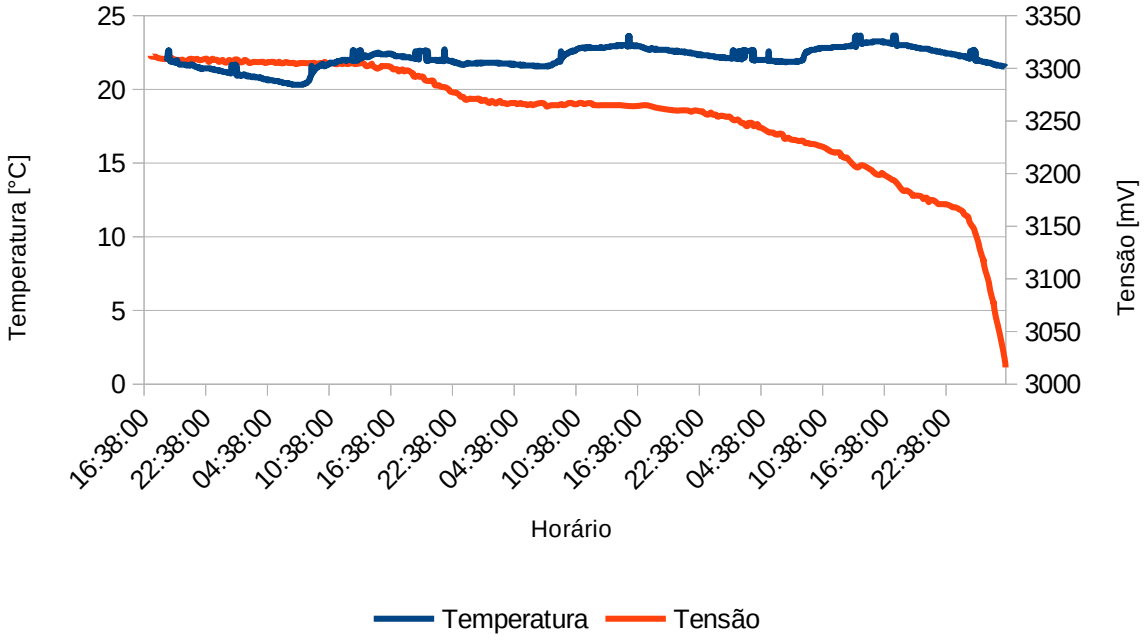
Fonte: o autor.

O ensaio foi iniciado às 22:08 do dia 27 de maio e encerrou-se às 10:26 do dia 31 do mesmo mês, perfazendo 5058 minutos, durante os quais foram enviados 1679 pontos de dados, equivalendo a uma média de 0,996 envios a cada 3 minutos. As transmissões cessaram após a tensão da bateria cair para 3045 mV.

Considerando-se que a capacidade da bateria seja de 1500 mAh, e sabendo que 5058 minutos equivalem a 84,3 horas, a corrente média consumida foi $I_{méd\,3} = \frac{1500\,mAh}{84.3h} = 17.8\,mA$.

Um quarto ensaio foi realizado nas mesmas condições do terceiro, durante o qual foram obtidos os dados mostrados na Figura 28.

Figura 28 - Ensaio estacionário 4.



Fonte: o autor.

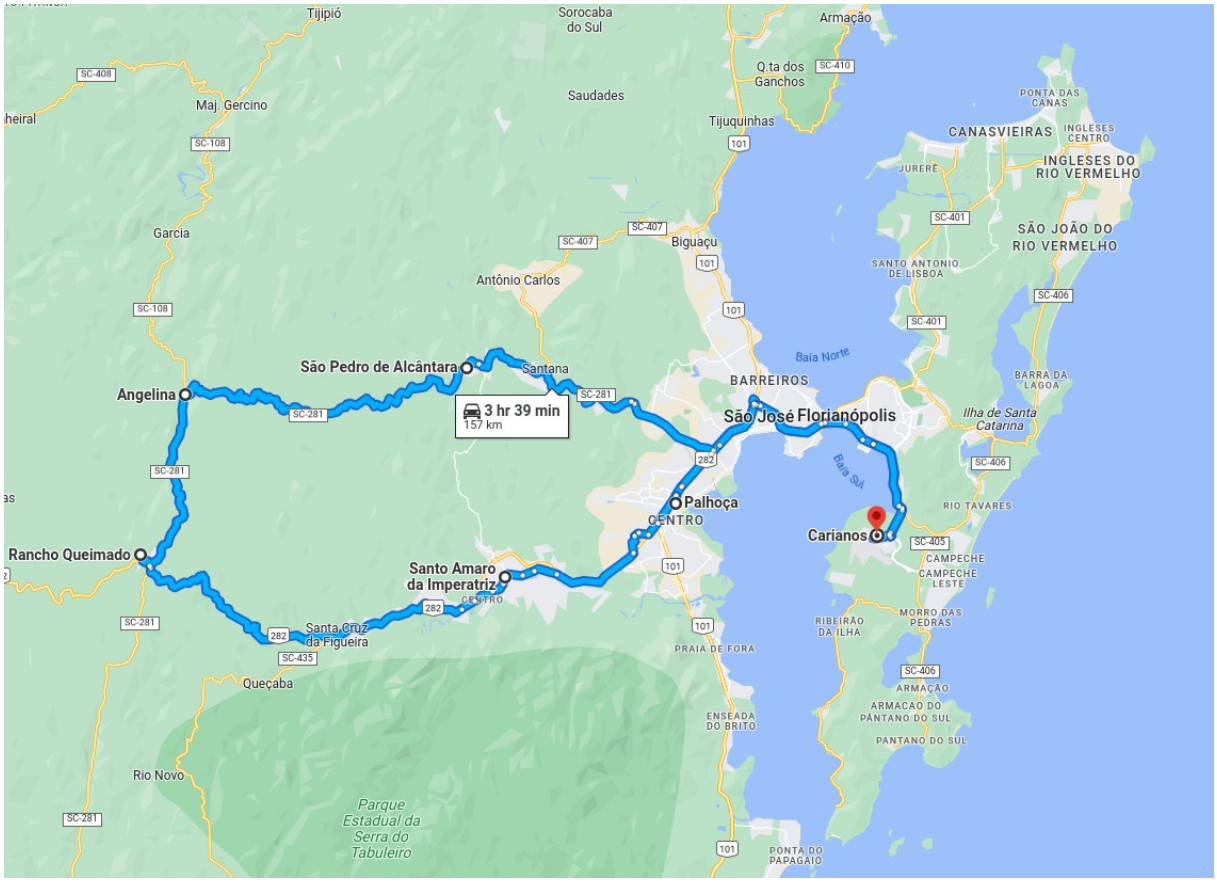
Este ensaio foi executado das 16:38 do dia 4 de junho às 4:26 do dia 8 do mesmo mês, totalizando uma autonomia de 5028 minutos. Durante o ensaio, foram enviadas 1708 mensagens de dados, ou 1,019 mensagens a cada 3 minutos. O último valor de tensão transmitido pelo protótipo foi de 3016 mV.

Fazendo as mesmas considerações dos ensaios anteriores, e com 5028 minutos equivalendo a 83,8 horas, a corrente média consumida foi $I_{méd\,4} = \frac{1500\,mAh}{83.8\,h} = 17.9\,mA$.

4.2 ENSAIO MÓVEL

Foi realizado, no dia 10 de junho, um ensaio móvel, transportando o protótipo por automóvel ao longo do itinerário mostrado na Figura 29. O dispositivo esteve configurado no regime de aquisições a cada 3 minutos e envios a cada 10.

Figura 29 - Itinerário do ensaio móvel.

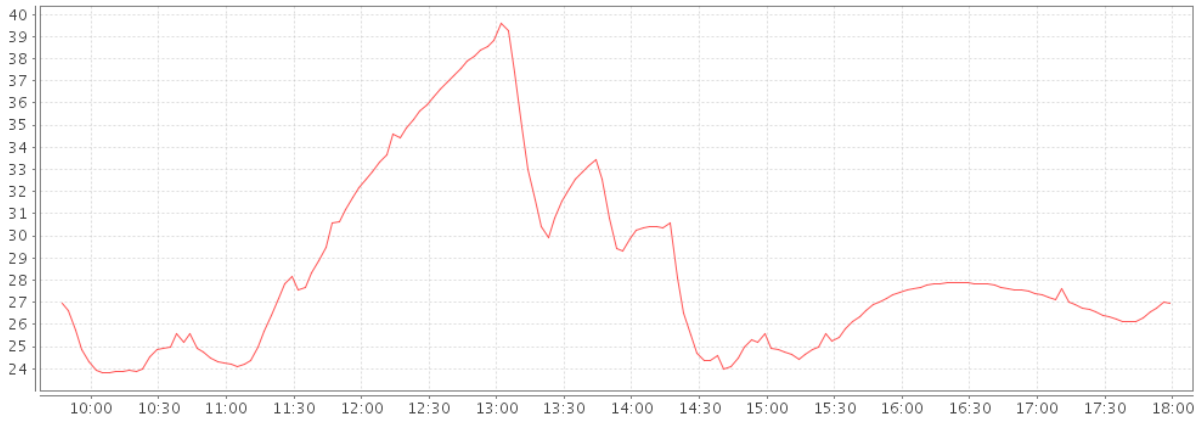


Fonte: o autor.

Este ensaio foi realizado com o intuito de verificar o comportamento do dispositivo, enquanto em modo LTE Cat M1, em condições de baixa cobertura de telefonia LTE, já que o trajeto escolhido atravessa áreas montanhosas e de floresta, afastadas de centros urbanos. O trajeto foi iniciado às 9:45 e encerrado às 18:00.

A Figura 30, obtida diretamente da plataforma Endrixx, mostra, de forma retrospectiva, as leituras de temperatura realizadas e registradas ao longo do trajeto. Deve-se notar que o dispositivo e a sonda de temperatura se encontravam no interior do veículo, o qual foi parado e deixado ao sol algumas vezes ao longo do ensaio, o que levou a flutuações consideráveis de temperatura, conforme registrado.

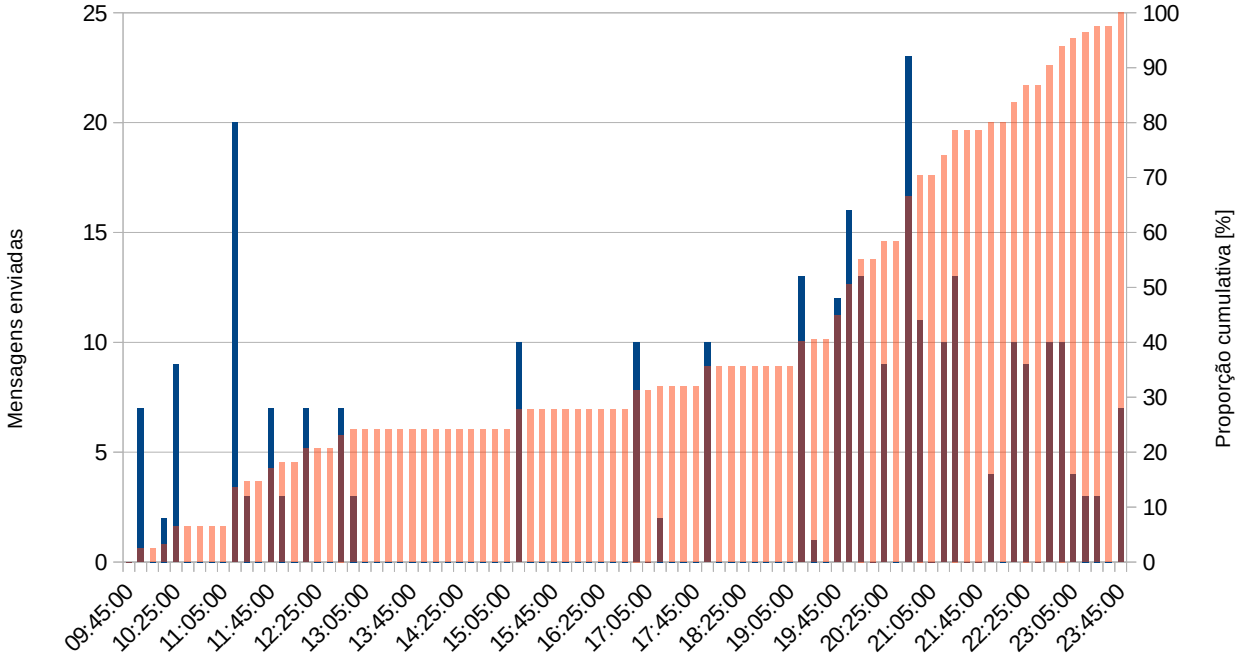
Figura 30 - Registro de temperatura durante o ensaio móvel.



Fonte: o autor.

Não se nota descontinuidades ou transições demasiado abruptas das leituras, que formam uma curva razoavelmente suave. É possível, através da plataforma Endrixx, acessar também os dados “crus” recebidos, organizados por data, de modo a determinar a quantidade de pontos de dados recebidos pelo servidor nos instantes em que se esperava haver envios pelo dispositivo. Como o dispositivo estava configurado para realizar envios a cada 10 minutos, e o primeiro envio foi realizado às 9:55, esperou-se que houvessem envios subseqüentes de, em média, 3 pontos de dados, às 9:55, 10:05, 10:15 e assim por diante. Um gráfico foi montado a partir desse levantamento, e é mostrado na Figura 31, com as quantidades absolutas enviadas mostradas em azul e a proporção cumulativa mostrada em vermelho.

Figura 31 - Quantidade de dados enviados ao longo do tempo durante o ensaio móvel.



Fonte: o autor.

Nota-se que menos de 40% dos dados referentes ao trajeto foram enviados ao longo deste: a maioria foi enviada gradualmente, após a chegada de volta a Florianópolis, ao longo de quase 6 horas, até as 23:45, completando 14 horas de operação, quando então haviam sido enviados 281 pontos de dados, representando uma média de 3,01 pontos enviados a cada 10 minutos, e o ensaio foi dado por encerrado.

5 DISCUSSÃO

Por praticidade, algumas figuras de mérito relativas aos ensaios estacionários são sumarizadas na Tabela 8:

Tabela 8 - Sumário da performance sob ensaios estacionários.

Ensaio	Intervalos [min]		Autonomia [h]	Envios	Corrente média [mA]	Cap. por envio [mAh]	Tensão de corte [mV]	Relação R/E
	Aquisição	Envio						
1	1	1	24.467	1752	1 58.9	0.856	3068	1.146
2	1	1	21.8	1665	1 68.8	0.901	3051	1.273
Média			**23.1335**	**1708.5**	**1 63.85**	**0.878**	**3059.5**	**1.2095**
3	3	10	84.3	1679	1 17.8	0.893	3045	0.996
4	3	10	83.8	1708	1 17.9	0.878	3016	1.019
Média			**84.05**	**1693.5**	**1 17.85**	**0.886**	**3030.5**	**1.007**

Fonte: o autor.

Pressupunha-se que, realizando múltiplos envios por ciclo, diminuir-se-ia o consumo de capacidade da bateria por envio, já que a conexão seria estabelecida uma única vez para múltiplos envios, ao invés de uma vez para cada envio, o que diminuiria *overheads* e, supunha-se, refletiria no consumo por envio, o que, no entanto, observando a Tabela 8, não ocorreu. Por outro lado, a relação entre a quantidade de envios realizados e a quantidade de envios esperados, denotada na Tabela 8 por “Relação R/E”, foi reduzida sob o regime de envios agrupados, o que significa que menos dados foram enviados redundantemente e, isto sim, impactou o consumo.

Dados podem ser enviados mais de uma vez caso ocorra um *timeout* enquanto se espera um retorno do servidor HTTP logo após um envio, sem que, contudo, tenha ocorrido erro algum: o programa simplesmente desistiu de esperar por uma resposta antes que esta tenha sido recebida do servidor HTTP, embora este tenha, em muitos casos, enviado essa resposta. Como o servidor HTTP

recebeu os dados, estes são registrados na plataforma; porém, como o dispositivo não recebeu a resposta “OK” do servidor, o dado é mantido em memória, e é retransmitido posteriormente. A “Relação R/E” serve de métrica da incidência de tais eventos.

Com efeito, embora o consumo por envio não tenha sido reduzido, a autonomia média foi aumentada de 3,63 vezes, ao invés de apenas 3, como se esperaria no caso de aquisições e envios a cada 3 minutos. Proporcionalmente, isto representa um ganho de autonomia de 21%, que corresponde proximamente à redução da relação R/E, de cerca de 20%.

Um fato talvez preocupante é a relação R/E ter, em um dos ensaios estacionários, caído para menos de 1: isto significa que foram recebidos menos dados do que o esperado para o período. A princípio, a menos que o espaço em memória se esgote, o que não ocorreu, já que a quantidade total de mensagens não excedeu a capacidade de memória, o dispositivo não pode simplesmente perder dados, já que as rotinas que lidam com eles foram desenvolvidas de modo a remover da memória apenas um dado cuja recepção por parte do servidor tenha sido confirmada. Os dados são, todavia, apagados da memória quando o microcontrolador tem seu *firmware* regravado, o que de fato ocorreu, como parte do ciclo normal de desenvolvimento. O mais provável é que, uma vez esgotada a bateria, tenham restado dados na memória, que não foram transmitidos posteriormente à recarga da bateria porque foram apagados quando o *firmware* foi regravado. Neste caso, trata-se, portanto, de uma falha do ensaio, e não do dispositivo, e que deve ser considerada e remediada em ensaios futuros.

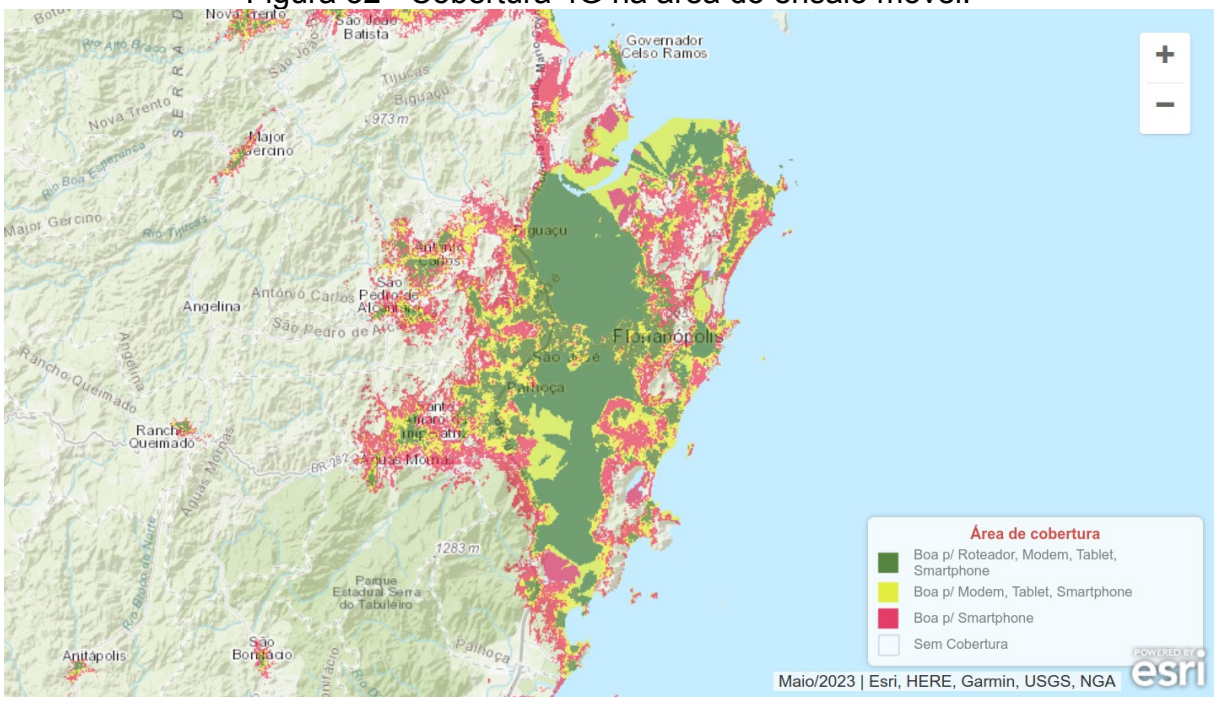
Observou-se que a tensão imediatamente antes da última transmissão do dispositivo variou entre 3016 e 3068 mV, um pouco acima do limiar de 3000 mV estabelecido no *firmware*. Isto pode se explicar de algumas maneiras diferentes: como mencionado, a tensão é medida imediatamente antes da transmissão, situação em que a corrente exigida é bastante baixa em relação àquela exigida durante uma transmissão. Assim, é plausível que, apesar dos capacitores de desacoplamento, a tensão terminal da bateria caia no momento da transmissão, interrompendo-a, seja pelo próprio modem ter detectado uma situação de subtensão e ter se desligado, ou pela rotina de detecção de subtensão implementada no *firmware* do MSP430FR5969 ter impedido a comunicação.

Outra explicação possível para o ocorrido é que haja uma diferença entre as medições da tensão da bateria realizadas pelo microcontrolador principal e pelo modem, devido a diferenças e tolerâncias de suas respectivas referências internas, que potencialmente fariam com que aquilo que é medido como acima de 3000 mV pelo microcontrolador principal seja medido como sendo inferior a esse valor pelo modem. Ainda, há que se considerar que o protótipo ensaiado está construído através de módulos fisicamente separados, conectados uns aos outros através de fios, o que não ajuda a assegurar que os diferentes componentes estejam alimentados a um mesmo potencial em todos os momentos.

Estas condições limítrofes devem ser ensaiadas mais especificamente, preferencialmente no protótipo em placa de circuito impresso.

Na Figura 31, relativa ao ensaio móvel, percebe-se que foram realizadas proporcionalmente poucas transmissões ao servidor remoto ao longo do trajeto percorrido, o que representa um prejuízo ao caráter de monitoramento em tempo real. Contudo, deve ser observado que o dispositivo operava somente em modo LTE Cat M1. Considerando-se o mapa de cobertura fornecido pela Claro e mostrado na Figura 32, para comunicações 4G, entre as quais se encontra o modo LTE Cat M1, tal resultado era de se esperar:

Figura 32 - Cobertura 4G na área do ensaio móvel.

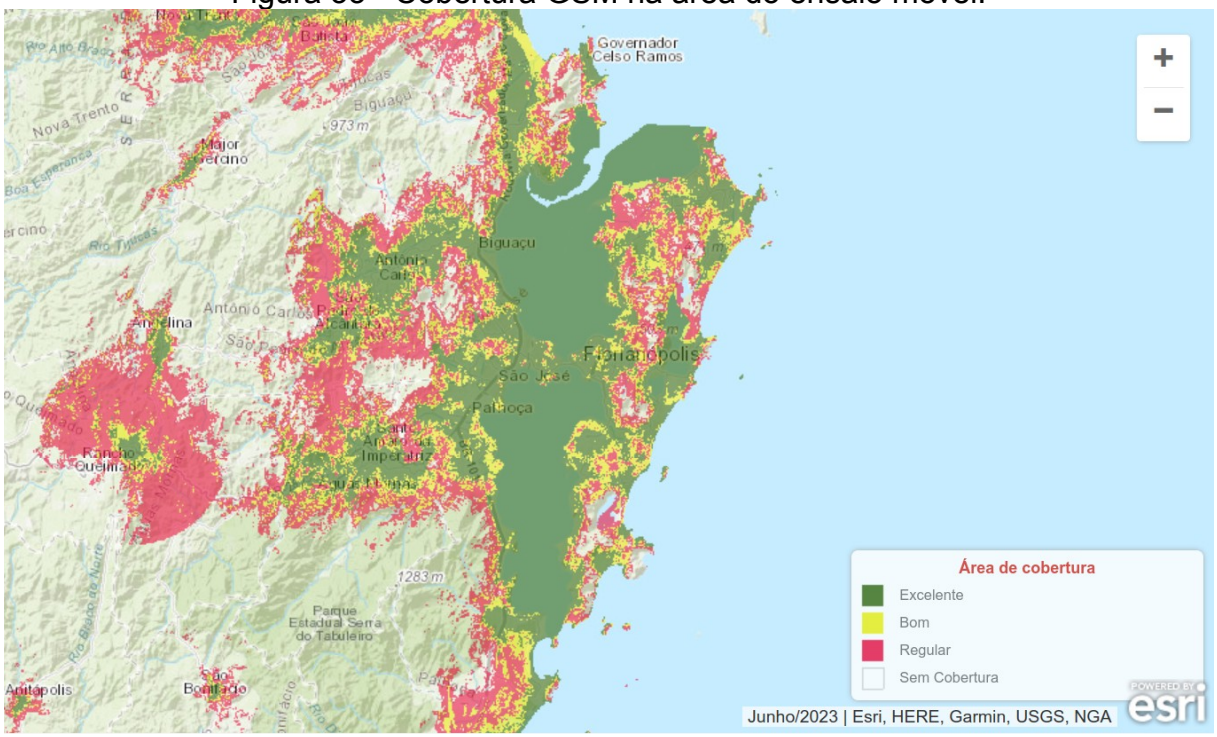


Fonte: https://www.claro.com.br/mapa-de-cobertura.

Nota-se, considerando o itinerário apresentado na Figura 29 e comparando-o com o mapa da Figura 32 que, de Santo Amaro da Imperatriz em diante, a cobertura 4G é restrita a uma pequena área do município de Rancho Queimado, e ausente por grande parte do trajeto subseqüente, até estar presente de forma esparsa na altura de São Pedro de Alcântara e de forma mais sólida apenas ao se alcançar o município de São José.

A habilitação de comunicação GSM no dispositivo provavelmente teria amenizado a situação, como sugere o mapa de cobertura GSM fornecido pela Claro, e que é mostrado na Figura 33:

Figura 33 - Cobertura GSM na área do ensaio móvel.



Fonte: https://www.claro.com.br/mapa-de-cobertura.

Os mapas mostrados pelas Figuras 32 e 33 evidenciam o estado ainda incipiente da implantação de infraestrutura 4G no Brasil, e a importância, ainda hoje, de contar com *fallback* para tecnologias de comunicação legadas, o que justifica a escolha do modem SIM7070G para este projeto, ainda que a substituição dessa infraestrutura legada por 4G seja inevitável, e apenas uma questão de tempo.

De toda forma, todos os dados correspondentes ao período do ensaio móvel foram, eventualmente, enviados, cumprindo com o objetivo de integridade dos dados.

Até o momento da escrita deste Trabalho, portanto, as especificações expostas na Tabela 9, a seguir, foram alcançadas:

Tabela 9 - Especificações alcançadas pelo projeto.

Sondas	Uma, externa, 1-Wire (DS18B20)
Comunicação	Wi-Fi, LTE
Alimentação	Externa, interna ($LiFePO_4$, 1500 mAh)
Autonomia	1700 envios, equivalentes a 23h (aquisições a cada 1 minuto), ou 84 horas (aquisições a cada 3 minutos)
Intervalo de aquisição	1 minuto (mínimo), configurável em passos de 1 minuto
Datalogger	640 aquisições
Dimensões	75x75 mm (apenas a placa)

Fonte: o autor.

6 CONCLUSÕES

Ainda que os dados tenham sido enviados, os ensaios realizados sugerem que talvez seja benéfico reconsiderar a lógica de envios. Como explicado anteriormente, na Seção 5.4.1, ao ocorrer um erro durante o ciclo de transmissão, nenhuma outra tentativa é realizada até um próximo ciclo. Talvez esta não seja a melhor lógica, e seja mais proveitoso persistir no envio por algumas vezes. Também pode ser benéfico adotar um *timeout* mais longo para as requisições, para buscar reduzir o envio redundante de dados, que se verificou ter impacto direto sobre a autonomia do dispositivo.

Também merece atenção, no sentido de reduzir a quantidade de energia gasta por envio, apesar dos resultados negativos obtidos até então, maneiras alternativas de se combinar dados de modo a reduzir *overheads*. Do modo como se encontra presentemente implementada, caso haja dados, a lógica de envio realiza múltiplas requisições GET sucessivas ao servidor, até que não restem dados ou até que ocorra um erro. Estas requisições poderiam ser combinadas, de modo a mandar, em uma única requisição GET, valores de uma mesma variável com *timestamps* diferentes, e deste modo talvez se alcançasse um gasto de energia por envio mais baixo. Além disso, é concebível a criação de uma lógica que omita o envio de eventuais repetições consecutivas de um mesmo valor de temperatura, com uma certa tolerância – por exemplo, a especificação de erro da sonda – a menos que já tenham se passado alguns minutos desde o último envio, de modo a evitar o disparo de alertas de falha de comunicação na plataforma Endrixx.

Outra questão que sem dúvida impacta a autonomia de bateria do protótipo ensaiado é o fato de que o módulo de desenvolvimento do SIM7070G usado é dotado de um LED vermelho que permanece aceso ininterruptamente. Com auxílio de um multímetro (Aneng AN8008), verificou-se que o módulo de desenvolvimento do modem, desligado, consome uma corrente de 11,4 mA, a maior parte da qual presume-se ser destinada à alimentação desse LED. A placa de circuito impresso desenhada não implementa nada semelhante: LEDs indicadores são controlados pelo microcontrolador principal e pelo circuito integrado carregador de bateria, que se encontrará desligado quando o dispositivo não estiver alimentado por fonte externa. Admitindo-se que o LED gaste 10 mA, a ausência deste, de modo a causar redução de 10 mA no consumo médio, considerando-se o cenário com intervalo de

aquisição de 3 minutos, representa uma redução proporcional, e conseqüente aumento de autonomia, de cerca de 80%, possivelmente levando a autonomia, nesse cenário, a 150 horas.

Um aspecto que não foi analisado nos ensaios para este Trabalho foi o chaveamento entre Wi-Fi e rede celular: os ensaios foram realizados com um meio de comunicação apenas. Isto será apreciado uma vez montado o protótipo em placa de circuito impresso dedicada, que infelizmente não pôde ser completado a tempo do término da escrita deste Trabalho, por dificuldades logísticas em se obter todos os componentes a tempo.

Também seria conveniente, no futuro, implementar atualizações de *firmware* e/ou parâmetros operacionais *over-the-air* (através do ar), sem necessidade de uma conexão física ao dispositivo; assim, atualizações poderiam ser realizadas remotamente, sem necessidade de deslocar um técnico ao local da instalação do equipamento, ou de tirar o equipamento de campo para realizar essa atualização. O MSP430FR5969 pode ter seu *firmware* escrito desta forma, através da UART que se encontra conectada ao ESP8266. Este poderia receber arquivos de *firmware* de um servidor e realizar a gravação, tanto de seu próprio *firmware,* algo a que tem suporte, quanto do microcontrolador principal.

Um *bug* de firmware que foi notado e que deve ser corrigido é que, durante a inicialização, o microcontrolador principal não realiza quaisquer verificações de nível de tensão de alimentação, e assim tenta inicializar os dispositivos de comunicação, que podem não se encontrar responsivos devido a um nível de alimentação aquém de 3,0 V, situação na qual o microcontrolador principal incorretamente os consideraria como ausentes, sendo necessário realizar um *reset* do microcontrolador após o nível de tensão subir além de 3,0 V para que a detecção funcione como deveria.

De modo geral, embora se tenha obtido um protótipo funcional e que cumpre o propósito do projeto, é necessário caracterizar versões vindouras do projeto mais profundamente, em testes a prazos mais longos, sob circunstâncias mais variadas e em situações reais de operação, além de implementar as melhorias e correções mencionadas, em um processo de melhoria contínua até a versão estável final.

REFERÊNCIAS

3GPP. **TS 27.007: AT Command Set for User Equipment.** 2022. Disponível em: https://www.3gpp.org/ftp/Specs/archive/27_series/27.007/. Acesso em: 28 jun 2023.

ANALOG DEVICES. **MAX40200 – Ultra-Tiny Micropower, 1A Ideal Diode with Ultra-Low Voltage Drop.** 2023. Disponível em: https://www.analog.com/media/en/technical-documentation/data-sheets/max40200.pdf. Acesso em: 23 jun. 2023.

ANVISA. Agência Nacional de Vigilância Sanitária. **Resolução RDC n° 430**, de 8 de outubro de 2020. Boas Práticas de Distribuição, Armazenagem e de Transporte de Medicamentos.

COMWINTOP. **CWT-L1T-DS.** Disponível em: https://store.comwintop.com/products/cwt-l1t-ds-wireless-gsm-3g-4g-sms-wifi-temperature-alarm-transmitter-data-logger. Acesso em: 8 fev. 2023

CONSONANCE. **1A LiFePO4 Battery Charger CN3058E.** 2022. Disponível em: http://www.consonance-elec.com/en/static/upload/file/20220425/1650867856106004.pdf. Acesso em: 23 jun. 2023.

EFENTO. **Wireless temperature and humidity logger.** Disponível em https://getefento.com/product/wireless-temperature-and-humidity-logger-lte-m-nb-iot/. Acesso em: 8 fev. 2022

ELITECH BRASIL. **RCW-360 WiFi Datalogger Temperatura e Umidade.** Disponível em https://www.elitechbrasil.com.br/rcw-360-wifi-datalogger-temperatura-e-umidade.html. Acesso em: 8 fev. 2023.

ELITECH. **RCW-360 Plus 4G Wireless Temperature and Humidity Data Logger with External Probe.** Disponível em http://www.elitechlog.com/elitech-rcw-360-plus-

4g-wireless-temperature-and-humidity-data-logger-with-external-probe/. Acesso em: 8 fev. 2023

ENIX ENERGIES. **26650 LIFEPO4 BATTERY.** 2014. Disponível em: https://docs.rs-online.com/4ad1/0900766b812fdd10.pdf. Acesso em: 23 jun. 2023.

ESPRESSIF SYSTEMS. **ESP8266EX Datasheet.** 2023. Disponível em: https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf. Acesso em: 23 jun. 2023.

ESPRESSIF SYSTEMS. **Q1 2023 Report.** 2023. Disponível em: https://www.espressif.com/sites/default/files/financial/Espressif%20Systems%202023%20Q1%20Results%20Announcement.pdf. Acesso em: 23 jun. 2023.

FIELDING, R. et al. **RFC 9110: HTTP Semantics**. 2022. Disponível em: https://www.rfc-editor.org/rfc/rfc9110. Acesso em: 23 jun. 2023.

FOSTER, Ian; ZHAO, Yong; RAICU, Ioan; LU, Shiyong. Cloud Computing and Grid Computing 360-Degree Compared. In: **2008 Grid Computing Environments Workshop**, 2008, Austin. **Proceedings…** Austin: IEEE, nov. 2008. 10 p. 1-10.

FUHS, Allen. **Hybrid Vehicles and the Future of Personal Transportation.** 1. ed. Boca Raton. Taylor & Francis, 2009.

GAMPERL, Jens. Three Ways Electronic Components Industry Leaders Can Prepare For What's Next. **Forbes**. 2023. Disponível em: https://www.forbes.com/sites/forbestechcouncil/2023/04/21/three-ways-electronic-components-industry-leaders-can-prepare-for-whats-next/. Acesso em: 23 jun. 2023.

KING PIGEON HI-TECH. **Cellular IoT RTU Quickly Selection Guide List.** Disponível em www.gprs-m2m.com/EnproductShow.asp?ID=432. Acesso em: 8 fev. 2023

KIZY TRACKING. **Supply Chain Tracker.** Disponível em: https://www.kizytracking.com/solutions/supply-chain-tracker/. Acesso em: 8 fev. 2022

LINDEN, David; REDDY, Thomas B. **Handbook of Batteries**. 3. ed. Estados Unidos da América. McGraw-Hill, 2002.

LITHIUMWERKS. **26650 Lithium Ion Power Cell.** 2021. Disponível em: https://store-cvt2jlgam8.mybigcommerce.com/content/LithiumWerks%20ANR26650M1-B%20Power%20Cell%20%28030921%29%20Data%20Sheet.pdf. Acesso em: 23 jun. 2023

MAXIM INTEGRATED. **DS18B20 Datasheet**. 2019. Disponível em: https://www.analog.com/media/en/technical-documentation/data-sheets/ds18b20.pdf. Acesso em: 23 jun. 2023.

MAXIM INTEGRATED. **Application Note 126**. 2002. Disponível em: https://pdfserv.maximintegrated.com/en/an/AN126.pdf. Acesso em: 23 jun. 2023.

MICROCHIP TECHNOLOGY INC. **MCP73831/2 Datasheet.** 2020. Disponível em: https://ww1.microchip.com/downloads/en/DeviceDoc/MCP73831-Family-Data-Sheet-DS20001984H.pdf. Acesso em: 23 jun. 2023.

NADEEM, Lubna et al. Integration of D2D, Network Slicing, and MEC in 5G Cellular Networks: Survey and Challenges. **IEEE Access,** Estados Unidos da América, v. 9, p. 37590 – 37612, mar. 2021.

NEXTCON. **DSBWIFI.** Disponível em: https://www.nextcon.com.br/MLB-1593622342-termmetro-para-geladeira-de-vacinas-wifi-Internet-_JM>. Acesso em: 8 fev. 2023.

NEXTCON. **DTHWIFI.** Disponível em: https://www.nextcon.com.br/MLB-1418388054-termometro-higrometro-digital-wifi-web-_JM. Acesso em: 8 fev 2023.

NOVUS. **Data logger Wi-Fi LogBox Wi-Fi.** Disponível em: https://www.novus.com.br/pt/logbox-wifi. Acesso em: 8 fev. 2023SMETRO. **Smetro Mobile.** Disponível em: https://site.smetro.io/produtos/sm-mb. Acesso em: 8 fev. 2023

OVERBAY, Caleb. FRAM or flash: how to select the right MCU for your application. **E2E design support**. 2017. Disponível em: https://e2e.ti.com/blogs_/b/process/posts/fram-or-flash-how-to-select-the-right-mcu-for-your-application. Acesso em: 23 jun. 2023.

PREGER, Yuliya et al. Degradation of Commercial Lithium-Ion Cells as a Function of Chemistry and Cycling Conditions. **Journal of The Electrochemical Society.** Estados Unidos da América, v. 167, n. 12, set. 2020.

RAYES, Ammar; SALAM, Samer. **Internet of Things From Hype to Reality**. 2. ed. Suíça: Springer Nature Switzerland AG, 2019.

SHIN, Donghee. A socio-technical framework for Internet of Things design: A human-centered design for the Internet of Things. **Telematics and Informatics**, Estados Unidos da América, v. 31, n. 4, p. 519-531, mar. 2014.

SIMCOM. **SIM7070G Hardware Design Guide**. 2020. Disponível em: https://microchip.ua/simcom/LTE-LPWA/SIM7070_SIM7080_SIM7090/SIM7070G_Hardware%20Design_V1.03.pdf. Acesso em: 23 jun. 2023.

SMETRO. **Smetro Mobile.** Disponível em: https://site.smetro.io/produtos/sm-mb. Acesso em: 8 fev. 2023

SMETRO. **Smetro WiFi + Ethernet.** Disponível em: https://site.smetro.io/produtos/sm-we. Acesso em: 8 fev. 2023

TEXAS INSTRUMENTS. **FRAM FAQs.** 2014. Disponível em: https://www.ti.com/lit/wp/slat151/slat151.pdf. Acesso em: 23 jun. 2023.

TEXAS INSTRUMENTS. **MSP430FR596x, MSP430FR594x Mixed-Signal Microcontrollers.** 2018. Disponível em: https://www.ti.com/lit/ds/symlink/msp430fr5969.pdf. Acesso em: 23 jun. 2023.

TEXAS INSTRUMENTS. **MSP430 Optimizing C/C++ Compiler v21.6.0.LTS User's Guide.** 2021. Disponível em: https://www.ti.com/lit/ug/slau132y/slau132y.pdf. Acesso em: 23 jun. 2023.

UBLOX. **LTE Cat 1**. 2023. Disponível em https://www.u-blox.com/en/technologies/lte-cat-1. Acesso em: 23 jun. 2023.

WI-FI ALLIANCE. **Wi-Fi Alliance 2022 Wi-Fi Trends.** 2022. Disponível em: https://www.wi-fi.org/news-events/newsroom/wi-fi-alliance-2022-wi-fi-trends. Acesso em: 23 jun 2023.

WHITMORE, Andrew; AGARWAL, Anurag; XU, Li Da. The Internet of Things – A survey of topics and trends. **Inf. Syst. Front.**, New York, v. 17 n. 2, p 261-274, abr. 2015.