

DESENVOLVIMENTO DE UM SISTEMA EMBARCADO PARA MONITORAMENTO DE SINAIS VITAIS COM RASPBERRY PI PICO W E INTEGRAÇÃO EM NUVEM PARA ANÁLISE DE DADOS

Felipe de Oliveira Gomes

Projeto final para o Programa de Capacitação Profissional em Sistemas Embarcados - Embarca Tech.

Grupo: SYSCORE.

**Fortaleza
2025**

Resumo

Este projeto visa o desenvolvimento de um sistema embarcado portátil para monitoramento de sinais vitais, denominado "Pico Health Hub", utilizando a Raspberry Pi Pico W. O objetivo principal é criar um dispositivo acessível para coleta de dados fisiológicos como pressão arterial, temperatura corporal, oximetria e sons cardíacos/respiratórios, com o intuito de promover o acompanhamento da saúde pessoal. O sistema emprega sensores periféricos, display OLED para interface com o usuário e conexão com plataforma na nuvem para armazenamento e análise dos dados. A metodologia inclui a integração de hardware, desenvolvimento de software para aquisição e processamento de sinais, criação de interface intuitiva para usuários e implementação da comunicação com a nuvem. Pretende-se obter um protótipo funcional que permita a usuários realizar testes, salvar e acessar seus dados, fomentando o autocuidado e o monitoramento contínuo da saúde de forma prática e de baixo custo. O projeto busca, assim, democratizar o acesso à tecnologia de monitoramento de saúde e contribuir para a prevenção e acompanhamento de condições médicas.

Palavras-Chave: Sistema embarcado; sinais vitais; Raspberry Pi Pico W; monitoramento; plataforma na nuvem; sensores; interface; saúde pessoal; autocuidado; baixo custo.

1. Introdução

A busca por uma vida saudável e a prevenção de doenças têm se tornado uma prioridade crescente na sociedade contemporânea. Nesse cenário, o monitoramento regular da saúde emerge como uma prática essencial para a detecção precoce de alterações e o acompanhamento de condições crônicas, contribuindo para a melhoria da qualidade de vida e a redução dos riscos de complicações. No entanto, a dificuldade de acesso a equipamentos de medição unificados, o ritmo acelerado da vida moderna e a crescente demanda por consultas médicas online têm evidenciado a necessidade de soluções inovadoras e acessíveis para o autocuidado da saúde.

O presente trabalho propõe o desenvolvimento do "Pico Health Hub", um sistema embarcado portátil e de baixo custo, baseado na plataforma Raspberry Pi Pico W, que visa integrar múltiplas funcionalidades de monitoramento de sinais vitais em um único dispositivo. O projeto surge como uma resposta a essa crescente necessidade, oferecendo uma ferramenta prática e intuitiva para que os indivíduos possam acompanhar sua saúde de forma regular, no conforto de suas casas e com a possibilidade de compartilhar informações com profissionais de saúde à distância.

O "Pico Health Hub" se destaca pela sua capacidade de coletar e processar dados fisiológicos de forma integrada, abrangendo medições de pressão arterial, temperatura corporal, saturação de oxigênio no sangue e auscultação de sons cardíacos e respiratórios. Para isso, o dispositivo emprega uma variedade de sensores periféricos de baixo custo, que são controlados pelo microcontrolador Raspberry Pi Pico W, conhecido por sua versatilidade e eficiência energética. A interface do usuário é facilitada por um display OLED, que fornece orientações claras e exibe os resultados das medições de forma concisa.

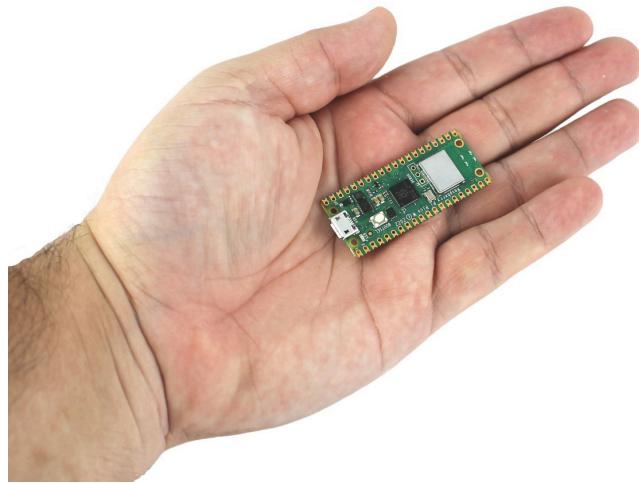


Figura 1. Microcontrolador raspberry pi pico w na palma da mão. Fonte: https://www.usinainfo.com.br/1034422-thickbox_default/raspberry-pi-pico-w-com-wifi-e-bluetooth.jpg

Uma das funcionalidades centrais do "Pico Health Hub" é a capacidade de armazenar os dados coletados em uma plataforma na nuvem. Essa integração permite que os usuários possam acessar seus históricos de medições de qualquer lugar e compartilhar as informações com seus médicos online, facilitando o acompanhamento remoto e o diagnóstico precoce de possíveis alterações. Além disso, o dispositivo oferece um sistema de autenticação de usuários, que garante a privacidade e a segurança dos dados armazenados.

O projeto também se preocupa em promover a acessibilidade da tecnologia para diferentes perfis de usuários. A interface intuitiva e as orientações claras exibidas no display OLED tornam o "Pico Health Hub" fácil de usar, mesmo para pessoas com pouca familiaridade com dispositivos tecnológicos. Além disso, a utilização de componentes de baixo custo e a natureza open-source do projeto contribuem para a democratização do acesso à tecnologia de monitoramento da saúde, tornando-o uma opção economicamente viável para diversas camadas da população.



Figura 2. Display OLED de 128 colunas por 64 linhas com informações de saúde. Fonte: Autor

Além das funcionalidades de medição e armazenamento de dados, o "Pico Health Hub" tem potencial para evoluir e integrar outras funcionalidades no futuro, como análise de sinais cardíacos e respiratórios por meio de algoritmos de processamento de sinais, notificações personalizadas para usuários com necessidades específicas e integração com aplicativos de saúde e bem-estar. O objetivo final é criar uma ferramenta completa e robusta para o autocuidado da saúde, que possa contribuir para a prevenção de doenças e a melhoria da qualidade de vida da população. Com o "Pico Health Hub", busca-se não apenas monitorar sinais vitais, mas também capacitar os indivíduos a assumirem um papel mais ativo no cuidado da sua própria saúde, empoderando-os com conhecimento e ferramentas para um futuro mais saudável.

2. Objetivos

2.1 Objetivo Geral

O objetivo geral deste projeto é desenvolver um sistema embarcado portátil e de baixo custo, denominado "Pico Health Hub", para monitoramento integrado de sinais vitais, visando fornecer uma ferramenta acessível e prática para o autocuidado e o acompanhamento da saúde pessoal, com a possibilidade de integração de dados em plataformas na nuvem para análise e compartilhamento com profissionais de saúde.

2.2 Objetivos Específicos

- Implementar um sistema de aquisição de dados fisiológicos utilizando a Raspberry Pi Pico W, integrando sensores para medição de pressão arterial, temperatura corporal, saturação de oxigênio, e captação de sons cardíacos e respiratórios;
- Desenvolver um protótipo funcional do "Pico Health Hub" que inclua uma interface de usuário intuitiva, utilizando um display OLED para orientar o usuário durante os testes e exibir os resultados;
- Estabelecer um sistema de armazenamento de dados em uma plataforma na nuvem, garantindo a segurança, privacidade e acessibilidade das informações coletadas pelos usuários;
- Criar um sistema de autenticação e identificação de usuários permitindo que cada indivíduo armazene e acesse seus dados de forma personalizada e segura;
- Validar a precisão e confiabilidade dos dados coletados pelo "Pico Health Hub", comparando os resultados com instrumentos de medição padrão e realizando testes de desempenho do sistema;
- Desenvolver um sistema de comunicação e transmissão de dados entre o "Pico Health Hub" e a plataforma na nuvem, garantindo a integridade e a segurança das informações;
- Projetar um sistema de fácil manuseio, permitindo que usuários com diferentes níveis de familiaridade com tecnologia possam utilizar o dispositivo de forma autônoma e intuitiva;
- Explorar a viabilidade econômica do projeto, utilizando componentes de baixo custo para tornar o "Pico Health Hub" uma solução acessível para a população em geral;
- Avaliar a usabilidade e a satisfação dos usuários com o "Pico Health Hub" através de testes práticos e questionários, buscando aprimorar o dispositivo e atender às necessidades dos usuários;

3. Justificativa

A saúde, bem mais valioso de cada indivíduo, demanda atenção e cuidados constantes. No cenário contemporâneo, onde o ritmo acelerado da vida e as exigências do cotidiano frequentemente nos afastam da rotina de cuidados, a importância do monitoramento contínuo da própria saúde emerge como um fator crucial para a prevenção de doenças e a garantia de uma vida plena e saudável. O projeto "Pico Health Hub" surge como uma resposta a essa necessidade, propondo uma solução tecnológica inovadora e acessível para o acompanhamento regular de sinais vitais e indicadores de saúde.

A relevância desse projeto se torna ainda mais evidente quando observamos os alarmantes números de mortes súbitas e inesperadas decorrentes da falta de monitoramento adequado da saúde. Muitas condições silenciosas, como a hipertensão arterial, arritmias cardíacas e alterações respiratórias, podem se desenvolver sem que o indivíduo perceba seus sintomas, levando a complicações graves e, em alguns casos, fatais. A ausência de um acompanhamento regular, que permita identificar precocemente essas alterações, contribui para a progressão dessas doenças e aumenta o risco de eventos adversos. O "Pico Health Hub", ao facilitar a medição periódica de sinais vitais, possibilita a identificação precoce de alterações e o encaminhamento para acompanhamento médico, podendo salvar vidas e melhorar a qualidade de vida da população.

Um dos desafios enfrentados atualmente é a dificuldade de acesso a equipamentos de medição unificados, que permitam realizar um check-up completo no conforto de casa. Frequentemente, é necessário recorrer a diversos dispositivos separados para aferir a pressão arterial, temperatura corporal, saturação de oxigênio e auscultar os sons cardíacos e respiratórios, gerando custos adicionais e dificultando a padronização dos procedimentos. O "Pico Health Hub" se propõe a solucionar esse problema, oferecendo um dispositivo integrado e portátil, que concentra em uma única plataforma todas as ferramentas necessárias para um monitoramento completo da saúde.

Além disso, a crescente adesão às consultas médicas online, intensificada pela pandemia de COVID-19, revela a necessidade de ferramentas que facilitem o acompanhamento remoto dos pacientes. O "Pico Health Hub" se destaca nesse cenário, permitindo que os usuários realizem as medições de sinais vitais em casa e compartilhem os dados com seus médicos online, agilizando o diagnóstico e o acompanhamento terapêutico. A disponibilidade dessas informações permite que os profissionais de saúde tenham uma visão mais clara do estado clínico de seus pacientes e tomem decisões mais assertivas, sem a necessidade de deslocamentos e consultas presenciais desnecessárias.

Ademais, este projeto possui uma relevância social inegável ao democratizar o acesso à tecnologia de monitoramento da saúde. A utilização de um microcontrolador acessível, como a Raspberry Pi Pico W, e a integração de componentes de baixo custo tornam o "Pico Health Hub" uma solução economicamente viável e acessível para diversas camadas da população. Além disso, o design do dispositivo e a interface do usuário são pensados para facilitar o uso, mesmo por pessoas com pouca familiaridade com tecnologia.

Portanto, a criação do "Pico Health Hub" se justifica pela necessidade de facilitar o monitoramento contínuo da saúde, contribuindo para a detecção precoce de doenças, a melhoria da qualidade de vida, o acesso facilitado às consultas médicas online, o empoderamento dos usuários sobre seus próprios cuidados e a democratização do acesso à tecnologia de saúde. O projeto, portanto, não apenas responde a uma demanda crescente, mas também representa uma contribuição significativa para a promoção do bem-estar e a prevenção de doenças.

4. Revisão Bibliográfica

Os exames de saúde de rotina desempenham um papel vital na detecção de potenciais problemas de saúde nas fases iniciais. Esta detecção precoce não só aumenta as probabilidades de sucesso do tratamento, mas também reduz significativamente os encargos financeiros associados a medidas reativas de cuidados de saúde. Os resultados positivos da monitorização regular da saúde não são meramente teóricos; são apoiados por histórias de sucesso e estatísticas que destacam o impacto transformador dos cuidados preventivos. De facto, estudos mostram que os indivíduos que são submetidos a monitorização regular da saúde têm 70% mais probabilidades de obter resultados positivos, destacando o impacto transformador dos cuidados preventivos conforme estudo de KROGSBØLL et al., (2019).

Em estudos realizados num projeto de inclusão digital para idosos realizado na Faculdade de Medicina de Ribeirão Preto (FMRP) da Universidade de São Paulo (USP), foi constatado que ao longo dos anos, a população brasileira tem incorporado o uso de equipamentos de monitoramento da saúde de uso doméstico e para a prática de atividades físicas, conforme ocorre há vários anos com a população dos países desenvolvidos.(SILVA, 2011)(SANTANA et al., 2012).

De acordo com um programa da prefeitura de São Paulo (Prefeitura de São Paulo, 2023), as pessoas têm sido cada vez mais responsabilizadas pelo autocuidado da saúde e, atualmente, o Ministério da Saúde disponibiliza monitores de glicemia, permitindo que diabéticos auto monitorarem seu índice glicêmico (tarefa que no início da década de 80 era somente feita em hospitais e/ou em clínicas especializadas). Considera-se que equipamentos tecnológicos como aferidor de pressão arterial, frequencímetro e medidor de glicemia podem auxiliar no autocuidado à saúde e podem ser aliados no controle de condições crônicas e também na qualidade de vida do usuário.



Figura 3. Aparelho de medição do índice glicêmico. Fonte: <https://down-br.img.susercontent.com/file/sg-11134201-7rbld-lmgkk7d622bvf4>

No Brasil, a Agência Nacional de Vigilância em Saúde (ANVISA) e seus Núcleos de Avaliação de Tecnologias em Saúde (NATS) têm importante ação neste

sentido, assim como os profissionais de saúde que irão orientar seu uso doméstico.(Agência Nacional de Vigilância Sanitária, 2024) Como a maioria dos aparelhos oferece respostas "acessíveis e diretas", o maior risco é o usuário tentar o autodiagnóstico e o autotratamento. "Eles podem ser excelentes aliados do monitoramento de alguns sinais, mas não devem ser usados sozinhos e sem supervisão".(VILELA R., 2018)

Destaca-se a importância de desenvolver a competência para o manejo desses equipamentos e a adequada orientação a quem vai utilizá-lo, buscando esclarecer que estes não substituem a necessidade dos exames clínicos e de equipamentos mais confiáveis para a fase de diagnóstico, nem das consultas regulares para a fase de acompanhamento.

O desenvolvimento de sistemas mais avançados para o cuidado à saúde que envolve o cuidado à distância perpassa, em primeira instância, a competência do sujeito para o manejo dessas interfaces. Assim, o uso de equipamentos de cuidados à saúde, dentre inúmeras possibilidades, implicam conhecer o âmbito das dificuldades e variáveis que interferem no uso de equipamentos.

No estudo de İlhan et al., (2016) um dispositivo de medição de pressão arterial sem fio usado com um smartphone foi desenvolvido. Com a interface desenvolvida, o dispositivo móvel foi utilizado tanto como indicador quanto como dispositivo de controle; os valores de medição da pressão arterial foram registrados no banco de dados do próprio aparelho. Assim, é possível efetuar consultas relacionadas com estes dados e os resultados dessas consultas podem ser enviados à pessoa relevante através de email ou SMS. O manguito que se comunica com o aparelho smartphone via Bluetooth foi projetado para medir a pressão arterial através do braço. Um filtro digital foi usado no manguito em vez de um circuito analógico de processamento e filtragem de sinais.

Num ambiente clínico, a saturação de oxigénio no sangue é um dos indicadores de sinais vitais mais importantes. Um oxímetro de pulso é um dispositivo que mede a saturação de oxigênio no sangue e a pulsação de pacientes com diversos distúrbios. No entanto, devido a preocupações éticas, os oxímetros de pulso disponíveis comercialmente são limitados em termos de calibração em pacientes gravemente enfermos, resultando numa taxa de erro significativa para medição na faixa crítica de saturação de oxigênio. A acessibilidade do dispositivo nos ambientes de saúde dos países em desenvolvimento também é limitada devido à portabilidade, às implicações de custos e à falta de necessidade reconhecida. No estudo de Nemomssa et al., (2022) foi desenvolvido um oxímetro de pulso confiável, de baixo custo e portátil, usando o microcontrolador ATMEGA328P, com maior precisão na faixa crítica de saturação de oxigênio.



Figura 4. Protótipo final do dispositivo de oxímetro de pulso proposto. Fonte: <https://pmc.ncbi.nlm.nih.gov/articles/PMC9084508/>

O trabalho de I. K. A. A. et al., (2021) mostra que, em geral, a saúde do corpo pode ser observada a partir do estado de temperatura, que é um sinal vital como indicação da atividade dos órgãos do corpo. Através da temperatura, será capaz de detectar se os órgãos do corpo estão funcionando bem ou não. Nesta pesquisa foi desenvolvido um aparelho para medição de temperatura corporal ou um termômetro sem contato direto. O módulo utilizado é um sensor e atuador onde estes dois módulos são controlados por um microcontrolador programado. O sistema que está sendo desenvolvido também utiliza o conceito de Internet das Coisas (IoT). Onde cada uma dessas ferramentas pode mais enviar dados para um banco de dados sem intervenção humana, é muito prático reduzir o contato direto entre humanos. A rede de comunicação utilizada nesta pesquisa é WiFi ou a chamada IEEE 802.11.

Os estetoscópios acústicos apresentam baixos níveis de som. O estetoscópio digital supera esse problema amplificando eletronicamente os sons do corpo. Como os sinais sonoros são transmitidos eletronicamente, ele pode ser sem fio e fornecer redução de ruído. O estetoscópio acústico pode ser transformado em um estetoscópio digital inserindo um microfone de capacidade elétrica em sua cabeça. Os sons cardíacos recebidos do microfone são processados, amostrados e os sinais sonoros são convertidos de analógico para digital e enviados sem fio usando técnicas da Internet das Coisas (IOT), para que vários médicos possam fazer auscultação e monitorar as condições do paciente, no estudo de Reddy et al.,(2018) um estetoscópio digital foi implementado onde os sinais podiam ser transmitidos através de IOT.

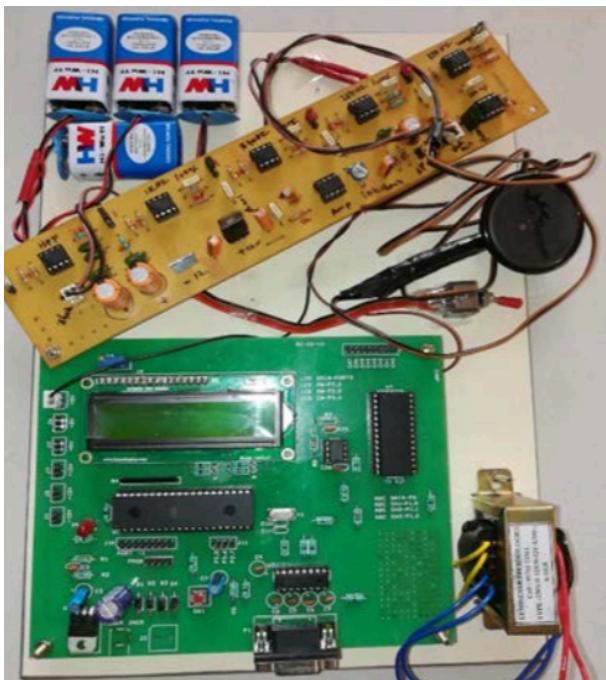


Figura 5. Protótipo final do dispositivo estetoscópio sem fio usando IOT. Fonte: <https://ijaas.iaescore.com/index.php/IJAAS/article/view/13521/>

Os batimentos cardíacos foram captados por meio de um microfone condensador e convertidos os sons de áudio em sinais elétricos. Os sinais elétricos convertidos são amplificados no circuito pré-amplificador. Os sinais amplificados são então filtrados com vários tipos de filtros passa-baixa e passa-alta para reduzir o ruído. Sinais de ruído reduzido são novamente amplificados com o circuito amplificador de potência. A entrada para este amplificador de potência é a saída do filtro passa-alto. Os sinais amplificados com amplificador de potência podem ser convertidos em digitais usando um conversor analógico para digital e o sinal digital é processado. O sinal processado é entregue ao microcontrolador e depois enviado para a nuvem ou internet.(REDDY et al., 2018).

Apesar dos projetos encontrados como referência, não foi encontrado um dispositivo unificado capaz de unir as tecnologias, melhorá-las através de algoritmos mais refinados e gerar dados cruzados conectando informações das diversas medições em prol de avaliação médica ou monitoramento de rotina pessoal, gerando ainda um banco de medições para tratamentos e acompanhamento mais assertivo.

5. Metodologia

As etapas de desenvolvimento deste projeto, "Pico Health Hub", será conduzida de forma sistemática e abrangente, visando a implementação de um sistema embarcado robusto e eficaz para o monitoramento de sinais vitais. Inicialmente, será realizada uma pesquisa minuciosa do estado da arte, com o objetivo de identificar as tecnologias mais recentes e as necessidades específicas da área de saúde. Em seguida, procederemos à seleção criteriosa dos componentes de hardware, especificando sensores, microcontrolador e demais periféricos, com base em critérios como custo, precisão e eficiência energética. A etapa seguinte será dedicada à definição das funcionalidades do software (firmware), incluindo o desenvolvimento de algoritmos de processamento de sinais e a implementação de protocolos de comunicação com a plataforma na nuvem.

Após a fase de especificação, o desenvolvimento prático será iniciado, envolvendo a programação do firmware na IDE e a depuração do sistema. Uma etapa crucial do projeto será a validação rigorosa, onde o desempenho do "Pico Health Hub" será avaliado por meio de testes de precisão, desempenho e usabilidade, com a comparação de resultados obtidos com equipamentos de referência. Finalmente, os resultados obtidos serão analisados e discutidos para identificar limitações e propor melhorias, culminando na elaboração de um relatório técnico detalhado e a apresentação das contribuições do projeto para o campo do monitoramento de saúde. Cada etapa será detalhada nas subseções seguintes, explicitando o processo de desenvolvimento e as escolhas metodológicas tomadas.

5.1 Pesquisa

No desenvolvimento do sistema embarcado espera-se a implementação de um sistema de controle que permita a integração de diversos sensores e interfaces de forma a atingir os objetivos especificados anteriormente. Visando organizar melhor tais componentes serão divididos em três grupos:

- Interfaces de comunicação com o usuário: Display OLED, Botões, Joystick, LED, Buzzers e Teclado bluetooth (opcional).
- Interfaces IOT: Webserver ou aplicação em tempo real, Banco de dados na nuvem para registro de medições e usuários.
- Sensores de medição: Sensor de pressão sanguínea, oxímetro, termômetro, módulo microfone estetoscópio.

A pesquisa será concentrada em encontrar projetos que possam servir como referência e embasamento para o desenvolvimento de cada um dos componentes de cada grupo.

5.1.1 Interfaces de comunicação com o usuário

Para desenvolvimento da interface OLED a biblioteca do github RPIPicoSSD1306Exp (DURRANT,2024) foi utilizada como referência onde é possível criar textos, animações e diferentes tipos de interação com o usuário.

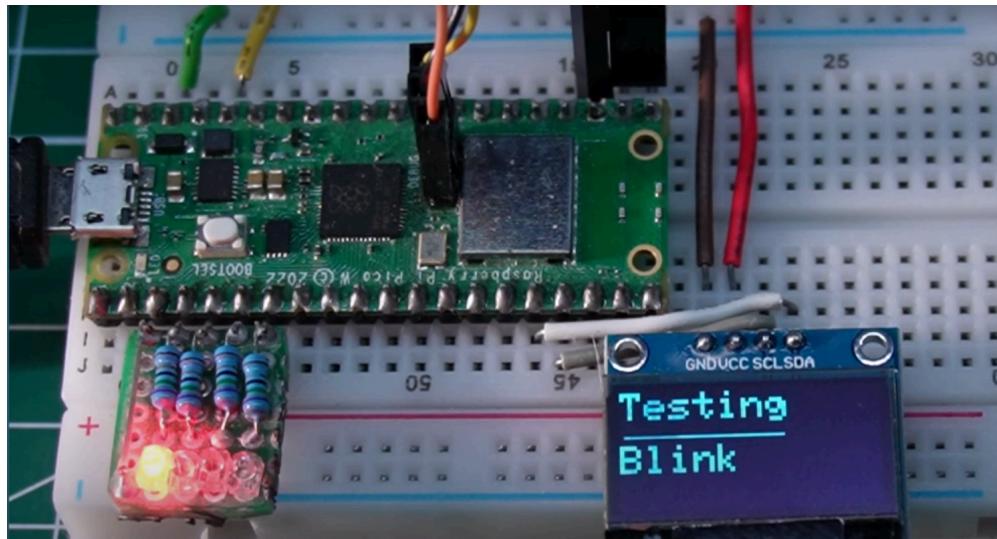


Figura 6. Demonstração de funcionamento da biblioteca de controle da interface I2C OLED.
Fonte: <https://www.youtube.com/watch?v=BvyYMIpI2R8>

Para desenvolvimento dos módulos que utilizam botões, LED e buzzer a principal referência de pesquisa foi a biblioteca do github de exemplos do SDK (Raspberry PI Pico SDK., 2024), onde é possível encontrar diversos exemplos de controle dos GPIOs do microcontrolador atendendo aos objetivos do projeto.

No desenvolvimento do módulo de joystick também foi utilizado um exemplo disponível no SDK da raspberry pi pico w(Raspberry PI Pico SDK., 2024) onde é possível interpretar os movimentos do joystick nos eixos x e y traduzindo os movimentos em ações que podem ser tomadas a partir do dado fornecido.

Na pesquisa sobre conectividade de um teclado bluetooth foram considerados os exemplos de bluetooth do SDK (Raspberry PI Pico SDK., 2024) e o artigo de Erich Styger., 2023 onde é possível entender em detalhes a implementação do módulo de bluetooth na raspberry pi pico w, sendo assim acessível receber os dados necessários para atender os objetivos do projeto também via bluetooth.

5.1.2 Interfaces IOT

Nos estudos de desenvolvimento do webserver os trabalhos de Geoff Simmons., (2022) e Xuyun Zeng., (2023) foram tomados como base pela vantagem de utilizarem o sdk da raspberry pi pico w e aproveitamento de código devido a similaridade com os objetivos do Pico Health Hub no tocante a interface iot web server.

Para criação de uma base de dados de armazenamento de usuários e seus respectivos dados, os trabalhos de Mahmood M. Shilleh., (2023) e Xuyun Zeng., (2023) serviram de base para criação de um ambiente seguro e de fácil desenvolvimento onde os dados pudessem ser armazenados.

O artigo "Build a cloud Raspberry Pi Pico W temperature/humidity logger" (ZENG., 2023) oferece um guia detalhado para criar um registrador de temperatura e umidade utilizando o Raspberry Pi Pico W, o sensor DHT22 e um display LED de 4 dígitos com o driver TM1637. O autor compartilha sua experiência pessoal ao enfrentar altos níveis de umidade e a necessidade de monitorar as condições de

umidade para proteger itens valiosos. O tutorial abrange desde os componentes necessários e esquemas de conexão até o código-fonte em Python para leitura dos dados e exibição no display. Além disso, o artigo explora como registrar esses dados na nuvem usando a plataforma Anvil, permitindo a análise de tendências ao longo do tempo. Este guia é uma excelente referência para entusiastas de eletrônica interessados em projetos práticos de monitoramento ambiental com conectividade à internet.

O artigo "Effortless Data Storage: MongoDB Database and Raspberry Pi Pico W Walkthrough" (SIMMONS., 2022) oferece um guia detalhado sobre como integrar o Raspberry Pi Pico W com o banco de dados MongoDB para armazenamento de dados. O tutorial abrange desde a configuração inicial do Raspberry Pi Pico W até a instalação e configuração do MongoDB, fornecendo instruções passo a passo para facilitar a compreensão. Este conteúdo é especialmente útil para explorar soluções de armazenamento de dados eficientes utilizando hardware de baixo custo, como no caso em questão e bancos de dados NoSQL. A abordagem prática do artigo permite que a implementação seja realizada facilmente com as instruções fornecidas em seus próprios projetos.

5.1.3 Sensores de medição

Este tópico aborda a pesquisa dos sensores de medição, componentes que atuam na interface entre o mundo físico e o sistema embarcado. A complexidade do projeto "Pico Health Hub" exige a integração de diversos sensores, cada um com sua especificidade, para monitorar uma variedade de sinais vitais. Nesta seção, apresentamos as tecnologias selecionadas para medir a temperatura corporal, a saturação de oxigênio, a pressão arterial e os sons cardíacos e respiratórios, detalhando suas funcionalidades e as bibliotecas que serão utilizadas de base para sua implementação.

5.1.3.1 Sensor de temperatura

O repositório pico-onewire (BOARDMAN., 2023) oferece uma biblioteca para o Raspberry Pi Pico que facilita a comunicação com dispositivos que utilizam o protocolo OneWire, como os sensores de temperatura DS18S20, DS18B20, DS1822, MAX31820 e MAX31826.

A biblioteca é baseada na liberação DS1820 de Erik Olieman para o mbed, com modificações nos tempos para compatibilidade com o Raspberry Pi Pico. Além disso, foram adicionados códigos de teste fora do dispositivo com mocks e uma configuração própria do CMake. A API foi modificada para um fluxo de trabalho em duas etapas: listagem de dispositivos com endereços para armazenamento e retorno de temperaturas para cada dispositivo por endereço.

A pico-onewire (BOARDMAN., 2023) é uma biblioteca útil para implementar comunicação OneWire no Raspberry Pi Pico, especialmente para leitura de sensores de temperatura. Sua base sólida e modificações específicas para o Pico tornam uma opção viável para o caso do sensor de temperatura no Pico Health Hub.

5.1.3.2 Oxímetro

A biblioteca max30102.h, disponível no repositório MAX30102 de Linnéa Edström., (2016) define uma biblioteca para o sensor de oximetria de pulso MAX30102, destinada ao ambiente Mbed. Este arquivo especifica os endereços dos registradores do sensor, incluindo registradores de status, FIFO, configuração e temperatura, além de definir o endereço I2C padrão do dispositivo.

A biblioteca é uma modificação da original para o MAX30100, adaptada para o MAX30102. Ela permite a leitura de dados brutos dos LEDs infravermelho e vermelho através da interface I2C, bem como a obtenção da temperatura interna do sensor em graus Celsius.

A biblioteca é um excelente ponto de partida. Porém, como o ambiente Mbed é diferente do Pico SDK, as chamadas de hardware (como I2C e delays) precisam ser substituídas pelas equivalentes no SDK do Raspberry Pi Pico. A lógica para leitura e configuração dos registradores pode ser aproveitada diretamente, reduzindo o esforço de implementação. O arquivo max30102.h fornece uma interface clara e direta para interagir com o sensor MAX30102, facilitando sua integração no Pico Health Hub, atendendo assim a necessidade de um oxímetro presente nos objetivos do projeto.

5.1.3.3 Aferidor de pressão arterial

O projeto "Blood Pressure Sensor Interfacing with Raspberry Pi" (VINARY.,2021) apresenta um sistema de monitoramento de pressão arterial que utiliza um sensor de pressão arterial de pulso integrado a um Raspberry Pi. O sistema é capaz de medir a pressão arterial e a frequência cardíaca do usuário, armazenar esses dados e enviá-los por e-mail em um arquivo de texto detalhado, contendo os valores de pressão sistólica, diastólica e frequência cardíaca.

Este projeto oferece uma abordagem prática para o monitoramento domiciliar da pressão arterial, permitindo que os usuários acompanhem e gerenciem seus valores diários de pressão arterial e frequência cardíaca. A integração com o Raspberry Pi possibilita a automação do processo de medição e a comunicação dos resultados, facilitando o acompanhamento remoto por profissionais de saúde ou familiares.

O projeto demonstra uma aplicação eficaz da tecnologia de código aberto no campo da saúde, oferecendo uma solução acessível para o monitoramento da pressão arterial e da frequência cardíaca, com recursos adicionais de armazenamento e comunicação de dados.

É possível adaptar o projeto para o Raspberry Pi Pico W no ambiente de desenvolvimento em C com o SDK. As principais etapas incluem a configuração do UART, a interpretação dos dados do sensor e a implementação de funcionalidades de rede para envio dos dados.

5.1.3.4 Estetoscópio

O projeto intitulado "Estetoscópio Eletrônico Usando Raspberry Pi" (BHARATH., 2021) foi desenvolvido por estudantes do Departamento de Engenharia Eletrônica e de Comunicação do G.M. Institute of Technology, Davanagere, sob a orientação do Sr. Chetan B V. Este projeto visa aprimorar o estetoscópio acústico tradicional, incorporando recursos digitais para melhorar a flexibilidade do dispositivo e reduzir a transmissão de patógenos.

O sistema utiliza um Raspberry Pi como unidade central de processamento. Um microfone condensador é acoplado à cabeça do estetoscópio para capturar os sons cardíacos, que são então processados e armazenados no Raspberry Pi. Além disso, um sensor de temperatura monitora a temperatura corporal do paciente. Os dados coletados são enviados por e-mail para profissionais médicos para posterior análise. O dispositivo também possui um sistema de esterilização por UV para desinfetar a superfície do estetoscópio após o uso.

O projeto oferece uma alternativa confiável e acessível ao estetoscópio acústico tradicional, incorporando funcionalidades digitais que aprimoram o diagnóstico médico e promovem a segurança tanto de pacientes quanto de profissionais de saúde.

Outra referência é o "Estetoscópio Digital Inteligente" (COLLINS., 2022) desenvolvido por uma equipe de design sênior da Universidade Estadual de Iowa, com o objetivo de aprimorar o monitoramento de pacientes com problemas cardíacos e pulmonares, minimizando o contato físico e aumentando a eficiência.

Os dois projetos foram tomados como referências no desenvolvimento do estetoscópio do Pico Health Hub, com uso de microfones acoplados ao auscultador de um estetoscópio tradicional e algoritmos para tratativas de áudio e detecção de doenças.

5.2 Especificação do hardware

Este capítulo detalha a especificação do hardware do "Pico Health Hub", explorando todos os componentes que viabilizam a coleta e o processamento de sinais vitais. Inicialmente, um diagrama de blocos apresentará a interligação dos módulos, seguido da descrição da função de cada bloco e sua respectiva configuração. Além disso, especificaremos os comandos e registros utilizados, a descrição da pinagem e o circuito completo do hardware, fornecendo uma visão detalhada da construção física do dispositivo.

5.2.1 Diagrama em bloco do hardware

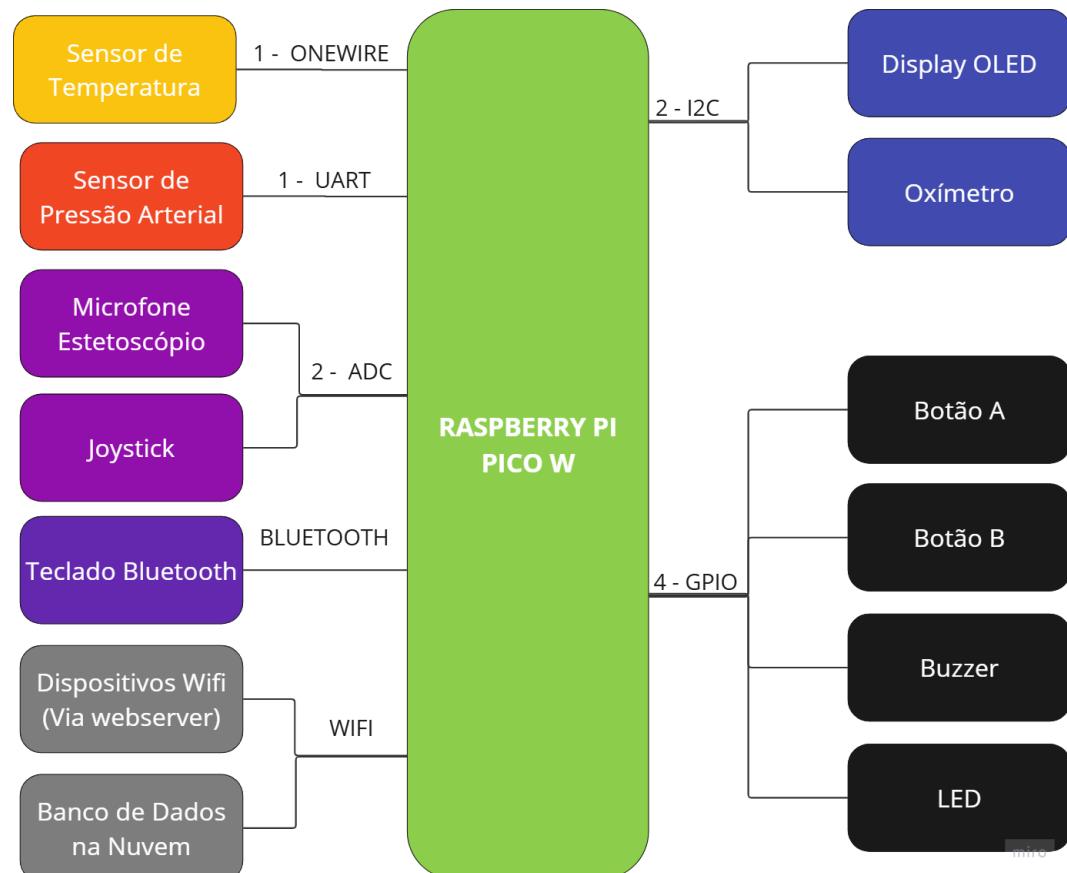


Figura 7. Diagrama de blocos do sistema de hardware do “Pico Health Hub”. Fonte: Autor

5.2.2 Função de cada bloco

- Raspberry Pi Pico W (Microcontrolador):

Unidade Central de Processamento: Atua como a unidade central de processamento de todo o sistema, responsável por executar o código do firmware, coordenar a funcionalidade de todos os outros blocos e gerenciar o fluxo de dados dentro do sistema.

Aquisição de Dados: Coleta dados brutos de vários sensores (temperatura, pressão arterial, oxímetro, microfone, joystick) através de suas respectivas interfaces (ONEWIRE, UART, I2C, ADC).

Processamento de Sinais: Processa os dados brutos dos sensores usando algoritmos de processamento de sinais digitais, filtrando ruídos e convertendo sinais dos sensores em medições significativas.

Lógica de Controle: Implementa a lógica do sistema e controla o display, botões, buzzer e LED, bem como outros aspectos da interface do usuário.

Comunicação: Facilita a comunicação com dispositivos externos via Wi-Fi e Bluetooth para interação com webserver, acesso a banco de dados na nuvem e funcionalidade do teclado Bluetooth.

Gerenciamento do Sistema: Gerencia o consumo de energia, o tempo do sistema e a estabilidade e confiabilidade geral do dispositivo.

- Sensor de Temperatura:

Medição da Temperatura Corporal: Mede a temperatura corporal do usuário usando um sensor baseado no protocolo OneWire e transmite esses dados para processamento pelo microcontrolador.

Monitoramento da Saúde: Fornece dados de temperatura ao sistema para fornecer informações sobre o estado de saúde do usuário.

- Sensor de Pressão Arterial:

Medição da Pressão Arterial: Mede a pressão arterial sistólica e diastólica do usuário usando um sensor baseado no protocolo UART e envia esses dados para processamento pelo microcontrolador.

Monitoramento Cardiovascular: Fornece dados de pressão arterial, que são um indicador importante da saúde cardiovascular.

- Microfone Estetoscópio:

Captura de Áudio: Grava os sons cardíacos e respiratórios do usuário através de um conversor analógico-digital (ADC) e envia os dados para o microcontrolador.

Auscultação: Usado para gravar e analisar sons do coração e pulmões, potencialmente detectando anormalidades e fornecendo uma forma de monitoramento por auscultação.

- Joystick:

Navegação e Controle: Permite que o usuário navegue pelos menus, faça seleções e controle o sistema. O usuário manipula o joystick e o ADC gera uma leitura que o microcontrolador pode traduzir em ações.

- Teclado Bluetooth:

Entrada de Texto: Fornece um meio para os usuários inserirem texto via uma interface sem fio Bluetooth para entradas de nome de usuário, notas ou outras formas de interação baseada em texto com o sistema.

Navegação e Controle: Permite que o usuário navegue pelos menus, faça seleções e controle o sistema.

- Display OLED:

Interface do Usuário: Fornece feedback visual dos dados adquiridos pelos vários sensores, resultados do processamento e permite que os usuários interajam com a interface do dispositivo por meio de menus.

Exibição de Dados: Exibe os dados de saúde medidos, mensagens de status do sistema e qualquer outra informação relevante.

- Oxímetro:

Medição da Saturação de Oxigênio no Sangue: Mede os níveis de saturação de oxigênio no sangue (SpO_2) do usuário usando um sensor que utiliza o protocolo de comunicação I2C.

Monitoramento Respiratório: Fornece dados sobre os níveis de oxigênio no sangue, o que é importante para o monitoramento respiratório.

- Botão A & Botão B:

Entrada do Usuário: Fornece uma interface para os usuários interagirem com o dispositivo por meio de pressionamentos de botões físicos, permitindo a seleção de opções, navegação em menus ou iniciação de ações. Os botões serão monitorados usando a interface GPIO.

- Buzzer:

Feedback Sonoro: Fornece feedback ou alertas sonoros ao usuário. Isso pode ser usado para indicar o início e o fim de uma medição ou para alertar sobre um evento específico do sistema. O buzzer será controlado usando uma interface GPIO.

- LED:

Feedback Visual: Fornece uma indicação visual do status do sistema (por exemplo, ligado, gravação de dados, erros) para o usuário. O LED será controlado usando a interface GPIO.

- Dispositivos Wifi (Via webserver):

Interface de Comunicação: Permite que o Raspberry Pi Pico W se comunique sem fio com dispositivos por meio de um webserver, que pode ser usado para configurar o dispositivo ou atuar como um painel de controle e análise de resultados.

- Banco de Dados na Nuvem:

Armazenamento e Análise de Dados: Fornece armazenamento remoto em nuvem para armazenamento de longo prazo de dados de saúde, além de permitir acesso remoto a dados e pode fornecer análise e visualização de dados.

5.2.3 Configuração de cada bloco

- Raspberry Pi Pico W (Microcontrolador):

Para a configuração geral o clock é configurado para operar na frequência padrão do Pico W (125 MHz), ou em uma frequência otimizada para baixo consumo de energia e precisão das leituras.

Na configuração da memória é utilizada a memória RAM disponível para o

armazenamento de dados temporários, buffers e pilha de execução do firmware.

Serão também inicializados os periféricos necessários, como I2C, UART, ADC, GPIO, Wi-Fi e Bluetooth.

O desenvolvimento do firmware é feito em C usando o SDK do Raspberry Pi Pico, incluindo as bibliotecas necessárias para comunicação com os sensores e a nuvem.

Serão também configurados os pinos GPIO para as linhas de dados (SDA) e clock (SCL) da interface I2C e definida a velocidade de comunicação I2C de 400 kHz para comunicação com o display OLED e o oxímetro.

Os pinos GPIO para as linhas de transmissão (TX) e recepção (RX) da interface UART serão configurados e a definição da velocidade de comunicação UART de 115200 bps para comunicação com o sensor de pressão arterial.

A configuração dos pinos GPIO para as entradas analógicas do ADC serão configuradas a fim de realizar a leitura do sinal do microfone e do joystick e também a definição da resolução do ADC e da taxa de amostragem (sample rate).

Os pinos GPIO para os botões, LED, e buzzer, também serão configurados definindo os modos de entrada ou saída.

O módulo Wi-Fi será inicializado para conectar-se à rede sem fio juntamente com a pilha TCP/IP para comunicação com webserver e a nuvem.

A Inicialização do módulo Bluetooth também será realizada para permitir a comunicação com o teclado Bluetooth.

- Sensor de Temperatura (DS18B20):

Para configurar o sensor de temperatura será feita a conexão do pino de dados do sensor DS18B20 a um pino GPIO do Raspberry Pi Pico W, implementação do protocolo OneWire no firmware do Pico W, usando bibliotecas existentes ou implementando uma solução própria, definição de resolução das medições (9, 10, 11 ou 12 bits), leitura dos dados de temperatura do sensor DS18B20 e conversão da temperatura para graus Celsius.

- Sensor de Pressão Arterial:

Na configuração do aferidor de pressão arterial será realizada a conexão do pino TX do sensor ao pino RX do Pico W, e pino RX do sensor ao pino TX do Pico W. Será utilizada a interface UART do Pico W para comunicação serial com o sensor., definição da velocidade de comunicação UART.

Também será necessário implementação da lógica de leitura e interpretação dos dados do sensor de pressão, cálculo da pressão sistólica e diastólica a partir dos dados recebidos.

- Microfone Estetoscópio:

Na configuração do auscultador será realizada a conexão do microfone acoplado em um estetoscópio com um pino analógico do ADC do Raspberry Pi Pico W. O circuito de interface inclui um amplificador operacional (op-amp) para amplificar o sinal do microfone.

Será feita a definição da resolução do ADC e taxa de amostragem, leitura dos dados do ADC a uma frequência apropriada para captura dos sons cardíacos e respiratórios, adaptação dos circuitos de leitura do microfone e os algoritmos de processamento de áudio para o ambiente do Raspberry Pi Pico W.

- Joystick:

Conexão dos eixos X e Y do joystick a dois pinos analógicos do ADC do Raspberry Pi Pico W, definição da resolução do ADC e taxa de amostragem, leitura dos valores dos eixos X e Y.

- Teclado Bluetooth:

Utilização da interface Bluetooth do Raspberry Pi Pico W para comunicação com o teclado e implementação do protocolo Bluetooth necessário para receber os dados do teclado.

- Display OLED:

Conexão do display OLED aos pinos I2C do Raspberry Pi Pico W utilização de uma biblioteca para o controle do display OLED (ex: RPIPicoSSD1306Exp), definição do endereço I2C do display e implementação de funções para exibir texto, imagens e gráficos no display.

- Oxímetro (MAX30102):

Conexão do sensor MAX30102 aos pinos I2C do Raspberry Pi Pico W, utilização da biblioteca max30102.h ou implementação de uma comunicação própria, definição do endereço I2C do sensor, configuração dos registradores do sensor para leitura dos dados de LEDs infravermelho e vermelho, configuração da taxa de amostragem e da resolução das leituras e cálculo da saturação de oxigênio (SpO2) e da frequência cardíaca a partir dos dados recebidos.

- Botão A & Botão B:

Conexão dos botões aos pinos GPIO configurados como entradas, utilização de resistores pull-up ou pull-down para garantir a leitura correta do estado dos botões e implementação de rotinas para detecção de eventos de clique nos botões.

- Buzzer:

Conexão do buzzer a um pino GPIO configurado como saída, utilização de um transistor para controlar a corrente no buzzer e implementação de rotinas para geração de sinais de áudio no buzzer.

- LED:

Conexão do LED a um pino GPIO configurado como saída, utilização de um resistor limitador de corrente e implementação de rotinas para acender e apagar o LED, ou para controlar seu brilho.

- Dispositivos Wifi (Via webserver):

Inicialização do módulo Wi-Fi do Raspberry Pi Pico W, configuração para conectar-se a uma rede Wi-Fi específica e desenvolvimento de um servidor web (webserver) que responde a requisições HTTP/HTTPS, para configurar o dispositivo e para comunicar dados.

- Banco de Dados na Nuvem:

Utilização do módulo Wi-Fi do Raspberry Pi Pico W para conectar-se à rede sem fio, implementação de protocolos de comunicação (HTTP/HTTPS ou MQTT) para comunicação com o banco de dados na nuvem e formatação dos dados para envio ao banco de dados na nuvem (JSON).

5.2.4 Comandos e registros

- Raspberry Pi Pico W (Microcontrolador):

Registros de Clock:

CLK_SYS_CTRL (para configurar o clock principal do sistema)

CLK_PERI_CTRL (para configurar o clock dos periféricos)

PLL_SYS_* (registros para configurar o PLL – Phase Locked Loop, para gerar o clock)

Registros GPIO:

IO_BANK0_GPIO*_CTRL (para configurar a função do pino)

GPIO*_OUT (para escrever valores nos pinos de saída)

GPIO*_IN (para ler valores dos pinos de entrada)

GPIO*_SET (para definir pinos de saída para alto)

GPIO*_CLR (para definir pinos de saída para baixo)

GPIO*_FSET (para definir pinos de saída para alto)

GPIO*_FCLR (para definir pinos de saída para baixo)

Registros I2C:

I2C*_CR (para habilitar/desabilitar o módulo I2C)

I2C*_SR (para verificar o status da comunicação)

I2C*_TXDATA (para enviar dados via I2C)

I2C*_RXDATA (para receber dados via I2C)

I2C*_ADDR (para configurar o endereço do slave I2C)

I2C*_CLKDIV (para definir a velocidade de comunicação I2C)

Registros UART:

UART*_CR (para habilitar/desabilitar o módulo UART)

UART*_LCR_H (para configurar o formato dos dados (bits, paridade, etc))

UART*_FR (para consultar flags de status da comunicação)

UART*_DR (para enviar/receber dados via UART)

UART*_IBRD (para configurar a parte inteira do divisor da taxa de transmissão)

UART*_FBRD (para configurar a parte fracionária do divisor da taxa de transmissão)

Registros ADC:

ADC_CS (para habilitar/desabilitar a conversão ADC)

ADC_RESULT (para ler o resultado da conversão ADC)

ADC_FCS (para controlar o conversor e ler os valores da conversão)

ADC_FIFO (para consultar o valor da conversão)

Registros Wi-Fi:

Registros específicos da biblioteca Wi-Fi (ex: cyw43_wifi_init(), cyw43_wifi_connect(), cyw43_tcpip_init())

Registros Bluetooth:

Registros específicos da biblioteca bluetooth (ex: bt_hci_init(), bt_adv_start(), etc.)

- Sensor de Temperatura (DS18B20):

Comandos de inicialização:

0xCC (Skip ROM)

0x44 (Convert Temperature)

0xBE (Read Scratchpad)

Registros do Scratchpad:

Bytes para armazenamento da temperatura.

- Sensor de Pressão Arterial:

O projeto utiliza um sensor de pressão arterial de pulso (um módulo GY-680) que se comunica via serial (UART). O sensor envia dados em um formato específico (um byte de start, seguido por bytes de dados, e um byte de stop). A comunicação UART envolve a configuração de baud rate (velocidade), bits de dados, paridade e stop bits.

```
uart_set_hw_flow(uart0, false, false);
uart_init(uart0, BAUD_RATE);
uart_set_fifo_enabled(uart0, true);
uart_set_format(uart0, DATA_BITS, STOP_BITS, UART_PARITY_NONE);
uart_set_baudrate(uart0, BAUD_RATE);
```

- Microfone Estetoscópio:

Comandos e Registros (ADC):

Registros de controle do ADC do Pico W (ex: ADC_CS, ADC_FCS).

Registros para leitura dos valores convertidos do ADC (ex: ADC_FIFO).

- Joystick:

Comandos e Registros (ADC):

Registros de controle do ADC do Pico W (ex: ADC_CS, ADC_FCS).

Registros para leitura dos valores convertidos do ADC (ex: ADC_FIFO).

- Teclado Bluetooth:

Comandos e Registros:

Registros e comandos da biblioteca Bluetooth para emparelhamento e recebimento de dados.

- Display OLED:

Comandos e Registros (I2C):

Comandos de inicialização do display (ex: configurar display, configurar resolução).

Comandos para desenho de textos e gráficos.

- Oxímetro (MAX30102):

Comandos e Registros (I2C):

Registros de controle e status do sensor (ex: REG_INTR_STATUS_1, REG_FIFO_WR_PTR).

Registros de configuração do sensor (ex: REG_MODE_CONFIG, REG_SPO2_CONFIG).

Registros para leitura dos dados FIFO (ex: REG_FIFO_DATA).

- Botão A & Botão B:

Comandos e Registros (GPIO):

Leitura dos pinos GPIO correspondentes aos botões (ex: GPIO*_IN).

- Buzzer:

Comandos e Registros (GPIO):

Controle do pino GPIO do buzzer (ex: GPIO*_OUT, GPIO*_SET, GPIO*_CLR).

- LED:

Comandos e Registros (GPIO):

Controle do pino GPIO do LED (ex: GPIO*_OUT, GPIO*_SET, GPIO*_CLR).

- Dispositivos Wifi (Via webserver):

Comandos e Registros (Wi-Fi/TCP):

Dependem da implementação do servidor web.

Funções de conexão Wi-Fi, criação de sockets, envio/recebimento de dados.

- Banco de Dados na Nuvem:

Comandos e Registros (HTTP/MQTT):

Dependem do serviço de nuvem utilizado.

Envolvem o envio de requisições HTTP/HTTPS ou mensagens MQTT.

5.2.5 Pinagem

GPIOs para I2C(Display OLED e OXÍMETRO):

GPIO14: SDA (Data)

GPIO15: SCL (Clock)

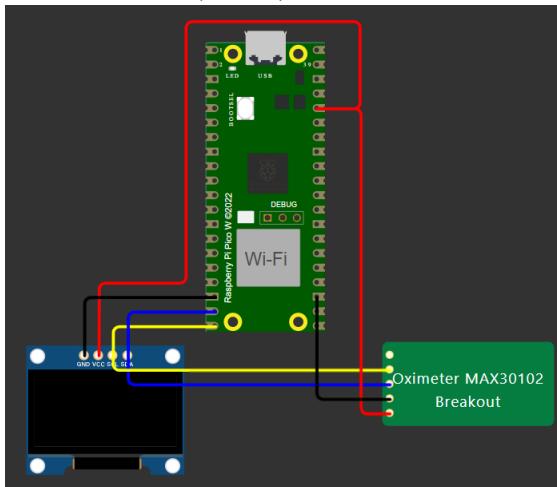


Figura 8. Pinagem do sistema de I₂C. Fonte: Autor

GPIOs para UART(Sensor de pressão arterial):

GPIO8: TX (Transmissão)

GPIO9: RX (Recepção)

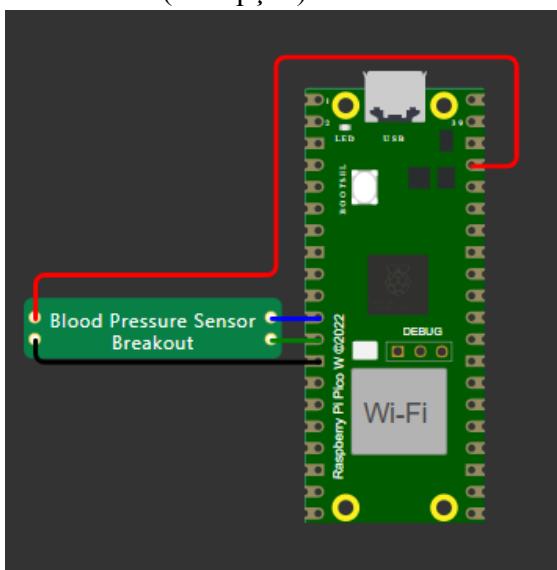


Figura 9. Pinagem do sistema UART. Fonte: Autor

GPIOs para ADC:

GPIO28: Entrada analógica para o microfone

GPIO26: Entrada analógica para o joystick

GPIO27: Entrada analógica para o joystick

GPIO22: Entrada analógica para o joystick

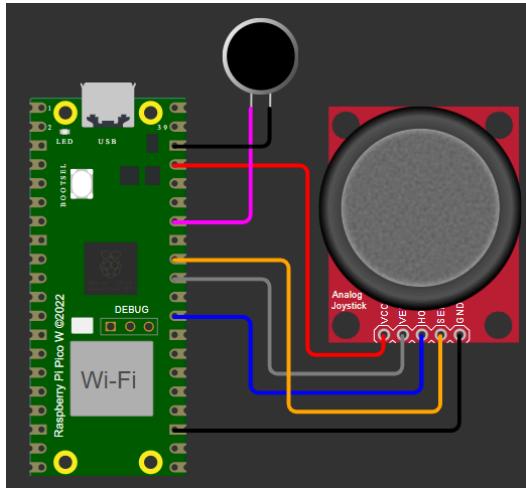


Figura 10. Pinagem do sistema ADC. Fonte: Autor

GPIOs para Botões:

GPIO5: Botão A

GPIO6: Botão B

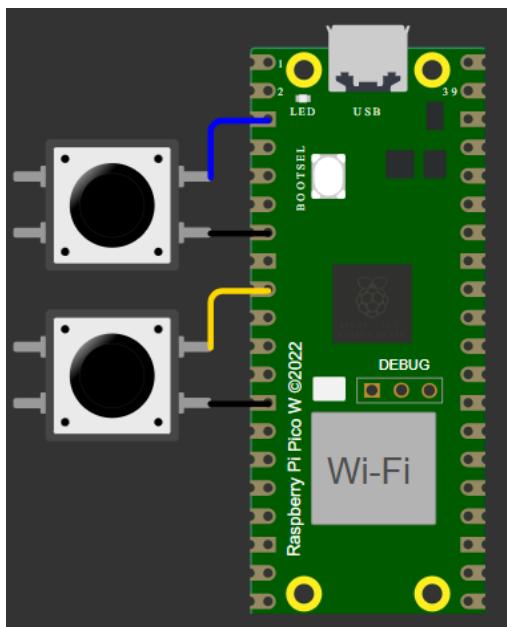


Figura 11. Pinagem do sistema de botões. Fonte: Autor

GPIO para LED:

GPIO13: LED_RED

GPIO12: LED_BLUE

GPIO11: LED_GREEN

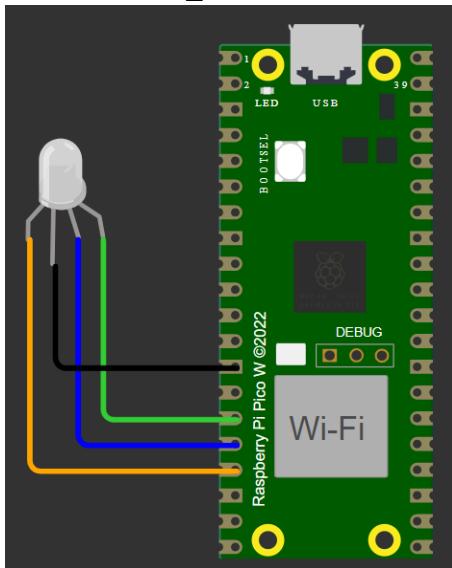


Figura 12. Pinagem do sistema de LED RGB. Fonte: Autor

GPIO para Buzzer:

GPIO10: Buzzer

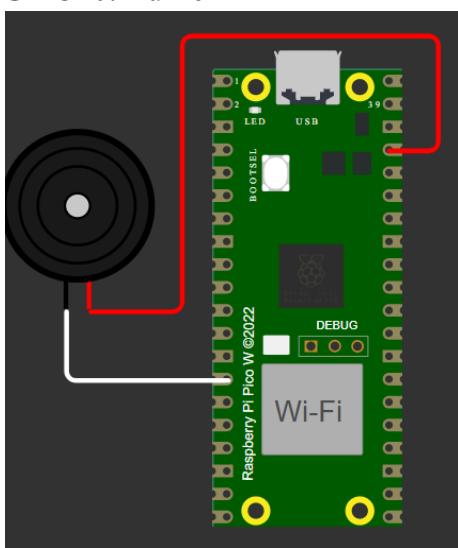


Figura 13. Pinagem do Buzzer. Fonte: Autor

Pinos para OneWire:

GPIO2: Pinos para dados do sensor de temperatura

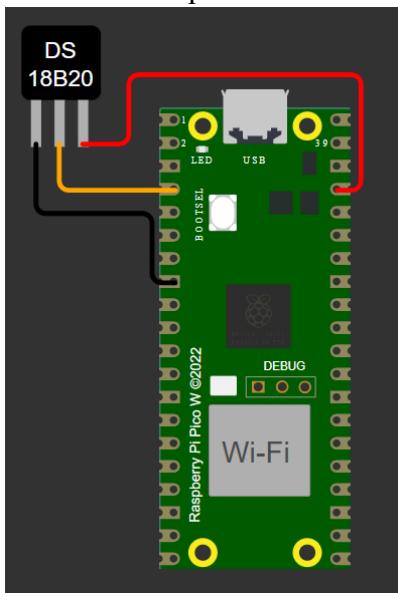


Figura 14. Pinagem do sensor de temperatura. Fonte: Autor

Pinos de alimentação e GND:

VSYS : Alimentação do sistema

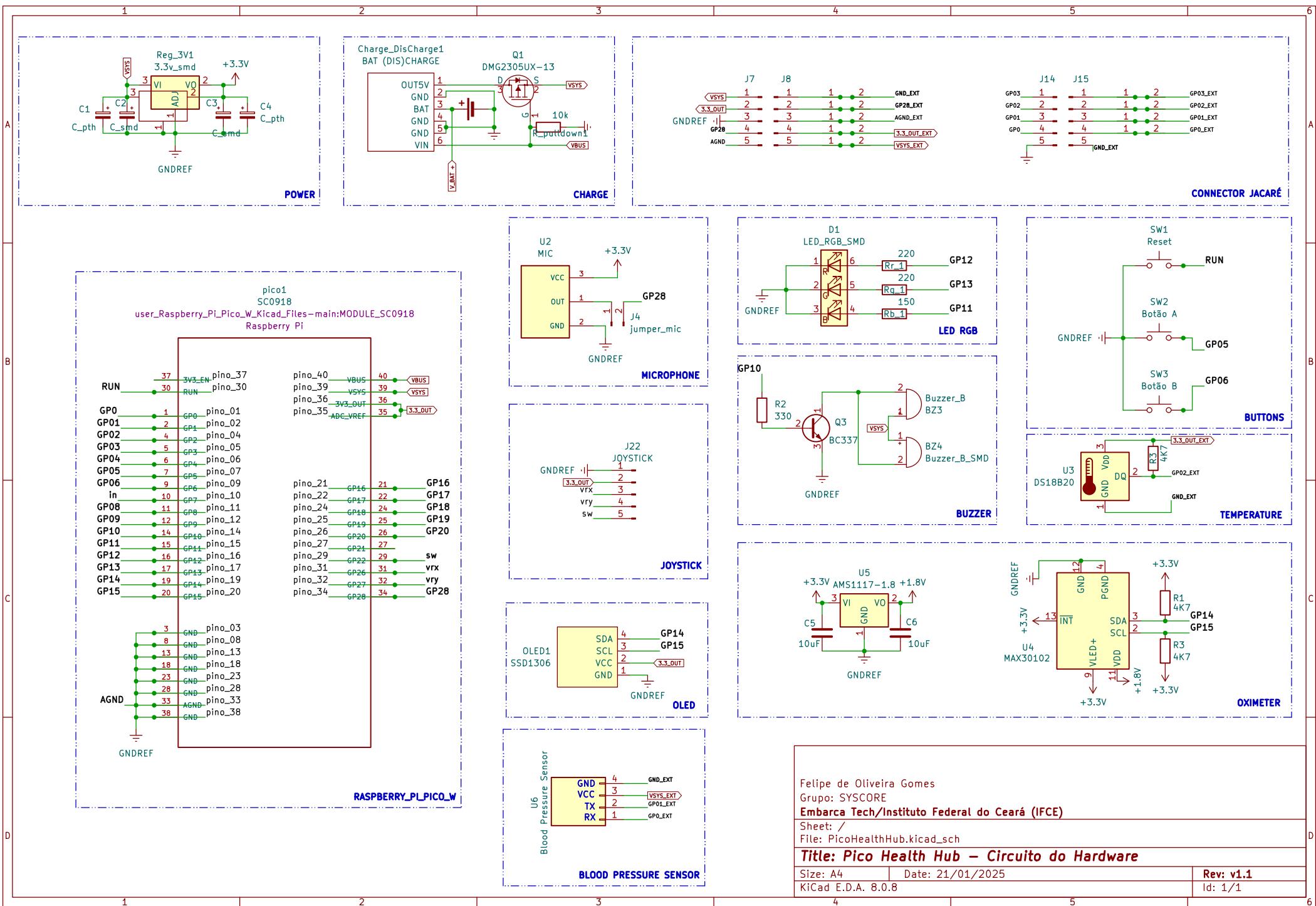
GND : Terra do sistema

Pinos Wi-Fi: Pinos internos do módulo Wi-Fi da Pico W.

Pinos Bluetooth: Pinos internos do módulo bluetooth da Pico W.

5.2.5 Circuito completo do hardware

https://github.com/FelipeOlliver/PicoHealthHub/blob/main/KiCAD/PicoHealthHub/pico_health_hub_circuito.pdf



5.2.6 Montagem

Para a montagem do protótipo foram realizadas a solda de fios com conectores microfit na placa bitdoglab visando a conexão dos periféricos e distância necessária para realização das medições. Conforme a figura 15 o microfone da placa foi removido e inserido um jack p2 fêmea no lugar facilitando assim a conexão do microfone estetoscópio.

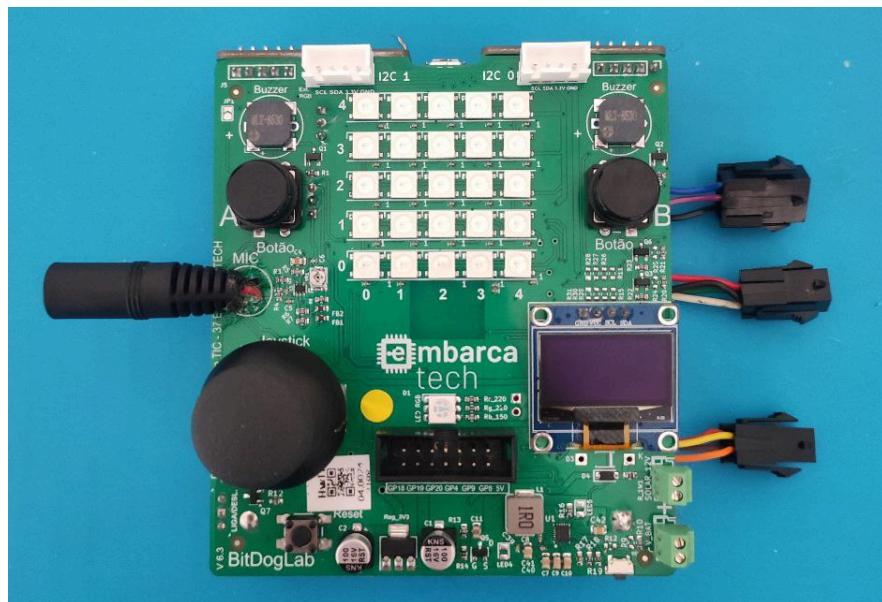


Figura 15. Frente da placa bitdoglab adaptada para funcionamento do pico health hub. Fonte: Autor

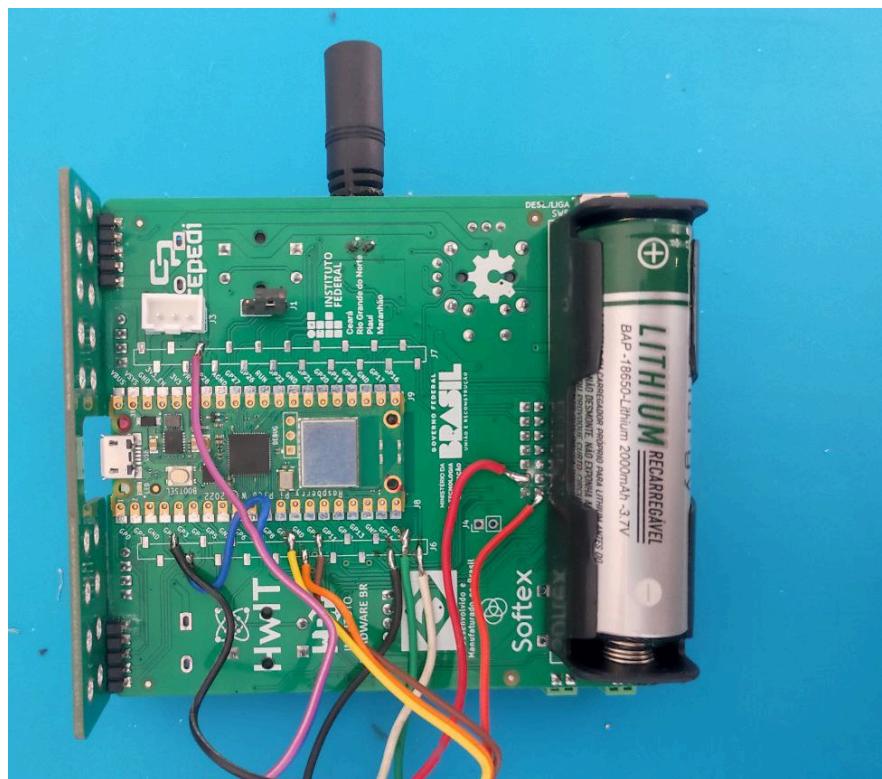


Figura 16. Verso da placa bitdoglab adaptada para funcionamento do pico health hub. Fonte: Autor

Para utilização do termômetro (sensor de temperatura) foi feita uma adaptação usando fios e um conector microfit 6 vias conforme a figura 17.

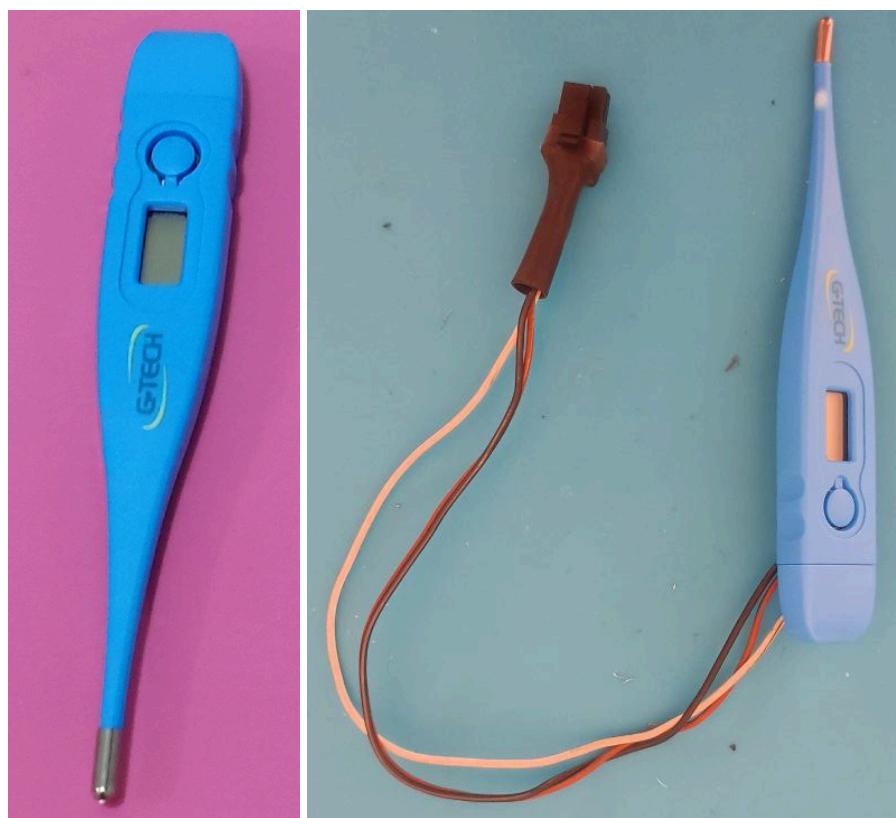


Figura 17. Termômetro adaptado para funcionamento com o pico health hub. Fonte: Autor

Para utilização do oxímetro foi feita uma adaptação usando fios e um conector microfit 4 vias conforme a figura 18.



Figura 18. Oxímetro adaptado para funcionamento com o pico health hub. Fonte: Autor

Para utilização do sensor de pressão arterial foi feita uma adaptação usando fios e um conector microfit 4 vias conforme a figura 20.



Figura 19. Sensor de pressão arterial. Fonte: Autor

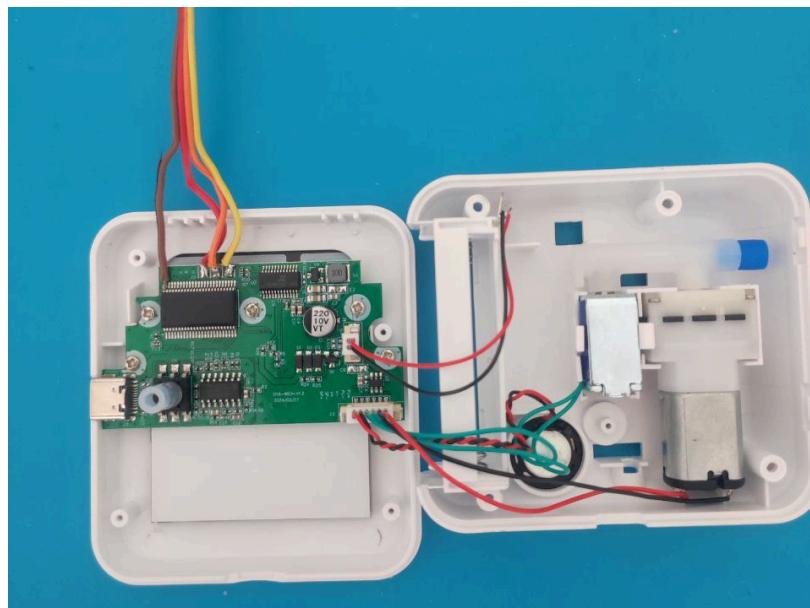


Figura 21. Sensor de pressão arterial adaptado para funcionamento com o pico health hub.
Fonte: Autor

Para utilização do microfone estetoscópio foi feita uma adaptação usando um estetoscópio tradicional e um microfone conforme as figuras 21 a 23.



Figura 21. Estetoscópio e microfone utilizados no pico health hub separados. Fonte: Autor



Figura 22. Estetoscópio e microfone utilizados no pico health hub unidos. Fonte: Autor



Figura 23. Estetoscópio e microfone adaptados para uso no pico health hub. Fonte: Autor

Finalmente com todos os componentes do projeto adaptados foi possível ser realizada a montagem completa do protótipo conforme a figura 24.

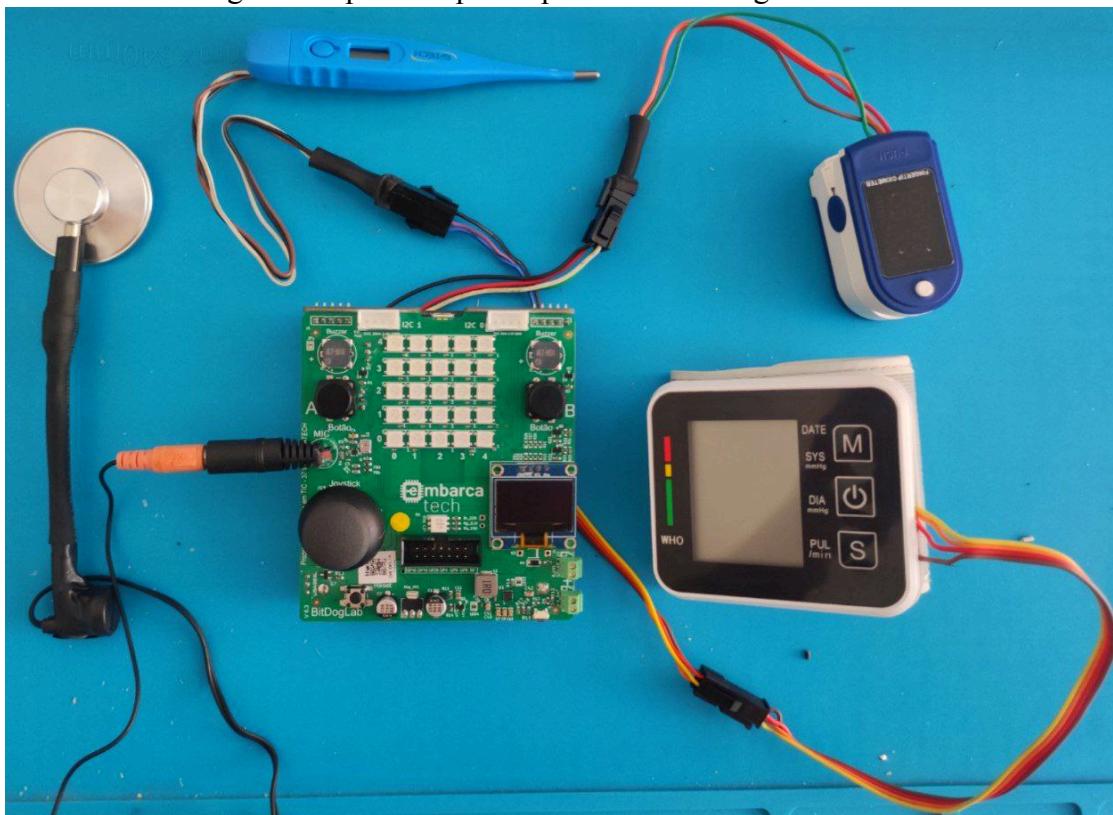


Figura 24. Protótipo do pico health hub com todos os periféricos conectados. Fonte: Autor

5.3 Especificação do firmware

Esta seção detalha a arquitetura e a funcionalidade do firmware que controla o 'Pico Health Hub'. O firmware, peça fundamental para o funcionamento do dispositivo, é responsável pela coleta de dados dos sensores, pelo processamento dos sinais, pela interação com o usuário através da interface, e pela comunicação com a plataforma na nuvem. Para melhor entendimento, apresentamos um diagrama das camadas do software, descrevendo suas respectivas funções, seguido de uma visão geral das funcionalidades de cada bloco de software. Em seguida, detalharemos as principais variáveis utilizadas, o fluxograma completo do software, o processo de inicialização, as configurações dos registros, a estrutura e o formato dos dados, a organização da memória, e os protocolos de comunicação implementados para garantir a operação eficiente e confiável do sistema, incluindo o formato dos pacotes de dados.

5.3.1 Diagrama em blocos do software

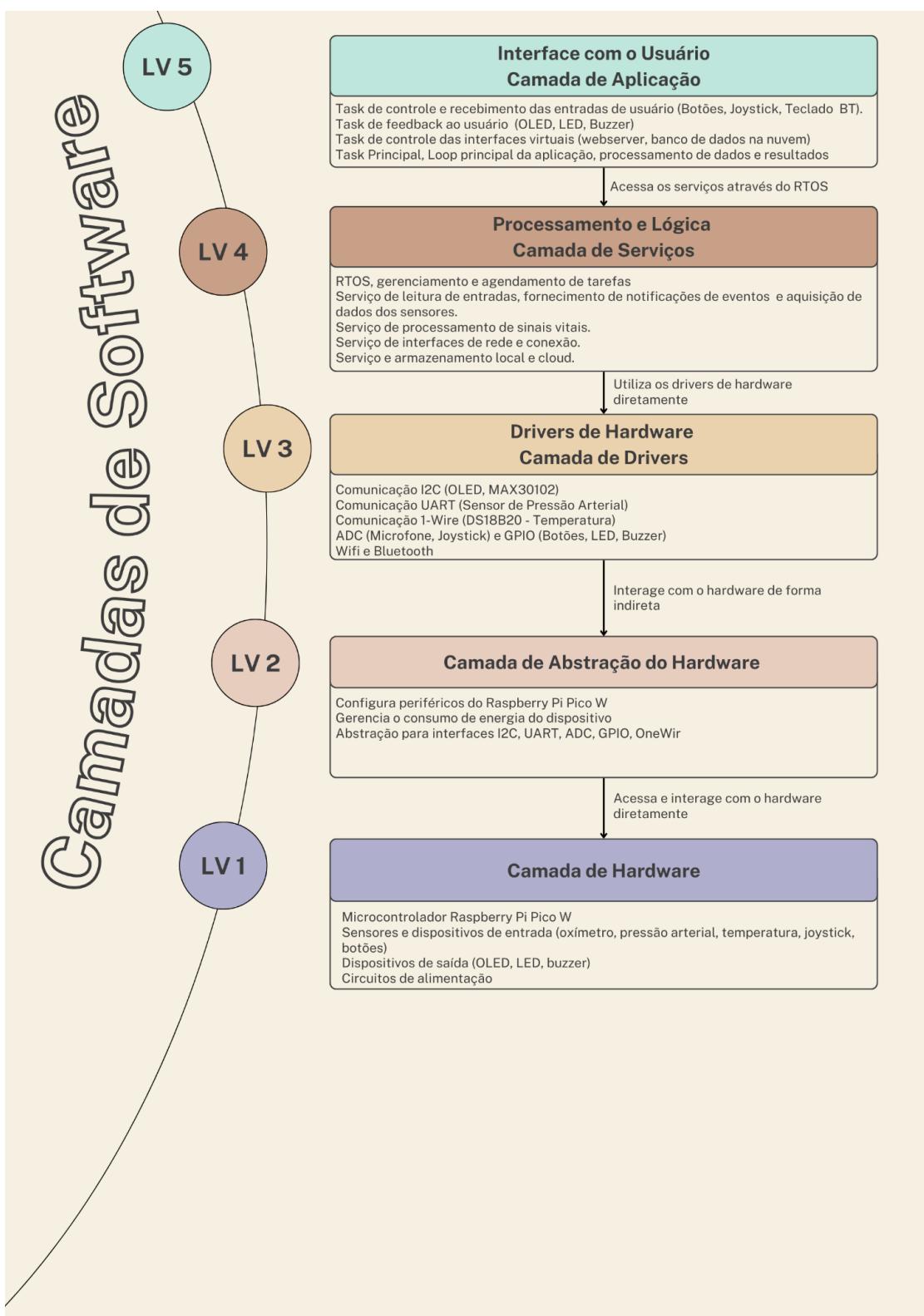


Figura 25. Diagrama de blocos funcionais do software. Fonte: Autor

5.3.2 Descrição das funcionalidades dos blocos

Cada camada tem um papel bem definido, garantindo um sistema modular, eficiente e escalável. A Camada de Hardware fornece os dados, os drivers gerenciam a

comunicação, a Camada de Serviços processa informações e a Camada de Aplicação apresenta os resultados ao usuário.

- **LV5 - Interface com o Usuário (Camada de Aplicação):**

Task de controle e recebimento das entradas do usuário (Botões, Joystick, Teclado BT):

Função: Responsável por monitorar as entradas do usuário, como cliques nos botões, movimentos do joystick e dados do teclado Bluetooth.

Descrição: Transforma eventos de entrada do usuário em comandos comprehensíveis pelo sistema. Utiliza o serviço de leitura de entradas da camada de serviços para obter os dados brutos e envia para a task principal.

Task de feedback ao usuário (OLED, LED, Buzzer):

Função: Gerencia os mecanismos de feedback visual e sonoro para o usuário.

Descrição: Controla o display OLED para exibir informações, o LED RGB para indicações visuais e o buzzer para alertas sonoros. Envia informações a serem exibidas ou acionadas para os serviços de feedback.

Task de controle das interfaces virtuais (webserver, banco de dados na nuvem):

Função: Gerencia a comunicação do dispositivo com o webserver e o banco de dados na nuvem.

Descrição: Envia dados do dispositivo para o banco de dados e recebe informações do webserver (se for o caso). Utiliza serviços de rede e conexão para realizar o processo.

Task Principal, Loop principal da aplicação, processamento de dados e resultados:

Função: Contém o loop principal do programa, que orquestra o funcionamento geral do dispositivo.

Descrição: Responsável por coordenar as outras tasks, processar os dados obtidos dos sensores, executar a lógica de negócios, e gerenciar o envio e recebimento de dados. Utiliza todos os serviços da camada de serviços.

- **LV4 - Processamento e Lógica (Camada de Serviços):**

RTOS, gerenciamento e agendamento de tarefas:

Função: Gerencia a execução das tarefas, alocando recursos e agendando a execução para garantir que as tasks funcionem de maneira eficiente e em tempo real.

Descrição: Oferece os serviços de agendamento e gerenciamento, que são utilizados pela camada de aplicação.

Serviço de leitura de entradas, fornecimento de notificações de eventos e aquisição de dados dos sensores:

Função: Abstrai a leitura de dados de hardware e fornece notificações de eventos para as camadas superiores.

Descrição: Utiliza drivers de baixo nível para ler os dados brutos dos sensores e notifica a camada de aplicação quando um evento ocorre.

Serviço de processamento de sinais vitais:

Função: Processa os dados brutos dos sensores para obter os sinais vitais (temperatura, pressão arterial, SpO2, sons cardíacos/respiratórios).

Descrição: Aplica algoritmos de processamento de sinais digitais para extrair informações significativas dos dados coletados.

Serviço de interfaces de rede e conexão:

Função: Gerencia a comunicação com a plataforma na nuvem e webserver (se for o caso).

Descrição: Implementa os protocolos de comunicação necessários para enviar e receber dados através do Wi-Fi e Bluetooth.

Serviço e armazenamento local e cloud:

Função: Gerencia o armazenamento local temporário dos dados, e o envio para a nuvem.

Descrição: Armazena dados temporariamente para que não sejam perdidos em caso de falha na comunicação com a nuvem e envia para a nuvem de forma assíncrona.

- **LV3 - Drivers de Hardware (Camada de Drivers):**

Comunicação I2C (OLED, MAX30102):

Função: Controla a comunicação I2C com o display OLED e o sensor de oximetria MAX30102.

Descrição: Implementa os protocolos de comunicação necessários para enviar comandos e receber dados dos dispositivos através do barramento I2C.

Comunicação UART (Sensor de Pressão Arterial):

Função: Controla a comunicação UART com o sensor de pressão arterial.

Descrição: Implementa os protocolos de comunicação para enviar comandos ao sensor e receber os dados da pressão arterial.

Comunicação 1-Wire (DS18B20 - Temperatura):

Função: Controla a comunicação 1-Wire com o sensor de temperatura DS18B20.

Descrição: Implementa os protocolos para enviar comandos e receber os dados de temperatura do sensor.

ADC (Microfone, Joystick) e GPIO (Botões, LED, Buzzer):

Função: Fornece interfaces para o conversor analógico digital e para as entradas e saídas digitais.

Descrição: Permite a leitura dos valores analógicos do microfone e joystick, o controle dos botões, o acionamento do LED e do buzzer.

Wifi e Bluetooth:

Função: Implementa a interface para conexões sem fio.

Descrição: Permite o dispositivo se comunicar com redes Wi-fi e dispositivos Bluetooth

- **LV2 - Camada de Abstração do Hardware:**

Configura periféricos do Raspberry Pi Pico W:

Função: Inicializa e configura os periféricos do Raspberry Pi Pico W.

Descrição: Configura os pinos GPIO, os controladores I2C, UART, ADC, e outros periféricos do microcontrolador.

Gerencia o consumo de energia do dispositivo:

Função: Otimiza o uso de energia para aumentar a autonomia do dispositivo.

Descrição: Monitora o consumo de energia e ajusta configurações para prolongar a duração da bateria.

Abstração para interfaces I2C, UART, ADC, GPIO, OneWire:

Função: Fornece uma camada de abstração para as interfaces de hardware, ocultando detalhes de baixo nível das camadas superiores.

Descrição: Simplifica o uso das interfaces de comunicação, fornecendo uma API consistente para as camadas de drivers.

- **LV1 - Camada de Hardware:**

Microcontrolador Raspberry Pi Pico W:

Função: Executa o firmware e controla todas as funções do dispositivo.

Descrição: É o núcleo do sistema, onde o código é executado e todas as operações são orquestradas.

Sensores e dispositivos de entrada (oxímetro, pressão arterial, temperatura, joystick, botões):

Função: Coleta dados fisiológicos e interações do usuário

Descrição: Converte grandezas físicas em sinais elétricos que podem ser interpretados pelo microcontrolador.

Dispositivos de saída (OLED, LED, buzzer):

Função: Fornecem feedback visual e sonoro para o usuário.

Descrição: Exibem informações no display OLED, emitem luz através do LED e sons pelo buzzer.

Circuitos de alimentação:

Função: Fornecem energia para o funcionamento do sistema

Descrição: Garante que todos os componentes do dispositivo recebam a energia necessária para o seu funcionamento

5.3.3 Definição das principais variáveis

1. Variáveis de Estado do Sistema:

```
// Variáveis de Estado do Sistema
enum SystemState { IDLE, MEASURING, SENDING_DATA, ERROR };
enum MeasurementStatus { NOT_STARTED, IN_PROGRESS, COMPLETE
};
```

```
SystemState systemState = IDLE;
MeasurementStatus measurementStatus = NOT_STARTED;
```

Representa o estado atual do sistema (ex: IDLE, MEASURING, SENDING_DATA, ERROR). Controla o fluxo principal da aplicação.

Indica o estado da medição atual (ex: NOT_STARTED, IN_PROGRESS, COMPLETE).

2. Variáveis de Sensores:

```
// Variáveis de Sensores
float temperature = 0.0;
int systolicPressure = 0;
int diastolicPressure = 0;
int spo2Level = 0;
int heartRate = 0;
uint16_t rawAudioData[1024]; // Buffer para dados de áudio
```

Armazena o valor da temperatura corporal em graus Celsius.

Armazena o valor da pressão sistólica em mmHg.

Armazena o valor da pressão diastólica em mmHg.

Armazena o nível de saturação de oxigênio no sangue (SpO2) em porcentagem.

Armazena a frequência cardíaca em batimentos por minuto (BPM).

Armazena os dados brutos do microfone (sons cardíacos/respiratórios) coletados pelo ADC.

3. Variáveis de Interface com o Usuário:

```
// Variáveis de Interface com o Usuário
char displayBuffer[128]; // Exemplo de buffer de texto
uint8_t buttonState = 0;
int joystickX = 0;
int joystickY = 0;
```

Buffer para armazenar o texto ou os gráficos que serão exibidos no display OLED.

Armazena o estado atual dos botões (ex: pressionado, não pressionado). Usada para identificar eventos de clique.

Armazena o valor da posição do joystick no eixo X.

Armazena o valor da posição do joystick no eixo Y.

4. Variáveis de Tempo:

```
// Variáveis de Tempo
uint32_t measurementStartTime = 0;
uint32_t systemUptime = 0;
```

Armazena o tempo em milissegundos em que uma medição foi iniciada. Utilizada para calcular o tempo decorrido e timeouts.

Armazena o tempo desde o início da execução do dispositivo.

5. Variáveis de Comunicação com a Nuvem:

```
// Variáveis de Comunicação com a Nuvem
char cloudDataBuffer[512];
enum CloudStatus { CONNECTED, DISCONNECTED, SENDING, ERROR };
CloudStatus cloudStatus = DISCONNECTED;
```

Buffer para armazenar os dados formatados para envio à nuvem. Formato JSON.
 Indica o estado da comunicação com a nuvem (ex: CONNECTED, DISCONNECTED, SENDING, ERROR).

6. Variáveis de Configuração:

```
// Variáveis de Configuração
int sampleRate = 100; // Hz
uint8_t i2cAddressOLED = 0x3C;
uint8_t i2cAddressMAX30102 = 0x57;
```

Armazena a taxa de amostragem dos sensores (em Hertz).
 Endereço I2C do display OLED.
 Endereço I2C do sensor de oximetria MAX30102.

7. Variáveis de Buffer de Dados

```
// Variáveis de Buffer de Dados
uint8_t uartRxBuffer[64];
uint8_t uartTxBuffer[64];
```

Buffer para armazenar dados recebidos pela interface UART (sensor de pressão arterial).
 Buffer para armazenar dados a serem transmitidos pela interface UART.

5.3.4 Fluxograma do software

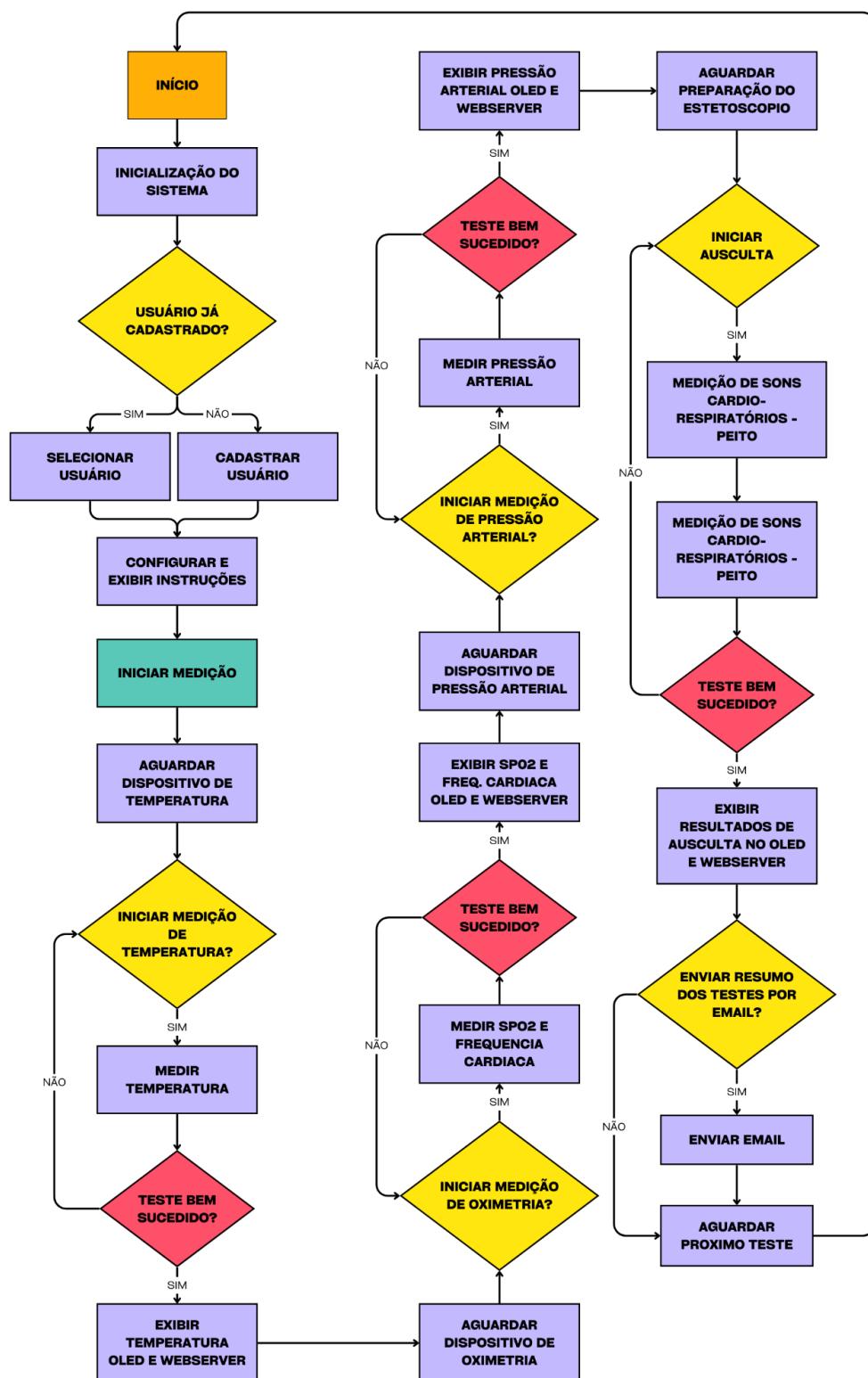


Figura 26. Fluxograma do software. Fonte: Autor

5.3.4.1 Descrição Detalhada do Fluxograma

Início:

O software é iniciado, começando a execução no nível da camada de aplicação (LV5).

Inicialização do Sistema:

A camada de abstração do hardware (LV2) configura os periféricos do Raspberry Pi Pico W.

Drivers são inicializados (I2C, UART, 1-Wire, GPIO, ADC).

O RTOS (LV4) é inicializado para gerenciar as tarefas.

Verificação do Usuário:

O software verifica se já existe um usuário cadastrado.

Seleção ou Cadastro de Usuário:

Se já existir um usuário, o software permite que o usuário selecione seu nome (usando botões/joystick).

Se não existir, o software inicia um processo de cadastro do usuário (usando botões/joystick e teclado Bluetooth, se disponível ou via webserver).

Após a seleção ou cadastro, o software avança para a próxima etapa.

Configurar e Exibir Instruções:

O software exibe instruções na tela OLED sobre como utilizar o "Pico Health Hub".

As instruções incluem a colocação dos sensores de temperatura, oxímetro e pressão arterial.

Iniciar Medição:

A task principal inicia o processo de medição.

Aguardar dispositivo de temperatura:

Aguarda o usuário colocar o dispositivo de temperatura, o LED é amarelo enquanto aguarda.

Medição de Temperatura:

O software pergunta ao usuário se ele já colocou o sensor de temperatura.

Se sim, O sensor de temperatura DS18B20 (via 1-Wire) é acionado.

O valor da temperatura é lido e armazenado em uma variável.

Exibir Temperatura:

O valor da temperatura é exibido no display OLED.

Aguardar dispositivo de oximetria:

Aguarda o usuário colocar o dispositivo de oximetria, o LED é amarelo enquanto aguarda.

Medição de Oximetria:

O software pergunta ao usuário se ele já colocou o sensor de oximetria.

Se sim, o sensor MAX30102 (via I2C) é acionado.

Os valores de SpO2 e frequência cardíaca são lidos e armazenados.

Exibir SpO2 e Frequência Cardíaca:

Os valores de SpO2 e frequência cardíaca são exibidos no display OLED.

Aguardar dispositivo de pressão arterial:

Aguarda o usuário colocar o dispositivo de pressão arterial, o LED é amarelo enquanto aguarda.

Medição de Pressão Arterial:

O software pergunta ao usuário se ele já colocou o sensor de pressão arterial.

Se sim, o sensor de pressão arterial (via UART) é acionado.

Os valores da pressão sistólica e diastólica são lidos e armazenados.

Exibir Pressão Arterial:

Os valores da pressão arterial são exibidos no display OLED.

Aguardar Preparação do Estetoscópio:

Aguarda o usuário preparar o estetoscópio, o LED é amarelo enquanto aguarda.

Medição de Sons Cardíacos/Respiratórios:

O software pergunta ao usuário se ele já está preparado para auscultar.

Se sim, O microfone do estetoscópio (via ADC) é acionado.

O usuário é orientado a realizar a medição no peito (5 locais) e nas costas (4 locais), respirando fundo e soltando o ar durante a medição.

Os dados de áudio são armazenados.

Exibir Resumo dos Testes:

Um resumo de todos os testes é exibido no display OLED.

O resumo também é enviado para o servidor web para visualização.

Enviar Teste por E-mail:

O software pergunta ao usuário se ele deseja enviar o resumo dos testes por e-mail.

Enviar E-mail:

Se o usuário concordar, um e-mail é enviado com os resultados.

Aguardar Próximo Teste:

O dispositivo aguarda para realizar o novo ciclo de testes.

Retorna ao começo do processo de medição (etapa 6)

LED e Buzzer:

LED RGB: Vermelho: Durante a medição (temperatura, SpO2, pressão arterial).

Amarelo: Indicando que o dispositivo está pronto para a medição, mas aguardando o usuário. Verde: Modo de espera, sem medição em andamento.

Buzzer: Um bipe curto indica medição concluída. Sons de alerta para medições fora dos padrões. Um som de erro se alguma medição falhar

Observações:

Sequencial: O fluxo do software é sequencial, guiando o usuário através de cada etapa do processo.

Interativo: O software interage com o usuário através da tela OLED, botões e LED.

Responsividade: O sistema aguarda o usuário colocar cada sensor e indica por meio de luz qual etapa está em andamento.

5.3.5 Inicialização do software

Ao ser ligado, o "Pico Health Hub" inicia um processo de inicialização que abrange diversas camadas de software e hardware. Primeiramente, o microcontrolador Raspberry Pi Pico W passa por um reset de energia, garantindo que todos os seus registradores e periféricos começam em um estado conhecido. Em seguida, o clock do sistema é configurado para operar na frequência desejada, e a memória RAM é inicializada, com a pilha (stack) sendo definida para alocar os recursos do programa. Os módulos de energia, como os gerenciadores de bateria e reguladores de tensão, são ativados, assegurando que todos os componentes recebem o suprimento de energia adequado.

Na sequência, a camada de abstração do hardware (LV2) entra em ação, configurando os periféricos do Raspberry Pi Pico W. Isso inclui a definição da funcionalidade dos pinos GPIO (entradas para botões e joystick, saídas para LEDs e buzzer), a inicialização do módulo I2C para comunicação com o display OLED e o sensor de oximetria, a inicialização do módulo UART para comunicação com o sensor de pressão arterial, a inicialização do módulo ADC para leitura dos dados do microfone e joystick e a inicialização do módulo OneWire para comunicação com o sensor de temperatura. Adicionalmente, os módulos de comunicação sem fio (WiFi e Bluetooth)

são inicializados. As funções de abstração para cada interface (I2C, UART, ADC, GPIO, OneWire, WiFi e Bluetooth) são preparadas para uso pelas camadas superiores, e as configurações para gestão de energia são definidas, otimizando o consumo e a autonomia do dispositivo.

A camada de drivers (LV3) é então inicializada, com seus drivers para cada componente sendo ativados. Isso compreende a definição das funções de leitura e escrita dos pinos GPIO (GPIO Driver), de envio e recebimento de dados via I2C (I2C Driver), de comunicação via UART (UART Driver), de leitura dos dados do ADC (ADC Driver), de comunicação via OneWire (OneWire Driver), e a comunicação sem fio é inicializada com seus respectivos drivers (WiFi and Bluetooth Drivers). Em seguida, os dispositivos de hardware conectados são testados e, caso necessário, inicializados (ex: comandos de inicialização do OLED, sensor de pressão arterial, MAX30102, DS18B20).

A camada de serviços (LV4) é ativada com a inicialização e configuração do RTOS (Real-Time Operating System). Isso inclui a criação de tarefas e definição de prioridades, com cada serviço sendo inicializado em ordem. O serviço de leitura de entradas é definido para obter dados dos botões e joystick, fornecendo notificações de eventos para a camada de aplicação. O serviço de processamento de sinais vitais é configurado, disponibilizando funções de leitura e processamento dos sensores. E o serviço de interfaces de rede e conexão ativa os drivers de WiFi e Bluetooth para iniciar a comunicação, e por fim o serviço de armazenamento local e cloud é configurado para armazenamento temporário de dados e envio para a nuvem.

Finalmente, a interface com o usuário (LV5) é inicializada, com uma tela de inicialização ou logotipo sendo exibida no display OLED e o LED RGB é aceso indicando que o dispositivo está inicializando. As variáveis globais que armazenam dados do usuário e dados do sistema são inicializadas, como systemState = IDLE e measurementStatus = NOT_STARTED. A tarefa principal da camada de aplicação é iniciada, executando um loop que verifica se o usuário já está cadastrado, solicitando o cadastro se necessário, exibindo instruções de uso no display OLED. Após este processo, o software entra em um loop principal, aguardando a interação do usuário para iniciar as medições, marcando o fim do processo de inicialização do dispositivo.

5.3.6 Configurações dos registros

Os registradores, localizados na memória do microcontrolador, são responsáveis por determinar o comportamento do hardware. O processo de configuração inicia-se com o controle do sistema e do clock, onde o registrador CLK_SYS_CTRL ajusta a frequência de operação do microcontrolador, enquanto CLK_PERI_CTRL define o clock para cada periférico, garantindo que funcionem dentro de suas especificações.

Em seguida, os registradores GPIO são configurados. O registrador IO_BANK0_GPIOx_CTRL determina a função de cada pino (entrada, saída ou função alternativa), permitindo que pinos sejam definidos como saídas para LEDs ou entradas para botões. Os registradores GPIOx_OUT, GPIOx_SET e GPIOx_CLR controlam o estado lógico dos pinos de saída, permitindo a escrita de valores diretamente nos pinos e o registro GPIOx_IN lê o estado lógico de cada pino configurado como entrada.

Para a comunicação I2C, o registrador I2Cx_CR habilita ou desabilita o módulo, enquanto I2Cx_SR lê o status da comunicação. Os registradores I2Cx_TXDATA e I2Cx_RXDATA são utilizados para enviar e receber dados, respectivamente. O registrador I2Cx_ADDR define o endereço do dispositivo escravo I2C, e o registrador

I2Cx_CLKDIV ajusta a velocidade da comunicação. De forma similar, a comunicação UART é configurada com o registro UARTx_CR que habilita ou desabilita o módulo, o registro UARTx_LCR_H que configura o formato dos dados e o registro UARTx_FR consulta as flags de status. Os registros UARTx_DR, UARTx_IBRD e UARTx_FBRD são utilizados para envio/recebimento de dados e para configurar a velocidade de comunicação.

A configuração do ADC é realizada por meio do registro ADC_CS, que habilita ou desabilita o conversor, definindo seu modo de operação e taxa de amostragem. O registro ADC_FIFO armazena os resultados da conversão. A comunicação OneWire é feita através de registros específicos da biblioteca utilizada, envolvendo a configuração de um pino GPIO como saída/entrada, com controle preciso do tempo de leitura/escrita para atender aos requisitos do protocolo. Para os módulos Wi-Fi e Bluetooth, os registros específicos de cada biblioteca são configurados, incluindo informações como o nome da rede (SSID), senha, tipo de autenticação (Wi-Fi), nome do dispositivo (Bluetooth), tipo de emparelhamento e serviços a serem utilizados.

Em resumo, a configuração dos registradores é um processo complexo, que envolve a manipulação de diversos bits para ativar ou desativar funcionalidades específicas do hardware. A documentação do microcontrolador e dos periféricos é essencial para entender a função de cada bit e para garantir o funcionamento correto do sistema. As configurações dos registradores são geralmente encapsuladas em funções de drivers, que abstraem os detalhes de hardware das camadas superiores, facilitando o desenvolvimento e a manutenção do firmware.

5.3.7 Estrutura e formato dos dados

Os dados dos sensores, da interface com o usuário e da comunicação com a nuvem são estruturados de maneira específica para otimizar o uso da memória e facilitar o processamento. Os dados dos sensores, por exemplo, são representados da seguinte forma: a temperatura, medida pelo sensor DS18B20, é armazenada como um valor de ponto flutuante (float) de 32 bits, representando a temperatura em graus Celsius; os valores de pressão arterial (sistólica e diastólica), medidos pelo sensor UART, são armazenados como inteiros (int) de 16 bits; os dados de oximetria (SpO2 e frequência cardíaca), obtidos do sensor MAX30102, são armazenados como inteiros (int) de 8 e 16 bits, respectivamente; e os dados de áudio (sons cardíacos/respiratórios), coletados pelo microfone via ADC, são armazenados como um array (uint16_t[]) de inteiros sem sinal de 16 bits.

Os dados da interface com o usuário incluem mensagens de texto, que são armazenadas como strings (char[]), o estado dos botões, que são armazenados como valores booleanos (uint8_t) e as posições do joystick, que são armazenadas como inteiros (int). O feedback visual do LED RGB é armazenado como um inteiro sem sinal (uint8_t) ou valores de enumeração, representando a cor do LED (vermelho, verde, azul, desligado).

Para os dados de configuração, a taxa de amostragem dos sensores é armazenada como um inteiro (int) de 16 bits, e os endereços I2C dos dispositivos são armazenados como inteiros sem sinal (uint8_t). Já os dados para comunicação com a nuvem são formatados em JSON (JavaScript Object Notation), um formato leve e fácil de analisar e transmitir em aplicações web. Os dados são organizados em pares chave-valor, como por exemplo, valores do sensor e dados de status do dispositivo, com campos

separados para cada sensor e dados do usuário (nome, email) quando da necessidade do envio para a nuvem.

Para melhorar a organização do código, são utilizadas estruturas de dados (struct) em C/C++, que agrupam dados relacionados, como por exemplo, dados de medidas (MeasurementData), dados de entrada do usuário (UserInputData), dados do usuário (UserData), dados de feedback (FeedbackData). Essas estruturas facilitam o acesso e a manipulação dos dados, tornando o código mais legível e fácil de manter.

A escolha dos tipos de dados é cuidadosamente feita com base na precisão e no intervalo de valores que cada variável precisa armazenar, otimizando o uso da memória do microcontrolador. Os tamanhos dos dados são definidos para evitar o desperdício de recursos, enquanto a endianness é considerada para assegurar a correta interpretação de dados de múltiplos bytes. O uso do formato JSON facilita a comunicação com a plataforma na nuvem, permitindo a interoperabilidade do sistema. As estruturas de dados (struct) agrupam informações relacionadas, melhorando a organização, legibilidade e manutenção do código. Além disso, a concisão no armazenamento e transmissão dos dados é uma prioridade para garantir a rapidez do sistema. Por fim, a segurança dos dados é primordial, evitando a exposição de dados sensíveis, e a robustez do sistema é assegurada com o tratamento de dados inválidos ou parciais, para prevenir possíveis falhas.

```

typedef struct {
    float temperature;
    int systolic;
    int diastolic;
    int spo2;
    int heartRate;
    uint32_t timestamp;
} MeasurementData;

typedef struct{
    uint8_t buttonAState;
    uint8_t buttonBState;
    int joystickX;
    int joystickY;
} UserInputData;

typedef struct {
    char name[50];
    char email[100];
    uint32_t userId;
} UserData;

typedef struct{
    uint8_t ledStatus;
}

```

```
    uint8_t buzzerState;
} FeedbackData;
```

5.3.8 Organização da memória

A Raspberry Pi Pico W possui uma arquitetura de memória dividida em regiões distintas, cada uma com uma função específica. A memória flash, uma área não volátil de 2MB, armazena o código do firmware, as constantes e outros dados que precisam ser mantidos mesmo quando o dispositivo é desligado. O código do programa, compilado a partir do código C/C++, é armazenado nessa região, junto com as constantes e a tabela de vetores de interrupção, que contém os endereços das rotinas de tratamento de interrupção.

Já a memória RAM, uma área volátil de 264KB, é utilizada para o armazenamento de variáveis globais e locais, a pilha (stack), o heap e outros buffers de trabalho. As variáveis globais, declaradas fora de qualquer função, são armazenadas nesta região, assim como as variáveis locais, declaradas dentro de funções. A pilha, também localizada na RAM, é utilizada para armazenar os endereços de retorno das chamadas de função e as variáveis locais, enquanto o heap permite a alocação dinâmica de memória (malloc/new). Buffers de dados são também armazenados na RAM, incluindo buffers para a leitura dos sensores, o texto a ser exibido na tela OLED e os dados formatados para envio à nuvem. As variáveis que controlam o estado do sistema também se encontram nesta região de memória.

Os registradores dos periféricos, mapeados em endereços de memória específicos fornecidos pela documentação do Raspberry Pi Pico W, permitem o controle do comportamento de cada periférico. Por exemplo, os registradores GPIO controlam a função de cada pino, enquanto os registradores I2C e UART controlam a comunicação com seus respectivos dispositivos. A tabela de vetores de interrupção, geralmente localizada no início da memória flash, armazena os endereços das rotinas de tratamento de interrupção, que são executadas quando ocorre uma interrupção.

5.3.9 Protocolo de comunicação

O "Pico Health Hub" utiliza uma variedade de protocolos de comunicação para garantir a troca eficiente de dados entre seus componentes e sistemas externos. A comunicação com os sensores é feita através de protocolos específicos para cada dispositivo, enquanto a comunicação com o servidor web e o banco de dados na nuvem, utilizando o Firebase, emprega o protocolo HTTP sobre TCP/IP.

A comunicação com o sensor de temperatura DS18B20 é realizada através do protocolo OneWire, um protocolo de comunicação serial que utiliza um único fio para a troca bidirecional de dados. O microcontrolador envia um pulso de reset seguido por um comando para solicitar a leitura da temperatura, com o sensor respondendo enviando os dados em um formato predefinido. Já o sensor de pressão arterial utiliza o protocolo UART (Universal Asynchronous Receiver-Transmitter), transmitindo e recebendo dados de forma serial através de um par de fios (TX e RX). O microcontrolador precisa configurar a taxa de transmissão e outros parâmetros da UART para garantir a comunicação correta com o sensor. O sensor de oximetria MAX30102 utiliza o protocolo I2C (Inter-Integrated Circuit), que requer dois fios (SDA e SCL), com o microcontrolador atuando como mestre, enviando o endereço do

dispositivo e comandos para leitura dos valores de SpO2 e frequência cardíaca, que são enviados de volta pelo sensor para o microcontrolador.

A comunicação com o microfone é feita através da leitura dos dados do conversor analógico-digital (ADC), onde o microcontrolador configura o ADC para ler o sinal analógico, convertendo-o em um sinal digital para armazenamento. Por outro lado, o controle dos pinos GPIO, que acionam botões, o LED RGB, e o buzzer, é realizado através de comandos diretos, ativando ou desativando os componentes e enviando os dados necessários. O joystick também tem sua posição lida através do ADC.

Para a comunicação com o servidor web e o banco de dados na nuvem, Firebase, o "Pico Health Hub" emprega o protocolo HTTP sobre TCP/IP. Para enviar os dados para o Firebase, o dispositivo cria uma requisição HTTP utilizando o método POST, contendo os dados formatados em JSON (JavaScript Object Notation), junto com um cabeçalho com informações sobre o tipo de conteúdo (application/json) e o endereço do servidor. O servidor Firebase, por sua vez, envia uma resposta HTTP, contendo um código de status e, opcionalmente, outros dados. O formato JSON é utilizado para facilitar a troca e a análise dos dados, permitindo a representação de diversos tipos de informações de maneira leve e eficiente. A utilização de protocolos padronizados garante que o "Pico Health Hub" consiga se comunicar de forma eficiente com todos os seus componentes e sistemas externos, possibilitando a interoperabilidade e o bom funcionamento do dispositivo.

Em resumo, são utilizados protocolos específicos para cada tipo de comunicação, desde a leitura dos dados dos sensores até a transmissão dos dados para a nuvem, utilizando protocolos padronizados e eficientes, e garantindo o bom funcionamento do dispositivo.

5.3.10 Formato do pacote de dados

A comunicação com a plataforma na nuvem e o servidor web utiliza o formato de dados JSON (JavaScript Object Notation), um padrão leve e versátil que facilita a interpretação e o tráfego de dados em aplicações web. Os pacotes de dados são formatados em JSON para garantir a compatibilidade e a eficiência na comunicação com o Firebase. O pacote de dados enviado para a nuvem contém informações sobre o usuário, o momento da medição e os dados dos sensores. O pacote inclui um userId, que identifica o usuário, um timestamp, que representa a data e hora da medição em formato ISO 8601, os valores da temperatura corporal (temperature em float), da pressão sistólica (systolic em inteiro), da pressão diastólica (diastolic em inteiro), da saturação de oxigênio (spo2 em inteiro), da frequência cardíaca (heartRate em inteiro) e dos sons cardíacos/respiratórios (rawAudioData), codificados em base64, além do estado do sistema (systemState como string).

Os dados são formatados para um pacote JSON com a seguinte estrutura: `{"userId": "123456", "timestamp": "2025-01-25T12:00:00Z", "temperature": 37.2, "systolic": 120, "diastolic": 80, "spo2": 98, "heartRate": 75, "rawAudioData": "base64encodedstring", "systemState": "IDLE" }.`. Para o envio dos dados, o firmware coleta todos os dados, formata-os em JSON e utiliza o método HTTP POST para o envio ao servidor Firebase, incluindo o pacote formatado no corpo da requisição e um cabeçalho específico definindo o tipo do conteúdo ("Content-Type": "application/json"). A resposta do servidor é recebida e interpretada para garantir que o processo de envio foi realizado com sucesso.

Adicionalmente, para o envio de dados dos usuários, como nome e e-mail, o formato JSON também é utilizado com a seguinte estrutura `{"userId": "123456", "name": "Felipe", "email": "felipe@teste.com"}`, onde cada campo contém um valor correspondente a cada informação do usuário. A estrutura do pacote de dados é flexível, permitindo a adição de outros campos no futuro, para a inclusão de novas informações e funcionalidades. O uso do formato JSON facilita a integração com a plataforma na nuvem, garantindo a padronização da troca de dados e a interoperabilidade entre os sistemas.

A padronização do formato dos pacotes de dados, especialmente com o uso do formato JSON, proporciona a organização e a eficiência necessárias para que a comunicação com a plataforma na nuvem ocorra de forma confiável e segura.

5.4 Inicialização, programação e depuração na IDE

O desenvolvimento do firmware para o "Pico Health Hub" utilizando o Visual Studio Code (VS Code) e o SDK do Raspberry Pi Pico W envolve uma série de etapas que vão desde a configuração do ambiente de desenvolvimento até o teste e validação do sistema. Inicialmente, o VS Code deve ser instalado, seguido da instalação do SDK do Raspberry Pi Pico W, que inclui as bibliotecas, as ferramentas de compilação e outros arquivos necessários para desenvolver para a placa. A instalação de extensões no VS Code, que oferecem suporte ao SDK, CMake e ferramentas de depuração, também é crucial para otimizar o processo de desenvolvimento. Além disso, o toolchain precisa ser configurado corretamente, definindo onde o VS Code pode encontrar o compilador, o linker e outros componentes do SDK.

A criação do projeto envolve a organização dos arquivos em uma estrutura lógica, com diretórios separados para código fonte, cabeçalhos, bibliotecas, arquivos compilados, e um arquivo CMakeLists.txt para definir o processo de compilação. A programação do firmware é feita em C/C++, utilizando as bibliotecas do SDK para interagir com o hardware e os protocolos de comunicação, com funções separadas para cada componente. O CMake é utilizado para configurar e gerar o arquivo .uf2.

A depuração é feita com o uso de pontos de interrupção (breakpoints) para pausar a execução em pontos específicos do código e analisar o valor das variáveis. O debugger do VS Code possibilita depurar o código em tempo real, conectado à placa Raspberry Pi Pico W, e, juntamente com ferramentas como osciloscópios, analisadores lógicos e multímetros, é possível inspecionar os circuitos eletrônicos. O envio do firmware para a placa é feito colocando a Raspberry Pi Pico W em modo BOOTSEL, arrastando e soltando o arquivo .uf2 na unidade de armazenamento removível, ou selecionando "RUN" na extensão da raspberry pi pico no VS Code.

A fase de testes e validação inclui testes de unidade para cada componente do firmware, testes de integração para validar a interação entre os componentes e testes de sistema para validar o comportamento geral. Um repositório foi criado no github (<https://github.com/FelipeOlliver/PicoHealthHub>) para facilitar o gerenciamento do código e a colaboração com outros desenvolvedores. A documentação do SDK é consultada para entender as bibliotecas e APIs disponíveis, o debugger do VS Code é utilizado para acompanhar o fluxo de execução, e o uso de testes garante a qualidade do firmware.

O processo de desenvolvimento do firmware do "Pico Health Hub" utiliza o VS Code com o SDK do Raspberry Pi Pico W como ferramentas principais, com uma combinação de configurações precisas, codificação cuidadosa, depuração detalhada e testes abrangentes, assegurando a qualidade e o bom funcionamento do sistema.

5.5 Código

O código do software/firmware de execução da aplicação do "Pico Health Hub" encontra-se integralmente disponível no repositório GitHub (<https://github.com/FelipeOlliver/PicoHealthHub/tree/main/Code>). Este capítulo não visa apresentar o código em sua totalidade, mas sim informar algumas características da sua arquitetura e as práticas de codificação adotadas para garantir a qualidade, a legibilidade e a capacidade de manutenção do projeto, visto que nos capítulos anteriores o software já foi especificado mais minuciosamente.

Visando a organização e a modularização do código, a estrutura do projeto foi dividida em diretórios distintos: src/, para os arquivos de código fonte (.c e .cpp), e include/, para os arquivos de cabeçalho (.h) com as declarações de funções e estruturas. Essa separação facilita a navegação pelo código e o entendimento da sua estrutura geral.

Além disso, foram utilizadas diversas técnicas de programação para garantir a qualidade do código, como a criação de estruturas de dados (structs) para agrupar informações relacionadas, o uso de comentários para documentar as funções e os algoritmos, a aplicação de convenções de nomenclatura consistentes e a adoção de práticas de programação defensiva para evitar erros e comportamentos inesperados. O objetivo foi criar um código claro, conciso e fácil de entender, que possa ser facilmente modificado e expandido no futuro, seguindo as boas práticas de desenvolvimento de software.

6. Testes de Validação

Para os testes realizados no 'Pico Health Hub' foram avaliados seu consumo de energia, conectividade, funcionalidade completa e usabilidade. Serão descritos os testes de consumo de energia, que visam quantificar o gasto energético do dispositivo em diferentes estados de operação, os testes de conectividade, que verificam a capacidade do sistema de se conectar e transmitir dados para servidores web e plataformas na nuvem, e os testes com usuários, que avaliam a funcionalidade completa do 'Pico Health Hub' em cenários reais de utilização, incluindo entrevistas para coletar informações sobre a usabilidade do dispositivo.

6.1 Teste de consumo

O objetivo deste teste é quantificar o consumo de energia do "Pico Health Hub" em diferentes estados de operação, permitindo avaliar sua eficiência energética e estimar a duração da bateria em uso contínuo. Para realizar este teste, utilizamos um multímetro digital de alta precisão, que permitirá medir a corrente consumida pelo dispositivo, e uma fonte de alimentação ajustável, que garantirá um fornecimento de energia estável e controlado.

O procedimento de teste consistirá em conectar o "Pico Health Hub" à fonte de alimentação e utilizar o multímetro para medir a corrente consumida pelo dispositivo em diferentes estados: estado ocioso (sem nenhuma medição em andamento), durante a medição dos sinais vitais (temperatura, pressão arterial e SpO2) e durante a transmissão dos dados para a plataforma na nuvem através da conexão Wi-Fi. As

medições serão registradas em uma tabela para posterior análise, permitindo identificar os estados de operação que consomem mais energia e avaliar o desempenho energético geral do dispositivo.

6.2 Teste de conectividade

O objetivo deste teste é verificar a capacidade do "Pico Health Hub" de estabelecer uma conexão estável e confiável com a rede Wi-Fi e de transmitir dados para o servidor web e a plataforma Firebase. Para realizar este teste, utilizaremos um roteador Wi-Fi com acesso à internet, que simulará o ambiente de rede do usuário, e um computador com acesso ao servidor web e ao Firebase, permitindo verificar a recepção dos dados enviados pelo dispositivo.

O procedimento de teste consistirá em configurar o "Pico Health Hub" para se conectar à rede Wi-Fi disponível, utilizando as configurações de rede previamente definidas. Em seguida, o dispositivo será instruído a iniciar o processo de medição e transmissão dos dados para o servidor web e o Firebase, simulando o envio das informações de saúde do usuário para a plataforma na nuvem. A verificação do sucesso da conexão e da transmissão dos dados será realizada através do monitoramento das atividades de rede no roteador Wi-Fi e da verificação da recepção dos dados no servidor web e no Firebase. O tempo de conexão e o sucesso da transmissão serão registrados para avaliar o desempenho da conectividade do "Pico Health Hub".

6.3 Testes de execução e usabilidade

O objetivo deste teste é avaliar a funcionalidade completa do "Pico Health Hub" em um cenário de uso real, envolvendo a participação de diferentes usuários com perfis distintos. Para tanto, selecionamos três participantes: um homem de 30 anos hipertenso, uma mulher de 28 anos com pressão baixa e um homem de 25 anos com pressão arterial regular. Cada participante utilizará o "Pico Health Hub" para realizar uma medição completa dos sinais vitais, seguindo as instruções fornecidas no dispositivo. Durante o teste, serão registradas as medições realizadas por cada participante, incluindo a temperatura corporal, a pressão arterial (sistólica e diastólica) e a saturação de oxigênio (SpO2).

Além disso, após a utilização do dispositivo, cada participante será entrevistado utilizando um questionário semiestruturado, com o objetivo de obter feedback sobre a usabilidade, a facilidade de uso, o conforto e as sugestões de melhoria para o "Pico Health Hub". As respostas dos participantes serão organizadas e analisadas para identificar padrões e tendências, fornecendo informações valiosas para o aprimoramento do dispositivo.

Perguntas:

- Em uma escala de 1 a 5 (1 sendo "nada confortável" e 5 sendo "extremamente confortável"), quanto confortável você se sentiu utilizando o "Pico Health Hub"?
- Em uma escala de 1 a 5 (1 sendo "muito difícil" e 5 sendo "muito fácil"), quanto fácil você achou de utilizar o "Pico Health Hub"?
- Em uma escala de 1 a 5 (1 sendo "nada intuitivo" e 5 sendo "extremamente intuitivo"), quanto intuitivo você achou o "Pico Health Hub"?
- Qual a sua opção favorita da plataforma?

- Que sugestões você daria para melhorar o "Pico Health Hub"?

7. Resultados

Este capítulo apresenta os resultados obtidos através da aplicação da metodologia e dos testes detalhados no capítulo anterior. Serão apresentados e analisados os dados referentes ao consumo de energia do dispositivo em diferentes estados de operação, à conectividade com a rede Wi-Fi e com a plataforma Firebase, e às medições realizadas com a participação de diferentes usuários, juntamente com o feedback coletado através de entrevistas.

A análise dos resultados permitirá avaliar o desempenho, a precisão e a usabilidade do 'Pico Health Hub', fornecendo informações valiosas para a validação dos objetivos do projeto e para a identificação de oportunidades de melhoria.

7.1 Resultado do teste de consumo

Estado	Componentes Ativos	Corrente (mA)	Notas
Ocioso	Raspberry Pi Pico W (mínimo)	10-20	Sem sensores ativos, Wi-Fi em modo de baixo consumo.
Medição (Temp)	Raspberry Pi Pico W, Sensor DS18B20	25-35	Sensor DS18B20 ativo durante a leitura.
Medição (Pressão)	Raspberry Pi Pico W, Sensor de Pressão (via UART)	30-40	Sensor de pressão arterial e comunicação UART ativos.
Medição (SpO2)	Raspberry Pi Pico W, Sensor MAX30102 (via I2C)	40-60	Sensor MAX30102 e comunicação I2C ativos
Medição (Microfone)	Raspberry Pi Pico W, Microfone (via ADC)	25-35	Microfone e ADC ativos durante a amostragem de áudio.
Transmissão Wi-Fi	Raspberry Pi Pico W, Módulo Wi-Fi	80-120	Módulo Wi-Fi em transmissão ativa (consumo depende da intensidade do sinal e da taxa de transferência).

Em espera	Raspberry Pi Pico W, Display OLED	10-20	Display OLED ligado mas exibindo uma tela escura
-----------	-----------------------------------	-------	--

Tabela 1. Testes de consumo. Fonte: Autor

O sistema é alimentado por uma bateria de 3.7V com capacidade de 2000 mAh, o que equivale a 7.4 Wh de energia armazenada. Considerando um consumo médio de 50 mA, que representa um uso misto do dispositivo em diferentes estados de operação (ocioso, medição e transmissão), a duração estimada da bateria é de aproximadamente 40 horas.

7.2 Resultado do teste de conectividade

Para o teste de conectividade do 'Pico Health Hub', foi utilizada uma rede Wi-Fi 802.11n de 2.4 GHz, localizada a 3 metros de distância sem obstáculos. Uma vez conectado, e realizados os testes de medição, foi possível o acesso e monitoramento através do webserver e também a verificação dos dados no firebase.

The screenshot displays the 'Pico Health Hub' web application. It has three main sections: 'Dados do Usuário' (User Data) showing ID 000001, Name Felipe Oliveira, and Email felipedeoliveiragomes01@gmail.com; 'Dados da Medição' (Measurement Data) showing Date/Hour 2025-02-03 10:30:00, Temperature 36.8 °C, Systolic Blood Pressure 150 mmHg, Diastolic Blood Pressure 90 mmHg, SpO2 98 %, Heart Rate 85 BPM, and System Status At Rest; and 'Ações' (Actions) with 'Exportar Dados' and 'Atualizar Dados' buttons.

Figura 27. Página do webserver. Fonte: Autor

Figura 28. Banco de dados Pico Health Hub firebase . Fonte: Autor

Resultados de conectividade:

Teste	Métrica	Resultado	Unidade
Conexão Wi-Fi	Tempo de Conexão	3-5	segundos
Transmissão para Servidor/Firebase	Taxa de Transmissão	100-200	KB/s
Transmissão para Servidor/Firebase	Sucesso da Transmissão	100	%
Requisição para Servidor	Latência	50-150	ms

Tabela 2. Testes de conectividade. Fonte: Autor

7.2 Resultados de execução e usabilidade

Para avaliar o desempenho do dispositivo em um cenário de uso real e coletar informações valiosas sobre a sua usabilidade, foram realizados testes de execução do programa completo com a participação de três usuários com perfis distintos: um homem de 30 anos hipertenso, uma mulher de 28 anos com pressão baixa e um homem de 25 anos com pressão arterial regular. Além de realizar as medições dos sinais vitais, os participantes foram entrevistados para fornecer feedback sobre o conforto, a facilidade de uso e a intuitividade do dispositivo, bem como sugestões de melhorias para o 'Pico Health Hub'. Os resultados destes testes serão apresentados e analisados a seguir, com o objetivo de identificar os pontos fortes e fracos do dispositivo e validar o seu potencial para o autocuidado da saúde.

Participante	Temperatura (°C)	Pressão Sistólica (mmHg)	Pressão Diastólica (mmHg)	SpO2 (%)	Frequência Cardíaca (BPM)	Tempo (s)
Felipe O.	36.8	150	90	98	85	65
Lizandra O.	36.7	100	70	98	82	58
Stelme S.	36.5	120	80	96	78	62

Tabela 3. Testes de execução. Fonte: Autor

Participante	Conforto	Facilidade	Intuitividade	Preferência	Sugestões
Felipe O.	4	4	5	Portabilidade	"Um estojo para guardar o aparelho seria útil."
Lizandra O.	5	5	5	Leitura Rápida	"Achei tudo muito fácil e intuitivo."
Stelme Souza	4	4	4	Display	"Gostaria de um histórico das minhas medições na tela."

Tabela 4. Testes de usabilidade. Fonte: Autor

REFERÊNCIAS

KROGSBØLL, Lasse T; JØRGENSEN, Karsten Juhl; GØTZSCHE, Peter C. General health checks in adults for reducing morbidity and mortality from disease. Cochrane Database Syst Rev. 2019;2019(1). PMC - PubMed. Disponível em: <https://PMC.ncbi.nlm.nih.gov/articles/PMC6353639>. Acesso em: 17 janeiro 2025

Silva DO. Uso de aparelhos eletrônicos por idosos em ambientes domésticos [dissertação]. São Paulo: Escola de Engenharia de São Carlos; Faculdade de Medicina de Ribeirão Preto; Universidade de São Paulo, Instituto de Química de São Carlos; 2011.

SANTANA MP, BERNARDES MS, RAYMUNDO TM, SANTANA CS. Instrumentalization program of elderly for the use of health care facilities in the domestic environment. Gerontechnology 2012;11(2):193.

Prefeitura Municipal de São Paulo, Secretaria Municipal da Saúde. Programa de Automonitoramento Glicêmico [Internet]. São Paulo: Prefeitura de São Paulo; 2023 Disponível em: <http://www.prefeitura.sp.gov.br/cidade/secretarias/saude/programas/index.php?p=6070#topo>. Acesso em: 17 janeiro 2025

Agência Nacional de Vigilância Sanitária . Boletim Informativo: Segurança do Paciente e Qualidade em Serviços de Saúde [Internet]. Brasília, DF: ANVISA; 2024 Disponível em: [http://portal.anvisa.gov.br/wps/wcm/connect/f72c20804863a1d88cc88d2bd5b3ccf0/B
OLETIM+I.PDF?MOD=AJPERES](http://portal.anvisa.gov.br/wps/wcm/connect/f72c20804863a1d88cc88d2bd5b3ccf0/BOLETIM+I.PDF?MOD=AJPERES). Acesso em: 17 janeiro 2025

VILELA R. Conheça sete inovações tecnológicas para monitorar sua saúde [Internet]. 2018 Disponível em: [http://www\[minhavida.com.br/saude/galerias/13164-conheca-sete-inovacoes-tecnologicas-para-monitorar-sua-saude](http://www[minhavida.com.br/saude/galerias/13164-conheca-sete-inovacoes-tecnologicas-para-monitorar-sua-saude). Acesso em: 17 janeiro 2025

İLHAN, İlhan et al. Development of a wireless blood pressure measuring device with smart mobile device. Comput. Methods Programs Biomed (2016). DOI <https://doi.org/10.1016/j.cmpb.2015.11.003>. Disponível em: <https://www.sciencedirect.com/science/article/abs/pii/S0169260715002965#preview-section-abstract>. Acesso em: 17 janeiro 2025

NEMOMSSA, Hundessa Daba et al. Development of Low-Cost and Portable Pulse Oximeter Device with Improved Accuracy and Accessibility. Cochrane Database Syst Rev. 2022;2022(1). Disponível em: <https://PMC.ncbi.nlm.nih.gov/articles/PMC9084508/>. Acesso em: 17 janeiro 2025

I. K. A. A. Aryanto, I. P. Wijaya, I. N. R. Hendrawan and K. Y. E. Aryanto, "A Prototype IoT based Technology for Body Temperature Monitoring," 2021 3rd International Conference on Cybernetics and Intelligent System (ICORIS), Makasar, Indonesia, 2021, pp. 1-6, doi: 10.1109/ICORIS52787.2021.9649631. Disponível em: <https://ieeexplore.ieee.org/document/9649631>. Acesso em: 17 janeiro 2025

REDDY, Revanth, MARJORIE, Roji. Design of an IOT Based Online Monitoring Digital Stethoscope. International Journal of Advances in Applied Sciences (2018). DOI: <http://doi.org/10.11591/ijaas.v7.i3.pp240-244>. Disponível em: <https://ijaas.iaescore.com/index.php/IJAAS/article/view/13521>. Acesso em: 18 janeiro 2025

DURRANT, Jon. Demonstrate OLED SSD1306 Display using Raspberry PI Pico. (2024). Github.com. Disponível em: <https://github.com/jondurrant/RPIPicoSSD1306Exp/tree/main>. Acesso em: 18 janeiro 2025

Raspberry PI Pico SDK. Raspberry Pi RP2350 Pico SDK Examples. Github.com (2024). Disponível em: <https://github.com/raspberrypi/pico-examples>. Acesso em: 18 janeiro 2025

STYGER, Erich. BLE with WiFi and FreeRTOS on Raspberry Pi Pico-W. MCU on eclipse 2023. Disponível em: <https://mcuoneclipse.com/2023/03/19/ble-with-wifi-and-freertos-on-raspberry-pi-pico-w/>. Acesso em: 18 janeiro 2025

SIMMONS, Geoff. picow-http-example. (2022). Gitlab.com. Disponível em: <https://gitlab.com/slimhazard/picow-http-example>. Acesso em: 19 janeiro 2025

SIMMONS, Geoff. picow_http (2022). Gitlab.com. Disponível em: https://gitlab.com/slimhazard/picow_http. Acesso em: 19 janeiro 2025

SHILLEH, Mahmood M. Effortless Data Storage: MongoDB Database and Raspberry Pi Pico W Walkthrough (2023). Shillehtek.com. Disponível em: https://shillehtek.com/blogs/news/effortless-data-storage-mongodb-database-and-raspberry-pi-pico-w-walkthrough-part-1?utm_source=youtube&utm_medium=product_shelf. Acesso em: 19 janeiro 2025

ZENG, Xuyun. Build a cloud Raspberry Pi Pico W temperature/humidity logger (2023). Xyzcreativeworks.com. Disponível em: <https://xyzcreativeworks.com/build-a-cloud-raspberry-pi-pico-w-temperature-humidity-logger/>. Acesso em: 19 janeiro 2025

BOARDMAN, Adam. Raspberry Pi Pico One Wire Library (2023). Github.com. Disponível em: <https://github.com/adamboardman/pico-onewire>. Acesso em: 19 janeiro 2025

EDSTRÖM, Linnéa. MAX30102 pulse oximeter library (2016). Os.mbed.com. Disponível em: <https://os.mbed.com/users/Filea/code/MAX30102/file/02f411fefe6f/max30102.h>. Acesso em: 19 janeiro 2025

VINARY, Y. N. Blood Pressure Sensor Interfacing with Raspberry Pi (2021). Hackster.io. Disponível em: <https://www.hackster.io/vinayyn/blood-pressure-sensor-interfacing-with-raspberry-pi-970f82>. Acesso em: 19 janeiro 2025

BHARATH, S. et al. Wireless transmission of data, body temperature, condenser microphone, auscultation (2021). G.M. Institute of Technology, Davanagere. Disponível em: https://www.kscest.org.in/spp/45_series/SPP45S/02_Exhibition_Projects/229_45S_BE_3229.pdf. Acesso em: 19 janeiro 2025

COLLINS, Austin et al. Smart Digital Stethoscope (2022). Yowa State University. Disponível em: <https://sdmay22-18.sd.ece.iastate.edu/docs/Final-draft.pdf>. Acesso em: 19 janeiro 2025