



Curso de Bacharelado em Engenharia de Computação
Disciplina: Sistemas Embarcados
Docente: Alexandre Sales Vasconcelos

Projeto Sistemas Embarcados 2023.1:
Monitoramento da temperatura e nível de água de reservatório com uso
do ESP32

Arthur Mauricio Thomaz Soares
Daniel da Silva Lemos
Jonatas da Silva Duarte

Campina Grande, 2023

Introdução

O projeto proposto tem como objetivo a construção de um controlador de temperatura e nível de água de um recipiente a partir do ESP32. O sistema será composto por um conjunto de sensores, para medição da temperatura da água e do nível do reservatório, além de atuadores para controlar a resistência e a bomba de água. Para torná-lo mais interativo e fácil de usar, uma interface homem-máquina (IHM) será desenvolvida para permitir ao usuário monitorar e controlar a aplicação em tempo real.

O hardware escolhido para implementação é o ESP32, um microcontrolador baseado em um processador dual-core Xtensa LX6 de 32 bits, que suporta conexão Wi-Fi e Bluetooth, possuindo recursos poderosos de processamento e conectividade, o que o torna ideal para projetos de IoT. Já no desenvolvimento do software, será utilizado o ambiente de desenvolvimento ESP-IDF (Espressif IoT Development Framework), uma plataforma de desenvolvimento oficial do fabricante do ESP32. O ESP-IDF inclui bibliotecas e ferramentas que simplificam o processo de desenvolvimento e permitem a criação de projetos robustos e eficientes.

Para garantir a qualidade do projeto, serão adotadas a divisão de todo o trabalho em etapas, tais quais: levantamento de requisitos para a definição de caso de uso, desenvolvimento deste relatório para descrever o projeto, desenvolvimento do hardware e software, testes e validação do sistema e, por fim, a entrega dos resultados. Com a conclusão do projeto, espera-se fornecer uma solução eficiente e confiável para o monitoramento e controle da temperatura e nível da água em um reservatório com o uso do ESP e podendo ser facilmente utilizada pelo usuário final.

Objetivos gerais do projeto

1. Desenvolver um sistema eficiente para controlar e monitorar a temperatura e nível d'água;
2. Proporcionar ao usuário o acesso às informações em tempo real sobre a temperatura da água e nível do reservatório;
3. Proporcionar ao usuário a capacidade de configurar as preferências do sistema, como a temperatura desejada e a frequência da bomba de água.

Objetivos específicos

1. Implementar a lógica de controle de temperatura, garantindo que a temperatura da água seja mantida dentro de uma faixa desejada pelo usuário;
2. Implementar a lógica de controle da bomba de água, garantindo que a água do reservatório aumente até o nível estipulado e pare caso atinja-o;
3. Implementar uma IHM através de um display LCD para monitoramento dos dados em tempo real;
4. Documentar todas as etapas do projeto, desde a seleção de componentes até a implementação do software, e disponibilizar essas informações em um repositório no GitHub para a disciplina.

Casos de uso

1. Medir e salvar temperatura da água no reservatório

Descrição: A temperatura da água dentro do reservatório deve ser medida com certa precisão pelo sensor e ser armazenada no cartão SD.

Requisitos funcionais:

- Medir a partir de sensores a temperatura
- Criar uma estrutura de dados que armazene a temperatura da água
- Armazenar esses dados em algum arquivo no cartão SD

Requisitos não funcionais:

- Registrar erro se houver algum problema pra salvar os dados no cartão SD
- Gerar um alarme de falha caso o erro no cartão SD persista

Atores:

- Cartão SD e sensores

Pré condições: nenhuma

Pós condições: o cartão SD deve conter mais uma entrada da temperatura de água no reservatório naquele instante de tempo

2. Medir e salvar nível da água no reservatório

Descrição: O nível da água dentro do reservatório deve ser medido pelo sensor e armazenado no cartão SD.

Requisitos funcionais:

- Medir a partir de sensores o nível de água
- Criar uma estrutura de dados que armazene o nível da água
- Registrar esses dados no cartão SD

Requisitos não funcionais:

- Registrar erro se houver algum problema pra salvar os dados no cartão SD
- Gerar um alarme de falha caso o erro no cartão SD persista

Atores:

- Cartão SD e sensores

Pré condições: nenhuma

Pós condições: o cartão SD deve conter mais uma entrada do nível da água no reservatório naquele instante de tempo

3. Tratamento de valores fora da faixa de variação

Descrição: Devemos tratar valores fora da faixa de variação dos sensores ao registrar os valores de temperatura e nível.

Requisitos funcionais:

- A partir de testes encontrar uma taxa de variação ideal para os sensores.
- Ao fim do registro de temperatura ou nível de água checar se os valores estão dentro da taxa de variação do sensor.
- Desconsiderar valores fora da taxa de variação e registrar isso em um arquivo de LOG

Requisitos não funcionais:

- Nenhum

Atores:

- Cartão SD

Pré-condições: Sensores funcionando e registrando temperatura e nível da água.

Pós condições: Se o valor medido estiver fora da taxa de variação eles devem ser desconsiderados, senão eles devem ser registrados normalmente.

4. Mostrar as informações salvas numa IHM

Descrição: Os valores registrados de temperatura e nível de água devem ser mostrados em um display.

Requisitos funcionais:

- Ler os dados do cartão SD e mostrar um histórico da temperatura e nível da água
- Ter alguma opção de atualizar os dados
- Mostrar essas informações em um display
- Mostrar médias dos valores

Atores:

- Cartão SD e IHM

Pré condições: Valores registrados no cartão SD e display funcionando

Pós condições: Display mostrando as informações citadas acima.

5. Registrar temperatura desejada a partir de uma IHM

Descrição: A temperatura desejada do reservatório deve ser registrada pelo usuário para que o sistema possa esquentar (ou não) a resistência.

Requisitos funcionais:

- Registrar a temperatura desejada no cartão SD

- Mostrar os dados de forma legível e explicativa
- Mostrar comparação em tempo real com os dados atuais

Requisitos não funcionais:

- Ter uma interface clara e de fácil acesso
- Registrar erro se houver algum problema pra salvar os dados no cartão SD

Pré condições: Display funcionando.

Pós condições: Um valor de referência de temperatura deve estar salvo no cartão SD para futuros ajustes.

Atores:

- Cartão SD, IHM

6. Registrar nível desejado a partir de uma IHM

Descrição: O nível desejado do reservatório deve ser registrado pelo usuário para que o sistema possa (ou não) ativar a bomba.

Requisitos funcionais:

- Registrar a o nível de água desejado no cartão SD
- Mostrar os dados de forma legível e explicativa
- Mostrar comparação em tempo real com os dados atuais

Requisitos não funcionais:

- Ter uma interface clara e de fácil acesso
- Registrar erro se houver algum problema pra salvar os dados no cartão SD

Pré condições: Display funcionando.

Pós condições: Um valor de referência de nível deve estar salvo no cartão SD para futuros ajustes.

Atores:

- Cartão SD, IHM

7. Modificar a temperatura da água

Descrição: A resistência deve esquentar se a temperatura atual do reservatório estiver menor do que a temperatura de referência registrada. Se não houver temperatura desejada registrada, a resistência não deve fazer nada.

Requisitos funcionais:

- Ler dados do cartão SD
- Fazer a resistência esquentar a água até o alvo baseado em parâmetros no cartão SD

Requisito não funcional:

- A temperatura atual sempre deve estar próxima a temperatura desejada.
- Registrar possíveis erros na leitura ou execução da ação

Atores:

- Cartão SD, sensores e resistência

Pré-condições: Temperatura de referência registrada, resistência funcionando e sensor funcionando.

Pós condições: Resistência funcionando baseado na temperatura atual e na temperatura de referência.

8. Modificar o nível da água

Descrição: A bomba deve ser ativada se o nível de água estiver abaixo do desejado, e ser desativada quando o reservatório estiver em um nível aceitável. Se não houver valor de referência registrado no cartão SD devemos ter algum valor padrão.

Requisitos funcionais:

- Ler dados do cartão SD
- A bomba deve ativar caso o reservatório esteja com um nível de água baixo
- A bomba deve ser desativada quando o reservatório estiver em um nível aceitável

Requisitos não funcionais

- O reservatório não pode transbordar nem ficar com um nível muito abaixo do desejado

Atores

- Cartão SD, sensor de capacidade e bomba

Pré-condições: Sensores funcionando, bomba funcionando e valor de referência salvo disponível.

Pós condições: Nível de água próximo ao nível desejado.

9. Sinalização de falha

Descrição: Devemos ter uma métrica de confiabilidade do nosso dispositivo. Um LED irá mostrar se o dispositivo (sensores, bomba, resistência e cartão SD) estão conseguindo executar suas funções.

Requisitos funcionais:

- Sinalizar em algum LED (vermelho) se o funcionamento do dispositivo está comprometido ou em verde se ele está funcionando corretamente
- Registrar LOGs que facilitem o debug do que deu errado no cartão SD

Atores:

- Cartão SD e todo o dispositivo

Pré condições: Nenhuma

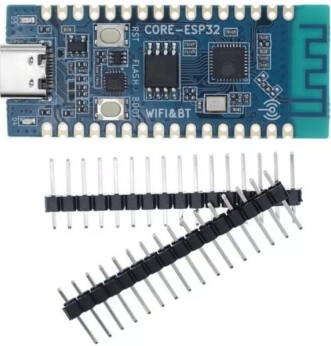
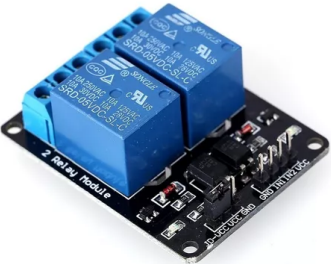


Pós condições: Estado do LED atualizado


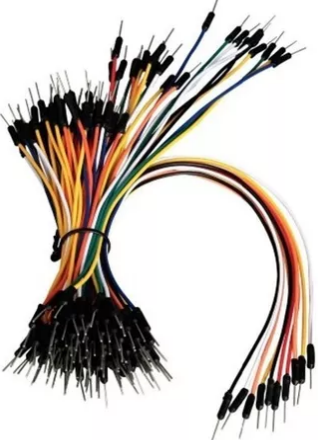


Materiais e Métodos


- **Materiais**

Lista dos componentes/materiais usados para o protótipo do projeto:

Imagem	Nome	Descrição	Quantidade	Datasheets
--------	------	-----------	------------	------------

	Esp32 C3	Placa De Desenvolvimento Esp32 C3 Wifi Bluetooth Micropython. É um microcontrolador de 32 bits, escolhido como o hardware central para este projeto.	1	https://www.espressif.com/sites/default/files/documentation/esp32-c3_datasheet_en.pdf
	Módulo Relé	Módulo Relé 3v 2 Canais Duplo. Será utilizado para controlar a resistência e a bomba de água do sistema.	1	https://www.elctruscomp.com.br/arquivos/1488543256_dados_tecnicos_p2_gbk_rele_duplo_10a.pdf
	Protoboard	Protoboard 400 Furos. Para a conexão entre os componentes eletrônicos do sistema, permitindo uma montagem organizada.	1	N/A
	Sensor De Temperatura	Sensor De Temperatura Ds18b20 Sonda À Prova D'água usada para medir a temperatura da água do reservatório.	1	https://www.elprocus.com/ds18b20-temperature-sensor/

	Sensor Ultrassônico	Sensor Ultrassônico De Distância Hc-sr04 usado para verificar o nível da água no reservatório.	1	https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf
	Jumper	Jumpers para facilitar a conexão entre os componentes eletrônicos do sistema: machos e fêmeas.	n/a	N/A
	Resistor	Resistores serão utilizados como parte do circuito, garantindo uma medição precisa e estável.	n/a	N/A
	Display LCD I2P com módulo I2C	Será utilizado como interface homem-máquina (IHM), permitindo ao usuário monitorar os dados em tempo real. O módulo auxilia na pinagem e facilita no código.	1	https://circuitdigest.com/article/16x2-lcd-display-module-pinout-datasheet

	Joystick analógico 3D	Será usado para fornecer ao usuário uma forma de interagir e controlar diferentes aspectos do sistema de forma precisa e personalizada.	1	https://components101.com/modules/joystick-module
---	-----------------------	---	---	---

Lista das bibliotecas e frameworks usados para o desenvolvimento do protótipo do projeto:

Nome	Descrição
ESP-IDF	Framework oficial da Espressif para o desenvolvimento de aplicações para toda a família ESP32.
stdio.h	Biblioteca de entrada/saída padrão em C, que fornece funções para leitura e escrita de dados em fluxos de entrada/saída.
FreeRTOS.h	Biblioteca que implementa um sistema operacional de tempo real (RTOS - Real-Time Operating System) de código aberto, usado para programação multitarefa em sistemas embarcados.
task.h	Biblioteca que fornece as funções e macros necessários para criar, controlar e gerenciar tarefas (threads) em um ambiente multitarefa usando o FreeRTOS.
gpio.h	Biblioteca que oferece funções para configurar e controlar as GPIOs em um microcontrolador, permitindo a leitura e escrita de sinais digitais em pinos específicos.
onewire.h	Biblioteca para comunicação usando o protocolo 1-Wire, que permite a transferência de dados serial utilizando apenas um fio de comunicação.

- **Métodos**

Na escolha dos componentes, consideramos o custo, a facilidade de compra e uso. Buscamos componentes com bom custo-benefício, facilmente disponíveis para compra e de fácil usabilidade no projeto. Em detalhes, temos:

- ESP32 C3: Foi escolhido como o padrão para o projeto, conforme orientação do professor.
- Módulo Relé: Atendeu aos requisitos de alimentação de 5V e oferecia os dois canais necessários.
- Protoboard: Optamos por uma protoboard de 400 furos, adequada ao tamanho necessário para a montagem.
- Sensor de Temperatura: Escolhemos um sensor à prova d'água para facilitar a instalação e uso no projeto.
- Sensor Ultrassônico: Optamos por um modelo de sensor ultrassônico devido à sua fácil utilização, entendimento e baixo custo.
- Jumpers: São necessários para realizar as conexões entre os componentes.
- Resistor: Utilizamos resistores como parte do circuito.
- Display LCD I2P: Escolhemos o display LCD I2P por sugestão do professor, pois sua aplicação é mais simples em comparação com outros tipos de displays.
- Joystick analógico 3D: Escolhemos o joystick como forma do usuário interagir e controlar o sistema, podendo alterar e setar valores para as medições de temperatura e distância.

Abaixo, temos a relação entre as conexões dos pinos GPIO do microcontrolador e os componentes que utilizamos para o projeto, permitindo a interação e o controle desses dispositivos:

1. DS_GPIO (GPIO03) - Este pino GPIO está conectado ao sensor de temperatura. Sendo responsável por receber os dados do sensor e fornecer informações sobre a temperatura captada.
2. TRIGGER_PIN (GPIO00) - Este pino GPIO está conectado ao sensor de distância. É usado como pino de disparo (trigger) para enviar um sinal ao sensor de distância, permitindo que ele realize a medição.
3. ECHO_PIN (GPIO01) - Este pino GPIO também está conectado ao sensor de distância. Sendo usado como pino de eco para receber o sinal de retorno do sensor de distância após ele ter realizado a medição. A duração desse sinal de retorno é usada para determinar a distância entre o sensor e o objeto em questão.
4. RELE_MOTOR_PIN (GPIO10) - Este pino GPIO está conectado ao primeiro módulo do relé. Sendo usado para ativar ou desativar o relé, ligando ou desligando o motor ou dispositivo correspondente.
5. RELE_RESIST_PIN (GPIO06) - Este pino GPIO está conectado ao segundo módulo do relé. Assim como o RELE_MOTOR_PIN, este pino também é usado para ativar ou desativar o relé, neste caso está controlando a resistência.
6. JOYSTICK_SW_PIN (GPIO12) - Este pino GPIO está conectado ao botão de joystick, que foi usado para interação dos valores a partir do display.
7. JOYSTICK_Y_PIN (GPIO02) - Este pino GPIO está conectado ao eixo Y do joystick, que é usado para detectar o movimento na direção vertical. Ele registra a posição da alavanca do joystick ao longo do eixo Y e envia essas informações ao ESP para que possam ser processadas e utilizadas no código.

Resultados e Discussões

Considerando o hardware desenvolvido junto ao código, assim como esta documentação, vamos falar como se deu os resultados de cada objetivo:

Objetivos gerais do projeto

- Desenvolver um sistema eficiente para controlar e monitorar a temperatura e nível d'água:
 - Atendido. A medição de temperatura é feita pelo sensor Ds18B20 para ser monitorada e o controle é feito pelo relé e resistência.
- Proporcionar ao usuário o acesso às informações em tempo real sobre a temperatura da água e nível do reservatório:
 - Atendido. Para esta etapa, utilizamos o display I2C, onde mostramos as informações de temperatura em C° e a porcentagem de nível do reservatório em tempo real. Também mostramos as informações pelo terminal.
- Proporcionar ao usuário a capacidade de configurar as preferências do sistema, como a temperatura desejada e a frequência da bomba de água:
 - Atendido. É possível que o usuário defina e altere valores preferenciais para temperatura e nível de água, utilizando o joystick. Escolhendo os valores mínimos de ativação da resistência e bomba, podendo também mudá-los em tempo real.

Objetivos específicos

- Implementar a lógica de controle de temperatura, garantindo que a temperatura da água seja mantida dentro de uma faixa desejada pelo usuário:
 - Atendido. Através do código, conseguimos manter a temperatura entre valores definidos pelo usuário fazendo uso da resistência.
- Implementar a lógica de controle da bomba de água, garantindo que a água do reservatório aumente até o nível estipulado e pare caso atinja-o:
 - Atendido. Através do código, conseguimos mensurar e manter o nível da água entre valores definidos pelo usuário fazendo uso da bomba d'água.

- Implementar uma IHM através de um display LCD para monitoramento dos dados em tempo real:
 - Atendido. Para esta etapa, utilizamos o display I2P, para monitoramento e controle das informações de temperatura e nível do reservatório em tempo real.
- Documentar todas as etapas do projeto, desde a seleção de componentes até a implementação do software, e disponibilizar essas informações em um repositório no GitHub para a disciplina:
 - Atendido. Toda a documentação e etapas foram dispostas adequadamente no GitHub da equipe. Lá dividimos os arquivos nas etapas de entrega, o código desenvolvido e este documento com todas as informações.

Conclusão

O projeto foi realizado durante a disciplina de Sistemas Embarcados, com uma duração aproximada de 3 meses. Desde o início, o grupo se empenhou em definir os passos e distribuir as atividades, visando garantir a qualidade das entregas. Para isso, adotamos a divisão do trabalho em etapas, que incluíram a seleção dos componentes para o hardware, levantamento de requisitos para definição dos casos de uso, desenvolvimento do software, realização de testes e validação do sistema. Ao longo dessas etapas, enfrentamos várias dificuldades, especialmente na escolha dos componentes, no entendimento do ambiente de desenvolvimento e, principalmente, na codificação do projeto em si. Questões como "Qual é o melhor sensor?", "Como utilizar as funções dessas bibliotecas?", "Como fazer essas conexões?" e etc, surgiram e nos guiaram para o entendimento. Apesar das dificuldades iniciais e da nossa falta de experiência, conseguimos superá-las e montar o hardware e desenvolver o software de medição de forma adequada e eficiente atendendo aos nossos objetivos iniciais.