**Destilador Inteligente**

Sérgio Manuel Figueira do Carmo

**TRABALHO DE FINAL DE CURSO**

Curso: **Engenharia Informática**

Orientador: Sérgio Correia

Ano Letivo: 2023 | 2024

Setembro | 2024

“Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam eget ligula eu lectus lobortis condimentum. Aliquam nonummy auctor massa. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Nulla at risus. Quisque purus magna, auctor et, sagittis ac, posuere eu, lectus. Nam mattis, felis ut adipiscing.”

Nome do autor (normal)*, Título do Trabalho (em Itálico)*.

*Lorem ipsum dolor sit amet*

**Agradecimentos**

.

**Resumo**

Este projeto, tem como finalidade modernizar um destilador. Este destilador, era completamente analógico, foi realizado uma modernização na máquina removendo-se todos os componentes antigos e substituídos por componentes digitais, tal como LEDs (*light emitting diode*), relê e um microprocessador chamado de ESP32.

Devido ao ESP32 ser um microprocessador potente, é assim possível ter o *firmware* e um servidor WEB em execução num único microprocessador.

O *firmware* controla todo o funcionamento da máquina, tal como a administração do tanque de água. O *firmware* é programado em linguagem de programação C

A página WEB, é uma interface intuitiva para controle e monitorização da máquina.

A página WEB é programada em linguagem de programação HTML, JavaScript e AJAX.

**Palavras-Chave**: Destilador, HTML, C, Arduino, HTML, JS, AJAX, ESP32.

**Abstract**

g.

**Keywords**: Lorem, Ipsum, Lorem, Ipsum, Lorem, Ipsum.

Lista de Abreviaturas, Siglas e Símbolos

AJAX - Asynchronous JavaScript and XML.

XML - Extensible Markup Language.

HTML - Hypertext Markup Language.

JS – JavaScript

CSS - Cascading Style Sheets

IDE - Integrated Development Environment

JSON - JavaScript Object Notation

SoC - System-on-Chip

SRAM - Static Random Acess Memory

LED – Light-emitting diode

AC - Alternating current

DC - Direct current

SPI - Serial Peripheral Interface

SRAM - *Static Random Acess Memory*

ÍNDICE GERAL

[1. Lista de Abreviaturas, Siglas e Símbolos vi](#_Toc174642018)

[2. ÍNDICE GERAL viii](#_Toc174642019)

[3. ÍNDICE DE ANEXOS ix](#_Toc174642020)

[4. ÍNDICE DE FIGURAS x](#_Toc174642021)

[5. ÍNDICE DE TABELAS xi](#_Toc174642022)

[1. INTRODUÇÃO 1](#_Toc174642023)

[2. Enquadramento e Justificação do Tema 1](#_Toc174642024)

[3. Requisitos Gerais e Específicos 1](#_Toc174642025)

[3.1. Requisitos Gerais 1](#_Toc174642026)

[3.2. Requisitos específicos 1](#_Toc174642027)

[4. Metodologia e Meios Utilizados 3](#_Toc174642028)

[6. Limitações da Pesquisa/Trabalho 3](#_Toc174642029)

[7. Estrutura Geral do Trabalho 3](#_Toc174642030)

[8. Hardware 5](#_Toc174642031)

[8.1. Arquitetura Geral 5](#_Toc174642032)

[8.2. Destiladora 6](#_Toc174642033)

[8.3. ESP32 6](#_Toc174642034)

[8.4. CPU e Memória 7](#_Toc174642035)

[8.5. Inputs/Outputs 7](#_Toc174642036)

[8.5.1. Inputs 8](#_Toc174642037)

[8.5.2. Outputs 8](#_Toc174642038)

[8.6. Circuito Elétrico 9](#_Toc174642039)

[9. CONCLUSÃO 11](#_Toc174642040)

[10. REFERÊNCIAS BIBLIOGRÁFICAS 13](#_Toc174642041)

[11. ANEXOS 14](#_Toc174642042)

[12. ANEXO 1 – “Circuito Elétrico” 14](#_Toc174642043)

[3. ANEXO 2 – “Instruções de Uso” 15](#_Toc174642044)

ÍNDICE DE ANEXOS

[Anexo 1 Circuito Elétrico 14](#_Toc174644360)

[Anexo 2 Instruções de Uso 15](#_Toc174644361)

ÍNDICE DE FIGURAS

[Figura 1 Arquitetura Geral da Destiladora 5](#_Toc174642045)

[Figura 2-Destiladora 6](#_Toc174642046)

[Figura 3 ESP32­WROVER­E 6](#_Toc174642047)

[Figura 4-GPIOs 8](#_Toc174642048)

[Figura 5 Substituir Circuito Elétrico 9](#_Toc174642049)

[Figura 6 Relês e ESP32 10](#_Toc174642050)

ÍNDICE DE TABELAS

**Não foi encontrada nenhuma entrada do índice de ilustrações.**

1. INTRODUÇÃO

Lorem, felis ut adipiscing.

1. Enquadramento e Justificação do Tema
2. Requisitos Gerais e Específicos
   1. Requisitos Gerais

1. Modo Automático

1.1. Liga/desliga a máquina com o temporizador

2. Modo Manual

2.1. Máquina sempre ligada em modo manual

3. Gestão de nível de água

3.1. Nível máximo desliga a bomba

3.2. Nível mínimo liga a bomba de água, liga resistência de calor e abre as válvulas

3.3. Nível de alarme desliga a resistência de calor, liga o led luminoso de alarme, fecha todas as válvulas

4. Controlo da destiladora via página WEB

* 1. Requisitos específicos

1. Uso do *ESP32 WROVER-E DevKitC V4 board*

1.1. Vantagens:

1.1.1. 26 I/O pins usáveis

1.1.2. WIFI integrado - 802.11n (2.4 GHz), up to 150 Mbps

1.1.3. CPU com 2 cores (core 0 corre *firmware* da máquina e core 1 o servidor WEB)

1.1.4. CPU trabalha a 40-MHz

1.1.5. *Firmware* desenvolvida em C

1.1.6. Arquitetura do CPU é *open* *source*, é possível programar ao nível do CPU se necessário, ao contrário se fosse usado, por exemplo, um raspberry pi

1.2. Desvantagens:

1.2.1. Memória para *firmware* + página WEB disponível muito limitada (8MB)

1.2.2. Por limitação de memória disponível no ESP32, velocidade de processamento e I/O disponíveis, apenas é possível fazer a página WEB em HTML usando CSS, e JavaScript, não havendo mais pins I/O disponíveis para ligar um módulo para cartão SD.

1.2.3. Limitação de CPU (40MHz)

1.3. Alternativas ao *ESP32 WROVER-E DevKitC V4*:

1.3.1. Raspberry pi

1.3.1.1. Uma opção boa seria usar um RP, teria muito mais memória e poder de processamento, mas os I/O não seriam suficientes (14 pins usáveis) para o projeto

1.3.2. Arduino Mega

1.3.2.1. O Arduino Mega tem muitos I/O (54 I/O), mas não tem WIFI e tem um CPU muito inferior (16MHz e 1 só core)

2. Página WEB

2.1. Página WEB vai ser desenvolvida em HTML, CSS, JavaScrip e AJAX

2.1.2. É usada a biblioteca “ESPAsyncWebServer.h” para criação e gestão de um servidor assíncrono WEB.

3. WIFI

3.1. É usada a biblioteca “WiFiManager.h” para gestão do WIFI

3.1.1. Com esta biblioteca, se não é conhecida a rede WIFI, ela cria um AP (*access point*) para o utilizador poder escolher, ligar e guardar as credenciais da nova rede WIFI na memória EEPROM e vai ler as credenciais quanda a máquina é ligada para se ligar à nova rede.

1. Metodologia e Meios Utilizados

O *firmware* da destiladora é programado em linguagem C, inclui os ficheiros de inclusão (ficheiros *header* com a extensão .h). A página *WEB* é programada em HTML com os respetivos ficheiros de CSS para adicionar estilos e os ficheiros JS, para executar JavaScript.

Todo o projeto foi elaborado no IDE *Visual Studio Code* com as extensões *PlatformIO IDE, GitHub Copilot* e *Doxygen.*

Todo o projeto foi também elaborado no sistema operativo Fedora 40 (Linux).

Limitações da Pesquisa/Trabalho

Não encontrei muitas dificuldades/limitações na pesquisa acerca do trabalho, a comunidade Arduino é muito grande e por consequência, há muita documentação disponível, talvez por haver tanta documentação, tenha havido uma certa dificuldade em filtrar o que realmente se adequasse ás minhas necessidades, tal como o método de comunicação (AJAX) entre o *firmware* e a página WEB.

Estrutura Geral do Trabalho

Na pasta raiz, estão as pastas “data”, “Documentation”, “include”, “lib” e “src”.

* data - Tem tudo o que é relacionado com a página de internet, tal como os ficheiros HTML, CSS e JS.
* Documentation – Contém tudo o que está relacionado com a documentação, tal como o relatório.
* Include – Contém todas os ficheiros *headers* necessários para o funcionamento do *software*.
* Lib – Esta pasta, contém as bibliotecas necessárias para o *software*.
* Src – Contém os ficheiros ‘.cpp’
* Na pasta raiz, estão ainda 2 ficheiros necessários para o funcionamento:
  + “partitions.csv”, que contém a configuração das partições da memória do ESP32.
  + “platformio.ini”, contém a configuração da extensão PlatformIO.

Hardware

## Arquitetura Geral

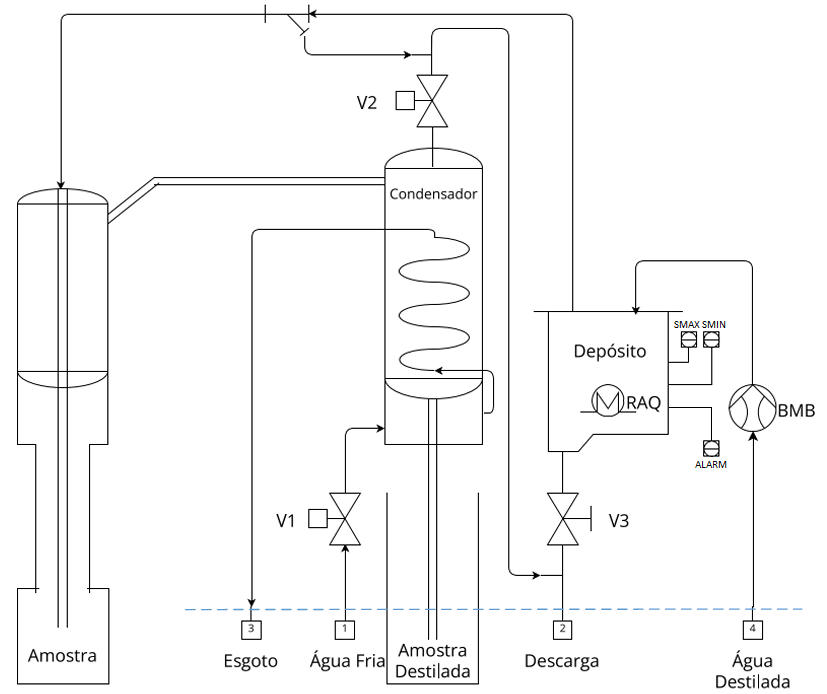


Figura 1 Arquitetura Geral da Destiladora

A Figura 1 representa o funcionamento da destiladora, em que há um depósito com água destilada em que é alimentado através de uma bomba de água e controlado com três sensores de nível de água, dentro do depósito há uma resistência de calor que vai evaporar a água destilada.

O vapor é controlado por uma válvula elétrica que direciona o vapor para destilar a amostra.

Há uma segunda válvula elétrica que abre para entrar água fria para o condensador, esta vai arrefecer o vapor vindo da amostra e condensar.

Há também uma terceira válvula que é manual, esta serve para despejar a água destilada existente no depósito.

## Destiladora

Uma imagem com máquina, texto, Eletrodoméstico, interior

Descrição gerada automaticamente

Figura 2-Destiladora

Foi usada a destiladora mostrada na Figura 2, era uma destiladora 100% analógica, temporizador, indicadores, botões, tudo era analógico operado a 250V, foi removido todos os componentes e trocados por componentes digitais, tal como releis, indicadores LED, um módulo para processamento ESP32 WROVER-E, fonte de alimentação de 250V AC para 5V DC.

## ESP32

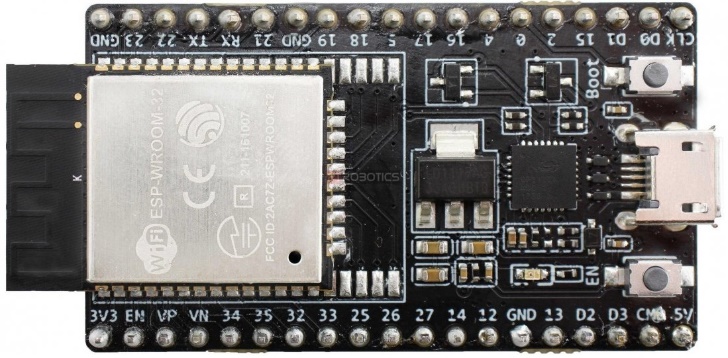


Figura 3 ESP32­WROVER­E

O módulo de desenvolvimento usado no projeto é um *ESP32 WROVER-E* [1], representado pela Figura 3, é um SoC (*System on a Chip*) desenvolvido pela empresa *fabless semiconductor company (Espressif Systems, SSE: 688018.SH)* [1]*.*

## CPU e Memória

O ESP32 tem um processador com 2 núcleos que podem ser controlados individualmente e a frequência pode ser ajustada a partir de 80MHz até 240MHz [2, p. 7], 520KB de SRAM (*Static Random Acess Memory*) e 4MB de memória FLASH.

O módulo tem também incorporado um módulo de Wi-fi e Bluetooth, sendo o Wi-Fi capaz de suportar os protocolos 802.11 b/g/n (802.11n até 150Mbps).

SRAM [3] é um tipo de memória volátil usada em sistema embedidos, esta memória usa circuitos *flip-flop* que assegura que os dados ficam presentes enquanto sejam alimentadas com energia elétrica. Este tipo de memória tem a vantagem de oferecer alta velocidade de acesso e ser energeticamente muito eficiente.

## Inputs/Outputs

O módulo tem 32 GPIO’s dos quais 19 são completamente usáveis como é constatado pela Figura 4, para melhor compatibilidade e fiabilidade, foram descartados os GPIO’s 0, 1, 3, 6-11, 34-39 por ou serem inputs e não outputs (GPIOs 3 e de 34 a 36 e 39) ou vice-versa (0, e 1) ou até mesmo, serem reservados para por exemplo, interação com *SPI flash* (GPIOs de 6 a 11) [2, p. 12].



Figura 4-GPIOs

### Inputs

Como inputs, foram usados os seguintes GPIOs declarados no *header* “OS.h”:

* GPIO 14 definido na constante “PIN\_SMAX” responsável pelo sensor de nível água
* GPIO 26 definido na constante “PIN\_SW\_MAN” responsável pelo Interruptor de modo manual;
* GPIO 25 definido na constante “PIN\_SALARM” responsável pelo sensor de nível de alarme.
* GPIO 27 definido na constante “PIN\_SMIN” responsável pelo sensor de nível água mínimo; máximo;
* GPIO 32 definido na constante “PIN\_SW\_AUTO” responsável pelo botão de modo auto;

### Outputs

Como outputs, foram usados os seguintes GPIOs declarados no *header* “OS.h”:

* GPIO 04 definido na constante “PIN\_IND\_MAX” responsável pelo indicador luminoso de nível água máximo;
* GPIO 05 definido na constante “PIN\_IND\_MIN” responsável pelo indicador luminoso de nível água mínimo;
* GPIO 15 definido na constante “PIN\_IND\_AUTO” responsável pelo indicador luminoso do modo auto;
* GPIO 18 definido na constante “PIN\_IND\_ALARM” responsável pelo indicador luminoso de nível de água de alarme;
* GPIO 19 definido na constante “PIN\_VALV\_WATER\_IN” responsável pela válvula de entrada de água fria;
* GPIO 21 definido na constante “PIN\_VALV\_WATER\_OUT” responsável pela válvula de descarga de vapor;
* GPIO 22 definido na constante “PIN\_BMB” responsável pela bomba de água;
* GPIO 23 definido na constante “PIN\_RAQ” responsável pela resistência de aquecimento;
* GPIO 33 definido na constante “PIN\_IND\_MAN” responsável pelo indicador luminoso do modo manual;

## Prototipagem em Protoboard

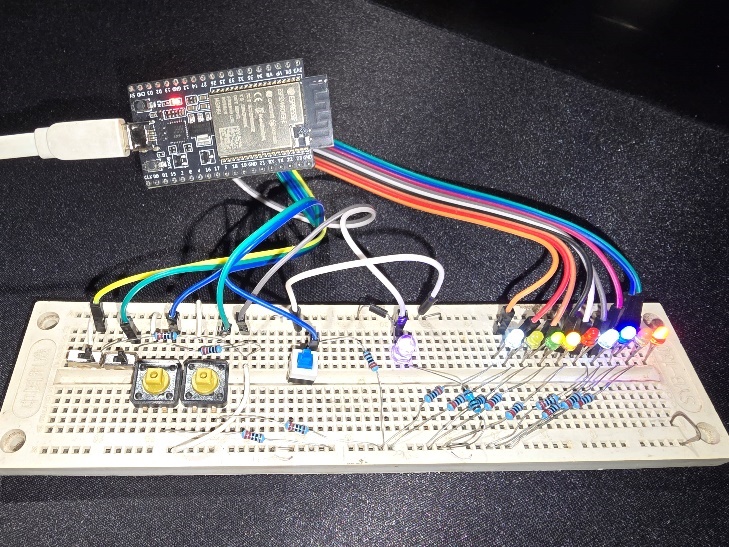


Figura 5 ESP32 na Protoboard

Para desenvolver o *Software* em casa, foi simulado toda a destiladora em uma *protoboard* com botões, interruptores e indicadores LED a simular os I/Os da máquina real, em que os botões e interruptores, simulam os *inputs* e os indicadores LED os *outputs*.

## Circuito Elétrico

Uma imagem com máquina, interior, eletrónica, Engenharia eletrónica

Descrição gerada automaticamente

Figura 6 Substituir Circuito Elétrico

Todo o circuito elétrico antigo e completamente analógico foi removido e substituído por um novo e digital, todos os cabos foram substituídos, como mostrado pela Figura 5, mas foram aproveitados alguns componentes, tal como os botões, válvulas, bomba de água e resistência de calor.

### Relês

Para controlo destes componentes, foram adicionados 2 módulos com 4 relês cada (ver Figura 6), o módulo da esquerda, está dedicado a controlar os componentes de 250V como a bomba de água, as 2 válvulas e a resistência de calor e o módulo da direita, controla os indicadores LED frontais (nível de água máximo, mínimo, alarme e modo automático), foi adicionado um disjuntor bipolar de 16Amperes, uma fonte de alimentação de 5V para fornecer energia ao microprocessador e componentes de baixa voltagem.

Uma imagem com Fios elétricos, Engenharia eletrónica, eletrónica, cabo

Descrição gerada automaticamente

Figura 7 Relês e ESP32

Para aproveitar o botão de manual, que tem um indicador embutido de néon de 250V, foi adicionado um relê dedicado somente ao botão de modo manual, em que ao ser ligado, aciona o relê para o microprocessador poder receber essa informação (ver Figura 8).

Uma imagem com diagrama, texto, file, Paralelo

Descrição gerada automaticamente

Figura 8 Esquema Elétrico Relê Modo Manual

### Filtro para os inputs

Como se está a trabalhar com alta tensão de onda alternada (~240V AC) juto com tensão baixa continua (5V DC) e o microprocessador é muito sensível a ruido nos sinais de *input*, foi detetado nos testes em que ao ligar/desligar o botão de modo manual, os *outputs* aleatoriamente trocavam de estado sem que fosse propositado, depois de muitos testes, foi detetado com recurso ao osciloscópio, que ao ser acionado botão de modo manual, causava ruido de alta frequência nos I/Os, já que os fios agem como antenas aos campos eletromagnéticos gerados pela alta tensão nos fios e pelas bobines dos relês [4].

Depois de pesquisado, foi encontrado um vídeo no Youtube [5] que representa o mesmo problema que estava presente no circuito da destiladora, foi a partir desse mesmo vídeo, que foi estudado o problema e resolvido com a implementação de um filtro passa-baixo passivo (ver Figura 9), que visa eliminar as altas frequências geradas pelos componentes e fios que operam em alta tensão

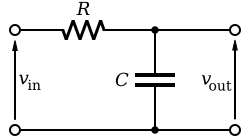


Figura 9 Filtro Passa-Baixo Passivo

Foi então aplicada a equação representada na Figura 10 e como foi usado (ver Figura 11) um condensador de tântalo de 220nF (*Farads*) e uma resistência de 47 OHMs, o resultado da equação é aproximadamente ~15392Hz, esta é a frequência máxima que o filtro permite.

Foi adicionado um filtro para cada *input*.

Foi adicionado também em paralelo um condensador eletrolítico de 16V e 1000uF e um condensador de tântalo de 220nF para ajudar na filtragem ligados diretamente na saída da fonte de alimentação.



Figura 10 Equação Filtro Passa-Baixo

Uma imagem com texto, diagrama, Esquema, file

Descrição gerada automaticamente

Figura 11 Esquema Elétrico ESP32 com Filtro Passa-Baixo

### Depósito e os sensores de nível

### Bombas e válvulas a 220v

### LED com rele a 5v

CONCLUSÃO

Lorem.

REFERÊNCIAS BIBLIOGRÁFICAS

|  |  |
| --- | --- |
| [1] | Espressif, “Espressif-About,” [Online]. Available: https://www.espressif.com/en/company/about-espressif. [Acedido em 08 2024]. |
| [2] | espressif, “espressif ESP32,” [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-wrover-e\_esp32-wrover-ie\_datasheet\_en.pdf. [Acedido em 08 2024]. |
| [3] | L. Harvie, “Embedded Systems Memory Types: Flash vs SRAM vs EEPROM,” [Online]. Available: https://medium.com/@lanceharvieruntime/embedded-systems-memory-types-flash-vs-sram-vs-eeprom-93d0eed09086. [Acedido em 08 2024]. |

ANEXOS

## Anexo 1 – "Circuito Elétrico”



Anexo 1 Circuito Elétrico

## Anexo 2 – “Instruções de Uso”



Anexo 2 Instruções de Uso