Relatório do projeto final

Nome: **José Adriano Filho**

Turma: **CodeBoard**

Neste relatório faremos a apresentação do projeto final do treinamento **EmbarcaTech**, apoiado pelo **IFCE – Instituto Federal do Ceará**, nele faremos apresentação, as especificações de hardware utilizado, o firmware desenvolvido, a execução do projeto e as referências pesquisadas como apoio para todo o trabalho.

1. ESCOPO DO PROJETO

1.1 – Apresentação do problema:

O projeto final nasceu da necessidade de uma amiga que tem uma casa na belíssima praia do Cumbuco, região metropolitana de Fortaleza capital do Ceará. Sabemos que as casas de praia são uma ótima oportunidade de ganho financeiro, principalmente em determinadas épocas, denominadas de alta estação. Estas casas são alugadas para períodos curtos como final de semana, feriados prolongados, local para festas dentre outras atividades.

Fig. 1 – Casa de praia - Foto feita pelo autor

Mas ao mesmo tempo que os locatários se divertem, são geradas algumas preocupações para o locador, como por exemplo, a gestão da caixa d'água e a aguação do gramado e plantas do imóvel. É muito inconveniente a falta de água na residência, mesmo que haja orientação para a pessoa que alugou, sempre tem aquele telefonema perguntando como ligar o motor para a caixa. Colocar uma pessoa para executar o serviço, passou a trazer, algumas vezes, até mais problemas.

Dai nasceu a ideia de um equipamento que juntasse a automação da irrigação do jardim bem como fazer o gerenciamento da caixa d'água do local.

Parece uma necessidade simples, mas quem tem este tipo de imóvel, já passou por diversas situações constrangedoras com inquilinos, bem como ter a grama queimada do sol forte que temos em nossa região, a dor existe e é real, podemos resolver com o desenvolvimento de um projeto que execute as duas funções ao mesmo tempo, a um custo relativamente baixo, já que utilizaremos o poder dos microcontroladores.

1.2 – Título do projeto:

O projeto foi batizado de “**AQUABOX – CUMBUCO**” – sistema de gestão de irrigação e caixa d’água para casas de praia.

1.3 – Objetivos do projeto:

Basicamente o projeto tem dois objetivos principais que são:

a. Executar o gerenciamento da caixa d’água do imóvel, fazendo com que a mesma esteja sempre disponível com volume de água suficiente para toda necessidade dos usuários;
b. Promover a irrigação do jardim/gramado em horário preestabelecido, deixando o mesmo sempre em boas condições para que o ambiente possa ser bem acolhedor para as pessoas que utilizem local.

Existem objetivos secundários como tirar a preocupação do dono do imóvel com caixa d’água e irrigação, o constrangimento do inquilino em ficar sem água durante sua estadia, o cuidado com o jardim durante os períodos em que a casa fica sem ninguém.

1.4 – Justificativa:

O projeto se justifica já que no mercado existem equipamento individuais para o tratamento de cada problema, ou seja, existe equipamento para o controle de caixa d’água e outro para o controle de irrigação do jardim.

Nossa proposta junta as duas atividades em um equipamento só, fazendo com que em um único dispositivo possamos automatizar os dois processos, inclusive ficando mais barata a solução.

Fig. 2 - Controlador de irrigação – retirado da internet https://produto.mercadolivre.com.br/

Fig. 3 – Controlador de caixa de água – retirado da internet https://redgtech.com.br/

Nas figuras 1 e 2 temos dois exemplos de produtos oferecidos pelo comércio para atuar nos problemas, só que de maneira individual.

Esta é uma dor real que existe, estamos desenvolvendo a cura para ela.

1.5 – Originalidade:

Como foi falado no item anterior, a originalidade se dá pela não existência de equipamento similar, conforme pesquisa realizada em sites de busca.

Quanto ao projeto em si, apesar de partes de hardware serem quase que padrões adotados, como o acionamento de relés, a utilização de botões para entrada de configurações, utilização de chip RTC para armazenamento de relógio, nosso projeto não se baseou em nenhum sistema já pronto, além dos requisitos do sistema. Todo o software foi desenvolvido levando em conta a necessidade desses mesmos requisitos, não utilizamos nenhuma biblioteca já pronta, preferimos fazer nossas próprias funções, para demonstrar essa originalidade do trabalho, tanto que preferimos, apesar de não ser uma boa prática, fazer todo firmware em um arquivo só, a exceção foi o uso do SDK – "Software Development Kit" que não tínhamos como ignorar o mesmo.

As pesquisas em site de busca não mostraram em momento algo parecido, encontramos sistemas de irrigação, projetos de medição de nível para caixa d'água, mais como forma de aprender, optamos por desenhar nossa própria solução, obviamente seguindo orientações dos datasheets dos componentes utilizados.

2. ESPECIFICAÇÃO DO HARDWARE

2.1 – Diagrama em blocos:

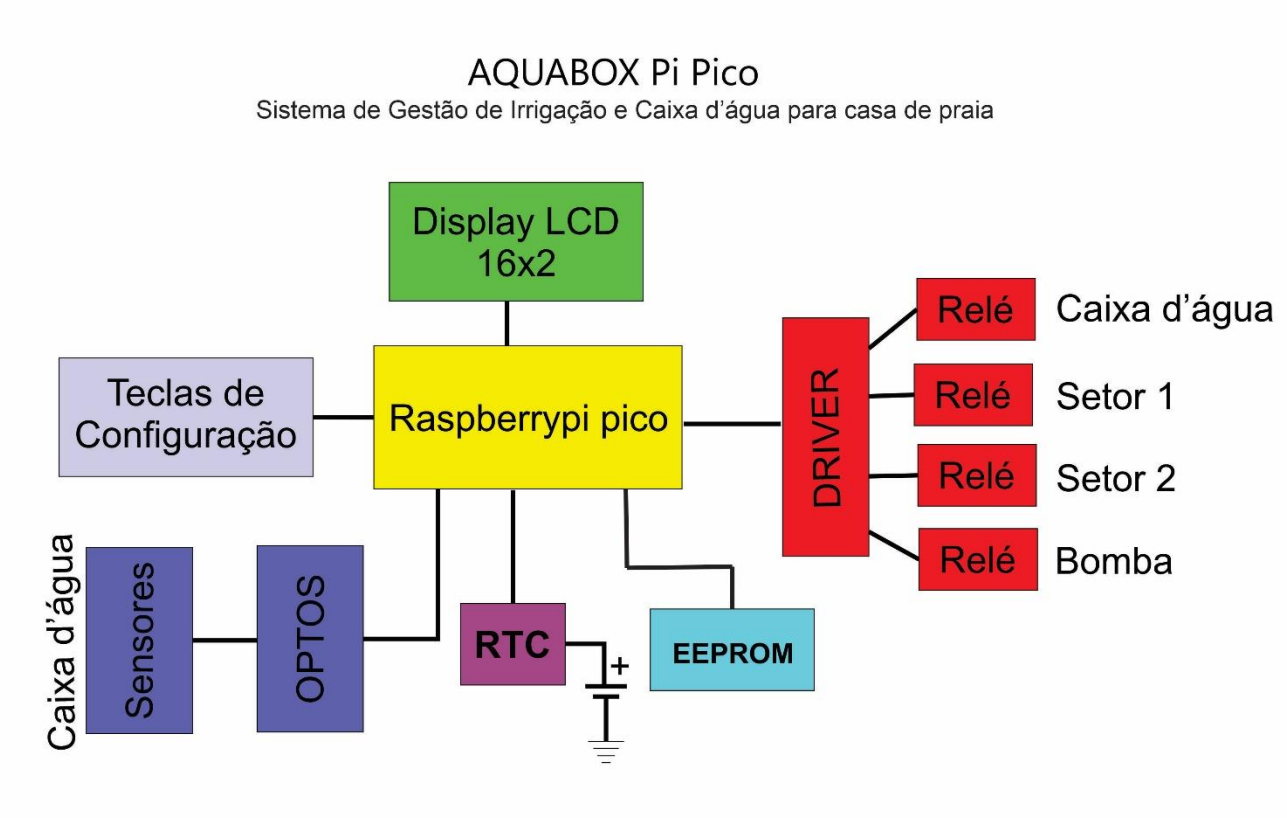


Fig. 4 – Diagrama de blocos do sistema Aquabox – desenhado pelo autor

Na figura 4 temos o diagrama de blocos do sistema, o mesmo mostra as conexões e componentes que fazem parte do projeto como um todo.

2.1.1 – Descrição do diagrama, comandos e registros

a. Raspberry Pi Pico – Plataforma de desenvolvimento baseado no microcontrolador RP2040, de baixo custo e alta performance. Tem dois cores de processadores ARM Cortex M0+, rodando até 133MHz, 264kB de memória SRAM e 2MB de memória Flash integrada a placa. Possui 26 GPIOs multifunção, com 2x SPI, 2x UART, 3x ADC de 12-bits e 16 canais PWM, clock preciso e timer on-chip, além de outras características.

b. **Display alfanumérico 16x2** – Lcd com 16 colunas e 2 linhas, possui controlador HD44780. O display terá a função de mostrar ao usuário os diversos estados do sistema como “enchendo a caixa”, “irrigando setor 1” ou “irrigando setor 2”, além de possibilitar executar a configuração do relógio ou a hora e duração da irrigação.

c. Relés - Conjunto de quatro relés para acionamento de válvulas solenoide para abertura do fluxo de água, bem como acionamento da bomba d’água.

d. Driver – Circuito baseado em transistor para fornecer corrente necessário para acionamento dos relés, sabendo que os pinos do microcontrolador não conseguem suprir essa necessidade.

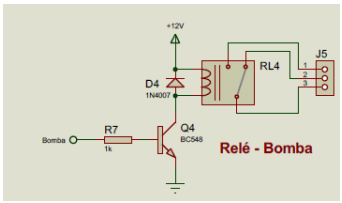


Fig. 5 – Driver para acionamento do relé – imagem do próprio autor

e. Teclas de configuração – Teclado composto de quatro teclas definidas como “Menu”, “Retorno”, “Seleção” e “Dado”, cuja a função é permitir a interação do usuário com o sistema, possibilitando que o mesmo possa configurar itens como acertar a hora do relógio, indicar em qual horário e duração da irrigação além de habilitar ou desabilitar em quais dias da semana a irrigação deverá acontecer.

f. Sensores – Existem dois sensores no sistema que devem ser colocados na caixa d’água, afim de informar a quantidade mínima e máxima de água na mesma. Estes sensores serão responsáveis pelo acionamento da bomba e válvula adequada para o enchimento dela.

g. **Optos** - Como estaremos alimentando os sensores com tensão de 12V, não poderemos conecta-los diretamente ao microcontrolador, assim estamos utilizando componentes denominados de optoacopladores que são responsáveis em nosso projeto de fazer uma conversão de nível, ou seja, converte de 12V para 3.3V e assim não danificamos o restante do sistema.

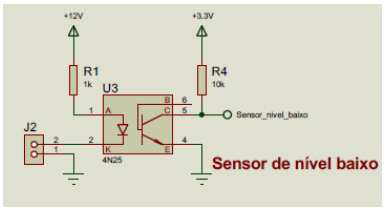


Fig. 6 – opto para converter nível de tensão para os sensores – imagem do próprio autor

h. **RTC** – Relógio de tempo real cuja a finalidade é armazenar calendário e relógio para o bom funcionamento do sistema, desta forma podemos configurar horário para a irrigação bem como saber qual o dia semana para podemos verificar se neste dia a irrigação está habilitada.

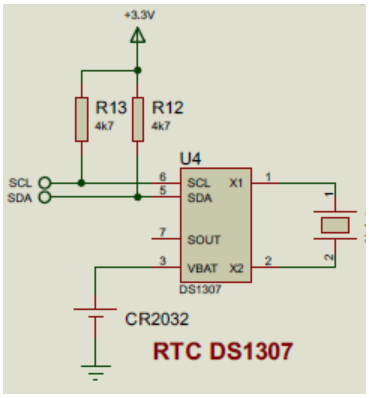


Fig. 7 – RTC – chip de relógio de tempo real com bateria – imagem do próprio autor

i. **EEPROM** – Memória de armazenamento permanente onde serão guardadas as configurações do sistema, em caso de falta de energia, como hora, minutos, duração da irrigação, bem como os comandos de habilitar ou desabilitar a mesma durante a semana.

Os blocos acima fazem um resumo do sistema, assim fica fácil ter uma ideia geral do funcionamento do projeto, bem como, facilita a escolha dos componentes utilizados para executar cada função, como por exemplo os optoacopladores, os transistores, dentre outros.

2.1.2 – Pinagem

Identificações de GPIOs			
id	**Componente**	**GPIO**	**Função**
1	LCD_16x2 - RS	2	Seleção Comando/Dado
2	LCD_16x2 - E	3	Enable
3	LCD_16x2 - D4	4	Bit 4
4	LCD_16x2 - D5	5	Bit 5
5	LCD_16x2 - D6	6	Bit 6
6	LCD_16x2 - D7	7	Bit 7
7	RTC DS1307 - SDA	20	Via de dados I2C
8	RTC DS1307 - SCL	21	Via de clock I2C
9	Relé - Caixa d'água	28	Acionamento da válvula solenóide para a caixa d'água
10	Relé - Setor 1	27	Acionamento da válvula solenóide para a irrigação do setor 1
11	Relé - Setor 2	26	Acionamento da válvula solenóide para a irrigação do setor 2
12	Relé - Bomba d'água	22	Acionamento da bomba d'água
13	Sensor de nível alto	15	Sensor de nível alto instalado na caixa d'água
14	Sensor de nível baixo	14	Sensor de nível baixo instalado na caixa d'água
15	Pushbutton vermelho	8	Botão para menu de opções
16	Pushbutton verde	9	Botão para retornar a tela inicial
17	Pushbutton amarelo	10	Botão para seleção de opções
18	Pushbutton azul	11	Botão para entrada de dados de configuração
19	EEPROM - SDA	18	Via de dados I2C
20	EEPROM - SCL	19	Via de clock I2C
21	Pushbutton RESET	RUN	Botão de reset

Fig. 8 – Tabela com a descrição e **GPIO**'s dos pinos utilizados no projeto – planilha do próprio autor

2.1.3 Esquemáticos

Nosso esquemático será apresentado em duas versões, uma completo com todos os componentes e conexões necessários ao real funcionamento do sistema e outro mais simplificado utilizado para a simulação na plataforma **Wokwi**.

Na simulação do **Wokwi**, utilizamos configuração dos resistores de pull-up, internos, mas no esquemático real, inserimos resistores externos, achamos melhor, uma outra diferença é que no projeto final temos o uso de uma memória EEPROM de 2k e na simulação não foi possível utiliza-la pois o componente não tem no simulador.

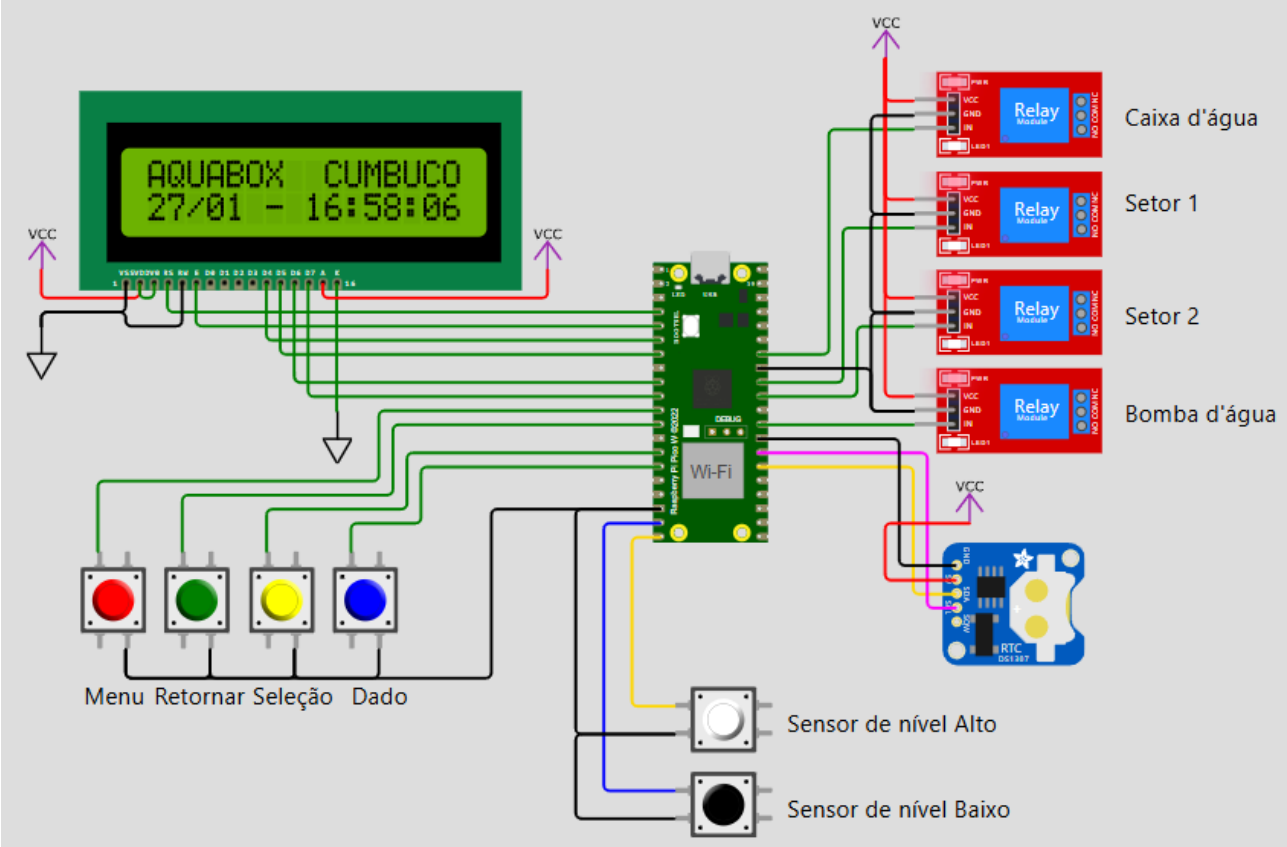


Fig. 9 – Diagrama esquemático do Wokwi – imagem do próprio autor

Na página seguinte temos nosso diagrama esquemático do projeto real, podemos notar os resistores de pull-up e a memória EEPROM de 2k. Nosso esquemático mostra os drivers para acionamento dos relés bem como o uso de optoacopladores para a conversão de nível dos sensores de nível alto e baixo da caixa d'água.

Baseado na plataforma de desenvolvimento da RaspberryPi pico w, temos todo o diagrama para conexão dos diversos componentes do sistema, um bloco com os reguladores de níveis de tensão, já que optamos por utilizar uma fonte externa de 12V DC, utilizamos circuitos para gerar a tensão de 5V e a tensão de 3.3V para alimentação dos diversos circuitos como o diaplay 16x2, RTC, EEPROM e o próprio raspberryPi Pico. A fonte também é utilizada em sua tensão padrão de 12V, para suprir o acionamento dos relés, para as válvulas solenoides externos, responsáveis por abrir e fechar o fluxo de água, bem como interligar os sensores que são utilizados para detectar os níveis de líquido dentro da caixa.

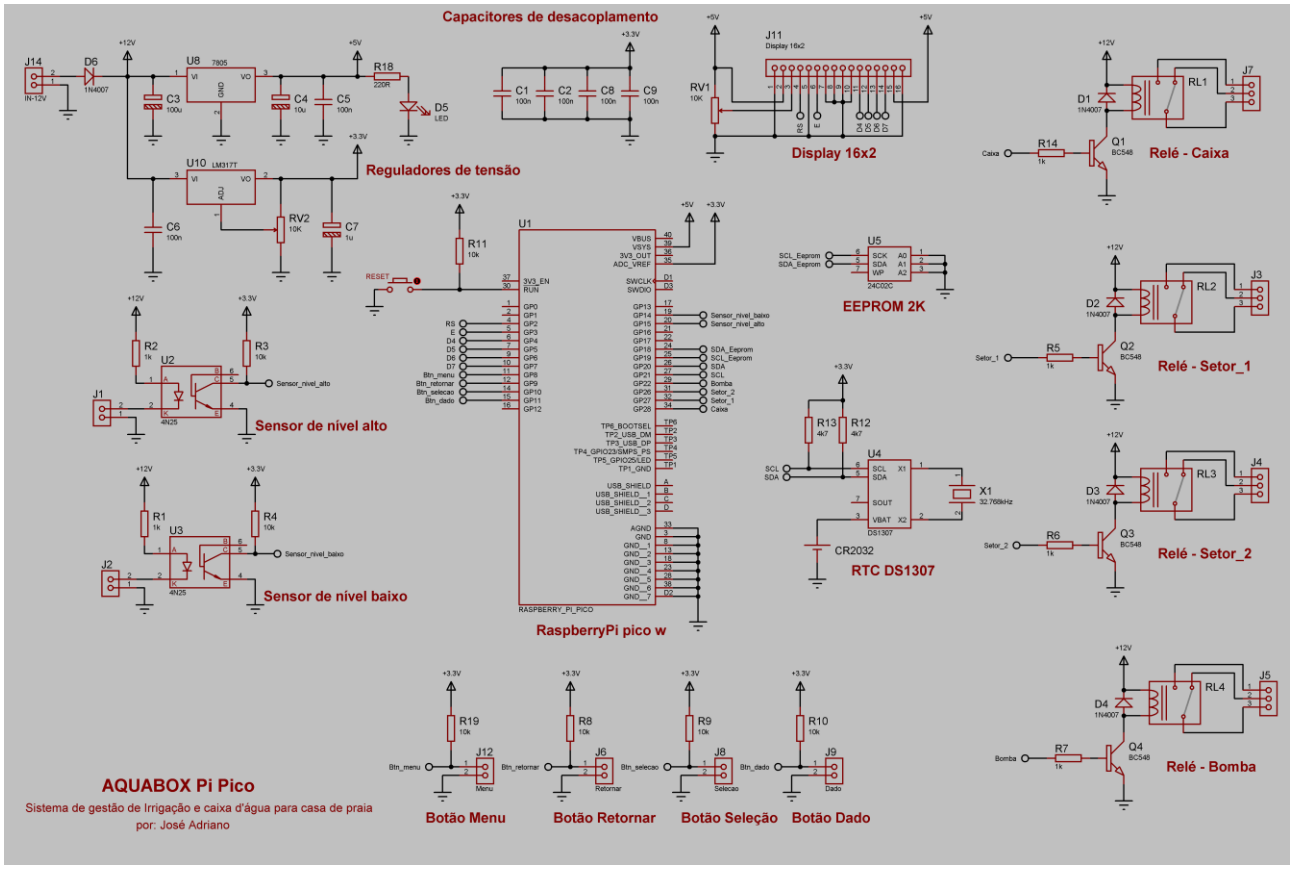


Fig. 10 – Diagrama esquemático do sistema real – imagem do próprio autor

Abaixo o tipo de sensor de nível que vamos utilizar em nosso sistema, o mesmo apresentou grande confiabilidade. Baseado em relé tipo ampola denominado de reed switch. É acionado quando tem aproximação de um campo magnético, neste caso um pequeno imã em sua haste móvel.

Fig. 11 – Sensores de nível de água – imagem retirada da internet

3. ESPECIFICAÇÃO DO FIRMWARE

3.1 – Blocos funcionais

Nosso firmware está baseado em máquina de estado, ou seja, dependendo do evento que seja requisitado, seja por sensores, horário pré-determinado, ou ações do usuário, temos um estado onde será executado um bloco de código, uma função específica como encher a caixa d'água. Optamos por esse método de desenvolvimento de sistema embarcado por se adequar bem a nossa necessidade. Na página seguinte temos um layout onde mostramos os blocos de cada função necessária.

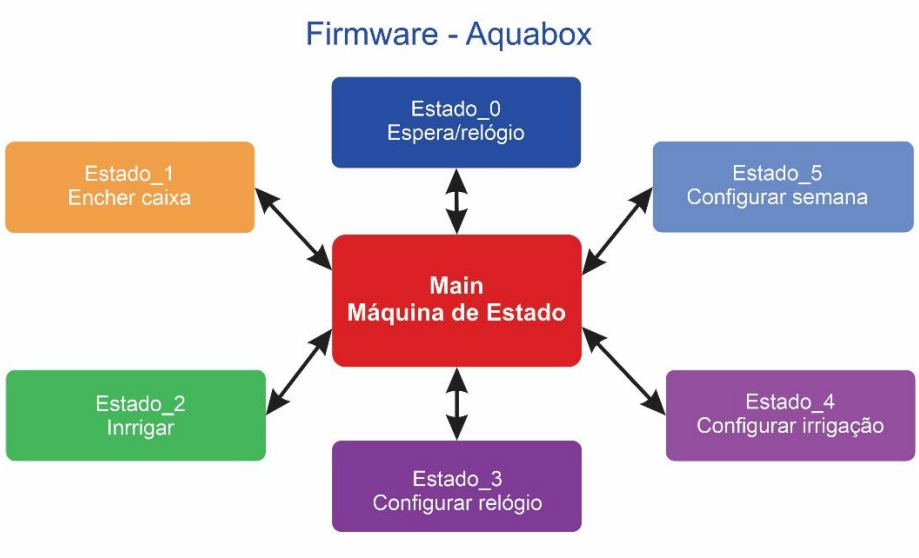


Fig. 12 – Blocos funcionais do firmware – imagem do próprio autor

1. **Main**: função principal do sistema, aqui executamos todas as inicializações necessárias ao bom funcionamento de todo projeto como inicialização das interfaces I2C, EEPROM, LCD, RTC, Timer, interface serial uart/usb para depuração de buges e o mais importante, a inicialização dos GPIO's, necessários para entradas e saídas de sinais necessários aos diversos eventos.
2. **Estado_0**: Neste estado o sistema fica em standby, ou seja, esperando que algum evento aconteça, nesta função também optamos por mostrar no display um relógio de tempo real com o dia, mês e a hora atual, inclusive com os segundos.
3. **Estado_1**: Este estado será acionado todas as vezes que o sensor de nível baixo que está alocado na caixa d'água, indicar o valor zero no Gpio14, ou seja, será acionado a válvula solenoide da caixa e logo em seguida será ligado a bomba, assim a caixa começará o processo de enchimento. Esta função ficará em estado de execução até que o sensor de nível alto seja acionado, isto indicará que a caixa está cheia, após o processo, o sistema retornará ao estado_0.
4. **Estado_2**: Da mesma forma que o estado anterior, este será chamado quando estiver no horário configurado para irrigação, assim será acionado a válvula do primeiro setor, decidimos dividir em dois setores para não sobrecarregar a bomba, após a conclusão da primeira irrigação, que é definida por uma variável **"duracao"**, será acionado o setor 2, que terá a mesma duração que o anterior, o sistema retornará para o estado_0.
5. **Estado_3**: Este estado será acionado por intervenção do usuário, nele será executado a configuração do RTC para data e hora atual, caso seja necessário tal intervenção. A entrada de dados será por intermédio de um conjunto de teclas que fazem parte do projeto, da mesma forma, ao concluir, será retornado ao estado_0.
6. **Estado_4**: Esta função é responsável pela configuração da hora, minutos e duração da irrigação, estes dados serão armazenados em uma eeprom externa de 2k, tamanho mais que suficiente para guardar estas informações. O uso da eeprom se fez necessário para evitar perda de configuração por uma falha de energia do sistema, a entrada de dados será pelo mesmo conjunto de teclas reportadas no item anterior, após a conclusão será retornado ao estado_0.

7. **Estado_5**: Sabemos que existem dias que não será necessária a irrigação, assim nesta função o usuário poderá configurar os dias da semana que será necessário irrigar e desabilitar os outros, por exemplo, na quadra chuvosa que temos em nosso estado no início do ano, se faz desnecessário o processo, assim podemos desabilitar todos os dias, quando a mesma passar, voltamos a habilitar a mesma. Ao final da configuração, retornamos ao estado_0.

3.2 – Definição de variáveis

Abaixo temos as principais variáveis utilizadas no código, como podemos notar, nosso sistema utiliza um timer de hardware para gerar uma interrupção a cada 100ms e assim fazer a leitura dos botões e sensores, optamos por utilizar sinalizadores na função de callback, assim sempre que há o estouro do timer, a função faz uma varredura e seta a variável flag correspondente, desta forma podemos no estado_0 efetuar a leitura e direcionar o código para executar a tarefa mais adequada.

3.2.1 – Variáveis mais importantes

- int8_t **funcao_ativa** = 0;
- volatile bool **nivel_baixo_flag** = false;
- volatile bool **nivel_alto_flag** = false;
- volatile bool **botao_menu_flag** = false;
- volatile bool **botao_retorno_flag** = false;
- volatile bool **botao_selecao_flag** = false;
- volatile bool **botao_dado_flag** = false;
- bool **enchendo_flag** = false;
- bool **irrigando_flag** = false;
- bool **configurando_flag** = false;

3.2.2 – Estruturas importantes

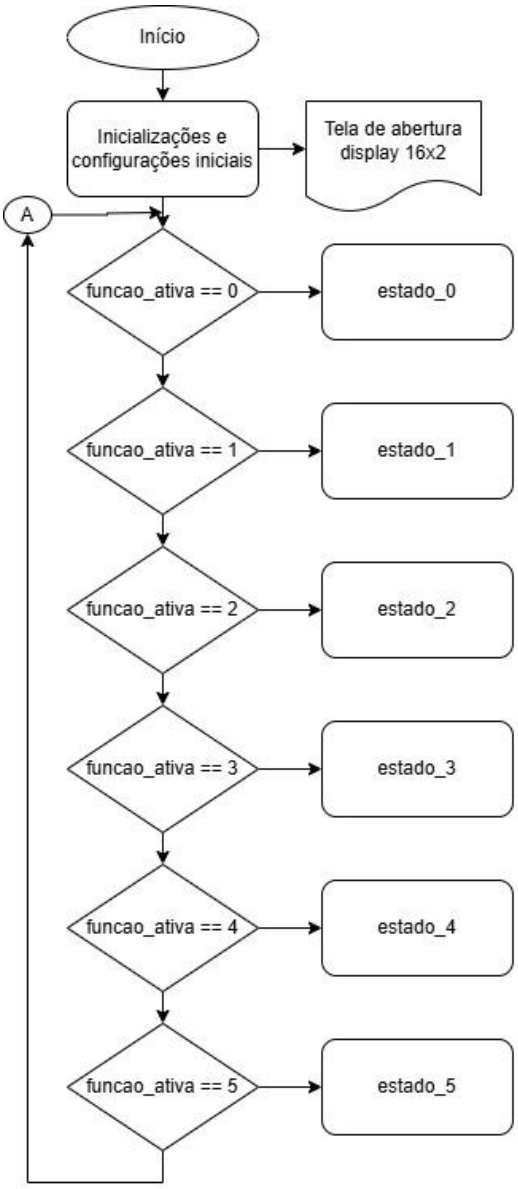
Abaixo temos duas estruturas criadas para facilitar o tratamento dos dados do relógio de tempo real, bem como as configurações do sistema de irrigação, nelas ficam armazenadas estas informações, bem como facilita salva-las no RTC DS-1307 e na EEPROM.

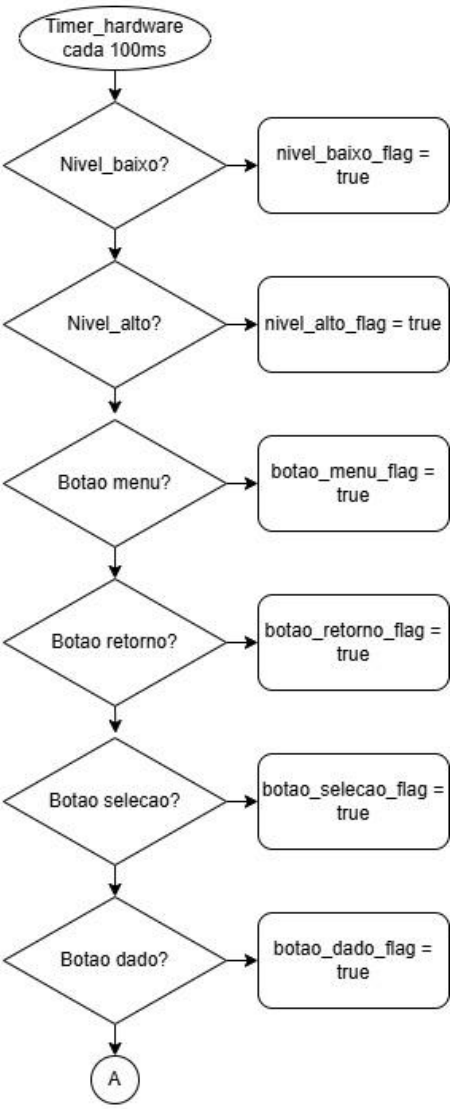
```
//Estrutura para controle do relógio
struct tempo
{
   uint8_t segundos;
   uint8_t minutos;
   uint8_t horas;
   uint8_t diaSemana;
   uint8_t dia;
   uint8_t mes;
   uint8_t ano;
};
```

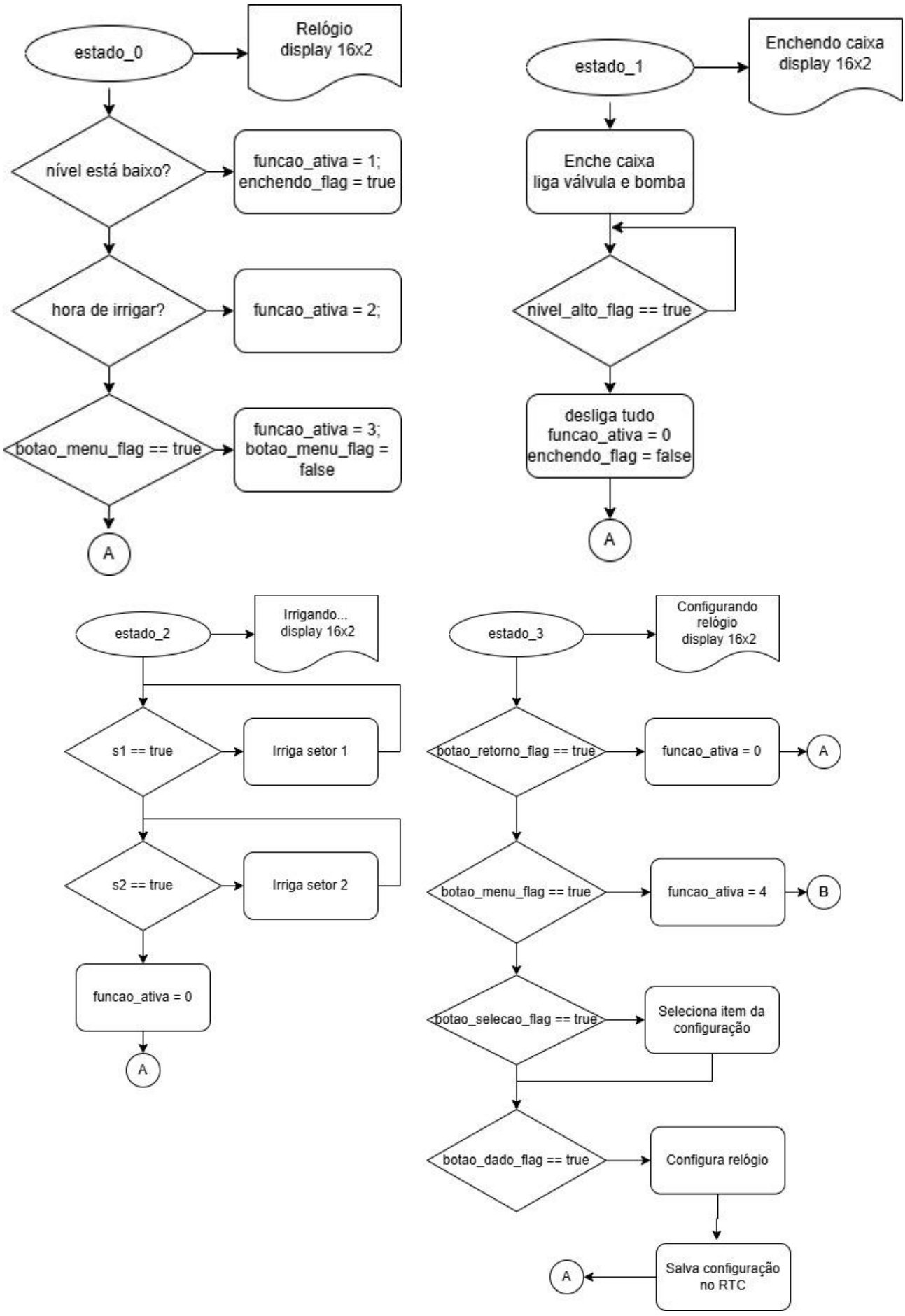
//Estrutura para controle da irrigação

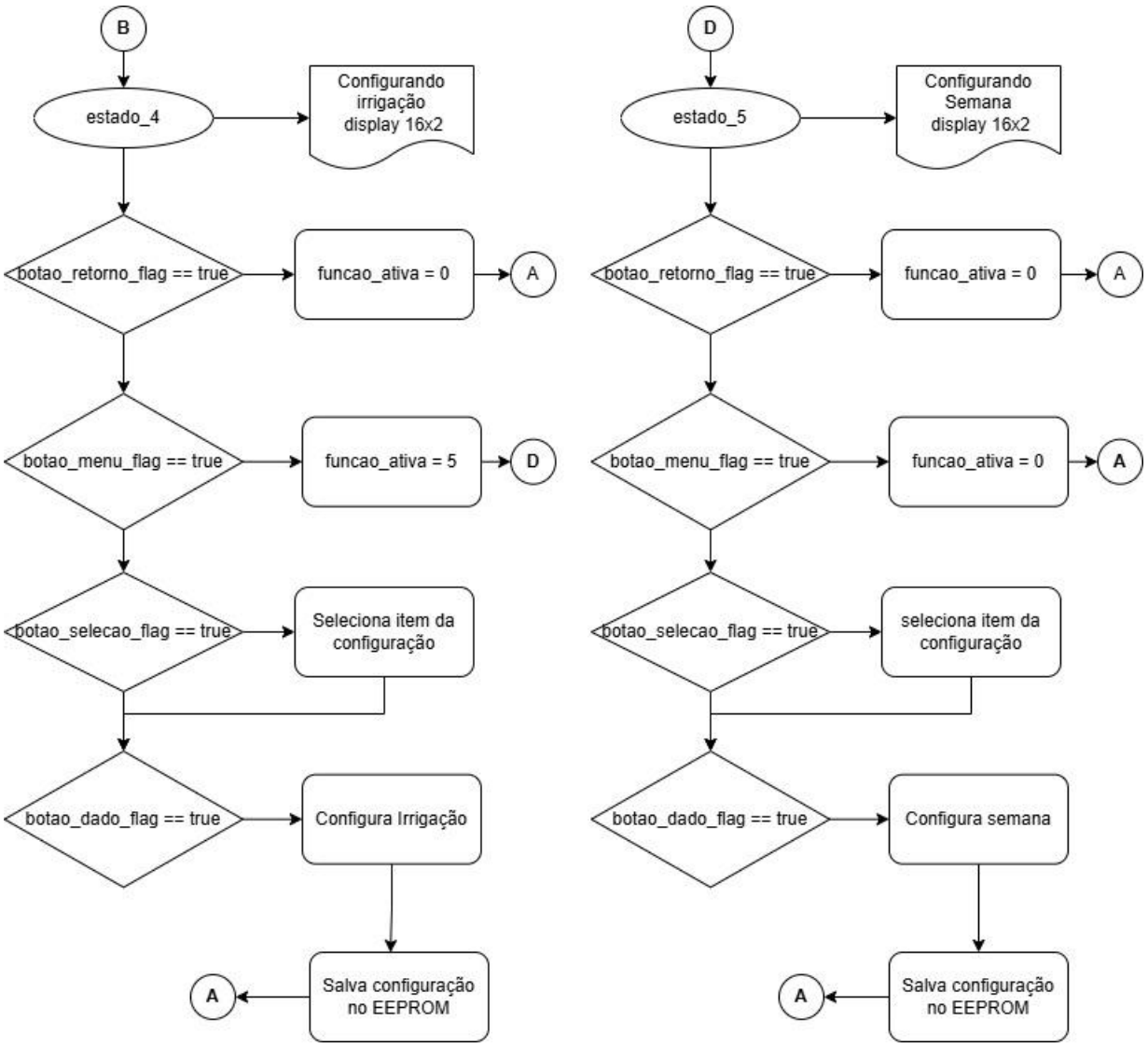
```
struct irriga
{
    uint8_t habilita;
    uint8_t hora;
    uint8_t minutos;
    bool dia_da_semana[7];
    uint8_t duracao;
};
```

3.2.3 – Fluxograma






estado_0
Relógio
display 16x2
nível está baixo?
funcao_ativa = 1;
enchendo_flag = true
hora de irrigar?
funcao_ativa = 2;
botao_menu_flag == true
funcao_ativa = 3;
botao_menu_flag = false
A
estado_1
Enchendo caixa
display 16x2
Enche caixa
liga válvula e bomba
nivel_alto_flag == true
desliga tudo
funcao_ativa = 0
enchendo_flag = false
A
estado_2
Irrigando...
display 16x2
s1 == true
Irriga setor 1
s2 == true
Irriga setor 2
funcao_ativa = 0
A
estado_3
Configurando
relógio
display 16x2
botao_retorno_flag == true
funcao_ativa = 0
A
botao_menu_flag == true
funcao_ativa = 4
B
botao_selecao_flag == true
Seleciona item da
configuração
botao_dado_flag == true
Configura relógio
Salva configuração
no RTC
A



4. EXECUÇÃO DO PROJETO

5, REFERÊNCIAS