UNIVERSIDADE FEDERAL DE ITAJUBÁ

RELATÓRIO:

VENTILADOR INTELIGENTE

Matheus Ferraz Capucho - 2020001439
Prof. Otávio de Souza Martins Gomes

2021

1 - Introdução

O presente relatório se destina a apresentar o projeto “Ventilador Inteligente”, feito para a avaliação final da disciplina de ECOP04 (programação embarcada) e ECOP14 (laboratório programação embarcada), da Universidade Federal de Itajubá.

Para realização do projeto, foi utilizado o ambiente de simulação do PICSimLab, com a placa PICGenios, onde é utilizado o microcontrolador Pic18f4520, visto na figura 1 (abaixo). Este microcontrolador faz parte da família da Microchip, contendo 256 bytes de memória EEPROM.

figura 1 - PICGenios

Todo o programa foi escrito na linguagem C, sendo que muitas das bibliotecas utilizadas foram disponibilizadas pelo professor Otávio de Souza Martins Gomes, e escritas pelo professor Rodrigo Maximiano Antunes de Almeida.

2 - Objetivos

O presente projeto se destina a implementar um sistema de ventilador programável, em que poderão ser utilizados 2 modos: O modo livre, que funciona como um ventilador comum, podendo ajustar a potência; e o modo automático, que consiste em um contador decrescente de tempo, até se desligar sozinho.

Essa funcionalidade foi pensada principalmente para pessoas que gostam de dormir com ventilador ligado, para que o mesmo não fique funcionando durante todo o sono.

3 - Projeto

3.1 - Ferramentas

> Simulador PICSimLab
> MPLAB X IDE v5.50
> XC8 (compilador)

3.2 - Componentes

> 1 Display LCD 16x2
> Teclado da Placa
> Ventoinha
> 4 displays de 7 segmentos
> Potenciômetro

4 - Funcionalidades

4.1 - Menu

O menu principal é composto é composto por 2 funcionalidades, visto na figura 2:

Figura 2 - Menu no LCD

> 1-Livre: Funciona como um ventilador normal, podendo ajustar sua potência conforme o desejado, e podendo retornar ao menu com o botão “*”;

> 2-Auto: Aqui, deve-se escolher o tempo desejado que o ventilador fica ligado (figura 3), podendo retornar ao menu ao colocar o potenciômetro na origem.

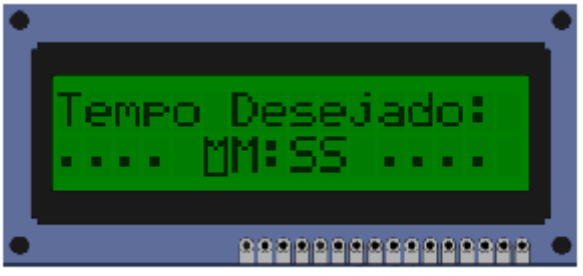

Figura 3 - Modo Auto

4.2 - Display de 7 Segmentos (ssd)

Ao selecionar o tempo desejado no modo Auto, o ssd apresentará um cronômetro decrescente de tempo, como pode ser visto um exemplo na figura 4:

Figura 4 - Cronômetro

4.3 - Ventoinha

A ventoinha (cooler) é o componente que simula o ventilador, e para controlar sua potência, é utilizado o primeiro potenciômetro da placa, sendo que seu valor (de 0 à 1000) é apresentado no display de LCD; um exemplo é mostrado a seguir, na figura 5.

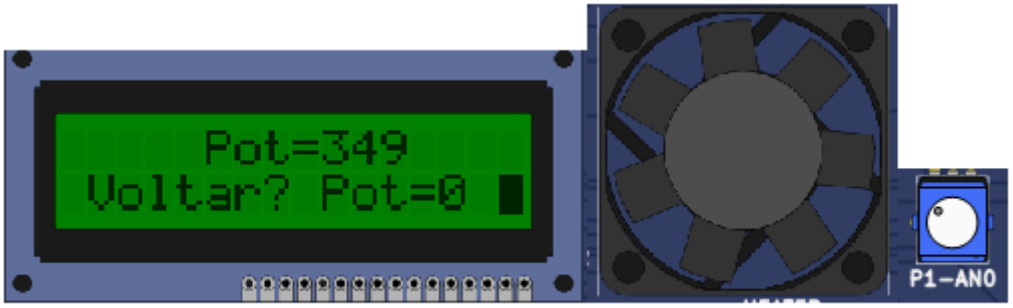

figura 5 - Componentes de controle do cooler

5 - Bibliotecas Utilizadas:

Para o funcionamento deste projeto, foram utilizadas as bibliotecas que controlam o PIC18F4520, do professor Rodrigo Maximiano Antunes de Almeida, e outras 2 foram criadas para a execução do programa: “ventilador” e “menu”. Todas as bibliotecas serão disponibilizadas no GitHub. A seguir, uma imagem do escopo das funções utilizadas em “ventilador.c” e “menu.c”

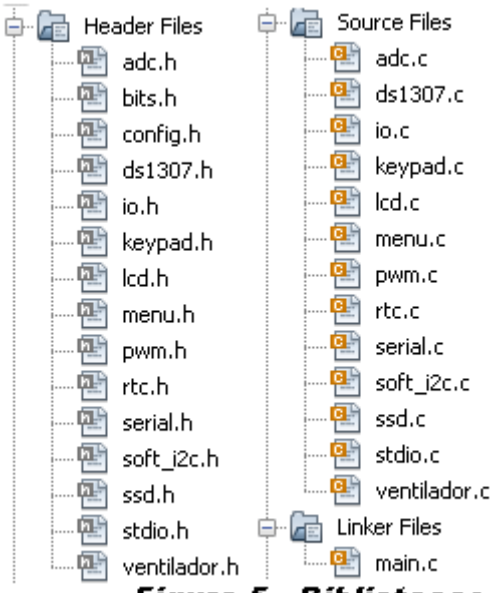

Figura 5 - Bibliotecas

5.1 - Menu e Ventilador

> start() : Configura a placa e mostra a mensagem de “Bem Vindo(a)” no LCD.

> func() : Realiza a leitura da tecla para a escolha do menu.

> livreVentilador() : Realiza a função livre.

>autoVentilador() : Realiza a função com temporizador.

```
#ifndef VENTILADOR_H
#define VENTILADOR_H

void livreVentilador();
void autoVentilador();

#endif  /* VENTILADOR_H */
```

```
#ifndef MENU_H
#define MENU_H

void start();
void func();

#endif  /* MENU_H */
```

Figura 6 - Escopo das Funções

A seguir, uma imagem da implementação das funções vistas acima.

```
#define L0 0x80
#define L1 0xC0
#define CLR 0x01
#define ON 0x0F

static int on = 0;

void start() {
    lcdInit();
    pwmInit();
    ssdInit();
    kpInit();

    lcdCommand(CLR);
    lcdCommand(L0);
    int i;
    for(i = 0; i< 16; i++)
        lcdData('.');
    lcdCommand(0xC0);
    for(i = 0; i< 16; i++)
        lcdData('.');
    lcdCommand(L0);
    printf("Bem Vindo(a)    ");
    lcdCommand(L1);
    printf("1-Livre  2-Auto");
}
```

```
void func() {

    unsigned char tecla;

    kpDebounce();

    if (kpRead() != tecla) {
        tecla = kpRead();
        if (bitTst(tecla, 3)) {
            livreVentilador();
        }
        if (bitTst(tecla, 7)) {
            autoVentilador();
        }
    }
}
```

Menu.c

```
void livreVentilador() { // ajusta a ventoi
    adcInit();
    pwmInit();
    kpInit();
    pwmFrequency(2000);
    pwmSet1(50);
    int valor, on = 1;
    while (on) {
        lcdCommand(0x80);
        valor = adcRead(0);
        pwmSet1(valor / 10);
        lcdData(' ');
        lcdData(' ');
        lcdData(' ');
        lcdData(' ');
        lcdData(' ');
        lcdData('P');
        lcdData('o');
        lcdData('t');
        lcdData('=');
        lcdData((valor / 100) % 10 + '0');
        lcdData((valor / 10) % 10 + '0');
        lcdData((valor / 1) % 10 + '0');
        printf("\nVoltar? *-Sim");
        kpDebounce();
        if (kpRead() != tecla) {
            tecla = kpRead();
            if (bitTst(tecla, 0)) {
                pwmSet1(0);
                on = 0;
```

```
void autoVentilador() { // temporiza o ven
    adcInit();
    pwmInit();

    int valor, on = 1, aux = 1;
    temporizador();
    pwmFrequency(2000);
    pwmSet1(50);

    while (on) {
        contador(cont);
        cont--;
        lcdCommand(0x80);
        valor = adcRead(0);
        pwmSet1(valor / 10);
        for (int x = 0; x < 5; x++)
            lcdData(' ');
        lcdData('P');
        lcdData('o');
        lcdData('t');
        lcdData('=');
        lcdData((valor / 100) % 10 + '0');
        lcdData((valor / 10) % 10 + '0');
        lcdData((valor / 1) % 10 + '0');
        for (int x = 0; x < 4; x++)
            lcdData(' ');
        printf("\n Voltar? Pot=0");

        if (cont == 0)
            on = 0;
```

6 - Resultados

Para demonstrar o funcionamento do projeto, foi disponibilizado um vídeo-tutorial, que pode ser acessado pelo seguinte link:

https://drive.google.com/file/d/1n_0lhKSW81eSC8HTwBS39DA7u-Zjys7v/view?usp=sharing

Todo o projeto está disponível publicamente no GitHub, no seguinte link:

https://github.com/MatheusCapucho/ProjetoFinalEmbarcado-Ventilador

7 - Conclusão

Ao desenvolver o projeto, foi possível observar o quão bem os componentes da placa funcionam em conjunto, desde que a configuração seja feita de acordo com o necessitado.

Todos os componentes utilizados atenderam o esperado, e ainda mais funcionalidades poderiam ser implementadas.