

# Projeto Final - Simulador MP3

GABRIEL B. VARGAS - 2019004680

Universidade Federal de Itajubá

gabriel.vargas@unifei.edu.br

8 de dezembro de 2020

## I. INTRODUÇÃO

O PIC18F4520 é um microcontrolador extremamente difundido, parte da popular família de 8 bits e núcleo de 14 bits, produzido pela MICROSHIP. De modo a se programar esse microcontrolador, utilizou-se a IDE MPLAB-X, desenvolvida também pela empresa MICROSHIP.

Para que os testes das aplicações fossem realizados, utilizou-se o software PICsimLab, desenvolvido pelo professor Luis Cláudio Gambôa. Tal placa possui suporte total ao PIC18F4520, e a placa de desenvolvimento simulada é a PICGenios.

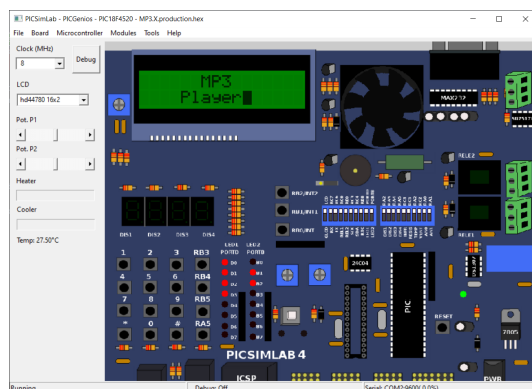


Figura 1: Placa PICGenios, simulada no software PICsimLab.

O projeto desenvolvido com essas ferramentas trata-se de um simulador de MP3 Player, e utiliza das seguintes funções da placa PICGenios:

- **Display LCD:** É utilizado para realizar a comunicação primária com o usuário, por meio de menus;
- **Display 7 Segmentos:** Apresenta informações nu-

méricas ao usuário, como duração restante da música;

- **Teclado:** Interface de comunicação do usuário com o sistema, utilizado para interagir com os menus;
- **Barramento de LEDs:** Demonstra o nível de volume das músicas;
- **Cooler:** É executado de modo a simular a caixa de som do aparelho de MP3.
- **Buzzer:** Aviso sonoro de que o processo finalizou, no caso, a música.

## II. DESENVOLVIMENTO

Inicialmente, foi realizado um planejamento do código, que seria separado em duas principais funções:

- Escolha da música;
- Execução da música.

Na primeira etapa, o menu inicial foi implementado, enquanto a segunda etapa seria responsável pelas funcionalidades propostas pelo projeto, que são executadas enquanto a música é tocada. Essas funções foram implementadas na biblioteca "songs.h", que foi desenvolvida especificamente para o projeto. Outras bibliotecas também foram utilizadas durante o desenvolvimento do projeto, que foram fornecidas pelo professor, e possuem funções para controlar, por exemplo, o teclado e o display LCD.

Além disso, de maneira a facilitar o controle de versionamento do código, foi utilizada a ferramenta git, sendo o projeto hospedado no GitHub.

## i. Inicialização da biblioteca

Durante o processo de criação do algoritmo, notou-se a necessidade de uma função exclusiva para a correta inicialização da biblioteca, assim como carregar as listas de músicas em uma struct com seus nomes e durações. Desse modo, a função "songsInit()" foi criada, assim como a struct "musica", que armazena uma string contendo o nome da música atual, e um inteiro, que armazena sua duração em segundos.

```

1 void songsInit(void) {
2     TRISC = 0x00;
3     for (unsigned int i = 0; i < 10; i++) {
4         musicas[i].duracao = duracoes[i];
5         strcpy(musicas[i].nome, (char*) nomes[i
6     ]);
7     }
8     return;
9 }
```

Acima está a implementação dessa função, que configura o TRISC, para futura ativação do buffer, e carrega as listas com os nomes das músicas e durações, em uma lista do tipo musica. Com isso, torna-se possível associar o nome de uma música com sua duração de maneira facilitada.

## ii. Escolher a Música

Sendo a biblioteca iniciada, e a lista de música prontas para ser utilizada, prosseguiu-se para a próxima etapa, que trata-se do menu principal de escolha da música. Essa etapa foi implementada na função "chooseSong()", que deve exibir no display LCD o nome da música atual, no display de 7 segmentos o índice da música, e realizar uma interação com o usuário pelo teclado.

```

1 void chooseSong(void) {
2     kpDebounce();
3     tecla = kpRead();
4     ssdUpdate();
5     if (bitTst(tecla, 3) || bitTst(tecla, 7)) {
6         flag = 1;
7         for (;;) {
8             ssdUpdate();
9             kpDebounce();
10            atraso_ms(10);
```

```

11            if ((kpRead() != tecla) || flag ==
12            1) {
13                tecla = kpRead();
14                if (bitTst(tecla, 3)) { //1
15                    if (indice == 0) {
16                        indice = 9;
17                    } else {
18                        indice -= 1;
19                    }
20                } else if (bitTst(tecla, 7)) {
21                    //2
22                    if (indice == 9) {
23                        indice = 0;
24                    } else {
25                        indice += 1;
26                    }
27                } else if (bitTst(tecla, 0)) {
28                    //3
29                    flag = 0;
30                    break;
31                }
32                lcdCommand(CLR);
33                lcdPosition(1, 0);
34                lcdStr("<-(1) (*) (2)->");
35                lcdPosition(0, 0);
36                lcdStr(musicas[indice].nome);
37                ssdDigit(indice, 3);
38                flag = 0;
39            }
40        }
41        playSong();
42    }
```

Acima encontra-se o código da função chooseSong(), que precisa que uma das teclas de seleção (1 ou 2) sejam pressionadas para que se prossiga para a próxima etapa. Com isso, entra-se em um for infinito, que realiza a interação com o usuário pelo teclado. A cada execução desse for, o display LCD é atualizado com as novas informações. Quando o usuário pressionar a teclada de seleção de música (\*), o for é interrompido, e a próxima função é executada.

## iii. Tocar a Música

Tendo a música sido selecionada, a próxima etapa é responsável por implementar as funcionalidades do sistema que dependem da execução da música, como Play/Pause e escolha do volume. Portanto, foi criada a função playSong(), que é responsável pelos métodos



Figura 2: Menu inicial de interação com o usuário, para escolha da música.

já citados, e também pelo buffer, cooler, e retorno ao menu principal.

```

1 void playSong() {
2     pwmInit();
3     lcdCommand(CLR);
4     lcdPosition(0, 0);
5     lcdStr(musicas[indice].nome);
6     lcdPosition(1, 0);
7     lcdStr("-(1)  (*)  (2)+");
8
9     tempo = musicas[indice].duracao;
10    pwmSet(100);
11    while (tempo != 0) {
12
13        minuto1 = (tempo / 60) % 10;
14        minuto2 = (tempo / 60) / 10;
15        segundo1 = (tempo % 60) % 10;
16        segundo2 = (tempo % 60) / 10;
17
18        ssdDigit(minuto2, 0);
19        ssdDigit(minuto1, 1);
20        ssdDigit(segundo2, 2);
21        ssdDigit(segundo1, 3);
22
23        for (unsigned char j = 0; j < 100; j++)
24        {
25            ssdUpdate();
26            atraso_ms(10);
27            kpDebounce();
28            tecla = kpRead();
29            if (bitTst(tecla, 3)) {
30                while(bitTst(tecla, 3)) {
31                    ssdUpdate();
32                    kpDebounce();
33                    tecla = kpRead();
34                }
35                alterarVolume(0);
36            }
37            else if (bitTst(tecla, 7)) {
38                while(bitTst(tecla, 7)) {
39                    ssdUpdate();
40                    kpDebounce();
41                    tecla = kpRead();

```

```

41        }
42        alterarVolume(1);
43    }
44    else if (bitTst(tecla, 0)) {
45        while(bitTst(tecla, 0)) {
46            ssdUpdate();
47            kpDebounce();
48            tecla = kpRead();
49        }
50        if (pause == 0) {pause = 1;}
51    else {pause = 0;}
52    }
53    else if (bitTst(tecla, 4)) {
54        while(bitTst(tecla, 4)) {
55            ssdUpdate();
56            kpDebounce();
57            tecla = kpRead();
58        }
59        ssdDigit(0, 0);
60        ssdDigit(0, 1);
61        ssdDigit(0, 2);
62        ssdDigit(0, 3);
63        return;
64    }
65    if (pause == 0) {
66        tempo -= 1;
67        pwmSet(100);
68    } else {
69        pwmSet(0);
70    }
71    }
72    TRISA=0x00;
73    pwmSet(0);
74    bitSet(TRISC, 1);
75    atraso_ms(500);
76    bitClr(TRISC, 1);
77    return;
78 }
79

```

Essa função realiza tais tarefas enquanto a música não tiver terminado, e, assim, é implementada em sua maior parte dentro de um loop while, que depende do tempo de duração da música. Nesse loop, o tempo é convertido para o display de 7 segmentos, e o LCD se mantém ativo com o nome da música e os comandos. A interação com o usuário é realizada pelo teclado, mas, nesse caso, foi utilizada uma estrutura if/while para cada uma das teclas utilizadas, de modo a evitar problemas no debouncing.

Essa estrutura está disposta dentro de um for que é executado 100 vezes, cada um com 10ms de atraso,

afim de atualizar o tempo ao fim desse ciclo, enquanto ainda mantém a interatividade com o usuário. Ao fim da música, os displays de 7 segmentos e o cooler são desligados, e o buffer ativado por meio segundo.



Figura 3: Menu de interação enquanto a música é executada.

#### iv. Alterar Volume

Por fim, houve a necessidade de outra função auxiliar, a `alterarVolume()`. Essa função é chamada pela `playSong()` quando a tecla 1 ou 2 é pressionada. Tal função soma, ou subtrai o volume atual, e o exibe no barramento de LEDs brevemente.

```

1 void alterarVolume(char opt) {
2     if (opt == 1) {
3         if (volume != 8) { volume += 1; }
4     } else {
5         if (volume != 0) { volume -= 1; }
6     }
7     unsigned char old_D, old_A;
8     old_D = TRISD;
9
10    PORTA = 0x00;
11    TRISD = 0x00;
12
13    if (volume == 0) {
14        PORTD = 0b00000000;
15    } else if (volume == 1) {
16        PORTD = 0b10000000;
17    } else if (volume == 2) {
18        PORTD = 0b11000000;
19    } else if (volume == 3) {
20        PORTD = 0b11100000;
21    } else if (volume == 4) {
22        PORTD = 0b11110000;
23    } else if (volume == 5) {
24        PORTD = 0b11111000;
25    } else if (volume == 6) {
26        PORTD = 0b11111100;
27    } else if (volume == 7) {
28        PORTD = 0b11111110;
29    } else if (volume == 8) {
30        PORTD = 0b11111111;

```

```

31    }
32    atraso_ms(500);
33    TRISD = old_D;
34 }
35

```

#### v. Main

Por fim, a função `main` foi implementada de maneira simples, apenas iniciando as bibliotecas importadas e preparando a apresentação do projeto. Além disso, a função `chooseSong()` é chamada dentro do `for` infinito.

```

1 void main(void) {
2
3     songsInit();
4     lcdInit();
5     ssdInit();
6     kpInit();
7     lcdCommand(0N);
8
9     lcdPosition(0, 6);
10    lcdStr("MP3");
11    lcdPosition(1, 4);
12    lcdStr("Player");
13    atraso_ms(5000);
14
15    lcdCommand(CLR);
16    lcdPosition(0, 0);
17    lcdStr("Escolha a musica");
18    for (;;) {
19        lcdPosition(1, 0);
20        lcdStr("<-(1) (*) (2)->");
21        chooseSong();
22    }
23 }
24

```