Despertador Inteligente
Sistema de Alarme que Gera Disposição ao Usuário

Vitor Jacinto Sulzbach
Universidade de Brasília - UNB
Faculdade do Gama - FGA
Gama - DF
05/07/2019
Email: vjsulzbach@hotmail.com
GitHub: https://github.com/Vitorsulzbach

1. Introdução

O capitalismo moderno traz consigo uma necessidade crescente de máxima eficiência em qualquer processo realizado, empresas de todos os portes estão buscando profissionais de alto rendimento para atender tal necessidade. Está pressão social amplifica problemas graves como stress crônico, depressão e até suicídio. Outros problemas menores que afetam diretamente a qualidade de vida da população também podem ter a causa relacionada a estresse e acumulo de trabalho, dentre eles estão a falta de tempo para a família ou hobbies e sedentarismo.

Em meio a tantos contras é difícil não ver esta cruzada moderna como vilã, porém ela se faz necessária para o progresso que sempre guiou a sociedade a tempos melhores. Nos resta então aprender a lidar com o stress da vida do novo milênio e descobrir maneiras de restaurar o equilíbrio do cotidiano com saúde e qualidade de vida.

Uma via saudável para alcançar esta eficiência é ter um bom ciclo diário de sono e acordar disposto para encarar os desafios diários, ter níveis controlados de cortisol, o hormônio do stress que nos ajuda a acordar e iniciar o dia, e melatonina, considerado o hormônio do sono que tem sua produção associada a menor incidência luminosa. Se atentando aos detalhes é possível aproveitar melhor a vida, com mais qualidade e saúde, menos esforço e sem deixar de lado o trabalho ou estudo. Tendo em vista as pessoas que tentam manter este equilíbrio nasce a proposta deste projeto.

2. Objetivos

Arquitetar e construir um sistema despertador otimizado para um acordar suave, porém efetivo. Objetivo principal do projeto é ser capaz de despertar ao máximo possível aqueles que sentem muita dificuldade de iniciar o dia devido a sonolência e preguiça matinal, porém também deve ser capaz de acordar aqueles que possuem sono pesado.

3. Projeto

Antes de montar qualquer projeto é necessário realizar uma pesquiza para averigurar quais métodos melhor se encaixam com os objetivos.

3.1. Despertar Suave

Para muitos é um esforço muito grande levantar ao acordar, aperta-se o botão soneca do despertador seguidamente e quando consegue sair da cama não há disposição alguma, ao passo que quando se acorda ao natural, como quando se acorda com a luz do dia atravessando a janela e o barulho que advém da mesma, não há resistência do corpo a iniciar o dia.

A grande diferença entre estes dois casos é a suavidade em se acordar, em um caso o usuário acorda com um barulho alto que o tira muitas vezes de um sono REM, o sono mais pesado e essencial, enquanto no outro caso o corpo tem tempo para recobrar a consciência, o organismo passa por estágios menores do sono e consegue recuperar o vigor. Com um despertar suave o usuário acorda com o sentimento de que o tempo de dormir acabou e é hora de levantar.

É necessário então escolher uma ferramenta que possuia saída de audio e possíbilidade de controle de uma lâmpada comum. Diversos microcontroladores são capazes de realizar tal tarefa, na maioria sendo necessário a utilização de periféricos, mas como também há o desejo de expandir o projeto para permitir a integração com mídias de vídeo, como acesso ao youtube após o alarme, foi escolhido a Single Board Computer Raspberry Pi, que conta com saída HDMI, saída de áudio e permite a utilização de um sistema operacional carregado de funcionalidades.

3.2. Levantar obrigatório

É sempre um incomodo sair da cama ao acordar, porém se manter em pé após levantar não é difícil. O ato de se levantar age como uma barreira de potencial que precisa ser quebrada para que o cotidiano se inicie.

Com o objetivo de quebrar está barreira o usuário deverá se levantar e se dirigir ao sistema e realizar o minigame proposto para desativar o alarme. Como o despertar será lento o usuário já estará bem desperto e após o jogo também está de pé.

4. Circuito

O circuito a ser utilizado se divide em 3 partes, um realiza a iluminação do ambiente, a outra é a circuitaria para desligar o alarme a última é a própria raspberry que junta os blocos com o software.

4.1. Iluminação

A iluminação gradual pode ser realizada através de um relé que recebe como entrada um GPIO da raspberry. Para controlar o rele é necessário usar um TIP131, de forma a enviar os 5 volts e 56mA necessários para polarizar a bobina do rele.

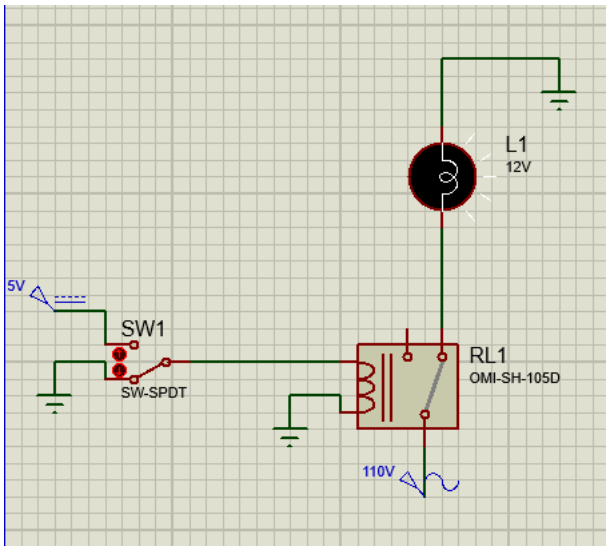


Figure 1. Relé

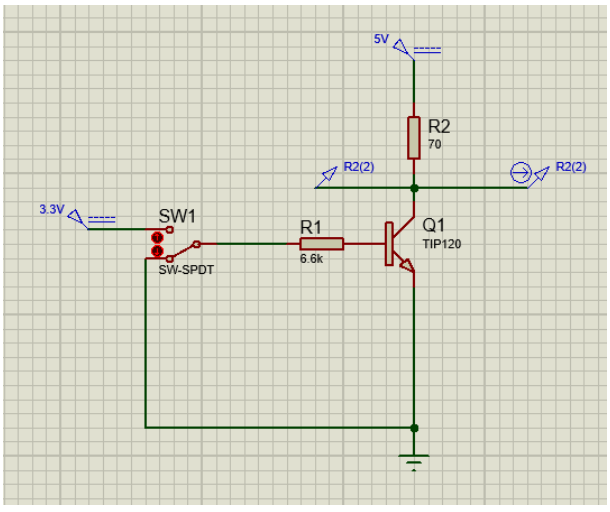


Figure 2. Drive para Relé

4.2. Minigame

O alarme é desligado por um jogo simples, composto por 5 LEDs. Gera-se um ciclo onde uma LED é acesa de cada vez, o usuário precisa pressionar o botão quando a LED do meio estiver acesa 3 vezes seguidas, então o alarme para e se inicia um vídeo pré-determinado para auxílio de produtividade pela amanhã.

4.3. Controle do despertador

O despertador é controlado via teclado e LCD, seu funcionamento não depende de circuito exterior além do seu próprio módulo. Não é necessário a utilização de mouse ou teclado, a conceito do controle é que a raspberry não seja uma Single Board Computer, mas um despertador que tem saida HDMI para execução de vídeos. O mesmo botão utilizado no mini game também é utilizado para parar o vídeo.

5. Software

O software construído é composto por dois arquivos, um que executa o toque de alarme e recebe um sinal para parar, e outro que controla quando este toque deve ocorrer.

5.1. Execução de Toque

Como mostra a figura abaixo o main é apenas a especificação de um sinal e o toque infinito de uma música.

```
int main(int argc, char *argv[]){
        signal(SIGUSR1, tratamento_SIGUSR1);
        while(1)
                system("aplay -q Music.wav");
        return 0;
}
```

Figure 3. Despertador.c

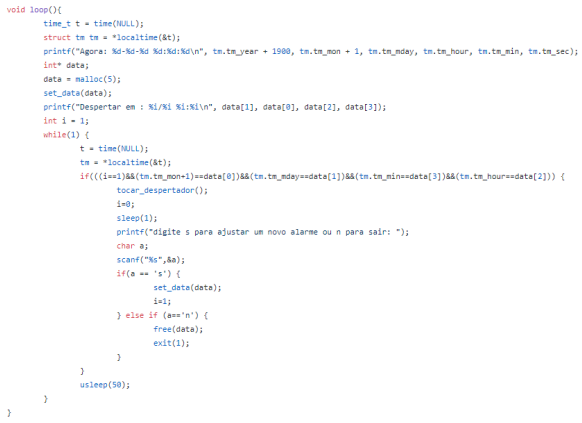
A parte de tratamento do sinal recebido é onde o alarme para, como a música toca através do comando system(aplay) é necessário obter o valor do pid do processo aplay através do comando pidof, então este valor é gravado em um arquivo, que é lido para então parar a música, depois basta finalizar a execução com um exit(1).

```c
void tratamento_SIGUSR1() {
        char *b;
        char dt[40] = "(kill -9 ";
        char f[] = ") &>/dev/null";
        system("rm -f a.txt");
        system("pidof aplay | tee a.txt");
        b = le_arq_texto("a.txt");
        strcat(dt,b);
        strcat(dt,f);
        system(dt);
        exit(1);
}
```

Figure 4. Sigint main

5.2. Controle

O controle é composto por várias funções, a main é responsável por determinar a hora atual através da biblioteca time, através de alocação dinâmica uma variável tipo int é preparada para passar por uma função que pergunta ao usuário a hora de despertar e armazena nesta variável, para então entrar em um loop de onde só sai quando a hora certa chega, onde se pode sair do programa ou programar um novo alarme.



```c
void loop(){
	time_t t = time(NULL);
	struct tm tm = *localtime(&t);
	printf("Agora: %d-%d-%d %d:%d:%d\n", tm.tm_year + 1900, tm.tm_mon + 1, tm.tm_mday, tm.tm_hour, tm.tm_min, tm.tm_sec);
	int* data;
	data = malloc(5);
	set_data(data);
	printf("Despertar em : %i/%i %i:%i\n", data[1], data[0], data[2], data[3]);
	int i = 1;
	while(1) {
		t = time(NULL);
		tm = *localtime(&t);
		if(((i==1)&&(tm.tm_mon+1)==data[0])&&(tm.tm_mday==data[1])&&(tm.tm_min==data[3])&&(tm.tm_hour==data[2])) {
			tocar_despertador();
			i=0;
			sleep(1);
			printf("digite s para ajustar um novo alarme ou n para sair: ");
			char a;
			scanf("%s",&a);
			if(a == 's') {
				set_data(data);
				i=1;
			} else if (a=='n') {
				free(data);
				exit(1);
			}
		}
		usleep(50);
	}
}
```

Figure 5. Loop.c

6. Apêndice

Despertador.c

```
#include
<stdio.h>
            #include <stdlib.h>
            #include <unistd.h>
            #include <errno.h>
            #include <signal.h>
            #include <string.h>

            char* le_arq_texto(char *nome_arquivo);
            int tam_arq_texto(char *nome_arquivo);
            void tratamento_SIGUSR1();

            int main(int argc, char *argv[]){
                    signal(SIGUSR1, tratamento_SIGUSR1);
                    while(1)
                            system("aplay -q Music.wav");
                    return 0;
            }

            void tratamento_SIGUSR1() {
                    char *b;
                    char dt[40] = "(kill -9 ";
                    char f[] = ") &>/dev/null";
                    system("rm -f a.txt");
                    system("pidof aplay | tee a.txt");
                    b = le_arq_texto("a.txt");
                    strcat(dt,b);
                    strcat(dt,f);
                    system(dt);
                    exit(1);
            }

            char* le_arq_texto(char *nome_arquivo){
                    FILE *arq = fopen(nome_arquivo, "r");
                    int long g = tam_arq_texto(nome_arquivo);
                    char *a;
                    a = malloc(g+3);
                    fread(a, 1, g, arq);
                    fclose(arq);
                    return a;
```

```
}

int tam_arq_texto(char *nome_arquivo){
        FILE *arq = fopen(nome_arquivo, "r");
        fseek(arq, 0, SEEK_END);
        int size = ftell(arq);
        return size;
}
```

Loop.c

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
#include <sys/wait.h>
#include <errno.h>
#include <signal.h>
#include <string.h>
#include <wiringPi.h>
#include <time.h>

int pularalarme =0;
void tocar_despertador();
void setup_io();
void loop();
int* set_data(int *data);
int t = 400000;
int a=0;
void win_animation();
void lose_animation();
void troca_led(int signum);
int n=0;

int main(int argc, char *argv[]){
        setup_io();
        loop();
        return 0;
```

```
}


void loop(){
	time_t t = time(NULL);
	struct tm tm = *localtime(&t);
	printf("Agora: %d-%d-%d %d:%d:%d\n", tm.tm_year + 1900, tm.tm_mon +
1, tm.tm_mday, tm.tm_hour, tm.tm_min, tm.tm_sec);
	int* data;
	data = malloc(5);
	set_data(data);
	printf("Despertar em : %i/%i %i:%i\n", data[1], data[0], data[2],
data[3]);
	fflush(stdout);
	int i = 1;
	while(1) {
		t = time(NULL);
		tm = *localtime(&t);

	if(((i==1)&&(tm.tm_mon+1)==data[0])&&(tm.tm_mday==data[1])&&(tm.tm_mi
n==data[3])&&(tm.tm_hour==data[2])) {
			tocar_despertador();
			i=0;
			sleep(1);
			printf("digite s para ajustar um novo alarme ou n para
sair: ");
			char b;
			scanf("%s",&b);
			if(b == 's') {
				set_data(data);
				i=1;
			} else if (b=='n') {
				free(data);
				exit(1);
			}
		}
		usleep(50);
	}
}

int* set_data(int *data){
	printf("Digite o mês para despertar: ");
	scanf("%d", &data[0]);
	if(data[0]==55)
		tocar_despertador();
	while(data[0]>12) {
```

```
		printf("Digite um mês válido: ");
		scanf("%d", &data[0]);
	}
	printf("Digite o dia para despertar: ");
	scanf("%d", &data[1]);
	while(data[1]>31) {
		printf("Digite um dia válido: ");
		scanf("%d", &data[1]);
	}
	printf("Digite a hora para despertar: ");
	scanf("%d", &data[2]);
	while(data[2]>24) {
		printf("Digite uma hora válida: ");
		scanf("%d", &data[2]);
	}
	printf("Digite o minuto para despertar: ");
	scanf("%d", &data[3]);
	while(data[3]>60) {
		printf("Digite um minuto válido: ");
		scanf("%d", &data[3]);
	}
	return data;
}

void setup_io() {
	if(wiringPiSetup() == -1){
		printf("deu ruim no setup!\n");
		exit(1);
	}
	pinMode(0,OUTPUT);
	pinMode(1,OUTPUT);
	pinMode(2,OUTPUT);
	pinMode(3,OUTPUT);
	pinMode(4,OUTPUT);
	pinMode(6,OUTPUT);
	pinMode(5,INPUT);
	digitalWrite(6,HIGH);
	signal(SIGALRM, troca_led);
}

void tocar_despertador() {
	printf("tocando alarme!\n");
	digitalWrite(6,LOW);
	int chilld;
	int i = 0;
	chilld = fork();
```

```
        if(childd==0){
                char *arg[] = {"./emb1",NULL};
                execvp(arg[0], arg);
                exit(1);
        }
        ualarm(t,0);
        while(i==0) {
                int d = 0;
                while(d==0){
                        if(digitalRead(5)==1){
                                ualarm(0,0);
                                if(a==2){
                                        win_animation();
                                        n++;
                                        if(n==1)
                                                t=200000;
                                        if(n==2)
                                                t=100000;
                                        if(n==3){
                                                kill(childd, SIGUSR1);
                                                digitalWrite(a,LOW);
                                                digitalWrite(6,HIGH);
                                                d=1;
                                        }
                                } else {
                                        lose_animation();
                                }
                                digitalWrite(a,LOW);
                                a=4;
                                if(pularalarme==1){
                                        d=1;
                                }
                                ualarm(t,0);
                        }
                }
                ualarm(0,0);
                int c;
                c=fork();
                if (c==0) {
                        system("chromium-browser
https://www.youtube.com/tv#/watch?v=k7BhL96l-jA");
                        exit(1);
                }
                sleep(5);
                while(1){
                        if(digitalRead(5)==1){
```

```
                        system("pkill -3 chromium");
                        sleep(2);
                        break;
                }
                usleep(100);
        }
        i++;
    }
}

void win_animation(){
    for(int i=0;i<25;i++){
        usleep(20000);
        digitalWrite(a,LOW);
        usleep(20000);
        digitalWrite(a,HIGH);
    }
}

void lose_animation(){
    for(int i=0;i<5;i++){
        usleep(40000);
        digitalWrite(a,LOW);
        usleep(40000);
        digitalWrite(a,HIGH);
    }
}

void troca_led(int signum){
    digitalWrite(a,LOW);
    a=(a+1)%5;
    digitalWrite(a,HIGH);
    ualarm(t,0);
}
```