Universidade Federal de Pernambuco
Centro de Tecnologia e Geociências
Departamento de Eletrônica e Sistemas

Relatório do Trabalho 2
Controle de iluminação e temperatura em Discoteca

Grupo:
Guilherme de Souza Bastos
Karl Vandesman de Matos Sousa

Disciplina:
Microcomputadores

Professor:
Mauro dos Santos

Recife - PE
Junho de 2017

1 Introdução

Este trabalho consiste no projeto de controle de temperatura e luminosidade de um ambiente de discoteca, em que foi utilizado o microcontrolador PIC16F877A, da Microchip. Para ter informações sobre a luminosidade do ambiente, é usado um LDR (do inglês, *Light Dependent Resistor*). Quanto à temperatura, é usado um sensor LM35.

O controle de temperatura é seguido segundo a Tabela 1. Tem-se à disposição quatro ares-condicionados e um conjunto de ventiladores que serão acionados pela saída PWM do microcontrolador.

Tabela 1: Valores comparativos para controle de temperatura.

Faixa de temperatura	**Ventiladores (PWM)**	**Ar-1**	**Ar-2**	**Ar-3**	**Ar-4**
$\leq$ **20ºC**	10% Pmax	desligado	desligado	desligado	desligado
$\geq$ **21ºC a** $\leq$ **25ºC**	30% Pmax	ligado	desligado	desligado	desligado
$\geq$ **26ºC a** $\leq$ **30ºC**	75% Pmax	ligado	ligado	desligado	desligado
$\geq$ **31ºC**	90% Pmax	ligado	ligado	ligado	ligado

Para o controle de luminosidade, são dispostas quatro luminárias no ambiente. O acionamento é feito segundo a relação da Tabela 2.

Tabela 2: Valores comparativos para controle de luminosidade.

Luz do ambiente	**Lum1**	**Lum2**	**Lum3**	**Lum4**
claro	desligada	desligada	desligada	desligada
sombra	ligada	desligada	ligada	desligada
escuro	ligada	ligada	ligada	ligada

2 Metodologia

2.1 Etapas para realização do projeto

Para realizar o desenvolvimento do projeto de acordo com o requisitado foi realizado o seguinte passo a passo:

- Realizar uma visão geral do projeto;
- Definir o modo de operação para a interface Homem/Máquina, a partir das entradas e saídas do microcontrolador;
- Criar um fluxograma para ilustrar a lógica central do programa;
- Configurar as portas do PIC16F877A de acordo com as especificações da sua folha de dados. Isso é necessário pois portas específicas possuem a habilidade de interrupção, outras são usadas como entrada analógica para o conversor A/D interno, entre outras coisas;
- Desenvolver o código fonte em *assembly*;
- Simular o funcionamento do sistema no Proteus.

2.2 Recursos utilizados

O MPLAB foi o *software* utilizado para desenvolver o código em *assembly* para o microcontrolador. O *software* Proteus foi utilizado a fim de simular o funcionamento do sistema, utilizando o código *assembly* no microcontrolador e as entradas e saídas necessárias.

O sistema principal de controle automático foi realizado a partir de um PIC 16F877A.

Os sensores utilizados foram o LM35, para medição de temperatura, e um LDR, que varia sua resistência conforme mudança na luminosidade incidente. Na Figura 1 estão representados ambos.

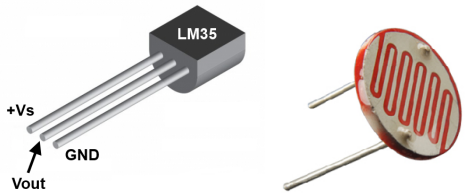

Figura 1: Sensores para medição de temperatura e luminosidade: LM35 e LDR.

A elaboração do projeto foi dividida em duas versões, cuja diferença está unicamente na programação do microcontrolador. Apesar de para ambas as versões o resultado ser identico para os usuários, há uma economia de energia do sistema ao serem utilizadas interrupções, pois isso possibilita o uso do comando *sleep*. Dessa forma, na segunda versão foram adicionadas interrupção *Timer 0*, interrupção externa por RB0 e a interrupção de periférico do conversor A/D.

3 Desenvolvimento

3.1 Versão 1 - Sem interrupções

Para os sensores foram feitas medições na simulação do Proteus para se verificar a tensão de saída conforme a variação no ambiente. A partir dos pontos registrados, foram gerados gráficos e feita uma reta para verificar a linearidade nos intervalos considerados. Os gráficos para ambos os sensores estão apresentados na Figura 2. A partir do registro desses valores, foi possível fazer uma análise dos valores digitais que seriam usados na programação do microcontrolador para as faixas de níveis de controle consideradas.

A partir dos sinais analógicos gerados pelos sensores, deve-se fazer um processamento de dados no microcontrolador. Para isso, é necessária a conversão Analógica Digital (A/D), tarefa que pode ser feita internamente ao PIC16F877A, que dispõe de um conversor A/D, cuja representação em diagrama de blocos está na Figura 3. É possível visualizar que realiza-se a conversão de apenas uma entrada analógica por vez, sendo preciso o controle por meio dos *bits* 5 a 3 (CHS2:CHS0) do registrador ADCON0.

Outra questão relevante é definir as portas como analógicas, por meio dos *bits* 3 a 0 (PCFG3:PCFG0). Definindo 0101 para esses bits, tem-se RA0 e RA1 como portas analógicas, e as restantes definidas como digitais, com AN3 sendo tensão de referência positiva ($V_{ref}+$).

A lógica de funcionamento do projeto de maneira mais simples está apresentada no fluxograma das figuras 4 e 5. Essa é uma forma de se definir a ideia geral do que será posteriormente implementado pela codificação no *assembly* no MPLAB.

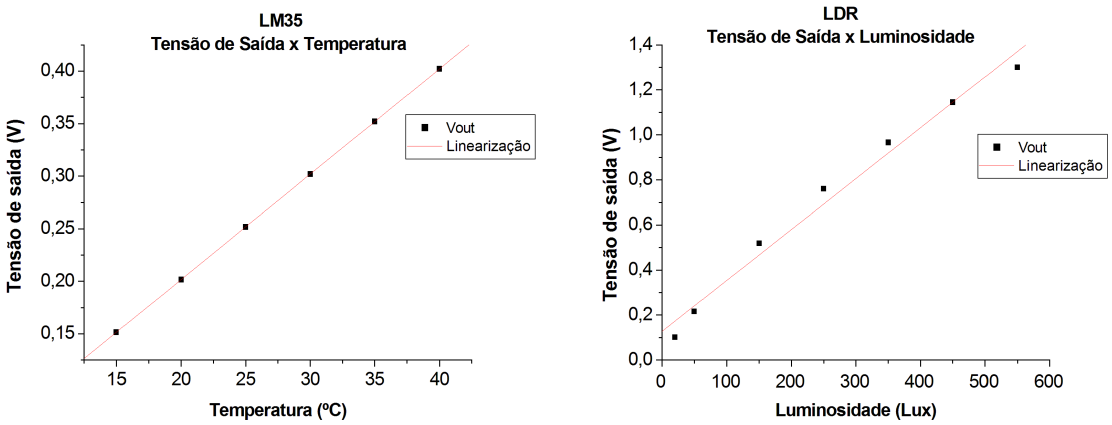

Figura 2: Gráfico para pontos medidos no Proteus dos sensores.

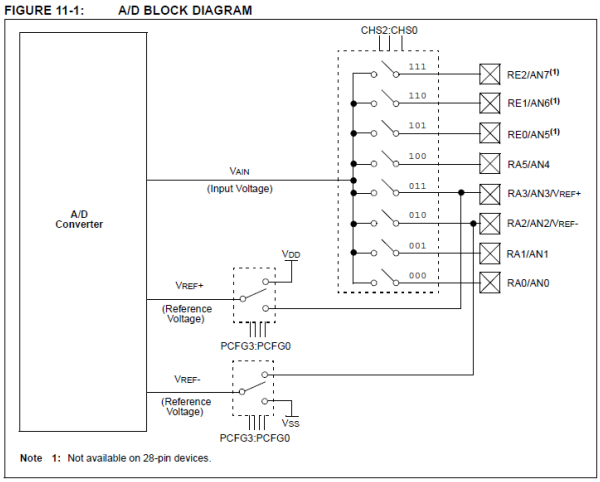

Figura 3: Diagrama de blocos do conversor A/D do PIC16F877A.

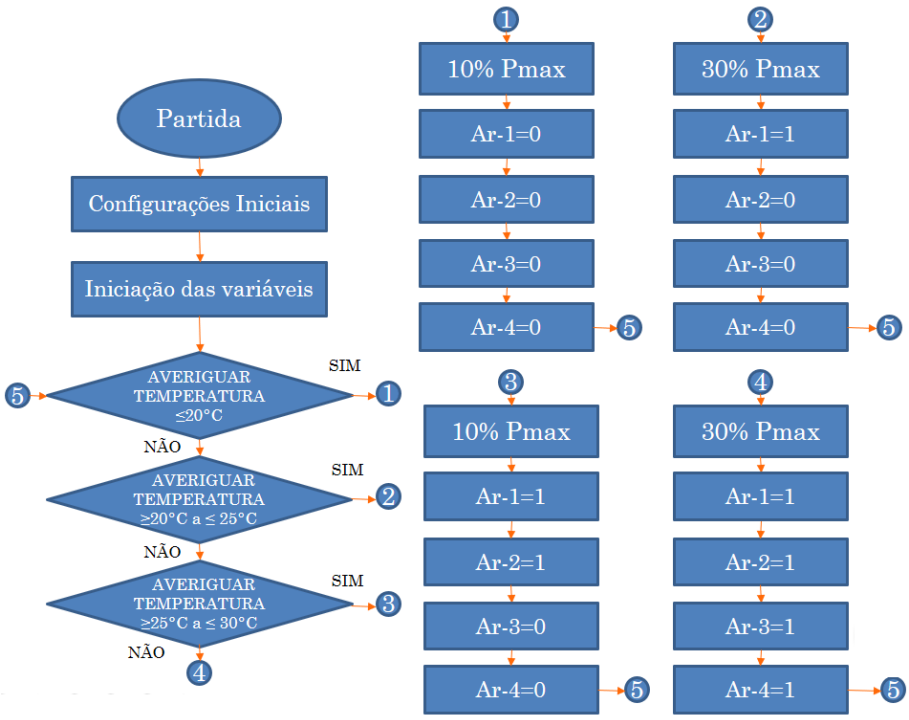

Figura 4: Primeira parte do fluxograma do projeto na versão 1.

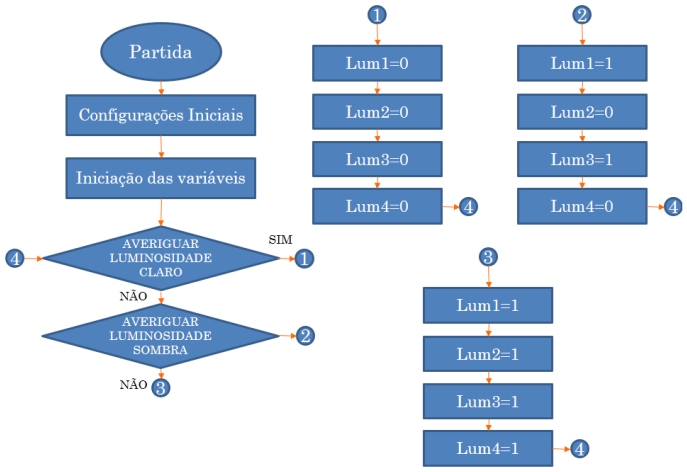

Figura 5: Segunda parte do fluxograma do projeto na versão 1.

Implementando-se esse código no MPLAB, chegou-se ao apresentado na Figura 6. Inicialmente é checada a presença de pessoas no ambiente, e caso não tenha ninguém, as saídas são desativadas pela rotina DESATIVA_TUDO e o comando *sleep* é acionado, e só poderá retornar ao funcionamento caso pessoas cheguem no ambiente e o sensor de presença gere a interrupção. No caso de houver pessoas na discoteca, inicia-se a leitura de dos sensores, e ativação dos respectivos controles necessários.

```
;*************************************************
;*               ROTINA PRINCIPAL                *
;*************************************************
MAIN_LOOP
    BTFSC   PRESENCA            ;Testa se o sensor de presença está ativado
    GOTO    INICIO_CONTROLE     ;Se tiver, começa o início da leitura dos sensores e controle

    CALL    DESATIVA_TUDO       ;Se não tiver, desativa todas as saídas
    SLEEP                       ; O MCU estará em Sleep, até que o sensor de presença
    NOP                         ; verifique que há alguém no ambiente

INICIO_CONTROLE
    BSF     LIGADO                  ;Aciona o LED avisando que o sistema está ligado

    CALL    LER_TEMPERATURA         ;Faz a leitura do sensor de temperatura
    CALL    CONTROLE_TEMPERATURA    ;A partir da leitura, realiza o controle de temperatura
                                    ;Fazendo controle do acionamento do conjunto de ventiladores
                                    ;e ares-condicionados

    CALL    LER_LUMINOSIDADE        ;Faz a leitura do sensor de luminosidade
    CALL    CONTROLE_LUMINOSIDADE   ;A partir da leitura, realiza o controle de temperatura
                                    ;Fazendo controle do acionamento das luminárias

    CALL    DELAY_TA                ;Depois de lidos os sensores, espera-se um tempo para a nova medição
                                    ;já que as variáveis mudam lentamente
    GOTO    MAIN_LOOP
```

Figura 6: Códigos em *assembly* da rotina principal.

3.2 Versão 2 - Com interrupções

As interrupções utilizadas para a segunda versão do projeto foram as do contador *Timer 0*, interrupção externa por RB0, e a do conversor A/D.

Para esse fim, foram configurados os registradores especiais INTCON e PIE1. Para ativação das interrupções desejadas, foram ativados no INTCON os *bits* de habilitação Geral (GIE), de periférico (PEIE) e interrupção externa por RB0 (INTE). Já no PIE1, foi

setado apenas o *bit* de habilitação da conversão A/D (ADIE). Com isso, a configuração da interrupção está realizada.

Um cuidado deve ser tomado quanto a interrupção do conversor. Utilizado o comando *sleep*, o oscilador principal de 20MHz do PIC16F877A para o funcionamento, logo, o microcontrolador não realiza o processo de conversão e nem pode gerar o sinal de *wake-up* para retornar ao funcionamento normal. Diante disso, para o conversor A/D poder operar no modo *sleep*, a fonte de *clock* do conversor deve ser o RC (setado nos bits ADCS1:ADCS0=11), e a instrução *sleep* deve ocorrer logo após a instrução de início de conversão, setando o *bit* GO/$\overline{DONE}$. No entanto, é utilizado como saída o PWM, que utiliza o *Timer 1* que não funciona durante o *sleep*, por isso, pode gerar eventuais erros no controle de velocidade dos ventiladores, e assim o *sleep* durante a conversão não foi utilizado.

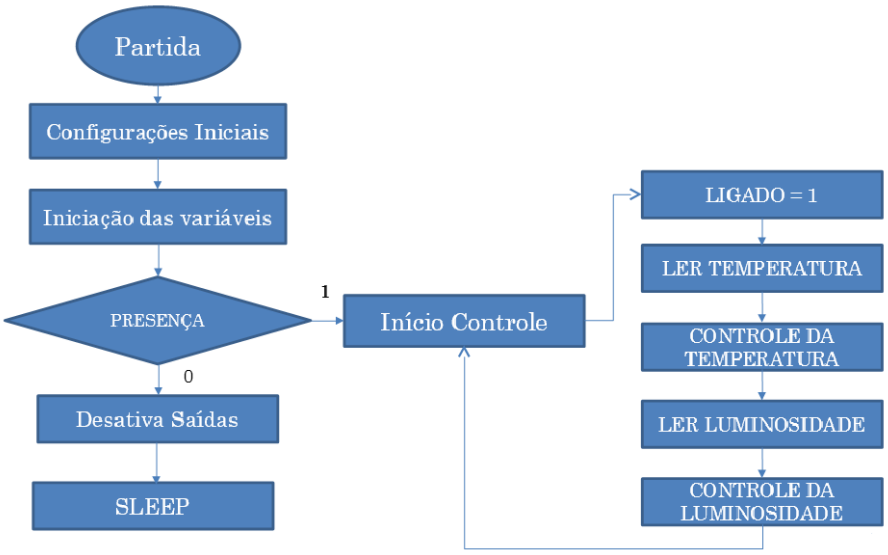

Figura 7: Primeira parte do fluxograma do projeto na versão 2.

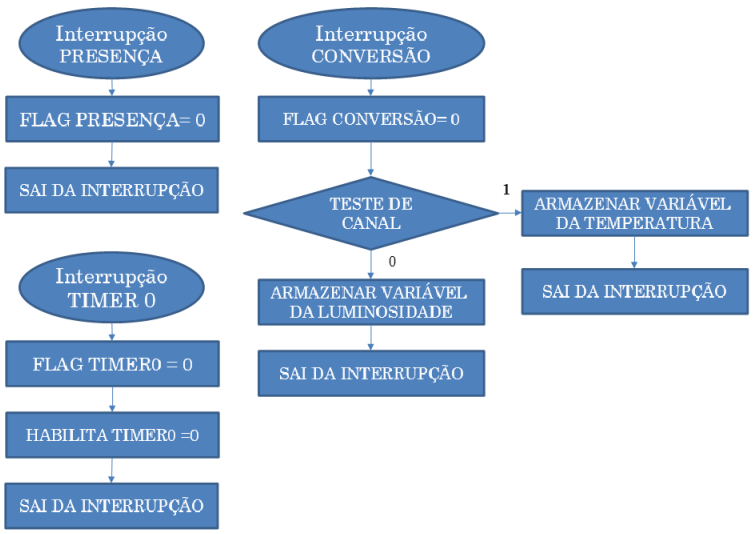

Figura 8: Segunda parte do fluxograma do projeto na versão 2.

4 Resultados

Na Figura 10 é apresentado o resultado final do sistema desenvolvido no Proteus. Vê-se uma divisão do sistema no Proteus em duas áreas: interface com os usuários, onde pode ser visto o resultado do controle (velocidade dos ventiladores, acendimento das luminárias e acionamento dos ares-condicionados) e a parte de controle, em que estão o microcontrolador, seu oscilador e os sensores.

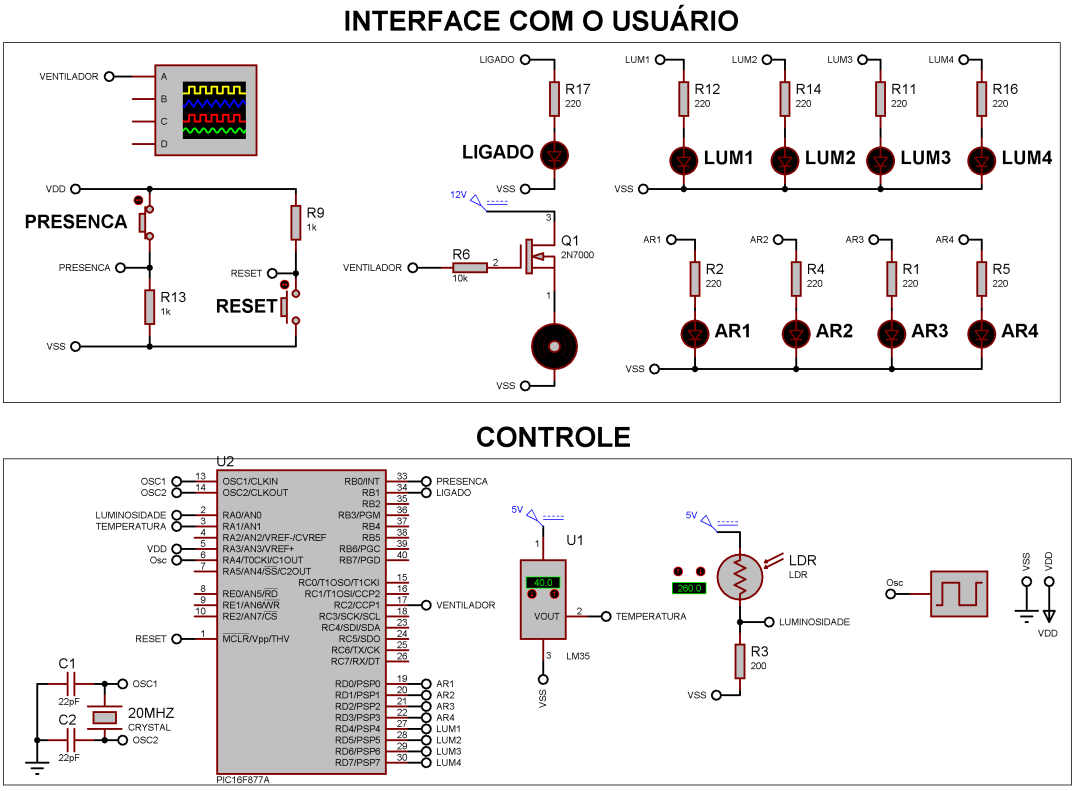

Figura 9: Visão geral do sistema implementado no Proteus.

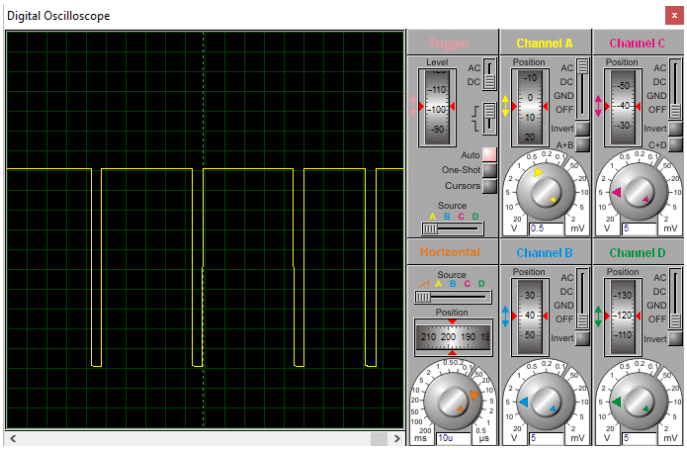

Figura 10: Saída PWM (temperatura $\geq$ 31ºC) vista pelo Oscilador Digital no Proteus.

5 Conclusões

Neste projeto foi realizado o controle automático de um sistema constituído por sensores de luminosidade e temperatura. Assim, como a informação dos sensores utilizados (LM35 e LDR) são variáveis analógicas, foi usado conversor Analógico-Digital para que o microcontrolador utilizado pudesse processar as informações e então enviasse os comandos de controle. Foi usado também uma forma de modulação bastante utilizado na eletrônica digital, a PWM, útil no controle mais suave partindo de um sistema digital.

Em um sistema eletrônico como o desenvolvido, um dos itens importantes a se considerar é a economia de energia. Considerando isso, o comando *sleep* do microcontrolador foi utilizado estrategicamente para que as leituras dos sensores fossem utilizadas em intervalos de tempo na escala de segundos, pois as variáveis consideradas (luminosidade e temperatura) tem variação lenta.

Em suma, o projeto desenvolvido apresentou sucesso quanto aos requisitos propostos, controlando de forma automática a luminosidade e temperatura do ambiente, visando a economia de energia e automatização do controle.