Engenharia de Controle e Automação
Universidade Federal de Santa Catarina
Arquitetura e Programação de Sistemas Microcontrolados
Professor: Werner Kraus Junior

Manual Técnico - Controle de Luminosidade

Fernando Battisti
Iago de Oliveira Silvestre
Igor Assis Rocha Yamamoto

.: developed by AUTOFIGI :.

2015

Sumário

Capítulo 1

Introdução

Neste manual serão descrito os aspectos técnicos do projeto implementado no microprocessador 8086 da Intel, idealizado e realizado pela equipe AutoFIGI. Serão discutidos aspectos tais quais como:

* Estruturamento do código

* Descritivo do código

* Interação com dispositivos externos.

O projeto de controle de luminosidade tem em vista automatizar sistemas de cortinas para ambientes com grandes variações de luminosidade externa e reduzir o custo em iluminação interna. O 8086 é um microprocessador de 16 bits da Intel criado para ser usado como CPU em um microcomputador, foi produzido de 1978 até a década de 90, este microprocessador foi muito importante para o sucesso da arquitetura x86, uma arquitetura que tinha em vista a retro-compatibilidade de processadores da Intel, e que foi utilizada em diversas CPUs.

Capítulo 2

Funcionamento

Apresentamos na Figura 2.1 um diagrama de funcionamento geral do sistema.

Nele podemos observar que, ao ser ligado, o programa reconhece o modo de operação (Manual ou Automático) e toma as devidas decisões.

Quando o modo for manual o programa pode subir a cortina ou descê-la, conforme solicitado pelo usuário.

Quando o modo for automático, *a priori* o programa verifica o sensor de luz externo. O sistema foi pensado de maneira que no período noturno a cortina deve estar fechada completamente e no período diurno ela deve controlar a luminosidade do ambiente interno com base no valor de referência requisitado pelo usuário. Assim, quando o valor da luminosidade externa está indicando escuridão total (luz externa está em 0%), a cortina desce totalmente e então retorna-se ao estado inicial de verificação. Mas por outro lado, se o valor não indicar ausência de luz (luz externa é maior que 0%), o controle do sistema passa para a próxima etapa de verificação.

O sensor de luz interno é então verificado. Se a diferença entre a luminosidade interna e a luminosidade desejada, tida como referência, é menor que o erro previamente estabelecido, retorna-se ao estado de verificação inicial. Porém se a diferença for maior que o erro, o programa verifica se o valor da luminosidade interna atual é maior ou menor que a desejada. Caso a luz do ambiente interno seja maior que o valor referência, a cortina deverá descer para que aquela diminua; se a luz interna for maior, a cortina deverá subir. Os motores são, então, acionados conforme o movimento da cortina.

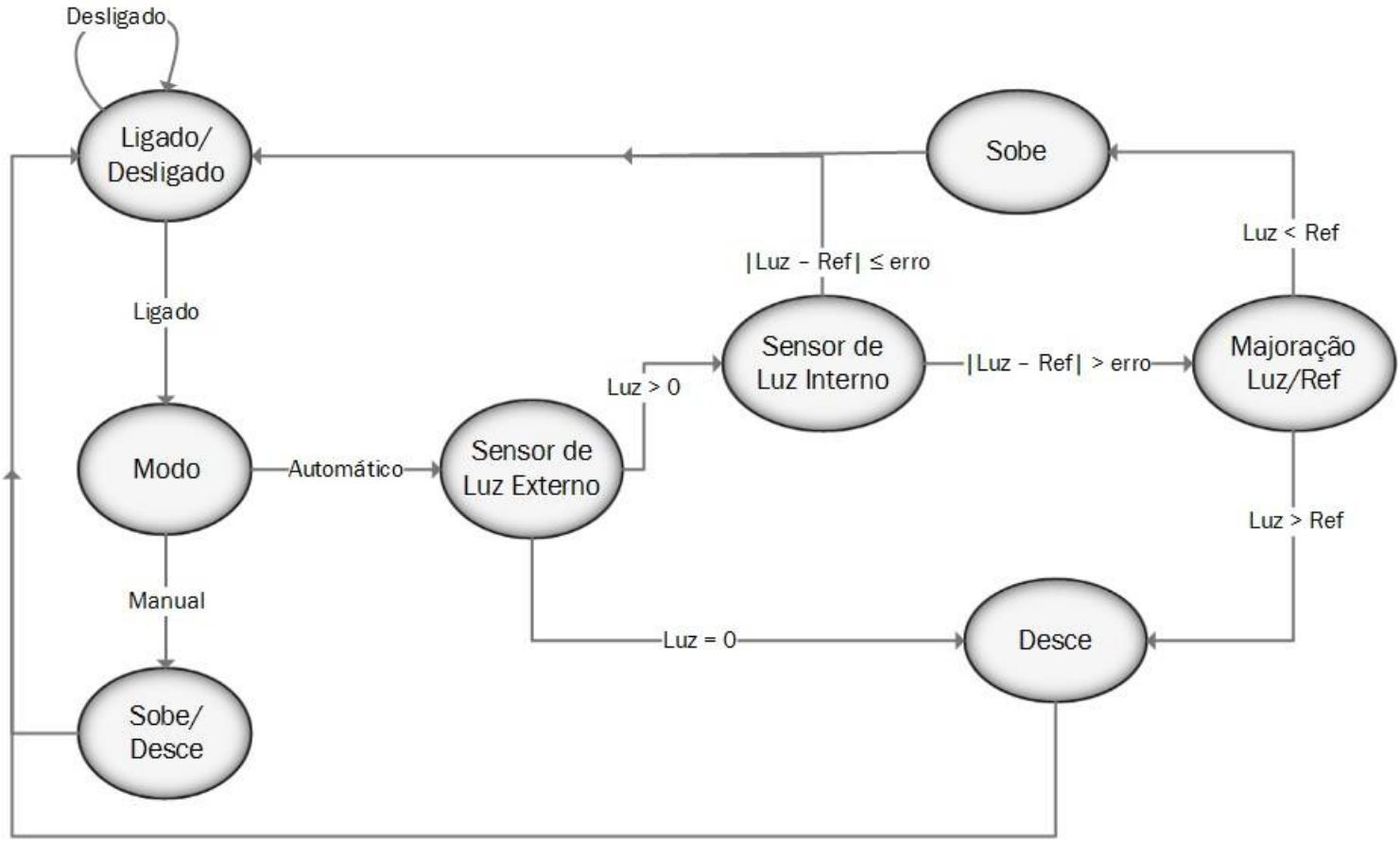

Figura 2.1: Funcionamento

Capítulo 3

Estruturamento do Código Fonte

Logo no inicio do código, são encontradas alguns comentários sobre a equipe que realizou o projeto, além da data quando o projeto foi finalizado entre outras informações. Após essas informações, temos a inicialização dos dispositivos externos que foram utilizados no projeto, o motor de passo e um dispositivo criado pelos integrantes do grupo que serão discutidos no tópico de interação com dispositivos externos. Além disso, temos a declaração das portas que são utilizadas no projetos para o controle da cortina automatizada. (Ver figura 3.1)

```
; Projeto da disciplina de Arquitetura e Programacao de Sistemas Microcontrolados - DAS5332
; Professor: Werner Kraus Junior
; Equipe: Fernando Battisti
;         Iago Silvestre de Oliveira
;         Igor Assis Rocha Yamamoto
; Abril de 2015

#start=Controle_de_Luminosidade.exe#
#start=Stepper_motor.exe#
LUMEXT equ 124           ; porta que le o valor da luminosidadde externa (sensor de luz externo)
LUM equ 125              ; porta que le o valor atual de luminosidade (sensor de luz interno)
BLOCK equ 126            ; porta com o valor de luminosidade bloqueada (giros do motor fazem essa definicao)
CTRLVAR equ 127          ; porta de controle
TARGET equ 128           ; porta com o valor desejado de referencia
```

Figura 3.1: Declarações Iniciais

Após as declarações iniciais, encontramos a segmentação do código. Os 3 segmentos que encontramos são:

* Segmento de Data: É utilizado para definirmos algumas variáveis globais que fornecem informações importantes para o sistema de controle da cortina, tais como o número máximo de voltas do motor e a margem de erro para luminosidade externa e interna. (Ver figura 3.2)

* Segmento de Interrupções: Este segmento está sendo usado para definir o que cada interrupção deve fazer (na Figura 3.2, o código foi retirado já que não é relevante a discussão de segmentação do código). (Ver figura 3.2)

* Segmento de Código: No começo deste estão as definições das interrupções na tabela de endereçoes do 8086, o que deve ser realizado primeiramente. Além disso, no segmente de código, também temos as sub-rotinas e funções que serão discutidas no descritivo do código. (Ver figura 3.2)

```
031 data segment
032
033     meiogiro db 20        ; variavel que controla os giros do motor
034
035     erro db 3             ; margem de erro para a luminosidade
036
037 ends
038
039
040 interrupt segment
041
042 trata90h:                ; tratador manual/automatico
043
044 trata91h:                ; tratador move pra baixo
045
046 trata92h:                ; tratador move para cima
047
048 trata93h:                ; tratador On/Off
049
050 trata94h:
051
052 trata95h:
053
054 ends
055
056
057 code segment
058 start:
059
060 def_trata90h:            ; define o tratador 90h na tabela de enderecos
061
062 def_trata91h:            ; define o tratador 91h na tabela de enderecos
063
064 def_trata92h:            ; define o tratador 92h na tabela de enderecos
065
066 def_trata93h:            ; define o tratador 93h na tabela de enderecos
067
068 def_trata94h:            ; define o tratador 94h na tabela de enderecos
069
070 def_trata95h:            ; define o tratador 95h na tabela de enderecos
071
```

Figura 3.2: Segmentação do Programa

Capítulo 4

Descritiva do Código Fonte

Para uma melhor compreensão do código fonte, algumas informações adicionais merecem destaque, como:

* Porta 124: Esta porta lê o valor da luminosidade externa através de um sensor de luminosidade localizado na parte externa do ambiente a ser utilizado o sistema. Ela mostra um valor na faixa de 0 a 100.

* Porta 125: Esta porta lê o valor da luminosidade interna através de um sensor de luminosidade localizado internamente no ambiente a ser utilizado o sistema. Ela mostra um valor na faixa de 0 a 100.

* Porta 126: Mostra o da luminosidade bloqueada, ou seja, o quanto a cortina está fechada (Os giros do motor fazem essa definição). Ela mostra um valor na faixa de 0 a 100.

* Porta 127: Porta de controle. Esta é a que merece mais enfoque, pois é ela quem indica se o sistema está ON/OFF, se está no modo MANUAL ou AUTOMATIC e se a cortina está subindo ou descendo. Como mostrado na figura 3.1, estão sendo usando 3 bits desta porta para o controle. O primeiro bit indica se o sistema está ON(xx1)/OFF(xx0); o segundo indica o movimento da cortina, que pode estar subindo(x1x) ou descendo(x0x); o terceiro indica o modo do sistema, que pode ser MANUAL(0xx)/AUTOMATIC(1xx).

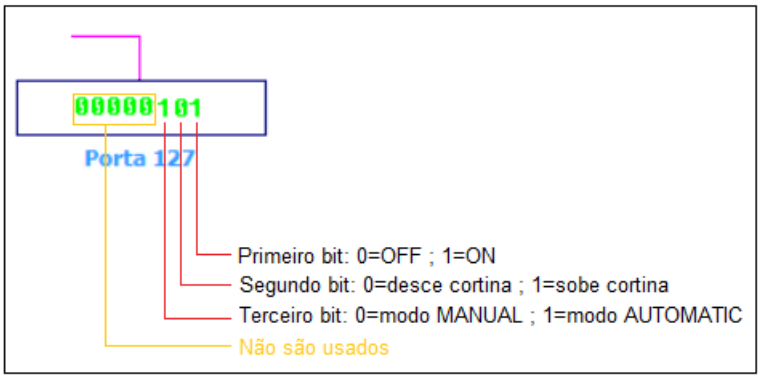

Figura 4.1:

* Porta 128: Esta porta mostra o valor desejado de referência, ou seja, o valor da luminosidade desejada pelo usuário. Ela mostra um valor na faixa de 0 a 100.

* Interrupção trata90h: Esta interrupção é usada pra selecionar o modo MANUAL/AUTOMATIC complementando o terceiro bit da porta 127. Associamos esta interrupção ao botão (2) da Figura 4.2.

* Interrupção trata91h: Esta interrupção é usada para mover a cortina para baixo, zerando o segundo bit da porta 127. Associamos esta interrupção ao botão (4) da Figura 4.2.

* Interrupção trata92h: Esta interrupção é usada para mover a cortina para cima, setando o segundo bit da porta 127. Associamos esta interrupção ao botão (3) da Figura 4.2.

* Interrupção trata93h: Esta interrupção é usada para ligar ou desligar o sistema (ON/OFF) complementando o primeiro bit da porta 127. Associamos esta interrupção ao botão (1) da Figura 4.2.

* Interrupção trata94h: Esta interrupção é usada para diminuir 5% o valor desejado. Para implementar, subtraimos 5 ao valor de TARGET caso ele já não esteja em 5% . Associamos esta interrupção ao botão (6) da Figura 4.2.

* Interrupção trata95h: Esta interrupção é usada para aumentar 5% o valor desejado. Para isso, adicionamos 5 ao valor de TARGET caso ele já não esteja em 95%. Associamos esta interrupção ao botão (5) da Figura 4.2.

Segue abaixo o código fonte implementado para realização do projeto.

```
; Projeto da disciplina de Arquitetura e Programacao de Sistemas Microcontrolados - DAS5332
; Professor: Werner Kraus Junior
; Equipe: Fernando Battisti
;         Iago Silvestre de Oliveira
;         Igor Assis Rocha Yamamoto
; Abril de 2015

#start=Controle_de_Luminosidade.exe#
#start=Stepper_motor.exe#

LUMEXT equ 124       ; porta que le o valor da luminosidadde externa (sensor de luz externo)
LUM equ 125          ; porta que le o valor atual de luminosidade (sensor de luz interno)
BLOCK equ 126        ; porta com o valor de luminosidade bloqueada (giros do motor fazem essa definicao)
CTRLVAR equ 127      ; porta de controle
TARGET equ 128       ; porta com o valor desejado de referencia

data segment
   meiogiro db 20    ; variavel que controla os giros do motor
   erro db 3         ; margem de erro para a luminosidade
ends
```

```
interrupt segment
trata90h:              ; tratador manual/automatico
   in al, CTRLVAR
   xor al, 100b        ; complementa o terceiro bit
   out CTRLVAR, al
   iret
trata91h:              ; tratador move pra baixo
   in al, CTRLVAR
   and al, 11111101b   ; zera o segundo bit
   out CTRLVAR, al
   iret
trata92h:              ; tratador move para cima
   in al, CTRLVAR
   or al, 010b         ; seta o segundo bit
   out CTRLVAR, al
   iret
trata93h:              ; tratador On/Off
   in al, CTRLVAR
   xor al, 001b        ; complementa o primeiro bit
   out CTRLVAR, al
   iret
trata94h:
   in al, TARGET
   cmp al, 5
   jae diminui_5
   iret
   diminui_5:
   sub al, 5
   out TARGET, al
   iret
trata95h:
   in al, TARGET
   cmp al, 95
   jbe aumenta_5
   iret
   aumenta_5:
   add al, 5
   out TARGET, al
   iret
ends

code segment
start:
def_trata90h:          ; define o tratador 90h na tabela de enderecos
   mov ax, interrupt
   mov ds, ax
   mov dx, offset trata90h
   mov ah, 25h
   mov al, 90h
   int 21h
def_trata91h:          ; define o tratador 91h na tabela de enderecos
   mov ax, interrupt
   mov dx, offset trata91h
   mov ah, 25h
   mov al, 91h
   int 21h
```

```
def_trata92h:          ; define o tratador 92h na tabela de enderecos
   mov ax, interrupt
   mov dx, offset trata92h
   mov ah, 25h
   mov al, 92h
   int 21h
def_trata93h:          ; define o tratador 93h na tabela de enderecos
   mov ax, interrupt
   mov dx, offset trata93h
   mov ah, 25h
   mov al, 93h
   int 21h
def_trata94h:          ; define o tratador 94h na tabela de enderecos
   mov ax, interrupt
   mov dx, offset trata94h
   mov ah, 25h
   mov al, 94h
   int 21h
def_trata95h:          ; define o tratador 95h na tabela de enderecos
   mov ax, interrupt
   mov dx, offset trata95h
   mov ah, 25h
   mov al, 95h
   int 21h
def_datasegment:       ; poe o segmento data em DS
   mov ax, data
   mov ds, ax
def_inicial:
   mov al, 30
   out TARGET, al      ; definicao do grau de luminosidade desejado
   mov al, 101b
   out CTRLVAR, al     ; valor inicial da variavel de controle
   mov al, ds:meiogiro
   out BLOCK, al       ; escreve o valor inicial do giro do motor na porta de bloqueio (20 meiogiros)
inicio:
   in al, CTRLVAR      ; le a variavel de controle
   and al, 001b        ; testa se o primeiro bit(on/off) esta ativo
   jnz controle        ; caso o bit esteja em 1(on) realiza o controle
   jmp inicio          ; caso contrario volta o inicio para realizar a verificacao novamente
controle:
   in al, CTRLVAR
   and al, 100b        ; testa se o terceiro bit(auto/manual) estao ativos
   jnz controle_auto   ; caso esteja pula para o controle automatico
controle_manual:
   in al, CTRLVAR
   and al, 010b        ; testa se o segundo bit(Up/Down) esta ativo
   jnz sobe            ; caso esteja vai para a 'funcao' que sobe a cortina
   jmp desce           ; caso contrario vai para a 'funcao' que desce a cortina
controle_auto:
   in al, LUMEXT       ; le o luz externa
   cmp al, 0           ; verifica se esta em zero (noite)
   je desce
   in al, TARGET       ; le o valor referencia(porta 128h)
   mov bl, al          ; move al para bl para poder ler outra porta
   in al, LUM          ; le o valor atual de luz(porta 125h)
   cmp al, bl          ; faz a verificacao de qual valor eh maior
   jb  LUMmenor
   sub al, bl          ; compara a diferenca do valor referencia com o valor atual para LUM > TARGET
   cmp al, ds:erro     ; verifica se a deiferenca esta dentro da margem de erro
```

```
   jb inicio          ; caso esteja, volta para o inico
   jmp desce          ; caso contrario, a cortina desce pois aqui o valor atual eh maior que a referencia
   LUMmenor:
   sub bl, al         ; compara a diferenca do valor referencia com o valor atual para LUM < TARGET
   cmp bl, ds:erro    ; verifica se a deiferenca esta dentro da margem de erro
   jb inicio          ; caso esteja, volta para o inico
                      ; caso contrario, a cortina sobe pois aqui o valor atual eh menor que a referencia
sobe:
   cmp ds:meiogiro, 0  ; compara os giros do motor com o valor minimo de giros
   je inicio          ; se o valor ja esta no minimo entao volta ao inicio
   in al, CTRLVAR
   or al, 010b        ;seta o segundo bit(sobe)
   out CTRLVAR, al
   mov cx, 4          ; repetindo a intrucao 10 vezes tem-se aproximadamente uma volta
   clkwise:           ; sequencia de instrucoes para girar o motor no sentido horario
   mov al, 001b       ; initialize.
   out 7, al
   mov al, 011b       ; half step 1.
   out 7, al
   mov al, 010b       ; half step 2.
   out 7, al
   mov al, 110b       ; half step 3.
   out 7, al
   loop clkwise
   dec ds:meiogiro    ; apos o motor ter girado no sentido horario, decrementa-se o valor do giro
   mov al, ds:meiogiro  ; move a variavel giro para AL para poder escrever na porta
   out BLOCK, al      ; escreve o valor do giro na porta de bloqueio de luz(cada giro que o motor da,
bloqueia 10% da luz que entra)
   jmp inicio         ; apos a rotacao, volta-se ao inicio para que o processo se repita
desce:
   cmp ds:meiogiro, 20  ; compara os giros do motor com o valor maximo de giros
   je inicio          ; se o valor ja esta no maximo entao volta ao inicio
   in al, CTRLVAR
   and al, 11111101b   ;zera o segundo bit(desce)
   out CTRLVAR, al
   mov cx, 4
   counterclkwise:
   mov al, 010b       ; initialize.
   out 7, al
   mov al, 110b       ; half step 1.
   out 7, al
   mov al, 001b       ; half step 2.
   out 7, al
   mov al, 011b       ; half step 3.
   out 7, al
   loop counterclkwise
   inc ds:meiogiro    ; apos o motor ter girado no sentido anti-horario, incrementa-se o valor do giro
   mov al, ds:meiogiro  ; move a variavel giro para AL para poder escrever na porta
   out BLOCK, al      ; escreve o valor do giro na porta de bloqueio de luz(cada giro que o motor da,
bloqueia 10% da luz que entra)
   jmp inicio         ; apos a rotacao, volta-se ao inicio para que o processo se repita
systemreturn:
   mov ax, 4c00h      ; exit to operating system
   int 21h
ends
end start             ; set entry point and stop the assembler
```

Capítulo 5

Interação com Dispositivos Externos

Os seguintes dispositivos foram utilizados no projeto de controle de luminosidade:

* Stepper-Motor

* Controle de Luminosidade

O Stepper-Motor, como o próprio nome diz, funciona como um motor de passo, que pode funcionar sobre as configurações half-step ou full-step, no caso do nosso projeto foi utilizado a configuração half-step para maior precisão da cortina. (Ver figura 5.1)

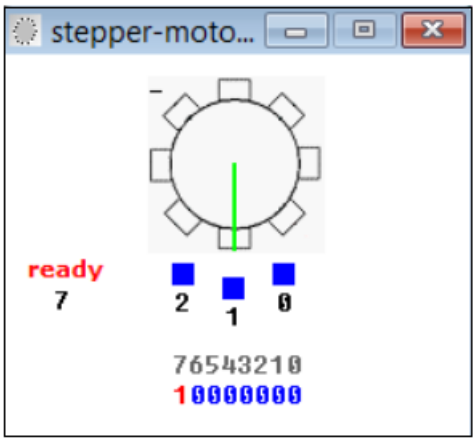

Figura 5.1: Dispositivo Stepper-Motor

A função do stepper-motor é fazer o controle da cortina, tanto para quando precisamos abrir a cortina, quanto quando precisamos fecha-la.No programa quando precisamos que a cortine realize algum movimento, é mandado um sinal de controle para o stepper-motor e baseado em quantas rotações ele deu, sabemos quanto a cortina se moveu. Ele funciona em cima de 3 bobinas e dependendo de quais estão ativadas ao mesmo tempo o posicionamento do motor muda, a maior diferença da configuração half-step para a full-step é que na configuração de meio passo 2 bobinas podem estar ligadas ao mesmo tempo.

O stepper-motor foi utilizado no projeto de maneira que quando o mesmo gira no sentido horário, a cortina sobe, e quando rotaciona no sentido anti-horário, a cortina desce. Além disto, para fechar ou abrir totalmente a cortina são necessárias 10 voltas do motor de passo.

O controle de luminosidade foi um dispositivo criado pela equipe em cima da linguagem Basic para realizar a simulação do ambiente externo. Nele, encontramos diversas informações importantes para a visualização da execução do programa, além de representações gráficas do sistema de controle.

O dispositivo criado para o projeto conta com uma simulação da luminosidade diária, fornecida em uma escala centesimal, onde 100% corresponde à total iluminação ao meio dia, enquanto 0% diz respeito à completa falta de iluminação solar (período noturno). No dispositivo pode-se ver, através do painel (13), o horário do dia, além disso é perceptível a mudança entre dia e noite pelas animações (5h começa o período diurno, enquanto 19h inicia-se o período noturno). Para uma representação mais fiel possível da luminosidade externa, foi pensada uma função matemática logarítmica fazendo-se uma interpolação do ponto menos luminoso do dia (5h -> 0%) com o mais luminoso (12h -> 100%). Assim, ao nascer do sol, a luminosidade cresce rapidamente no início, mantém um ritmo de crescimento desacelerado durante o dia até 12h, e decaimento desacelerado após, até que ao entardecer caí rapidamente.

Além da função de simulação externa, para que então possa ser desenvolvida toda a parte de controle do sistema no assembler, o dispositivo fornece informações atualizadas constantemente a respeito das 5 portas utilizadas no projeto e contém os 6 botões de interrupções do sistema. (Ver figura 5.2)

Figura 5.2: Interface