

LABORATÓRIO DE ARQUITETURA DE COMPUTADORES

Prof. Wilian França Costa

“Uso de macros”

Exercícios de laboratório: Displays de 7-Segmentos (multiplexagem)

Os segmentos num display de 7 segmentos podem ser selecionados de modo a obtermos quaisquer dos caracteres hexadecimais de 0 a F, um de cada vez, conforme se pode ver na animação:



É possível o visionamento de números com vários dígitos, utilizando displays adicionais. Embora seja mais confortável trabalhar com displays LCD (displays de cristal líquido), os displays de 7 segmentos continuam a constituir um standard na indústria. Isto devido à sua robustez em relação à temperatura, visibilidade e amplo ângulo de visão. Os segmentos são representados pelas letras minúsculas: a, b, c, d, e, f, g, dp, em que dp representa o ponto decimal.

Os 8 LEDs contidos no display podem estar dispostos nas configurações de ânodo comum ou de cátodo comum. Nos displays de cátodo comum, o cátodo comum deve ser ligado à massa e para que os leds acendam, é preciso aplicar uma tensão positiva aos respectivos ânodos (1 lógico). Os displays de ânodo comum apresentam o ânodo comum ligado a +5V e acendem quando se aplica um nível lógico zero aos cátodos respectivos. O tamanho do display é medido em milímetros, que corresponde à altura do display propriamente dito (não do encapsulamento mas sim do dígito!). No mercado, estão disponíveis displays com tamanho de 7, 10, 13.5, 20 ou 25mm. Podem também aparecer em diversas cores como vermelho, laranja e verde.

A maneira mais simples de alimentar um display é utilizando um ‘display driver’. Estes estão disponíveis para até 4 displays.

Alternativamente, os displays podem ser atuados por intermédio de um microcontrolador e, se necessitarmos de mais que um display, podemos utilizar o método de ‘multiplexagem’.

A principal diferença entre estes dois métodos, consiste no número de linhas utilizadas para fazer as ligações aos displays. Um ‘driver’ especial, pode necessitar apenas de uma linha de “clock” e será o chip que o contém que irá aceder aos segmentos e incrementar o display.

Se o microcontrolador for alimentar um único display, então apenas serão necessárias 7 linhas ou mais uma se utilizarmos o ponto decimal. Se utilizarmos vários displays, então precisamos de uma linha adicional para cada display.

Para construirmos displays de 4, 5 ou 6 dígitos, devemos ligar em paralelo todos os displays de 7 segmentos.

A linha de cátodo comum (no caso de displays de cátodo comum) é tratada separadamente e é posta a nível baixo durante um curto espaço de tempo para acender o display.

LED_Disp2 macro número

número é o número de 0 a 99 que vai ser mostrado nos dígitos MSD e LSD.

Exemplo: LED_Dis2 0x34

Neste caso, vai aparecer o número 34 nos displays. Neste caso, o número 34, vai aparecer nos dois displays. A implementação da macro é apresentada a seguir.



7-seg.inc

```
;***** Macros *****

LED_Init macro
    call InitPorts
    call InitTimers
endm

LED_Dis2 macro num
    movlw num
    movwf LO
    call UpdateDisplay
endm

;***** Subprogramas *****

InitPorts
    BANK1
    clrf LEDtrisA           ; Pinos RA0-4 de saída
    clrf LEDtrisB           ; Porto B de saída
    BANK0
    clrf LEDportA           ; Todos os bits a
    clrf LEDportB           ; nível lógico 0
    bsf LEDportA,3          ; Activar display MSD

    RETURN

InitTimers
    BANK1
    movlw B'10000100'       ; Prescaler atribuído a TMR0
    movwf OPTION_REG        ; e igual a 32
    BANK0
    movlw B'00100000'       ; Habilitar interrupção por TMR0
    movwf INTCON             ;
    movlw .96                ;
    movwf TMR0               ; Iniciar o temporizador

    RETFIE

;*****ISR - Rotina de Serviço de Interrupção*****

ISR
    bcf INTCON,GIE           ; Inibir todas as interrupções
    btfsc INTCON,GIE         ; Verificar se estão inibidas
    goto ISR

    movlw .96                ; Iniciar TMR0
    movwf TMR0
    bcf INTCON,TOIF          ; Limpar a flag TOIF
    call UpdateDisplay        ; atualizar o display

    RETFIE
```

```

UpdateDisplay
    movf LEDportA,W           ; Estado dos displays -> registro w
    clrf LEDportA             ; Apagar todos os displays de 7 segmentos
    andlw 0x0f                ; Isolar os quatro bits menos significativos
    movwf TempC               ; Guardar o estado dos displays em TempC
    bsf TempC,4               ; Estado inicial do display menos significativo
    rrf TempC,F               ; Estabelecer estado do display seguinte
    btfss STATUS,C            ; c=1 ?
    bcf TempC,3               ; Se não, apagar display menos significativo
    btfsc TempC,0             ; Se sim, verificar estado do display + significativo
    goto UpdateMsd            ; Se display activado, mostrar o byte + significativo

UpdateLsd
    call ChkMsdZero           ; dígito + significativo = 0 ?
    btfss STATUS,Z            ; sim, ignorar inst. seguinte
    movf LO,W                 ; dígito - significativo -> W
    andlw 0x0f                ; mascarar o que não interessa
    goto DisplayOut           ; Visualizar no display

UpdateMsd
    swapf LO,W                ; dígito + significativo -> W
    andlw 0x0f                ; mascarar o que não interessa
    btfsc STATUS,Z            ; dígito + significativo diferente de 0?
    movlw 0x0a                ; Se sim, ignorar

DisplayOut
    call LedTable              ; Obter código de acendimento
    movwf LEDportB            ; Código de acendimento p/ Porto B
    movf TempC,W              ; Acender o display
    movwf LEDportA

RETURN

LedTable
    addwf PCL,F               ; código de acendimento de '0'
    retlw B'00111111'         ; código de acendimento de '1'
    retlw B'00000110'         ; código de acendimento de '2'
    retlw B'01011011'         ; código de acendimento de '3'
    retlw B'01100110'         ; código de acendimento de '4'
    retlw B'01101101'         ; código de acendimento de '5'
    retlw B'01111101'         ; código de acendimento de '6'
    retlw B'00000111'         ; código de acendimento de '7'
    retlw B'01111111'         ; código de acendimento de '8'
    retlw B'01101111'         ; código de acendimento de '9'
    retlw B'00000000'         ; display apagado.....

ChkMsdZero
    movf LO,W                 ; Verificar zero à esquerda
    btfss STATUS,Z            ; dígito + significativo -> W
                                ; = 0 ?
    RETURN                    ; Se não for = 0
    retlw .10                 ; Se for = 0, regressar com W= .10

```




UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática



O programa que se segue, exemplifica a utilização de macros num programa. Este programa faz aparecer o número '21' nos dois displays de 7 segmentos.

**LED.asm**

```
;***** Escolher e configurar o microcontrolador *****  
  
PROCESSOR 16f84  
#include "p16f84.inc"  
  
__CONFIG _CP_OFF & _WDT_OFF & _PWRTE_ON & _XT_OSC  
  
;***** Declarar variáveis *****  
  
Cblock 0x0C ; Início da Ram  
TempC ; Pertence à macro "LED_Dis2"  
LO  
endc  
  
;***** Declarar o hardware *****  
  
LEDtrisA equ TRISA  
LEDportA equ PORTA  
  
LEDtrisB equ TRISB  
LEDportB equ PORTB  
  
;***** Estrutura da memória de programa *****  
  
ORG 0x00 ; Vector de reset  
goto Main  
  
ORG 0x04 ; Vector de interrupção  
goto ISR ; A rotina de interrupção encontra-se  
; no ficheiro 7-seg.inc  
  
#include "bank.inc" ; ficheiros auxiliares  
#include "7-seg.inc"  
  
Main ; Início do programa  
LED_Init  
  
LED_Dis2 0x21 ; Visualizar nos dois displays de 7 segmentos  
; o número "21"  
loop goto loop ; Permanecer aqui  
  
End ; Fim do programa
```



Exercício para entrega no moodle:

Modificar o exemplo para que funcione no 18F4550A, placa MCLab2 no PICSimLab.
Enviar arquivo nomeado com 7seg.asm