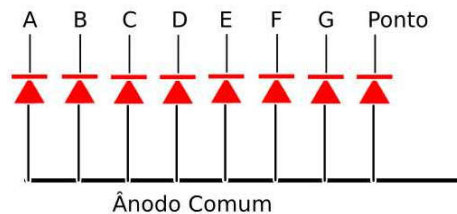
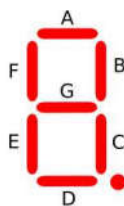


EEL7030 Microprocessadores – Laboratório 3

Prof. Raimes Moraes

Subrotinas

A tarefa executada pelo programa a seguir é mostrar o caractere 1 no *display* 0 do EDSIM51 (<http://www.edsim51.com/>), sendo que o mesmo ilustra o emprego de subrotina. Para permitir a escrita no *display* 0 do EDSIM51, deve-se fazer CS='1', END0=END1='0'. A subrotina CONVERTE recebe, como parâmetro, o valor contido no acumulador (1 no exemplo abaixo) que se deseja mostrar no display (de 0 a FH); a subrotina retorna, no acumulador, o valor a ser escrito no *display* de 7 segmentos (através da porta P1) para mostrar o valor passado como parâmetro para a subrotina (1 no exemplo abaixo). Compile o programa no Keil e o carregue no EDSIM51 (arquivo .hex) para observar a sua execução. Obs: Modifique o campo "Update Frequency" do EDSIM para que a velocidade de execução não seja muito lenta.



```
;Programa      MOSTRA1.asm
CS      EQU    P0.7
END0     EQU    P3.3;
END1     EQU    P3.4;
```

```
ORG      0h
CLR      END0
CLR      END1
SETB     CS
MOV      A,#1
CALL     CONVERTE
MOV      P1,A
JMP      $
```

```
CONVERTE:  INC A
           MOVC A,@A+PC
           RET
```

```
TABELA: DB 40H, 79H, 24H, 30H, 19H, 12H, 02H, 78H, 00H, 10H, 08H, 03H, 46H,
          21H, 06H, 0EH
          END
```

Exercícios:

- 1) Modifique o programa anterior para mostrar valor lido das chaves (0 a FH) conectadas à porta P2 (*Switch Bank*) no display 0. Valor lido de chave aberta equivale a '1'; de chave fechada, '0'. Portanto, complemente (instrução CPL A) o valor lido de P2 antes de apresentá-lo à subrotina de conversão.
- 2) Modifique o programa anterior para mostrar no *display* 1, o nro de chaves fechadas. OBS: Para tal, use a instrução RLC A e teste o *flag* de *carry* para contar o número de chaves fechadas. Fazer isto ciclicamente.
- 3) Modifique o programa anterior para realizar uma contagem decrescente (9 a 0) no *display* 0. Para tal, seu código deve possuir rotina de atraso entre apresentações da contagem, tal que seja possível visualizar a mesma. Veja sugestão de como implementar tal subrotina:

```
                ATRASO EQU 0FEH
                ...
                MOV  A,#ATRASO
                CALL  DELAY
                ...
DELAY:          DJNZ  Acc,DELAY    ; subrotina DELAY
                RET
```

Obs: Modifique o campo "Update Frequency" do EDSIM caso o atraso seja baixo ou alto.

- 4) Modifique o programa anterior para que a subrotina de atraso passe a não alterar qualquer registrador.
- 5) Há também leds conectados à Porta P1. Faça um programa com duas subrotinas. Uma para inserção de atraso. A outra rotina deve rotacionar um led aceso (obs: cor do led vai para branco) da direita para a esquerda inserindo atraso entre rotações. Fazer como

procedimento cíclico. OBS: Inibir CS do *display*. Utilizar instrução RL A.

- 6) Faça um programa com duas subrotinas. Uma para inserção de atraso. A outra rotina deve ir acendendo todas os 8 leds, um a um (da direita para a esquerda) com inserção de atraso; quando todos os leds estiverem acesos, ir apagando um a um (da esquerda para a direita). Fazer como procedimento cíclico. OBS: Utilizar as instruções RRC e RLC, além de setar, limpar ou testar o *flag* de *carry*. Tenha em mente que algumas instruções podem afetar o *flag* de *carry*.

Obs: Modifique o campo "Update Frequency" do EDSIM caso o atraso seja baixo ou alto.

- 7) Faça um programa que teste a chave conectada a P2.7. Se a mesma estiver fechada (P2.7='0'), rotacione um led aceso para a esquerda (P2.6='1') ou para a direita (P2.6='0') pelo número de vezes especificado pelo complemento do valor das 4 chaves menos significativas (P2.3 a P2.0). Se a chave P2.7 estiver aberta, fique aguardando ser alterada.

- 8) Faça um programa que identifique a tecla pressionada no *keypad* conectado à porta P0 e a apresente no *display* 0. A rotina para identificar a tecla pressionada é apresentada a seguir; este código coloca uma das linhas do teclado (P0.3 a P0.0) em nível lógico baixo, uma após a outra. Quando uma das linhas é colocada em nível lógico baixo, as colunas são testadas (P0.6 a P0.4) para verificar se, em alguma delas, '0' é lido (observe que a tecla pressionada fecha contato da linha com coluna). Se '0' for lido, a tecla pressionada corresponde àquela combinação de linha e coluna. Modificar a tabela da função converte, conforme abaixo. Obs: Modifique o campo "Update Frequency" do EDSIM caso o atraso seja baixo ou alto. Usar teclado no modo "PULSE"

```
; Subrotina que retorna em R0 o valor do dígito pressionado no teclado do EDSIM51
; (Obs: retorna A para * e C para #)
```

```
; teclado
;
;                                linhas
;      +----+----+----+
;      | 1 | 2 | 3 |   P0.3
;      +----+----+----+
;      | 4 | 5 | 6 |   P0.2
;      +----+----+----+
;      | 7 | 8 | 9 |   P0.1
;      +----+----+----+
;      | A | 0 | C |   P0.0
;      +----+----+----+
;      colunas  P0.6 P0.5 P0.4
```

KEYPAD:

```
ORL  P0,#7Fh      ; escreve '1' em 7 pinos da porta P0
CLR  F0           ; limpa flag que identifica tecla pressionada
MOV  R0, #0       ; limpa R0 – retorna o número da tecla foi pressionada

; varre primeira linha
CLR  P0.3         ; coloca '0' na linha P0.3
CALL colScan      ; chama rotina para varredura de coluna
JB   F0, finish   ; se flag F0 = '1', tecla identificada => retorna

; varre segunda linha
SETB P0.3         ; seta linha P0.3
CLR  P0.2         ; coloca '0' na linha P0.2
CALL colScan      ; chama rotina para varredura de coluna
JB   F0, finish   ; se flag F0 = '1', tecla identificada => retorna

; varre terceira linha
SETB P0.2         ; seta linha P0.2
CLR  P0.1         ; coloca '0' na linha P0.1
CALL colScan      ; chama rotina para varredura de coluna
JB   F0, finish   ; se flag F0 = '1', tecla identificada => retorna

; varre quarta linha
SETB P0.1         ; seta linha P0.1
CLR  P0.0         ; coloca '0' na linha P0.0
CALL colScan      ; chama rotina para varredura de coluna
JB   F0, finish   ; se flag F0 = '1', tecla identificada => retorna

JMP  KEYPAD       ; se flag F0 = '0', tecla não identificada => repete varredura
```

finish:

```
RET
```

```
; Subrotina que varre as colunas para identificar a qual pertence a tecla pressionada
; o registrador R0 é incrementado a cada insucesso de forma a conter o nro. da tecla
; pressionada
```

colScan:

```
JNB P0.6, gotKey  ; tecla pressionada pertence a esta coluna – retornar
INC  R0
JNB P0.5, gotKey  ; tecla pressionada pertence a esta coluna – retornar
INC  R0
```

```

        JNB P0.4, gotKey      ; tecla pressionada pertence a esta coluna – retornar
        INC R0
        RET                  ; tecla pressionada não pertence a esta linha – retornar
gotKey:
        SETB F0              ; faz flag F0 = '1' => tecla identificada
        RET

; Subrotina converte com tabela modificada para o exercício solicitado
CONVERTE: INC A
          MOVC A,@A+PC
          RET
TABELA:  DB 79H, 24H, 30H, 19H, 12H, 02H, 78H, 00H, 10H,08H,40H, 46H

```