Universidade de São Paulo

Danilo Ortiz Rodrigues - 11232521
Eduardo Souza Caproni - 11279537
Enzo Rocha Paranaguá - 10746773
Fernando Barboza de Deus Fonseca - 11232201

SEL0336 - Aplicação de Microprocessadores

Projeto

São Carlos, Brasil
Abril / 2024

1 Objetivos

Desenvolvimento de um projeto em linguagem Assembly para 8051 que explore os seguintes recursos no simulador EdSim51: registradores GPR e SFR, contagem de tempo, detecção de eventos, pilha, sub-rotinas, portas de entradas e saídas, e interfaces externas (botões, LEDs e displays de 7 segmentos).

2 Diagrama do Hardware

As conexões das portas do 8051 com os hardwares disponíveis foram feitas da seguinte forma:

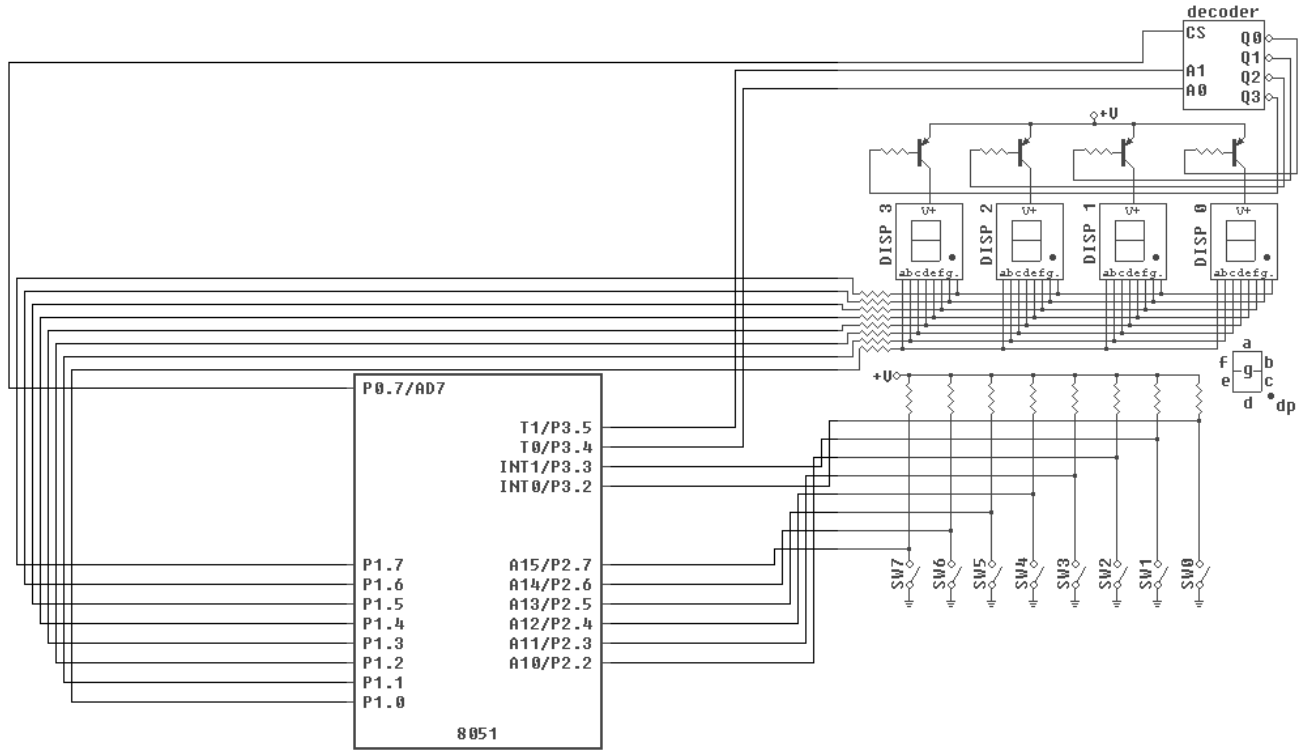

Figura 1: Diagrama das conexões.

Com o objetivo de melhor fixar os conteúdos apresentados em aula, escolhemos utilizar interrupções neste projeto. É importante destacar que os **switches 0 e 1** foram ligados a portas diferentes do usual, sendo colocados nas **portas P3.2 e P3.3**, respectivamente. Isso foi feito para tornar possível o uso de interrupções externas.

3 Explicação do código

O código produzido é dividido em alguns blocos principais. Os de maior importância a serem explicados são os blocos *setup, main, delay e print*. Além disso, o código faz uso de interrupções externas a partir de switches e realiza a contagem de tempo a partir de decremento de registradores.

O bloco *setup* é resposável por ativar as interrupções externas 0 e 1, iniciar os registradores que são utilizados no código para seus valores default, por exemplo, R0 = 00h (delay igual a zero) e A = FFh (display desligado).

O bloco *main* é onde fica o loop infinito e que chama dentro dele duas subrotinas importantes: o delay e o print.

O bloco *delay* começa por analisar o valor de R0 para saber qual será o tempo de espera:

- Se R0 = 00h, não há delay e a subrotina é encerrada
- Se R0 = 01h, o delay é de 25ms
- Se R0 = 02h, o delay é de 1s

Uma vez determinado o tempo de delay, os valores dos registradores R1, R2 e R3 são alterados de forma que, quando decrementados até zero, ocasionaram o delay desejado.

- 25ms: R1 = 255, R2 = 255 e R3 = 2. Ou seja, 130050 operações, que equivalem a 260100μs (2μs por operação).
- 1s: R1 = 255, R2 = 255 e R3 = 8. Ou seja, 520200 operações, que equivalem a 1040400μs (2μs por operação).

No final desse bloco, o valor do acumulador é incrementado.

O bloco *print* analisa o valor no acumulador e printa o valor correspondente em P1. Por exemplo, se o valor 2 se encontra no acumulador, o valor de 0A4h é transferido para a porta P1 afim de visualizar o valor 2. A Tabela 1 mostra a relção entre os dígitos a serem visualizados e os valores que são necessários na porta P1. Caso o valor do acumulador seja FFh, o display é desligado.

Valor Display	0	1	2	3	4	5	6	7	8	9
Registrador P1	0C0h	0F9h	0A4h	0B0h	99h	92h	82h	0F8h	80h	98h

Tabela 1: Tabela de valores do digito visualizado no display com o valor correspondente no registrador P1

Por fim, as interrupções Externas 0 e 1, ligadas as portas P3.2 e P3.3, respectivamente, são acionadas em borda de descida para regularem o valor do delay. Se o botão switch 0 é pressionado, R0 recebe o valor 01h (25ms de delay) e se o botão switch 1 é pressionado, acontece a interrupção externa 1 e R0 recebe o valor 02h (1s de delay).

Na imagem abaixo é possível visualizar o diagrama do código:

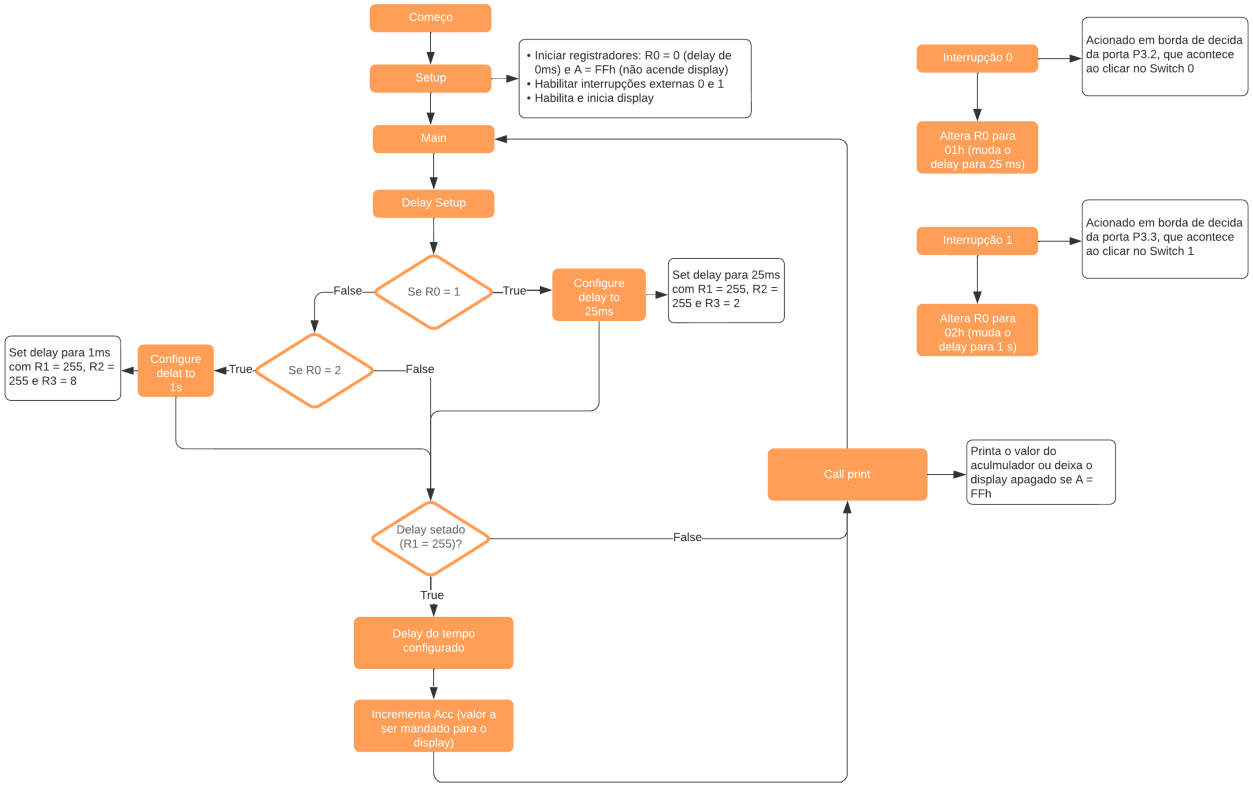

Figura 2: Diagrama Código

4 Código

```
org 0000h    ; Coloca o programa na posicao 0000h (reset)
JMP setup    ; Salta para o setup do programa

org 3h       ; Coloca o programa na posicao 0003h (IE0)
MOV R0, #1h ; Altera R0 para 1 (cronometro com periodo de 25 ms)
CLR IE0      ; Limpa a flag da IE0
RETI         ; Retorna da interrupcao

org 13h      ; Coloca o programa na posicao 0003h (IE1)
MOV R0, #2h ; Altera R0 para 2 (cronometro com periodo de 1 s)
CLR IE1      ; Limpa a flag da IE1
RETI         ; Retorna da interrupcao

org 33h          ; Coloca o programa na posicao 0033h
setup:           ; Setup do programa
MOV R0, #0h      ; Reseta R0 (cronometro parado)
MOV A, #0FFh     ; Altera o digito no display (0xFFh = apagado)
SETB EA          ; Habilita as interrupcoes
SETB EX0         ; Habilita a interrupcao externa 0
SETB EX1         ; Habilita a interrupcao externa 1
SETB IT0         ; Interrupcao externa 0 no modo falling-edge
SETB IT1         ; Interrupcao externa 1 no modo falling-edge
SETB P0.7        ; Habilita o display
CLR P3.4         ; Seleciona o display da direita
CLR P3.5         ; Seleciona o display da direita

main:        ; Loop principal do programa
CALL delay   ; Chama a subrotina de delay
CALL print   ; Chama a subrotina que exibe o digito atual
JMP main     ; Salta novamente para o loop principal

delay:                      ; Subrotina de delay
CALL setup_delay            ; Chama a subrotina de setup do delay
CJNE R1, #255, delay_end    ; Caso o timer nao tenha sido setado,
   nao executa o delay
run_delay:                  ; Loop principal do delay
DJNZ R1, $                  ; Decrementa R1 ate zera-lo
MOV R1, #255                ; Reseta R1
DJNZ R2, run_delay          ; Decrementa R2 ate zera-lo
MOV R2, #255                ; Reseta R2
DJNZ R3, run_delay          ; Decrementa R3 ate zera-lo
ADD A, #1h                  ; Uma vez o timer finalizado,
   incrementa o digito a ser mostrado
CJNE A, #0Ah, delay_end     ; Caso o proximo digito nao seja 10,
   finaliza o delay
MOV A, #0h              ; Caso o proximo digito seja 10, altera
   para 0
delay_end:
RET                     ; Finaliza a subrotina de delay

setup_delay:            ; Subrotina de setup do delay
MOV R1, #0              ; Reseta R1
```

```
49 MOV R2, #0                ; Reseta R2
50 MOV R3, #0                ; Reseta R3
51 CJNE R0, #1h, case_2      ; Caso R0 nao seja 1 (25 ms), verifica o
      proximo modo (1 s)
52 MOV R1, #255              ; Altera R1 para 255
53 MOV R2, #255              ; Altera R2 para 255
54 MOV R3, #2                ; Altera R3 para 2
55 case_2:                   ; Modo 2
56 CJNE R0, #2h, setup_delay_end   ; Caso R0 nao seja 2 (1 s),
      finaliza o setup do delay
57 MOV R1, #255              ; Altera R1 para 255
58 MOV R2, #255              ; Altera R2 para 255
59 MOV R3, #8                ; Altera R3 para 8
60 setup_delay_end:
61 RET                       ; Finaliza a subrotina de setup do delay
62
63 print:                 ; Subrotina de print
64 CJNE A, #0FFh, 3h     ; Caso A nao seja 0xFFh, verifica o proximo
      digito
65 MOV P1, #0FFh         ; Caso A seja 0xFFh, apaga o display
66 CJNE A, #0, 3h        ; Caso A nao seja 0, verifica o proximo digito
67 MOV P1, #0C0h         ; Caso A seja 0, exibe no display
68 CJNE A, #1, 3h        ; Caso A nao seja 1, verifica o proximo digito
69 MOV P1, #0F9h         ; Caso A seja 1, exibe no display
70 CJNE A, #2, 3h        ; Caso A nao seja 2, verifica o proximo digito
71 MOV P1, #0A4h         ; Caso A seja 2, exibe no display
72 CJNE A, #3, 3h        ; Caso A nao seja 3, verifica o proximo digito
73 MOV P1, #0B0h         ; Caso A seja 3, exibe no display
74 CJNE A, #4, 3h        ; Caso A nao seja 4, verifica o proximo digito
75 MOV P1, #99h          ; Caso A seja 4, exibe no display
76 CJNE A, #5, 3h        ; Caso A nao seja 5, verifica o proximo digito
77 MOV P1, #92h          ; Caso A seja 5, exibe no display
78 CJNE A, #6, 3h        ; Caso A nao seja 6, verifica o proximo digito
79 MOV P1, #82h          ; Caso A seja 6, exibe no display
80 CJNE A, #7, 3h        ; Caso A nao seja 7, verifica o proximo digito
81 MOV P1, #0F8h         ; Caso A seja 7, exibe no display
82 CJNE A, #8, 3h        ; Caso A nao seja 8, verifica o proximo digito
83 MOV P1, #80h          ; Caso A seja 8, exibe no display
84 CJNE A, #9, 3h        ; Caso A nao seja 9, verifica o proximo digito
85 MOV P1, #98h          ; Caso A seja 9, exibe no display
86 RET                   ; Retorna da subrotina de print
87
88 end                   ; Fim logico do programa
```