Universidade de São Paulo
Escola de Engenharia de São Carlos

Projeto de um cronômetro digital usando Assembly e 8051

SEL0614 - Aplicação de Microprocessadores
Prof. Pedro Oliveira

Bárbara Fernandes Madera, 11915032
Matheus dos Santos Inês, 12546784
Victor Gabriel Miranda Rosa, 11232114

São Carlos
2023

Sumário

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. Programa Comentado

```
ORG 0000h

MOV DPTR, #SEGMENTOS ; Carrega o endereço da tabela de segmentos em DPTR
JMP inicio ; Salta para o inicio

ORG 0033h

inicio:
 JNB P2.0, start ; Salta para start se o SW0 não estiver pressionado, caso contrário, continua aguardando
 JMP inicio ; Continua aguardando até que o SW0 seja pressionado


; Inicializa a variável R2 com 0
start:
 MOV R2, #00

loop:
 MOV A, R2 ; Move o valor de R2 para o acumulador
 MOVC A, @A+DPTR ; Acessa o número na tabela que será exibido no display
 MOV P1, A ; Exibe o número no display
 INC R2 ; Incrementa R2 para obter o próximo número
 JMP checagem_delay ; Verifica qual atraso deve ser aplicado

pos_delay:
 CJNE R2, #10, LOOP ; Se o valor de R2 ainda não chegou a 10 -> continue a exibição
 JMP start ; Caso contrário, reinicie a contagem a partir de zero

checagem_delay:
 JNB P2.0, delay_025 ; Se o SW0 estiver pressionado, use o atraso de 0,25 segundos
 JNB P2.1, delay_1 ; Se o SW1 estiver pressionado, use o atraso de 1 segundo
 JMP checagem_delay ; Aguarde até que uma das portas esteja pressionada

delay_025:
 MOV R1, #500 ; Configura o valor de R1 para um atraso de 0,25 segundos
 JMP loop_delay ; Inicia o atraso
```

```
delay_1:
 MOV R1, #2000 ; Configura o valor de R1 para um atraso de 1 segundo
 JMP loop_delay ; Inicia o atraso

loop_delay:
 MOV R0, #250 ; Configura R0 para um valor de atraso
 DJNZ R0, $ ; Gasta 0,0005 segundos a cada iteração
 DJNZ R1, loop_delay ; Repete o atraso até que o tempo seja atingido
 JMP pos_delay ; Retorna à contagem após o atraso

; Tabela de segmentos para exibição de dígitos no display de 7 segmentos
SEGMENTOS:
DB 0c0h
DB 0f9h
DB 0a4h
DB 0b0h
DB 99h
DB 92h
DB 82h
DB 0f8h
DB 80h
DB 90h
```

3. Explicação sobre o Programa

O programa é projetado para exibir números em um display com base na seleção de botões e com atrasos diferentes, dependendo das opções selecionadas em um microcontrolador ou microprocessador. O código é escrito em linguagem Assembly para o programa EDSim51, e a lógica exata de como os números são exibidos é feita com na tabela de 7 segmentos e do hardware subjacente.

Neste sentido, inicia-se carregando o endereço de uma tabela de segmentos em um registrador armazenador de memória conhecido como Ponteiro de Dados (DPTR) , o qual acessa o endereço de memória do programa. Ele então realiza o salto para o rótulo “início” e começa a execução. Com isso, o código aguarda que o botão SWO(assumido como P2.0) seja pressionado antes de prosseguir para a inicialização. A variável R2, por sua vez, é inicializada com valor nulo no rótulo “start”.

O programa, então, entra em um loop que exibe números de 0 a 9 na tela, acessando a tabela de segmentos e atualizando o display. Após cada exibição, R2 é incrementado e o programa realiza a verificação de qual atraso deve ser aplicado tendo como base a seleção dos botões SW0 e SW1. Assim, o atraso é executado em “loop_delay”, após o qual o programa retorna à contagem no rótulo “pos_delay”.

É importante ressaltar que o código contém instruções para aplicar atrasos específicos, em que R0 e R1 são configurados para controlar os botões de atrasos

respectivos de 0,25 segundos e 1 segundo. Ademais, a forma de como os atrasos são feitos é por meio de ciclos de loop que aguardam até que o tempo especificado seja atingido, para o seguinte programa isso ocorre quando R2 atinge o valor de uma dezena-momento esse em que a contagem é reiniciada.

4. Diagramas Solicitados

- Diagrama esquemático do microcontrolador 8051 com a ligação das interfaces de entrada e saída usadas no projeto (segundo a estrutura disponível no EdSim51)

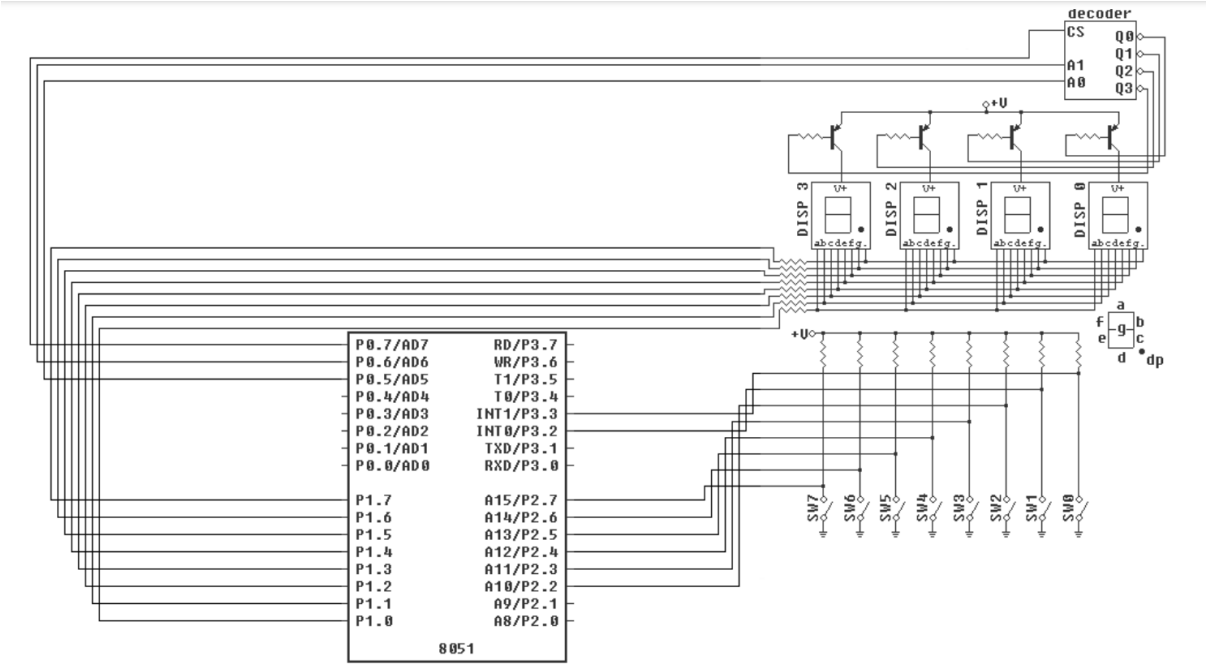

Figura 1: Esquemático do circuito (note que algumas portas estão diferentes em relação à configuração padrão do simulador)

- Diagrama ou tabela de como é feita a varredura no display de 7 segmentos disponível no EdSim51 para acender números de 0 a 9 usando os 8 bits do registrador da Porta P1.

Número	Portas	Binário	Código Hexadecimal
0	a, b,c, d, f	11000000	C0
1	b, c	11111001	F9
2	a, b, d, e, g	10100100	A4
3	a, b, c, d, g	10110000	B0
4	b, c, f, g	10011001	99
5	a, c, d, f, g	10010010	92
6	a, c, d, e, f, g	10000010	82
7	a, b, c	11111000	F8
8	a, b, c, d, e, f, g	10000000	80
9	a, b, c, d, f, g	10010000	90