UNIVERSIDADE DE SÃO PAULO

Escola de Engenharia de São Carlos

Aplicação de Microprocessadores

Pedro Oliveira

Projeto 1 - Cronômetro digital usando Assembly e 8051

Diogo Barboza de Souza - nºUSP 12745657

São Carlos

Outubro de 2023

Introdução

Nesse projeto foi feito um cronômetro programado em assembly para o MCS-51, esse cronômetro faz uma contagem de 0 a 9 usando o display do dispositivo, contendo funções para utilizar botões para mudar o intervalo de tempo entre números e da contagem como um todo.

Desenvolvimento

Nos requisitos deste projeto foi dado que o programa deve contar de 0 a 9 e ao apertar um botão o intervalo deve mudar, com isso, não foi compreendido totalmente se era o intervalo do final do programa ou ainda o intervalo entre cada número, sendo assim, foram feitos os dois modos.

Ao iniciar o programa o mesmo ficará esperando os apertos de botões, foram usados os botões P2.0, P2.1, P2.2 e P2.3, como explicado anteriormente há dois modos desse programa.

O primeiro modo usa os botões P2.0 e P2.1, esses botões foram implementados com o intuito de mudar o tempo entre cada número da contagem, isso é, o tempo decorrido entre 0 e 1 por exemplo, sendo que ao apertar P2.0 o tempo decorrido será de 0,25s e ao apertar P2.1 o tempo será de 1s.

Agora para o segundo modo, foi feito a seguinte interpretação, cada botão muda o tempo entre cada contagem, da forma que quando a contagem de 0 a 9 terminar o tempo para se iniciar uma nova contagem dependerá do botão apertado, nesse caso foram usados os botões P2.2 e P2.3, usando o mesmo conceito do modo anterior, o P2.2 terá um tempo de 0,25s para se iniciar outra contagem e o botão P2.3 terá um tempo de 1s, porém ambos terão na passagem de cada número 1s, isto é, entre 3 e 4 haverá um tempo de 1s.

Visto isso, o funcionamento dos botões do MSC-51 podem levar a confusão já que eles continuam pressionados após sua ativação, com isso em mente, para o funcionamento correto do programa ao acionar um botão e em seguida ao acionar outro botão deve-se desativar o botão anterior, deixando apenas um botão desejado ativado. Além disso, deve ser colocado a frequência de atualização (“Update Freq.”) em 10000.

Display

Para o funcionamento do display se foi usado o DPTR (Data Pointer) que é um ponteiro de 16 bits para área de dados em memória RAM interna e externa, ele foi usado para armazenar os valores de cada número na contagem correspondente com o valor binário do número para ser colocado no display. Visto isso, foi usado o ACC para percorrer os números e colocá-los no display p1.

Cálculo de cada tempo no delay

Para os determinados delays foram usados a seguinte forma de programação, por exemplo:

```
delay:
    mov r1, #500
    inicio:
        mov r2, #250
        DJNZ r2, $
        DJNZ r1, inicio
        RET
```

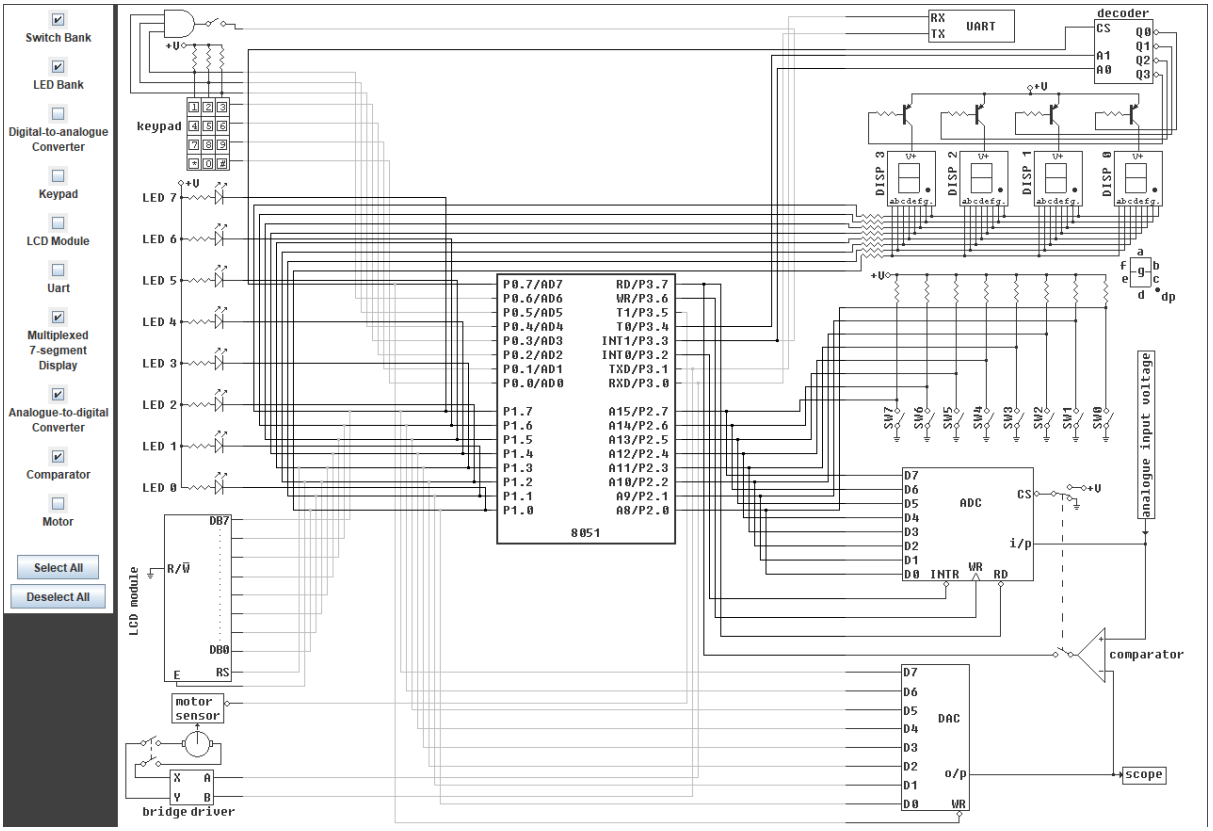
Sendo assim, o cálculo no exemplo será: 500 * 250 * 2us = 0,25s , esse que foi o valor pedido também ao apertar um determinado botão. Outro valor pedido foi o de 1s, para isso foi feito o seguinte cálculo: 1000 * 500 * 2us = 1s.
Observação: sendo 2us o valor usado pela instrução de 2 ciclos de máquina (2 * 10^-6).

Apertando um botão no decorrer da contagem

Enquanto o programa está fazendo sua contagem após apertar o botão P2.0 ou P2.1, se um outro botão for apertado e o anterior desativado, deve-se ocorrer a mudança no período de tempo entre cada número, vale lembrar que isso apenas vale ao primeiro modo, como explicado anteriormente apenas para os botões P2.0 e P2.1.

Para fazer esse funcionamento foi usado a função “botoes” que é chamada logo após um delay, assim verificando em passagem de número da contagem se outro botão foi apertado e assim mudando o intervalo de tempo, se caso for o P2.0 será de 0,25s e se for o P2.1 será de 1s.

Diagrama lógico do EdSim51

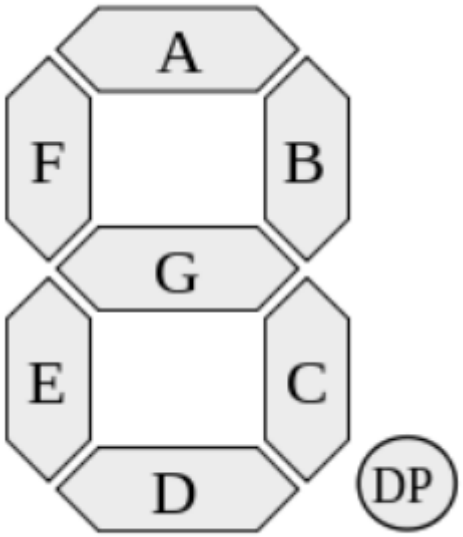


Varredura dos display conectados ao 8051 no simulador EdSim51

Observação: bit = 1 o led está desligado e bit = 0 o led está ligado. Além disso, os valores usados para a implementação foram em binário.

Display	Quais segmentos acender (bit = 0)	Valor (Bits do registrador - PORTA P1)	Valor em Hexadecimal
0	a,b, c, d, f	#11000000b	#0C0h
1	b,c	#11111001b	#0F9h
2	a,b, d, e, g	#10100100b	#0A4h
3	a,b, c, d, g	#10110000b	#0B0h
4	b,c, f, g	#10011001b	#99h
5	a,c, d, f, g	#10010010b	#92h
6	a,c, d, e	#10000010b	#82h
7	a, b, c	#11111000b	#0F8h
8	a,b,c,d,e,f,g	#10000000b	#80h
9	a,b,c, d, f	#10010000b	#90h

Display de 7 segmentos:



Código assembly

```
org 0000h ;origem no endereço 0000h

main:
;inicializando as variaveis
        mov r0, #0
        mov r1, #0
        mov r2, #0;
        mov r3, #0
        mov r4, #0;
        mov r5, #3
        acall botoes_inicio ;chamada da função que esepra um botão ser apertado
        mov dptr, #segmentos ; move para dptr todos os numeros que serão usados na
        contagem
        acall contagem ;chama função da contagem

botoes_inicio:
;Verificação se foi apertado um dos botões e pulando para uma label que coloca o valor que
será usado no delay
        jnb p2.0, zero
        jnb p2.1, um
        jnb p2.2, um
        jnb p2.3, um
        jmp botoes_inicio
        ret
```

```
botoes: ;função para mudar de botão em meio a contagem
        cjne r5, #1, Bum ;jmp apenas se o botão 1 não for apertado, r5 = 1 significa botão 1
        apertado
        cjne r5, #0, Bzero ;;jmp apenas se o botão 1 não for apertado, r5 = 0 significa botão
        0 apertado

        Bzero:
                jnb p2.0, zero ;verifica se p2.0 foi apertado
                jmp um ;jmp para recarregar novamente os valores caso seja apenas p2.1
                apertado
                ret
        bum:
                jnb p2.1, um ;verifica se p2.0 foi apertado
                jmp zero ;jmp para recarregar novamente os valores caso seja apenas p2.1
                apertado
                ret

zero: ;coloca os valores adequados que serão usados no delay
;0,25s
        mov r1, #100 ;se foi colocado 100 para a melhor visualização no display, mas para
        ser 0,25s como pedido deve se colocar 500
        mov r2, #250
        mov r5, #0
        ret

um: ;coloca os valores adequados que serão usados no delay
;1s
        mov r1, #1000
        mov r2, #500
        mov r5, #1
        ret

contagem: ;função da contagem em si
        mov r4, #10 ;contador para auxiliar na contagem
        continue:

        mov A, r3 ;r3 = 0, posicionando o inicio da contagem
        movc A, @A+dptr ;usando A como um ponteiro para percorrer dptr
        mov p1, A ;atriindo o valor do numero para o display p1
        acall delay ;chamada de delay entre numeros
        inc r3 ;aumentando o valor de r3 para ser usado na proxima contagem
        DJNZ r4, continue ;diminuindo o valor de r4 e retomando a contagem
        mov r4, #1 ;colocando 1 em r4 para ser usado na função "acabou"
        acall acabou ;chamada da função, sinalizando que a contagem acabou
        jmp main

acabou: ;função usada quando a contagem acabou
        mov p1, #10111111b ;acendendo apenas o led do meio do numero do display
```

```
        acall delay_final ;chamando delay do fim da contagem
        djnz r4, acabou ;diminui r4 e ser for 0 continua, caso contrario começa função
        "acabou" novamente, sendo assim, mudando o valor de r4 pode-se fazer o delay
        entre uma contagem e outra maior.
        acall desligar ;chamada de fuunção para desligar display
        ret

segmentos: ;lista de numeros que serão usados em binario
        db 11000000b ;0
        db 11111001b ;1
        db 10100100b ;2
        db 10110000b ;3
        db 10011001b ;4
        db 10010010b ;5
        db 10000010b ;6
        db 11111000b ;7
        db 10000000b ;8
        db 10010000b ;9

desligar: ;desliga o led p1
        mov p1, #11111111b
        ;setb p2.0
        ret

delay_final: ;função usada para o delay quando se acabar a contagem
        jnb p2.2, dois ;verifica se p2.2 esta apertado
        jnb p2.3, tres ;verifica se p2.3 esta apertado
        jnb p2.0, delay ;verifica se p2.0 esta apertado
        jnb p2.1, delay ;verifica se p2.1 esta apertado

        dois: ;delay para se p2.2 for apertado, tendo delay de 1s no final da contagem
                mov r6, #1000
                iniciod:
                        mov r7, #500
                        djnz r7, $
                        djnz r6, iniciod
                        ret
        tres: ;delay para se p2.2 for apertado, tendo delay de 0,25s no final da contagem
                mov r6, #100 ;500
                iniciot:
                        mov r7, #250
                        djnz r7, $
                        djnz r6, iniciot
                        ret

delay: ;delay padrão usado entre cada numero e/ou no final da contagem se apertado p2.0
e p2.1
        mov A, r2
```

```
        inicio:
                mov r2, A
                DJNZ r2, $
                DJNZ r1, inicio
                acall botoes ;chamada de função para verificar se um botão foi apertado
                durante a contagem
                RET

end
```