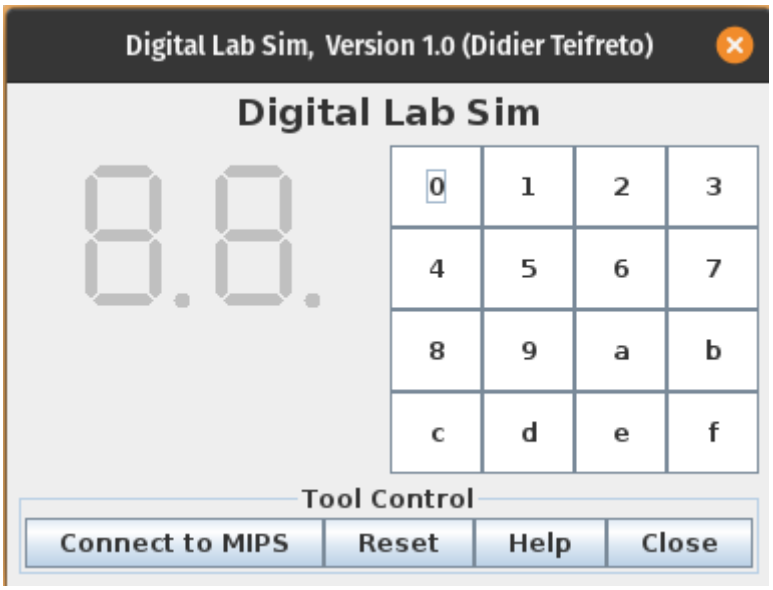
Relatório do Laboratório 02

Alunos: Vinicius Henrique Ribeiro (23200351) e Lucas Furlanetto Pascoali (23204339)
Professor: Marcelo Daniel Berejuck
Disciplina: Organização de Computadores I

Questão 1

A primeira questão solicitava um programa em MARS que escrevesse, sequencialmente, os números 0, 1, 2, 3, 4, 5, 6, 7, 8 e 9 em qualquer um dos displays de sete segmentos disponíveis na ferramenta *Digital Lab Sim*, do próprio simulador.



A estratégia utilizada para resolver esse problema foi a seguinte: ao ler a aba *Help* do *Digital Lab Sim*, descobrimos que para interagir com o *display* de sete segmentos, devemos escrever um número de 7 *bits* um endereço específico da memória, no nosso caso, esse endereço é 0xffff0010 (*display* da direita). Cada *bit* do número a ser escrito no endereço, é responsável por ligar um dos segmentos do *display*. Com isso em mente, a área *.data* do nosso programa realiza um mapeamento do valor (em decimal) que deve ser escrito na memória para que o display escreva cada um dos números. Confira abaixo:

```
.data
        _0_: .word 63   # 0111111
        _1_: .word 6    # 0000110
        _2_: .word 91   # 1011011
        _3_: .word 79   # 1001111
        _4_: .word 102  # 1100110
        _5_: .word 109  # 1101101
        _6_: .word 125  # 1111101
        _7_: .word 7    # 0000111
        _8_: .word 127  # 1111111
        _9_: .word 111  # 1101111
```

Na área *.text* do programa, realizamos apenas o carregamento de cada *word* do *.data* para um registrador temporário e logo em seguida escrevemos o valor do registrador no endereço do *display* de sete segmentos (0xffff0010). Realizamos esses passos para cada um dos números a serem escritos.

```
.text
        lw      $t0, _0_
        sw      $t0, 0xffff0010

        lw      $t0, _1_
        sw      $t0, 0xffff0010

        lw      $t0, _2_
        sw      $t0, 0xffff0010

        lw      $t0, _3_
        sw      $t0, 0xffff0010

        lw      $t0, _4_
        sw      $t0, 0xffff0010

        lw      $t0, _5_
        sw      $t0, 0xffff0010

        lw      $t0, _6_
        sw      $t0, 0xffff0010

        lw      $t0, _7_
        sw      $t0, 0xffff0010

        lw      $t0, _8_
        sw      $t0, 0xffff0010

        lw      $t0, _9_
        sw      $t0, 0xffff0010
```

Nessa questão já podemos perceber um comportamento repetitivo que pode ser transformado em um procedimento/função.

Questão 2

Na segunda questão foi solicitado um programa que fosse capaz de ler o teclado alfanumérico da ferramenta *Digital Lab Sim* e exibi-lo no *display* de sete segmentos dela. Inicialmente, definimos dois vetores na área *.data*, o primeiro deles possui os valores que o teclado alfanumérico retorna para cada dígito pressionado, enquanto o outro possui o valor necessário a ser escrito no endereço do *display* de sete segmentos para que o número seja escrito. A posição desses valores representa o número que eles se referem, por exemplo, o primeiro elemento do primeiro vetor é o valor retornado ao pressionar o número 0 no teclado alfanumérico e o primeiro elemento do segundo vetor é o valor que deve ser escrito no endereço do *display* para que o número zero seja exibido.

```
.data
        key:            .word   0x11, 0x21, 0x41, 0x81, 0x12, 0x22, 0x42, 0x82, 0x14, 0x24, 0x44, 0x84, 0x18, 0x28, 0x48, 0x88
        d7_code:        .word   63 6 91 79 102 109 125 7 127 111 119 124 57 94 121 113
```

Na área .text, iniciamos com um *label* "main", que carrega as configurações do teclado alfanumérico da ferramenta *Digital Lab Sim*, isso é, carrega o endereço da linha de leitura para o registrador $s0 e o endereço no qual o código da tecla pressionada no teclado é armazenado para o registrador $s1. Em seguida, pulamos para o *label* "CAPTURA_TECLA".

```
.text
MAIN:
        li      $s0, 0xffff0012         # salva o endereço linha para leitura do teclado
        li      $s1, 0xffff0014         # salva o endereço do código da tecla
        j       CAPTURA_TECLA
```

Esse *label* "CAPTURA_TECLA" apenas inicializa o registrador $t0 com o valor 1, ele servirá como o contador da linha que estamos lendo. Após isso temos mais 3 *labels* definidos. Começando pelo "LOOP_LER_TECLA", primeiramente armazenamos o byte do nosso contador $t0 no endereço de $s0, ou seja informamos a linha que desejamos ler. Após isso carregamos o valor da memória armazenado em $s1, caso o valor de $t1 seja diferente de zero vamos para o outro *label* "CLICOU_TECLA", ou seja, verificamos se o usuário clicou em alguma tecla dessa linha. Caso o valor seja zero, ou seja o usuário não clicou em nenhuma tecla dessa linha, o valor do contador $t0 vira a soma dele com ele mesmo, ou seja, estamos dobrando seu valor, pois para ler a primeira linha é 1 (0001), a segunda é 2 (0010), a terceira é 4 (0100) e a última é 8 (1000). Caso após essa soma o valor de $t0 seja 16, ou seja, passamos da última linha, voltamos para o *label* "CAPTURA_TECLA", senão ocorre um *jump* para "LOOP_LER_TECLA".

```
10  CAPTURA_TECLA:
11          li      $t0, 1
12          LOOP_LER_TECLA:
13                  sb      $t0, 0($s0)
14                  lw      $t1, 0($s1)
15                  bnez    $t1, CLICOU_TECLA # sai do loop caso precione alguma tecla
16
17                  add     $t0, $t0, $t0   # 1, 2, 4, 8
18                  beq     $t0, 16, CAPTURA_TECLA
19
20                  j       LOOP_LER_TECLA  # se não apertar nada, continua esperando
21
```

Continuando, o *label* "CLICOU_TECLA" que é alcançado quando o usuário, como o próprio nome diz, clica em uma tecla do teclado alfanumérico, começa carregando o endereço da lista "key" definida em .data para o registrador $t2 e o valor 0 para o registrador $t3. Depois definimos o *label* "LOOP_QUAL_TECLA", que carrega a palavra no endereço $t2 para o registrador $t4. Subtraímos $t4 com $t1 e salvamos o valor da subtração em $t5. Caso o valor de $t5 seja zero, significa que

$t4 e $t1 são iguais, então vamos para o *label* "IMPRIMIR_D7SEG". Caso eles sejam diferentes, continuamos somando 4 ao registrador $t2, para que na próxima execução de *LOOP_QUAL_TECLA*, possamos pegar o próximo elemento da lista "KEY". Depois somamos 4 também ao registrador $t3, pois ele futuramente vai servir pra sabermos qual a posição da tecla que o usuário apertou na lista "KEY". Por fim, ocorre um *jump* para "LOOP_QUAL_TECLA".

```
22          CLICOU_TECLA:
23                  la      $t2, key
24                  li      $t3, 0
25                  LOOP_QUAL_TECLA:
26                          lw      $t4, 0($t2)
27                          sub     $t5, $t4, $t1
28                          beq     $t5, $zero, IMPRIMIR_D7SEG
29                          addi    $t2, $t2, 4
30                          addi    $t3, $t3, 4
31                          j       LOOP_QUAL_TECLA
```

O último *label* é "IMPRIMIR_D7SEG" que apenas carrega em $t4, o elemento na posição $t3 da lista "d7_code". Por fim carregamos esse valor de $t4 para o endereço de memória 0xffff0010 que vai ligar o *display* de 7 segmentos da direita com o carácter pressionado pelo usuário. Confira abaixo uma foto de todo o *.text*, para melhor entendimento:

```
4   .text
5   MAIN:
6           li      $s0, 0xffff0012         # salva o endereço linha para leitura do teclado
7           li      $s1, 0xffff0014         # salva o endereço do código da tecla
8           j       CAPTURA_TECLA
9
10  CAPTURA_TECLA:
11          li      $t0, 1
12          LOOP_LER_TECLA:
13                  sb      $t0, 0($s0)
14                  lw      $t1, 0($s1)
15                  bnez    $t1, CLICOU_TECLA # sai do loop caso precione alguma tecla
16
17                  add     $t0, $t0, $t0   # 1, 2, 4, 8
18                  beq     $t0, 16, CAPTURA_TECLA
19
20                  j       LOOP_LER_TECLA  # se não apertar nada, continua esperando
21
22          CLICOU_TECLA:
23                  la      $t2, key
24                  li      $t3, 0
25                  LOOP_QUAL_TECLA:
26                          lw      $t4, 0($t2)
27                          sub     $t5, $t4, $t1
28                          beq     $t5, $zero, IMPRIMIR_D7SEG
29                          addi    $t2, $t2, 4
30                          addi    $t3, $t3, 4
31                          j       LOOP_QUAL_TECLA
32          IMPRIMIR_D7SEG:
33                  lw      $t4, d7_code($t3)
34                  sw      $t4, 0xffff0010
35
36
```