Relatório da resolução do Trabalho Prático 4 "Especificação geral e Especificação 3 em MIPS, Assembly"

(22111296) Vitor R. Borges* Faculdade de Informática — PUCRS

5 de fevereiro de 2024

Resumo

Este artigo mostra a resolução descrita para a especificação geral, e a especificação 3, de acordo com o que fora pedido

1 Elaboração

Para a resolução do projeto, será mostrado aqui o relatório usando diversas imagens que demonstram sua elaboração e sua resolução.

1.1 Pseudocódigo

Utilizando Java, podemos criar o seguinte código, note que é a especificação geral e a especificação 3 em conjunção.

```
public int smulti(int[] A, int[] B){
    int[] C = new int[10], D = new int[10];

    for (int i=0; i<10; i++){
        C[i] = A[i] + B[i];
        D[i] = A[i] - B[i];

    }

    for (int i=0; i<10; i++){
        System.out.print(C[i]+" ");

}

for (int i=0; i<10; i++){
        System.out.print(D[i]+" ");

}

for (int i=0; i<10; i++){
        System.out.print(D[i]+" ");

}

int max=0, Cc=0, Dd=0, SM;

for (int i=0; i<10; i++){
        if (C[i] > max) max = C[i];
        if (D[i] > max) max = D[i];
        Cc = Cc + C[i];
        Dd = Dd + D[i];

}

SM = (Dd + Cc) * max;
    return SM;

}
```

Figura 1: Código em Java relevante às especificações.

^{*}v.rafael02@edu.pucrs.br,

1.2 Relação com MIPS

Aqui realizaremos a comparação entre o código em Java e seus equivalentes em MIPS.

```
public int smulti(int[] A, int[] B){
                                                                    $t0,$t0,$t0
                                                                                    #inicializa $t0 em 0
            int[] C = new int[10], D = new int[10]
                                                            xor
                                                                                    #carrega e inicializa $t1 com valor de n(10)
           for (int i=0; i<10; i++){
                                                                    $t1, 0($t1)
                                              D[i] = A[i] - B[i];
                                                            la
                                                                    $t2. A
                                                                                   #carrega A para $t2
                                                            la
                                                                    $t3, B
                                                                                    #carrega B para $t3
                                                            1a
                                                                    $t8, C
                                                                                    #carrega C para #t8
            for (int i=0: i<10: i++){
                System.out.print(C[i]+" ");
10
11
12
13
                                                                    $t4, 0($t2)
                                                                                   #$t4 carrega valor de A[i]
                                                            lw
                                                                   $t5, 0($t3)
                                                                                    #$t5 carrega valor de B[i]
                                                            lw
            for (int i=0; i<10; i++){
14
                System.out.print(D[i]+" ");
                                                                    $t6, $t4, $t5  #$t6 recebe A[i] + B[i]
15
                                                            SW
                                                                    $t6, 0($t8)
                                                                                    #guarda valor de $t6 em C[i]
17
            int max=0, Cc=0, Dd=0, SM;
18
                                                                   $t0, $t0, 1
                                                            addi
                                                                                    #incrementa i
            for (int i=0; i<10; i++){
19
                                                                    $t2, $t2, 4
                                                                                    #próximo valor de A
                if (C[i] > max) max = C[i];
                                                            addi
20
                                                                    $t3, $t3, 4
                                                                                    #próximo valor de B
                                                            addi
                if (D[i] > max) max = D[i];
21
22
                                                            addi
                                                                   $t8, $t8, 4
                                                                                    #próximo valor de C
                Cc = Cc + C[i];
                Dd = Dd + D[i];
                                                            blt $t0, $t1, soma
                                                                                    #auxiliar de loop
24
25
            SM = (Dd + Cc) * max:
27
            return SM;
```

Figura 2: Comparação da adição.

```
public int smulti(int[] A, int[] B){
                                                                     $t0.$t0.$t0
                                                                                    #reinicializa $t0 em (
                                                                     $t6,$t6,$t6
                                                                                    #inicializa $t6 em 0
            int[] C = new int[10], D = new int[10];
                                                             xor
                                                             la
                                                                     $t1, n
                                                                                    #reinicializa $t1 em n(10) por consistência
             for (int i=0; i<10; i++){
               C[i] = A[i] + B[i];
5
               D[i] = A[i] - B[i];
6
7
                                                             la
                                                                     $t2. A
                                                                                    #carrega A para $t2
8
                                                             la
                                                                     $t3, B
                                                                                    #carrega B para $t3
             for (int i=0; i<10; i++){
                                                             la
                                                                     $t8, D
                                                                                    #carrega D para $t8
                System.out.print(C[i]+" ");
11
                                                     sub:
                                                                     $t4, 0($t2)
                                                                                    #carrega A[i] para $t4
12
                                                                    $t5, 0($t3)
                                                                                    #carrega B[i] para $t5
                                                             lw
13
            for (int i=0; i<10; i++){
                System.out.print(D[i]+" ");
14
                                                                    $t6, $t4, $t5 #$t6 recebe A[i] - B[i]
15
16
17
            int max=0, Cc=0, Dd=0, SM;
                                                                     $t6, 0($t8)
                                                                                    #D[i] recebe $t6
18
                                                                    $t0, $t0, 1
19
             for (int i=0; i<10; i++){
                                                             addi
                                                                                    #incrementa i
                                                             addi
                                                                    $t2, $t2, 4
                                                                                    #A recebe próximo valor
                if (C[i] > max) max = C[i];
                                                                    $t3, $t3, 4
                                                                                    #B recebe próximo valor
                                                             addi
                if (D[i] > max) max = D[i];
                                                             addi
                                                                    $t8, $t8, 4
                                                                                    #D busca próximo valor
22
                Cc = Cc + C[i]:
                Dd = Dd + D[i];
23
                                                             blt $t0, $t1, sub
                                                                                    #auziliar de loop
24
25
26
            SM = (Dd + Cc) * max;
27
            return SM;
```

Figura 3: Comparação da subtração.

```
1 | public int smulti(int[] A, int[] B){
                                                                   $t0.$t0.$t0
                                                                                   #reinicializa $t0 em 0
            int[] C = new int[10], D = new int[10];
                                                            xor
                                                                   $t6,$t6,$t6
                                                                                   #reinicializa $t6 em 0
                                                            xor $t7.$t7.$t7
                                                                              #inicializa $t7 em 0
            for (int i=0; i<10; i++){
               C[i] = A[i] + B[i];
                                                           la
                                                                   $t1. n
                                                                                   #carrega endereco e valor de n(10), por consistência
                Dril = Aril - Bril;
                                                                   $t1, 0($t1)
                                                           1w
                                                                   $t2, C
                                                                                   #carrega C para $t2
            for (int i=0; i<10; i++){
                                                            1a
                                                                   $t3, D
                                                                                   #carrega D para $t3
                System.out.print(C[i]+" ");
10
11
                                                    somacd
                                                           1w
                                                                   $t4. 0($t2)
                                                                                   #carrega C[i]
13
            for (int i=0; i<10; i++){
                                                           1w
                                                                   $t5, 0($t3)
                                                                                   #carrega D[i]
               System.out.print(D[i]+" ");
14
15
                                                                   $t6, $t6, $t4 #$t6 recebe $t6 + C[i] CC
                                                            add
16
17
18
                                                            add
                                                                   $t7, $t7, $t5 #$t6 recebe $t7 + D[i] Dd
            int max=0, Cc=0, Dd=0, SM;
                                                                   $t0, $t0, 1
19
            for (int i=0; i<10; i++){
                                                            addi
                                                                   $t2, $t2, 4
                                                                                   #busca próximo valor de C
20
                if (C[i] > max) max = C[i];
                                                            addi
                                                                   $t3, $t3, 4
                                                                                   #busca próximo valor de D
21
                if (D[i] > max) max = D[i];
           Cc = Cc + C[i];
22
                                                                                          #auxiliar de loop
                                                           blt
                                                                   $t0. $t1. somacd
23
              ■ Dd = Dd + D[i];
24
25
26
27
                                                        add $t6, $t6, $t7 #$t6 recebe $t6 + $t7 SM = Cc + Dd (TEMPORÁRIO)
            SM = (Dd + Cc) * max;
                                                                   $t0, SM
                                                                                   #carrega SM
            return SM;
                                                                   $t6, 0($t0)
                                                                                   #SM recebe valor de $t6 (somatório de C[i] + somatório de D[i])
```

Figura 4: Comparação da primeira soma entre somatórios de C e D.

```
1 | public int smulti(int[] A, int[] B){
                                                                      $t0.$t0.$t0
                                                                                     #reinicializa $t0
            int[] C = new int[10], D = new int[10]
                                                                     $t6. $t6. $t6 #reinicializa #t6
                                                              xor
             for (int i=0; i<10; i++){
                                                                     $t1, n
                                                                                     #carrega endereço e valor de n(10), por consistência
                                                              la
                 C[i] = A[i] + B[i];
                                                                     $t1, 0($t1)
                D[i] = A[i] - B[i];
                                                                     $t2, C
                                                                                     #carrega C
                                                              1a
                                                                     $t3, D
                                                                                     #carrega D
            for (int i=0; i<10; i++){
                System.out.print(C[i]+" ");
10
12
                                                         1w
                                                                 $t4, 0($t2)
                                                                                     #carrega C[i]
                                                                 $t5, 0($t3)
                                                                                     #carrega D[i]
            for (int i=0; i<10; i++){
                                                         lw
13
                 System.out.print(D[i]+" ");
                                                                 $t4, $t6, maxb1
                                                                                     #branch para ver se valor atual é menor que valor de C[i]
                                                                 $t6, $t4, 0
                                                                                     #se C[i] for maior, pega este valor
            int max=0, Cc=0, Dd=0, SM;
17
                                                      maxb1:
                                                         blt
                                                                 $t5, $t6, postmax \, #branch para ver se valor atual \acute{e} menor que valor de D[\emph{i}]
            for (int i=0; i<10; i++){
19
                                                         addi
                                                                $t6, $t5, 0
                                                                                     #se D[i] for maior, pega este valor
                if (C[i] > max) max = C[i];
20
21
22
                if (D[i] > max) max = D[i];
                 Cc = Cc + C[i];
                                                      postmax:
                                                                 $t0, $t0, 1
                                                         addi
                                                                                     #incrementa i
23
                 Dd = Dd + D[i];
                                                          addi
                                                                                      #próximo valor de C
                                                                 $t3, $t3, 4
                                                                                     #próxiom valor de D
             SM = (Dd + Cc) * max:
                                                         blt
                                                                 $t0, $t1, max
                                                                                     #auxiliar de loop
27
             return SM:
28
                                                                 $t5, MAX
                                                                                     #carrega endereço de MAX
                                                         SW
                                                                 $t6, 0($t5)
                                                                                     #MAX recebe $t6
```

Figura 5: Comparação da procura do valor máximo de C e D.

```
public int smulti(int[] A, int[] B){
           int[] C = new int[10], D = new int[10];
            for (int i=0; i<10; i++){
               C[i] = A[i] + B[i];
               Dril = Aril - Bril;
            for (int i=0; i<10; i++){
               System.out.print(C[i]+" ");
                                                               $t0,$t0,$t0
           for (int i=0; i<10; i++){
               System.out.print(D[i]+" ");
                                                               $t2,$t2,$t2
                                                       xor
15
                                                                                  #$t5 carrega endereço e valor de MAX
           int max=0, Cc=0, Dd=0, SM;
                                                               $t5, 0($t5)
18
            for (int i=0; i<10; i++){
                                                               $t1. SM
                                                                              #$t1 carrega endereço e valor de SM
19
                                                             $t1, 0($t1)
                if (C[i] > max) max = C[i];
               if (D[i] > max) max = D[i];
                                                   multism:
               Cc = Cc + C[i];
                                                          add $t2, $t2, $t1
               Dd = Dd + D[i];
                                                           sub $t5, $t5, 1
                                                           bne $t5, $t0, multism
            SM = (Dd + Cc) * max;
26
27
                                                                   $t3. SM
                                                                                  #carrega endereço de SM em $t2
            return SM;
                                                                   $t2, 0($t3)
                                                                                  \#guarda valor final de SM em SM
```

Figura 6: Comparação da multiplicação de somatório de C e D pelo valor máximo.

1.3 Área de dados

Aqui podemos ver a área de dados que fora criada para a elaboração das especificações.

```
129 .data
130 n: .word 10
131 A: .word 33 31 44 59 52 39 17 68 -73 -14
132 B: .word 39 53 1 90 33 54 29 90 -7 -68
133 C: .word 0 0 0 0 0 0 0 0 0 0
134 D: .word 0 0 0 0 0 0 0 0 0 0
135 SM: .word 0
136 MAX: .word 0
```

Figura 7: Área de Dados em MIPS.

Esperamos que:

```
C = {72, 84, 45, 149, 85, 93, 46, 158, -80, -82}.
D = {-6, -22, 43, -31, 19, -15, -12, -22, -66, 54};
```

- SM {80896}.
- $MAX = \{158\}.$

2 Utilização do MARS

Fora utilizado MARS para a elaboração do projeto, e será demonstrado suas áreas de dados antes e depois da execução. Levamos as seguintes cores para interpretação:

- Amarelo para *n*.
- Vermelho para A.
- Azul para *B*.
- Roxo para C.
- Verde para *D*.
- Preto para SM.
- Rosa para MAX.

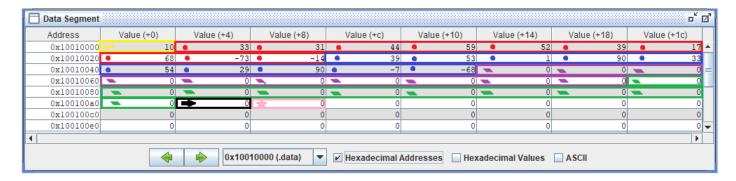


Figura 8: Área de dados antes da execução.

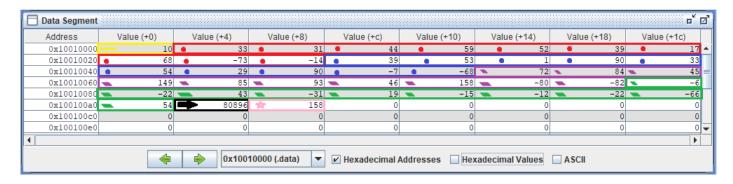


Figura 9: Área de dados após a execução.

3 Utilização do ModelSim

Para a utilização do ModelSim, foram necessárias algumas modificações no arquivo *wave.do* que podem ser observadas na Figura 10.

```
quietly virtual signal -install /cpu tb/Data mem {
                                                                                                                                         (context /cpu tb/Data mem )(RAM(143) &RAM(142) &RAM(141) &RAM(140) )} mem35
                                                                                                                                                                                                                                                                                                 &RAM(144)
quietly virtual signal -install /cpu_tb/Data_mem {
                                                                                                                                          (context /cpu_tb/Data_mem )(RAM(147) &RAM(146)
                                                                                                                                                                                                                                                                      &RAM(145)
quietly virtual signal -install /cpu tb/Data mem {
                                                                                                                                         (context /cpu tb/Data mem )(RAM(151) &RAM(150)
                                                                                                                                                                                                                                                                      &RAM(149)
                                                                                                                                                                                                                                                                                                 &RAM(148)
                                                                                                                                                                                                                                                                                                                                   mem37
quietly virtual signal -install /cpu_tb/Data_mem
                                                                                                                                         (context /cpu_tb/Data_mem ) (RAM(155) &RAM(154)
quietly virtual signal -install /cpu_tb/Data_mem {
                                                                                                                                         (context /cpu_tb/Data_mem )(RAM(159) &RAM(158) &RAM(157)
                                                                                                                                                                                                                                                                                                &RAM(156)
quietly virtual signal -install /cpu tb/Data mem { (context /cpu tb/Data mem ) (RAM(163) &RAM(162) &RAM(161) &RAM(160)
Quietly virtual signal -install /cpu_tb/Data_mem ( (context /cpu_tb/Data_mem ) (RAM(167) &RAM(166) &RAM(165) &RAM(164) add wave -noupdate -color purple -itemcolor purple -radix decimal /cpu_tb/Data_mem/mem35 add wave -noupdate -color purple -itemcolor purple -radix decimal /cpu_tb/Data_mem/mem35 add wave -noupdate -color purple -itemcolor purple -itemc
add wave -noupdate -color purple -itemcolor purple -radix decimal /cpu_tb/Data_mem/mem37 add wave -noupdate -radix decimal /cpu tb/Data_mem/mem38
add wave -noupdate -color purple -itemcolor purple -radix decimal /cpu_tb/Data_mem/mem39
add wave -noupdate -radix decimal /cpu_tb/Data_mem/mem40
add wave -noupdate -color purple -itemcolor purple -radix decimal /cpu_tb/Data_mem/mem41
add wave -noupdate -radix decimal /cpu_tb/Data_mem/mem42
```

Figura 10: Modificações feitas em wave.do.

Essas modificações foram feitas através do estudo do código dos arquivos criados para utilização com um arquivo *mips.txt*. Agora demonstraremos a Área de Dados quando utilizadas no ModelSim. Levamos em consideração os seguintes valores de *mem* para identificação:

- 0 para *n*.
- 1 a 10 para A.
- 11 a 20 para *B*.
- 21 a 30 para *C*.
- 31 a 40 para *D*.
- 41 para *SM*.
- 42 para *MAX*.

Consideramos também as seguintes cores para identificação, são diferentes das utilizadas nas Figuras 8 e 9:

- Verde para *n*.
- Vermelho para A.
- Azul para B.

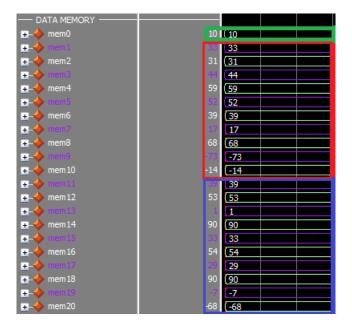


Figura 11: Elementos n, $A \in B$.

Para a próxima Figura, consideramos as seguintes cores para identificação:

- Amarelo para C.
- Roxo para D.
- Branco para SM.
- Verde para MAX.

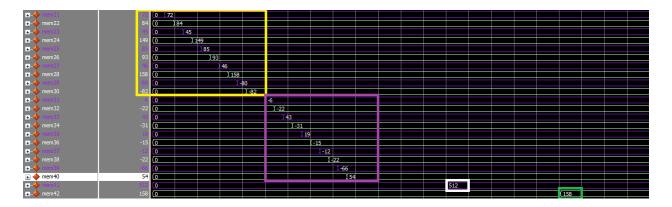


Figura 12: Elementos C, D, Valor temporário de SM e MAX.

Vemos agora o valor de SM quando no final da simulação. Levamos em consideração as mesmas cores utilizadas na Figura 12.

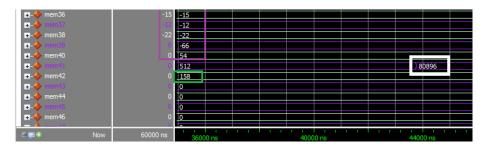


Figura 13: Elemento SM em seu valor final.

Agora mostraremos a área de simulação completa quando utilizada no ModelSim. Levamos em conta que os tempos estão como *aproximadamente*, também consideramos as seguintes cores como equivalentes em relação ao tempo de execução de certas especificações:

- Vermelho para C[i] = A[i] + B[i].
- Verde para D[i] = A[i] B[i].
- Amarelo para Somatório de C + Somatório de D.
- Roxo para identificação do valor máximo C e D.
- Branco para multiplicação do Somatório de C + Somatório de D pelo valor máximo de C e D.

A Figura 14 pode ser encontrada na próxima página, devido ao seu tamanho.

Todas as figuras utilizadas para criação deste relatório podem ser encontradas em alta definição no seguinte acervo do **GOOGLE DRIVE**¹ de Vitor Rafael.

A versão em LATEX do relatório pode ser encontrada no acervo do **OVERLEAF**² de Vitor Rafael.

¹https://abre.ai/fundsisvrafael

ou https://drive.google.com/drive/folders/1kDszM2yhHpP99I3bMTFv97-pJbznRLWz?usp=sharing

²https://abre.ai/latexfundsisvrafael

ou https://www.overleaf.com/read/spngqxcnwqsxe2536a

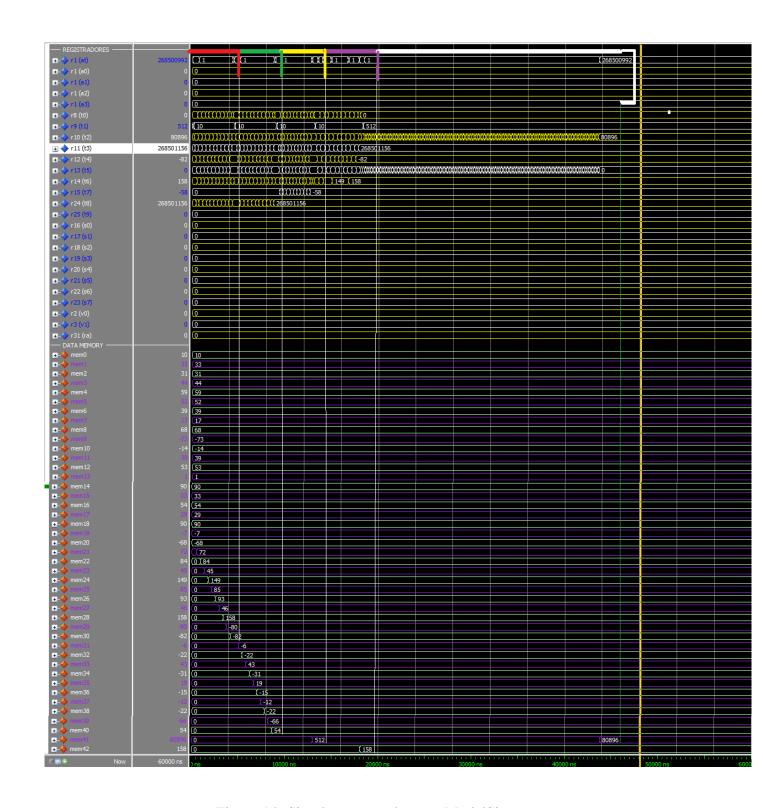


Figura 14: Simulação completa em ModelSim.