DUE IN CLASS

STUDENTS IDENTIFICATION:

Number:	Name:
102908	Luana Ferraz
103091	Pedro Curto
103095	Daniel Nunes

2.1 Simple execution, without data forwarding techniques

e)

Clock cycles	18	Instructions	7	Average CPI	2.571

f)

Clock cycles	174		Stalls: - Data	101
Instructions	61		- Structural	0
Average CPI	2.852		- Branch Taken	8

g) Na execução do programa, pode-se observar que ocorrem 8 Branch Taken stalls, num total de 9 ciclos executados; após cada instrução "bne", a instrução seguinte começa a ser executada mas não termina; logo, é flushed uma vez que a branch prediction foi incorreta. Logo, a branch prediction policy adotada pelo simulador é, então, Predict not taken, o que provoca com que ocorram consistentemente stalls do pipeline uma vez que o branch é, na verdade, taken.

2.2 Application of data forwarding techniques

c)

Clock cycles	136		Stalls: - Data	63
Instructions	61		- Structural	9
Average CPI	2.230		- Branch Taken	8

d)

$$speedup = \frac{t_{antigo}}{t_{novo}} = \frac{\#cycles_{antigo} \times \cancel{t_{cycle}}}{\#cycles_{novo} \times \cancel{t_{cycle}}} = \frac{174}{136} \simeq 1.279$$

2.3 Source code optimization: minimization of data and structural hazards

a) Attach a copy of the new assembly program.

c)

Clock cycles	118		Stalls: - Data	36
Instructions	61		- Structural	9
Average CPI	1.934		- Branch Taken	8

d)

$$speedup = \frac{t_{antigo}}{t_{novo}} = \frac{\#cycles_{antigo} \times \cancel{t_{cycle}}}{\#cycles_{novo} \times \cancel{t_{cycle}}} = \frac{174}{118} \simeq 1.475$$

2.4 Source code optimization: loop unrolling

a) Attach a copy of the new assembly program.

c)

Clock cycles	91
Instructions	49
Average CPI	1.857

Stalls:	- Data	51
	- Structural	9
	- Branch Taken	2

d)

$$speedup = \frac{t_{antigo}}{t_{novo}} = \frac{\#cycles_{antigo} \times \cancel{t_{cycle}}}{\#cycles_{novo} \times \cancel{t_{cycle}}} = \frac{174}{91} \simeq 1.912$$

2.5 Source code optimization: branch delay slot

a) Attach a copy of the new assembly program.

d)

Clock cycles	101
Instructions	61
Average CPI	1.656

Stalls:	- Data	27
	- Structural	9
	- Branch Taken	0

e)

$$speedup = \frac{t_{antigo}}{t_{novo}} = \frac{\#cycles_{antigo} \times \cancel{t_{cycle}}}{\#cycles_{novo} \times \cancel{t_{cycle}}} = \frac{174}{101} \simeq 1.723$$

Table 1: Pipeline time diagram, with data forwarding techniques.

	INSTRUCTIONS	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
1	lw $12, 0($1)	F	D	X	M	W																																			
2	dmul $12, $12, $9		F	D	M_0	R	M_1	M_2	M_3	M_4	M_5	M_6	M	W																											
3	dadd $9, $9, $12			F	D	R	X	R	R	R	R	A	S	M	W																										
4	daddi $5, $5, 1				F	F	D	D	D	D	D	D	D	X	M	W																									
5	daddi $1, $1, 8						F	F	F	F	F	F	F	D	X	M	W																								
6	bne $6, $5, loop													F	D	X	M	W																							
7	sw $9, mult ($0)														F																										
8	lw $12, 0($1)															F																									
9																																									
10																																									
11																																									
12																																									
13																																									
14																																									
15																																									
16																																									
17																																									
18																																									
19																																									
20																																									
21																																									
22																																									
23																																									
24																																									
25																																									
26																																									
27																																									
28																																									
29																																									
30																																									

Table 2: Pipeline time diagram, with minimization techniques to reduce the data and structural hazards.

	INSTRUCTIONS	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
1	lw $12, 0($1)	F	D	X	M	W																																			
2	daddi $5, $5, 1		F	D	X	M	W																																		
3	dmul $12, $12, $9			F	D	M_0	M_1	M_2	M_3	M_4	M_5	M_6	M	W																											
4	daddi $1, $1, 8				F	D	X	M	W																																
5	dadd $9, $9, $12					F	D	X	R	R	R	R	S	M	W																										
6	bne $6, $5, loop						F	D	D	D	D	D	D	X	M	W																									
7	sw $9, mult($0)							F	F	F	F	F	F																												
8	lw $12, 0($1)													F																											
9																																									
10																																									
11																																									
12																																									
13																																									
14																																									
15																																									
16																																									
17																																									
18																																									
19																																									
20																																									
21																																									
22																																									
23																																									
24																																									
25																																									
26																																									
27																																									
28																																									
29																																									
30																																									

Table 3: Pipeline time diagram: usage of loop unrolling minimization techniques to reduce the control hazards.

	INSTRUCTIONS	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
1	dmul $16,$12,$9	F	D	M_0	M_1	M_2	M_3	M_4	M_5	M_6	M	W																													
2	daddi $1,$1,8		F	D	X	M	W																																		
3	lw $12,16($1)			F	D	X	M	W																																	
4	dadd $9,$9,$16				F	D	X	R	R	R	S	M	W																												
5	dmul $17,$13,$9					F	D	R	R	R	R	M_0	M_1	M_2	M_3	M_4	M_5	M_6	M	W																					
6	daddi $1,$1,16						F	F	F	F	F	D	X	M	W																										
7	lw $13,8($1)											F	D	X	M	W																									
8	dadd $9,$9,$17												F	D	X	R	R	R	S	M	W																				
9	dmul $18,$14,$9													F	D	R	R	R	R	M_0	M_1	M_2	M_3	M_4	M_5	M_6	M	W													
10	daddi $5,$5,1														F	F	F	F	F	D	X	M	W																		
11	lw $14,16($1)																			F	D	X	M	W																	
12	dadd $9,$9,$18																				F	D	X	R	R	R	S	M	W												
13	bne $6,$5,loop																					F	D	D	D	D	D	X	M	W											
14	sw $9,mult($0)																						F	F	F	F	F														
15																																									
16																																									
17																																									
18																																									
19																																									
20																																									
21																																									
22																																									
23																																									
24																																									
25																																									
26																																									
27																																									
28																																									
29																																									
30																																									

Table 4: Pipeline time diagram: usage of branch delay slot techniques to reduce the control hazards.

	INSTRUCTIONS	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
1	lw $12, 0($1)	F	D	D	D	D	D	X	M	W																															
2	daddi $5, $5,1		F	F	F	F	F	D	X	M	W																														
3	dmul $12,$12,$9							F	D	M_0	M_1	M_2	M_3	M_4	M_5	M_6	M	W																							
4	daddi $1, $1,8								F	D	X	M	W																												
5	bne $6, $5, loop									F	D	X	M	W																											
6	dadd $9, $9,$12										F	D	X	R	R	R	S	M	W																						
7	lw $12,0($1)											F																													
8																																									
9																																									
10																																									
11																																									
12																																									
13																																									
14																																									
15																																									
16																																									
17																																									
18																																									
19																																									
20																																									
21																																									
22																																									
23																																									
24																																									
25																																									
26																																									
27																																									
28																																									
29																																									
30																																									

Table 5: Pipeline time diagram, without data forwarding techniques.

	INSTRUCTIONS	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
1	lw $12, 0($1)	F	D	X	M	W																																			
2	dmul $12, $12, $9		F	D	R	R	M_0	M_1	M_2	M_3	M_4	M_5	M_6	M	W																										
3	dadd $9, $9, $12			F	F	F	D	R	R	R	R	R	R	R	R	X	M	W																							
4	daddi $5, $5, 1						F	F	F	F	F	F	F	F	F	D	X	M	W																						
5	daddi $1, $1, 8															F	D	X	M	W																					
6	bne $6, $5, loop																F	D	R	X	M	W																			
7	sw $9, mult($0)																	F	F																						
8	lw $12, 0($1)																			F																					
9																																									
10																																									
11																																									
12																																									
13																																									
14																																									
15																																									
16																																									
17																																									
18																																									
19																																									
20																																									
21																																									
22																																									
23																																									
24																																									
25																																									
26																																									
27																																									
28																																									
29																																									
30																																									

Code for exercise 2.3:

```
                .data
A:              .word   1, 3, 1, 6, 4
                .word   2, 4, 3, 9, 5
mult:   .word   0

        .code
        daddi   $1, $0, A       ; *A[0]
        daddi   $5, $0, 1       ; $5 = 1 ;; i
        daddi   $6, $0, 10      ; $6 = N ;; N = 10
        lw      $9, 0($1)       ; $9 = A[0]  ;; mult
        daddi   $1, $1, 8       ;

loop:   lw      $12, 0($1)      ; $12 = A[i]
        daddi   $5, $5, 1       ; i++
        dmul    $12, $12, $9    ; $12 = $12*$9 ;; $12 = A[i]*mult
        daddi   $1, $1, 8       ; moves to next A element
        dadd    $9, $9, $12     ; $9 = $9 + $12  ;; mult = mult + A[i]*mult
        bne     $6, $5, loop    ; Exit loop if i == N

        sw      $9, mult($0)    ; Store result
        halt

;; Expected result: mult = f6180 (hex), 1008000 (dec)
```

Code for exercise 2.4:

```
                .data
A:              .word   1, 3, 1, 6, 4
                .word   2, 4, 3, 9, 5
mult:   .word   0

        .code
        daddi   $1, $0, A       ; $1 = &A[0]
        daddi   $5, $0, 1       ; $5 = i (começa a 1 e vai até 4, para o loop correr 3x)
        daddi   $6, $0, 4       ; $6 = N (para comparar no bne)
        lw      $9, 0($1)       ; $9 = A[0] (mult)
        lw      $12, 8($1)      ; $12 = A[1]
        lw      $13, 16($1)     ; $13 = A[2]
        lw      $14, 24($1)     ; $14 = A[3]
        daddi   $1, $1, 8       ; $1 = &A[1] ()

loop:
        dmul    $16, $12, $9    ; $16 = A[i]*mult
        daddi   $1, $1, 8       ; avança 1 elemento de A (fica em &A[i+1])
        lw      $12, 16($1)     ; $12 = A[i+3]
        dadd    $9, $9, $16     ; mult = mult + A[i+1]*mult

        dmul    $17, $13, $9    ; $16 = A[i+1]*mult
        daddi   $1, $1, 16      ; Avança 2 elementos de A (fica em &A[i+3])
        lw      $13, 8($1)      ; $12 = A[i+4]
        dadd    $9, $9, $17     ; mult = mult + A[i+1]*mult

        dmul    $18, $14, $9    ; $16 = A[i+2]*mult
        daddi   $5, $5, 1       ; i+=1
        lw      $14, 16($1)     ; $12 = A[i+5]
        dadd    $9, $9, $18     ; mult = mult + A[i+2]*mult

        bne     $6, $5, loop    ; Exit loop if index == N ;; 1ª e 2ª iter bne checks (2 stalls), 3ª iter bne falha -> guarda valor

        sw      $9, mult($0)    ; Store result
        halt

;; Expected result: mult = f6180 (hex), 1008000 (dec)
```

Code for exercise 2.5:

```
                .data
A:              .word   1, 3, 1, 6, 4
                .word   2, 4, 3, 9, 5
mult:   .word   0

        .code
        daddi   $1, $0, A       ; *A[0]
        daddi   $5, $0, 1       ; $5 = 1 ;; i
        daddi   $6, $0, 10      ; $6 = N ;; N = 10
        lw      $9, 0($1)       ; $9 = A[0]  ;; mult
        daddi   $1, $1, 8       ;

loop:   lw      $12, 0($1)      ; $12 = A[i]
        daddi   $1, $1, 8       ;
        dmul    $12, $12, $9    ; $12 = $12*$9 ;; $12 = A[i]*mult
        daddi   $5, $5, 1       ; i++
        bne     $6, $5, loop    ; Exit loop if i == N
        dadd    $9, $9, $12     ; $9 = $9 + $12  ;; mult = mult + A[i]*mult

        sw      $9, mult($0)    ; Store result
        halt

;; Expected result: mult = f6180 (hex), 1008000 (dec)
```