**Domanda 1**

Considerando il processore MIPS64 e l’architettura descritta in seguito:

|  |  |  |
| --- | --- | --- |
| * + Integer ALU: 1 clock cycle   + Data memory: 1 clock cycle   + FP multiplier unit: pipelined 6 stages | * + FP arithmetic unit: pipelined 3 stages   + FP divider unit: not pipelined unit that requires 6 clock cycles   + branch delay slot: 1 clock cycle, and the branch delay slot disabled | * + forwarding enabled   + it is possible to complete instruction EXE stage in an out-of-order fashion. |

Usando il frammento di codice riportato, si calcoli il tempo di esecuzione dell’intero programma in colpi di clock e si completi la seguente tabella.

; for (i = 0; i < 100; i++) {

; v4[i] = (v1[i] / v2[i]) + v3[i];

; v5[i] = v4[i] \* v3[i]; }

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | Clock cycles |
| .data |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| V1: .double “100 values” |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| V2: .double “100 values” |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| V3: .double “100 values”  …  V5: .double “100 zeros” |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| .text |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| main: daddui r1,r0,0 | F | D | E | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 5 |
| daddui r2,r0,100 |  | F | D | E | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |
| loop: l.d f1,v1(r1) |  |  | F | D | E | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |
| l.d f2,v2(r1) |  |  |  | F | D | E | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |
| div.d f4,f1,f2 |  |  |  |  | F | D | s | / | / | / | / | / | / | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 7 |
| l.d f3,v3(r1) |  |  |  |  |  | F | s | D | E | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 0 |
| dadd.d f4,f4,f3 |  |  |  |  |  |  |  | F | D | s | s | s | s | + | + | + | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 3 |
| s.d f4,v4(r1) |  |  |  |  |  |  |  |  | F | s | s | s | s | D | E | s | S | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |
| mul.d f5,f4,f3 |  |  |  |  |  |  |  |  |  |  |  |  |  | F | D | s | s | \* | \* | \* | \* | \* | \* | M | W |  |  |  |  |  |  |  |  |  |  |  | 6 |
| s.d f5,v5(r1) |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | s | s | D | E | s | s | s | s | S | M | W |  |  |  |  |  |  |  |  |  |  | 1 |
| daddui r1,r1,8 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | D | s | s | s | s | s | E | M | W |  |  |  |  |  |  |  |  |  | 1 |
| daddi r2,r2,-1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | s | s | s | s | s | D | E | M | W |  |  |  |  |  |  |  |  | 1 |
| bnez r2,loop |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | s | D | E | M | W |  |  |  |  |  |  | 2 |
| Halt |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | - | - | - | - |  |  |  |  |  | 1 |
| TOTAL | 6+100\*25 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 2506 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**Domanda 2**

Considerando il programma precedente, cosa succede se i salti vengono svolti durante la fase di EXE e non durante la fase di DECODE? Motivare la risposta.

In questo caso non dovrei aggiungere lo stallo a livello della D e avrei 1 clock in meno ad iterazione (la branch avrebbe 1 clock invece di 2, in quanto lo stallo non è più necessario); inoltre in questo caso verrebbe fatto in più la decode della halt finale, sprecando 1 clock in + ad ogni iterazione.

**Domanda 3**

Considerando il programma precedente e l’architettura del processore superscalare descritto in seguito; completare la tabella relativa alle prime 3 iterazioni.

Processor architecture:

• Issue 2 instructions per clock cycle

• jump instructions require 1 issue

• handle 2 instructions commit per clock cycle

• timing facts for the following separate functional units:

i. 1 Memory address 1 clock cycle

ii. 1 Integer ALU 1 clock cycle

iii. 1 Jump unit 1 clock cycle

iv. 1 FP multiplier unit, which is pipelined: 6 stages

v. 1 FP divider unit, which is not pipelined: 6 clock cycles

vi. 1 FP Arithmetic unit, which is pipelined: 3 stages

• Branch prediction is always correct

• There are no cache misses

• There are 2 CDB (Common Data Bus).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| # iteration |  | Issue | EXE | MEM | CDB x2 | COMMIT x2 |
| 1 | l.d f1,v1(r1) | 1 | 2ea | 3 | 4 | 5 |
| 1 | l.d f2,v2(r1) | 1 | 3ea | 4 | 5 | 6 |
| 1 | div.d f4,f1,f2 | 2 | 6d-11d | / | 12 | 13 |
| 1 | l.d f3,v3(r1) | 2 | 4ea | 5 | 6 | 13 |
| 1 | dadd.d f4,f4,f3 | 3 | 13a-15a | / | 16 | 17 |
| 1 | s.d f4,v4(r1) | 3 | 5ea | / | / | 17 |
| 1 | mul.d f5,f4,f3 | 4 | 17m-22m | / | 23 | 24 |
| 1 | s.d f5,v5(r1) | 4 | 6ea | / | / | 24 |
| 1 | daddui r1,r1,8 | 5 | 6i | / | 7 | 25 |
| 1 | daddi r2,r2,-1 | 5 | 7i | / | 8 | 25 |
| 1 | bnez r2,loop | 6 | 9j | / | / | 26 |
| 2 | l.d f1,v1(r1) | 7 | 8ea | 9 | 10 | 26 |
| 2 | l.d f2,v2(r1) | 7 | 9ea | 10 | 11 | 27 |
| 2 | div.d f4,f1,f2 | 8 | 12d-17d | / | 18 | 27 |
| 2 | l.d f3,v3(r1) | 8 | 10ea | 11 | 12 | 28 |
| 2 | dadd.d f4,f4,f3 | 9 | 19a-21a | / | 22 | 28 |
| 2 | s.d f4,v4(r1) | 9 | 11ea | / | / | 29 |
| 2 | mul.d f5,f4,f3 | 10 | 23m-28m | / | 29 | 30 |
| 2 | s.d f5,v5(r1) | 10 | 12ea | / | / | 30 |
| 2 | daddui r1,r1,8 | 11 | 12i | / | 13 | 31 |
| 2 | daddi r2,r2,-1 | 11 | 13i | / | 14 | 31 |
| 2 | bnez r2,loop | 12 | 15j | / | / | 32 |
| 3 | l.d f1,v1(r1) | 13 | 14ea | 15 | 16 | 32 |
| 3 | l.d f2,v2(r1) | 13 | 15ea | 16 | 17 | 33 |
| 3 | div.d f4,f1,f2 | 14 | 18d-23d | / | 24 | 33 |
| 3 | l.d f3,v3(r1) | 14 | 16ea | 17 | 18 | 34 |
| 3 | dadd.d f4,f4,f3 | 15 | 25a-27a | / | 28 | 34 |
| 3 | s.d f4,v4(r1) | 15 | 17ea | / | / | 35 |
| 3 | mul.d f5,f4,f3 | 16 | 29m-34m | / | 35 | 36 |
| 3 | s.d f5,v5(r1) | 16 | 18ea | / | / | 36 |
| 3 | daddui r1,r1,8 | 17 | 18i | / | 19 | 37 |
| 3 | daddi r2,r2,-1 | 17 | 19i | / | 20 | 37 |
| 3 | bnez r2,loop | 18 | 21j | / | / | 38 |

**Domanda 4**

Considerando il segmento di codice presentato nella tabella precedente, se assumiamo che ci sia un unico Common Data Bus, qual è la prima istruzione che dovrebbe stallare durante l’esecuzione del programma? motivare la risposta.

Se CDB passa da 2 a 1, la prima istruzione che stalla è quella con il CDB ripetuto per prima, ovvero la

|  |
| --- |
| l.d f3,v3(r1) |

Della 2^ iterazione (in quanto abbiamo già la

|  |
| --- |
| div.d f4,f1,f2 |

Che occupa la posizione 12 del CDB).