**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 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| daddui r2,r0,100 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| loop: l.d f1,v1(r1) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| l.d f2,v2(r1) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| div.d f4,f1,f2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| l.d f3,v3(r1) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| dadd.d f4,f4,f3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| s.d f4,v4(r1) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| mul.d f5,f4,f3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| s.d f5,v5(r1) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| daddui r1,r1,8 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| daddi r2,r2,-1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| bnez r2,loop |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Halt |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| TOTAL |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**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.

**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 | l.d f2,v2(r1) |  |  |  |  |  |
| 1 | div.d f4,f1,f2 |  |  |  |  |  |
| 1 | l.d f3,v3(r1) |  |  |  |  |  |
| 1 | dadd.d f4,f4,f3 |  |  |  |  |  |
| 1 | s.d f4,v4(r1) |  |  |  |  |  |
| 1 | mul.d f5,f4,f3 |  |  |  |  |  |
| 1 | s.d f5,v5(r1) |  |  |  |  |  |
| 1 | daddui r1,r1,8 |  |  |  |  |  |
| 1 | daddi r2,r2,-1 |  |  |  |  |  |
| 1 | bnez r2,loop |  |  |  |  |  |
| 2 | l.d f1,v1(r1) |  |  |  |  |  |
| 2 | l.d f2,v2(r1) |  |  |  |  |  |
| 2 | div.d f4,f1,f2 |  |  |  |  |  |
| 2 | l.d f3,v3(r1) |  |  |  |  |  |
| 2 | dadd.d f4,f4,f3 |  |  |  |  |  |
| 2 | s.d f4,v4(r1) |  |  |  |  |  |
| 2 | mul.d f5,f4,f3 |  |  |  |  |  |
| 2 | s.d f5,v5(r1) |  |  |  |  |  |
| 2 | daddui r1,r1,8 |  |  |  |  |  |
| 2 | daddi r2,r2,-1 |  |  |  |  |  |
| 2 | bnez r2,loop |  |  |  |  |  |
| 3 | l.d f1,v1(r1) |  |  |  |  |  |
| 3 | l.d f2,v2(r1) |  |  |  |  |  |
| 3 | div.d f4,f1,f2 |  |  |  |  |  |
| 3 | l.d f3,v3(r1) |  |  |  |  |  |
| 3 | dadd.d f4,f4,f3 |  |  |  |  |  |
| 3 | s.d f4,v4(r1) |  |  |  |  |  |
| 3 | mul.d f5,f4,f3 |  |  |  |  |  |
| 3 | s.d f5,v5(r1) |  |  |  |  |  |
| 3 | daddui r1,r1,8 |  |  |  |  |  |
| 3 | daddi r2,r2,-1 |  |  |  |  |  |
| 3 | bnez r2,loop |  |  |  |  |  |

**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.