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 2 stages   + FP divider unit: not pipelined unit that requires 8 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++) {

; v5[i] = (v1[i]/v2[i]\*v3[i]) + v3[i]/v4[i];

;}

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| .data |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | Clock  cycles |
| V1: .double “100 values” |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| V2: .double “100 values” |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| V3: .double “100 values”  …  V5: .double “100 zeros” |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| V4: .double “100 values” |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| V5: .double “100 values” |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| .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 f3,v3(r1) |  |  | F | D | E | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |
| l.d f4,v4(r1) |  |  |  | F | D | E | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |
| div.d f5,f3,f4 |  |  |  |  | F | D | s | : | : | : | : | : | : | : | : | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 9 |
| l.d f1,v1(r1) |  |  |  |  |  | F | s | D | E | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 0 |
| l.d f2,v2(r1) |  |  |  |  |  |  |  | F | D | E | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 0 |
| mul.d f6,f2,f3 |  |  |  |  |  |  |  |  | F | D | s | X | X | X | X | X | X | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 2 |
| div.d f6,f1,f6 |  |  |  |  |  |  |  |  |  | F | s | D | s | s | s | s | s | : | : | : | : | : | : | : | : | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 8 |
| add.d f5,f6,f5 |  |  |  |  |  |  |  |  |  |  |  | F | s | s | s | s | s | D | s | s | s | s | s | s | s | A | A | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 2 |
| s.d f5,v5(r1) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | s | s | s | s | s | s | s | D | E | s | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |
| daddui r1,r1,8 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | D | s | E | M | W |  |  |  |  |  |  |  |  |  |  |  |  | 1 |
| daddi r2,r2,-1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | s | D | E | M | W |  |  |  |  |  |  |  |  |  |  |  | 1 |
| bnez r2,loop |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | s | D | E | M | W |  |  |  |  |  |  |  |  |  | 2 |
| Halt |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | - | - | - | - |  |  |  |  |  |  |  |  | 1 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Total |  |  |  |  | 6+29\*100 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 2906 |

**Domanda 3**

Considerando il programma precedente, quale sarebbe il tempo di esecuzione del programma se il processore avesse abilitato il Branch Delay slot? motivare la risposta.

35 (=29+6) perché con il Branch Delay Slot viene sempre eseguita l’istruzione immediatamente successiva alla branch ossia la halt che determina la fine dell’esecuzione del programma.

**Domanda 4**

Considerando il programma precedente, quali sono le copie di istruzioni che beneficiano principalmente dell’architettura Harvard del processore e perché? motivare la risposta.

Nell’architettura Harvard ci sono due memorie separate per istruzioni e dati

**Domanda 5**

Considerando il programma precedente, e in particolare la copia di istruzioni:

l.d f2,v2(r1)

mul.d f6,f2,f3

come viene attivato e qual è il cammino di forwarding che partecipa alla loro esecuzione? motivare la risposta.

**Domanda 6**

Considerando il programma precedente e l’architettura del processore superscalare descritto in seguito; completare la tabella relativa alle prime 2 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:
    1. 1 Memory address 1 clock cycle
    2. 1 Integer ALU 1 clock cycle
    3. 1 Jump unit 1 clock cycle
    4. 1 FP multiplier unit, which is pipelined: 6 stages
    5. 1 FP divider unit, which is not pipelined: 8 clock cycles
    6. 1 FP Arithmetic unit, which is pipelined: 2 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 f3,v3(r1) | 1 | 2m | 3 | 4 | 5 |
| 1 | l.d f4,v4(r1) | 1 | 3m | 4 | 5 | 6 |
| 1 | div.d f5,f3,f4 | 2 | 6d | - | 14 | 15 |
| 1 | l.d f1,v1(r1) | 2 | 4m | 5 | 6 | 15 |
| 1 | l.d f2,v2(r1) | 3 | 5m | 6 | 7 | 16 |
| 1 | mul.d f6,f2,f3 | 3 | 8x | - | 14 | 16 |
| 1 | div.d f6,f1,f6 | 4 | 22d | - | 30 | 31 |
| 1 | add.d f5,f6,f5 | 4 | 31a | - | 33 | 34 |
| 1 | s.d f5,v5(r1) | 5 | 6m | - | - | 34 |
| 1 | daddui r1,r1,8 | 5 | 6i | - | 7 | 35 |
| 1 | daddi r2,r2,-1 | 6 | 7i | - | 8 | 35 |
| 1 | bnez r2,loop | 7 | 9j | - | - | 36 |
| 2 | l.d f3,v3(r1) | 8 | 9m | 10 | 11 | 36 |
| 2 | l.d f4,v4(r1) | 8 | 10m | 11 | 12 | 37 |
| 2 | div.d f5,f3,f4 | 9 | 14d | - | 22 | 37 |
| 2 | l.d f1,v1(r1) | 9 | 11m | 12 | 13 | 38 |
| 2 | l.d f2,v2(r1) | 10 | 12m | 13 | 15 | 38 |
| 2 | mul.d f6,f2,f3 | 10 | 16x | - | 22 | 39 |
| 2 | div.d f6,f1,f6 | 11 | 30d | - | 38 | 39 |
| 2 | add.d f5,f6,f5 | 11 | 39a | - | 41 | 42 |
| 2 | s.d f5,v5(r1) | 12 | 13m | - | - | 42 |
| 2 | daddui r1,r1,8 | 12 | 13i | - | 15 | 43 |
| 2 | daddi r2,r2,-1 | 13 | 14i | - | 16 | 43 |
| 2 | bnez r2,loop | 14 | 17j | - | - | 44 |

**Domanda 7**

Considerando il segmento di codice presentato nella tabella precedente, se assumiamo che il ROB ha una dimensione di 16 elementi, qual è la prima istruzione che dovrebbe stallare durante la esecuzione del programma? motivare la risposta.

**Domanda 8**

Considerando il segmento di codice presentato nella tabella precedente, se assumiamo che si vorrebbero migliorare le prestazioni del programma duplicando una delle seguenti unità funzionali:

* FP multiplier unit
* FP divider unit
* FP Arithmetic unit

Quale permetterebbe di ottenere un miglioramento maggiore nelle prestazioni del processore? motivare la risposta.