**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 7 stages | * + FP arithmetic unit: pipelined 3 stages   + FP divider unit: not pipelined unit that requires 7 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];

; v5[i] = (v3[i]/v1[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 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 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 8 |
| s.d f4,v4(r1) |  |  |  |  |  | F | s | D | E | s | s | s | s | s | S | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |
| l.d f3,v3(r1) |  |  |  |  |  |  |  | F | D | s | s | s | s | s | s | E | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |
| div.d f6,f3,f1 |  |  |  |  |  |  |  |  | F | s | s | s | s | s | s | D | s | / | / | / | / | / | / | / | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 8 |
| add.d f7,f6,f4 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | s | D | s | s | s | s | s | s | + | + | + | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 3 |
| s.d f7,v5(r1) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | s | s | s | s | s | s | D | E | s | S | M | W |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |
| daddi r2,r2,-1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | D | s | s | E | M | W |  |  |  |  |  |  |  |  |  |  |  |  | 1 |
| daddui r1,r1,8 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | s | s | D | E | M | W |  |  |  |  |  |  |  |  |  |  |  | 1 |
| bnez r2,loop |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | D | E | M | W |  |  |  |  |  |  |  |  |  |  | 1 |
| Halt |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | F | - | - | - | - |  |  |  |  |  |  |  |  |  | 1 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Total |  |  |  |  | 6+100\*28 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 2806 |

**Domanda 2**

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

div.d f4,f1,f2

…

div.d f6,f3,f1

che tipo di hazard potrebbe crearsi e come viene risolto? motivare la risposta.

Si potrebbe creare uno STRUCTURAL HAZARD (ovvero stessa unità, non pipelined, usata da 2 istruzioni contemporaneamente)

**Domanda 2**

Si consideri una BHT di 1K elementi con contatori di saturazione a 2 bit. Usando il frammento di codice proposto, si calcoli lo stato finale della BHT al completamento dell’esecuzione del programma proposto. Si calcoli inoltre, la *misprediction rate* complessiva. Lo stato iniziale della BPU è indicato nella tabella.

Assunzioni generali:

* R10 è il registro di controllo del loop ed è inizializzato a 100
* R3 e R7 sono registro di riferimento con valore 5 (R3, R7 = 5)
* R13 e R17 sono i registri in ingresso
  + R13 riceve un valore sempre maggiore a 5 (R13 > 5)
  + R17 riceve un valore una volta minore a 5, e la volta successiva maggiore a 5: (R17 < 5, R17 > 5) [alterno]
* Gli altri registri sono inizializzati a 0 fuori dal ciclo.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Address** | **Instruction** | | **BHT (2-bit)** | **Prediction** | **misP. counter** |
| **0x0000** | **L0:** | **…** | 2 | T |  |
| **…** | ; | ***Reading input values*** | 2 | T |  |
| **0x0010** |  | **SLT R1, R3, R13**  **[sempre 1, R3 < R13]** | 2 | T |  |
| **0x0014** |  | **BEQZ R1, L1**  **[sempre not taken]** | 2-1-0 | T-NT-NT… | 1-…-1 |
| **0x0018** |  | **DADDI R11, R0, 10**  [R11 = 10] | 2 | T |  |
| **0x001C** | **L1:** | **SLT R4, R7, R17**  **[0, 1, 0, 1, …]** | 2 | T |  |
| **0x0020** |  | **BEQZ R4, L2**  **[taken, not taken,…]** | 2-3-2-3-2-… | T-T-NT-T-NT-… | 0-1-1-2-2…-50 |
| **0x0024** |  | **DADD R12, R12, R4**  [r12 = 0, 1, 1, 2, 2, 3, 3, …]  [servono 20 iterazioni per arrivare a r12 = 10] | 2 | T |  |
| **0x0028** | **L2:** | **SLT R5, R4, R1**  **[1, 0, 1, 0, …]** | 2 | T |  |
| **0x002C** |  | **BEQZ R5, L3**  **[not taken, taken,…]** | 2-1-2-1-2-… | T-NT-T-NT-T | 1-2-3-…-100 |
| **0x0030** |  | **DADD R14, R0, R5** | 2 | T |  |
| **0x0038** | **L3:** | **SLT R6, R12, R11**  **[per 20 iterazioni 1, poi sempre 0]** | 2 | T |  |
| **0x003C** |  | **BEQZ R6, L4**  **[not taken (x20), taken (x80)]** | 2-1-0-0-…-1-2-3-3… | T-NT-NT-…-T-T-T | 1-1-…-2-3-…-3 |
| **0x0040** |  | **DADDI R15, R0, 10** | 2 | T |  |
| **0x0044** | **L4:** |  | 2 | T |  |
| **0x0048** |  | **DADDI R10, R10, #-1** | 2 | T |  |
| **0x004C** |  | **BNEZ R10, L0**  **[taken (x99), not taken]** | 2-3-…-3-2 | T-T-…-NT | 0-0-0-…-1 |
| **0x0050** |  |  |  |  |  |
|  |  |  |  |  |  |

Note:

SLT R1,R2,R3 ;IF (R2 < R3) R1 🡨 1

;ELSE R1 🡨 0

Misprediction rate = (1+50+100+3+1)/5\*100=0.31=31%

**Domanda 4**

Considerando il programma precedente, che vantaggio/svantaggio potrebbe avere se la BHT includesse contatori di saturazione a 3-bit. Motivare la risposta.

Il primo salto non cambia (satura solo più lentamente, ma da 4 passo a 3 e quindi solo 1 misprediction)

Il secondo nemmeno (4-5-4-5-4-5- 🡪 50 MP)

Il terzo nemmeno (4-3-4-3-4…🡪 100 MP)

Il quarto invece cambia (4-3-2-1-0-…-0-1-2-3-4-5-6-7…🡪 1 MP + 4 MP al posto di solo 2MP nella seconda parte)

Il quinto non cambia (1 MP)