**Domanda 1**

Si completi il programma per un processore di tipo MIPS riportato in tabella in modo che rispecchi i valori dei iniziali e finali dei predittori della Branch Prediction Unit (BPU) basata su Branch History Table (BHT) riportata in tabella.

Nel caso specifico, si consideri una BPU di tipo BHT con contatori a saturazione di 2-bit e 1K elementi. Si assuma che lo stato iniziale della BHT è indicato nella colonna **BHT INIT** e lo stato finale dopo l’esecuzione del programma nella colonna **BHT END**.

Il programma deve essere in grado di portare la BHT dalla colonna di inizio allo stato finale indicato, prima di arrivare all’istruzione HALT.

Assunzioni Generali:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Address** | **Instruction** | | **BHT INIT** | **BHT END** |
| **0x0000** |  |  | **--** | -- |
| **0x0004** |  |  | **--** | **--** |
| **0x0008** |  |  | **--** | **--** |
| **0x000C** |  |  | **--** | **--** |
| **0x0010** | Lab0: |  | **--** | **--** |
| **0x0014** |  | B… … Lab2 | **0** | **3** |
| 0x0018 |  |  | -- | -- |
|  |  |  |  |  |
| **0x0018** | Lab1: |  | **--** | **--** |
| **0x001C** |  | B… … Lab2 | **0** | **1** |
| **0x0020** |  |  | **--** | **--** |
| **0x0024** | Lab2: |  | **--** | **--** |
| **0x0028** |  |  | **--** | **--** |
| **0x002C** |  |  | **--** | **--** |
| **0x0030** |  | B… … Lab3 | **0** | **2** |
| **0x0038** |  |  | **--** | **--** |
| **0x003c** | Lab3: |  | **--** | **--** |
| **0x0040** |  | B… … Lab4 | **0** | **1** |
| **0x0044** |  |  | **--** | **--** |
| **0x0048** | Lab4: |  | **--** | **--** |
| **0x004c** |  |  | **--** | **--** |
| **0x0050** |  |  | **--** | **--** |
| **0x0054** |  | B… … Lab0 | **0** | **2** |
| **0x0058** |  | HALT | **--** | **--** |

**Domanda 2**

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

* + Unità intera ALU: 1 clock cycle
  + Data memory: 1 clock cycle
  + FP multiplier unit: pipelined 8 stages
  + FP arithmetic unit: pipelined 4 stages
  + FP divider unit: not pipelined unit that requires 10 clock cycles
  + branch delay slot: 1 clock cycle
  + data forwarding is enabled
  + it is possible to complete instruction EXE stage in an out-of-order fashion,

e usando il seguente fragment di codice:

; \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* MIPS64 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

; v7[i] = (v1[i]\*v2[i]+v3[i]\*v4[i]);

;}

Si ottimizzi il programma per migliorare le sue prestazioni cercando di eliminare la maggior degli stalli dovuti ai hazard presenti nel programma. Si usino le seguenti tecniche di ottimizzazione: instruction rescheduling, register renaming, loop unrolling (di massimo 2 iterazioni) e abilitazione del branch delay slot.

|  |  |  |  |
| --- | --- | --- | --- |
| V1- V2-V3-V4:  .double “100 values” | Clock cycles - Originali | Versione codice ottimizzato | Clock cycles – Ottimizzati |
| main: daddui r1,r0,0  daddui r2,r0,100  loop: l.d f1,v1(r1) |  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| l.d f2,v2(r1) |  |  |  |
| mul.d f7,f1,f2 |  |  |  |
| l.d f3,v3(r1) |  |  |  |
| l.d f4,v4(r1) |  |  |  |
| mul.d f8,f3,f4 |  |  |  |
| add.d f10,f8,f7 |  |  |  |
| s.d f10,v7(r1) |  |  |  |
| daddui r1,r1,8 |  |  |  |
| daddi r2,r2,-1 |  |  |  |
| bnez r2,loop |  |  |  |
| halt |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| Total |  |  |  |