**Test 8 curs CN1**

**SUBIECT MIPS IN BANDA DE ASAMBLARE**

In urmatorul tabel, sunt prezentate instructiunile ce se vor executa, plus semnificatia lor, avand in vedere ca BEQ este True o data si apoi Not True:

|  |  |  |
| --- | --- | --- |
| Nr. instr. | Instructiune | Semnificatie |
| 1 | ADD $s7, $s6, $s5 | $s7 = $s6 + $s5 |
| 2 | LW $s1, 0($s1) | $s1 = MEM[$s1 + 0] |
| 3 | AND $s1, $s1, $s2 | $s1 = $s1 & $s2 |
| 4 | ET1:  LW $s2, 0($s1) | $s2 = MEM[$s1 + 0] |
| 5 | BEQ $s2, $s0, ET1 | if($s2==$s0) go to ET1  (presupunem TRUE) |
| 6 | ET1:  LW $s2, 0($s1) | $s2 = MEM[$s1 + 0] |
| 7 | BEQ $s2, $s0, ET1 | if($s2==$s0) go to ET1  (presupunem FALSE) |
| 8 | OR $s2, $s2, $s3 | $s2 = $s2 | $s3 |
| 9 | SW $s2, 0($s3) | MEM[$s3 + 0] = $s2 |

1. **Gasiti toate dependentele din aceasta secventa de instructiuni specificand tipul hazardurilor gasite.**

In seria de instructiuni data, exista urmatoarele dependinte si pot aparea urmatoarele hazarde:

* intre instructiunea 2 si instructiunea 3 (prin s1):

Instructiunea 3 depinde de valoarea registrului s1, in care se scrie in instructiunea 2. Instructiunea 2 LW scrie in registrul $s1 la pasul 5 din pipeline, dar instructiunea 3 AND are nevoie de valoarea registrului la pasul 2.

Deci poate apare un hazard RAW (instructiunea 3 incearca sa citeasca din $s1 inainte ca instructiunea 1 sa scrie in $s1).

In cazul acestor instructiuni, poate aparea si un hazard WAR (daca instructiunea 3 s-ar termina inainte de 2, atunci 3 ar scrie in $s1 inainte ca 2 sa citeasca valoarea de la adresa din $s1).

* intre instructiunile 2,3 si instructiunile 4 (prin s1):

Instructiunea 4 citeste valoarea de la adresa indicata de $s1, iar cele 2 instructiuni de dinainte scriu in acest registru. Poate aparea un hazard RAW (instructiunea 4 sa incerce sa citeasca din $s1 inainte ca instructiunile 2 si 3 sa fi scris in $s1).

* intre instructiunea 3 si instructiunea 4 (prin s2):

Instructiunea 4 scrie in $s2 dupa ce instructiunea 3 citeste din $s2. Poate aparea un hazard WAR (instructiunea 4 LW sa scrie in $s2 inainte ca 3 AND sa citeasca din $s2).

* intre instructiunea 4 si instructiunea 5 (prin s2):

Instructiunea 5 BEQ citeste valoarea din $s2, iar instructiunea 4 LW scrie in $s2. Deci poate aparea un hazard RAW.

* intre instructiunea 5 si instructiunea 6 (prin s2):

Poate aparea un hazard WAR (instructiunea 6 LW sa scrie in $s2 inainte ca 5 BEQ sa citeasca din $s2.

* intre instructiunea 4 si instructiunea 6 (prin s2):

Poate aparea un hazard WAW (ambele instructiuni scriu in registrul $s2 si se poate intampla ca instructiunea 6 sa se termine inainte instructiunii 4).

* intre instructiunea 6 si instructiunea 7 (prin s2):

Poate aparea un hazard RAW (acelasi caz ca pt instructiunile 4 si 5).

* intre instructiunea 6 si instructiunea 8 (prin s2):

Poate aparea un hazard RAW (instructiunea 8 OR sa citeasca din $s2 inainte ca instructiunea 6 LW sa scrie in $s2).

Poate aparea si un hazard WAW (instructiune 8 OR sa scrie in $s2 inainte ca instructiunea 6 LW sa scrie tot in $s2).

* intre instructiunea 8 si instructiune 9 (prin s2):

Poate aparea un hazard RAW (instructiunea 9 SW sa citeasca din $s2 inainte ca instructiunea 8 OR sa scrie in $s2).

Pe langa aceste hazarde, mai pot aparea:

* hazarde de control, datorate folosirii instructiunii BEQ:

Nu avem garantia ca in pipeline dupa instructiunea BEQ este adusa instructiunea corecta (asta depinde de rezultatul lui BEQ).

* hazarde structurale:

Acestea pot aparea atunci cand doua sau mai multe instructiuni ce se afla in pipeline folosesc aceeasi resursa (de exemplu instructiunea 1 ADD si instructiunea 3 AND, ambele folosesc unitatea aritmetica logica).

1. **Daca nu exista hardware de Forwarding sau pentru detectie hazarduri, sa se corecteze executia doar prin inserare de NOP-uri pentru eliminarea hazardurilor de la punctul a).**

ADD $s7, $s6, $s5

LW $s1, 0($s1)

NOP

NOP

AND $s1, $s1, $s2

NOP

NOP

ET1: LW $s2, 0($s1)

NOP

NOP

BEQ $s2, $s0, ET1

NOP

OR $s2, $s2, $s3

NOP

SW $s2, 0($s3)

1. **Repetati punctul b) dar utilizand NOP-uri doar cand un hazard nu poate fi evitat prin schimbarea sau rearanjarea acelor instructiuni.**

LW $s1, 0($s1)

ADD $s7, $s6, $s5

NOP

AND $s1, $s1, $s2

NOP

NOP

ET1: LW $s2, 0($s1)

NOP

NOP

BEQ $s2, $s0, ET1

NOP

OR $s2, $s2, $s3

NOP

SW $s2, 0($s3)

1. **Sa se repete punctul b) utilizand STALL-uri in loc de NOP-uri.**

Am notat cu -> Stall.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instr. | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| ADD $s7, $s6, $s5 | IF | ID | EX | ME | WB |  |  |  |  |  |  |  |  |  |  |  |  |  |
| LW $s1, 0($s1) |  | IF | ID | EX | ME | WB |  |  |  |  |  |  |  |  |  |  |  |  |
| AND $s1, $s1, $s2 |  |  | IF | ID | -> | EX | ME | WB |  |  |  |  |  |  |  |  |  |  |
| ET1: LW $s2, 0($s1) |  |  |  | IF | -> | ID | -> | EX | ME | WB |  |  |  |  |  |  |  |  |
| BEQ $s2, $s0, ET1 |  |  |  |  | -> | IF | -> | ID | -> | EX | ME | WB |  |  |  |  |  |  |
| ET1: LW $s2, 0($s1) |  |  |  |  |  | -> | -> | -> | -> | IF | ID | EX | ME | WB |  |  |  |  |
| BEQ $s2, $s0, ET1 |  |  |  |  |  |  | -> | -> | -> | IF | ID | -> | -> | EX | ME | WB |  |  |
| OR $s2, $s2, $s3 |  |  |  |  |  |  |  | -> | -> | -> | -> | -> | -> | IF | ID | EX | ME | WB |
| SW $s2, 0($s3) |  |  |  |  |  |  |  |  | -> | -> | -> | -> | -> | IF | ID | -> | -> | EX |

... ME WB

1. **Daca exista hardware de Forwarding sau pentru detectie hazarduri, sa se corecteze executia precizand cand si cum are loc Forwarding.**

Am ilustrat Forwarding prin .

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instr. | 0 | 1 | 2 | 3 | 4 | 6 | 8 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| ADD $s7, $s6, $s5 | IF | ID | EX | ME | WB |  |  |  |  |  |  |  |  |  |  |
| LW $s1, 0($s1) |  | IF | ID | EX | ME | WB |  |  |  |  |  |  |  |  |  |
| AND $s1, $s1, $s2 |  |  | IF | ID | EX | ME | WB |  |  |  |  |  |  |  |  |
| ET1: LW $s2, 0($s1) |  |  |  | IF | ID | EX | ME | WB |  |  |  |  |  |  |  |
| BEQ $s2, $s0, ET1 |  |  |  |  | IF | ID | EX | ME | WB |  |  |  |  |  |  |
| ET1: LW $s2, 0($s1) |  |  |  |  |  | -> | IF | ID | EX | ME | WB |  |  |  |  |
| BEQ $s2, $s0, ET1 |  |  |  |  |  |  | IF | ID | -> | EX | ME | WB |  |  |  |
| OR $s2, $s2, $s3 |  |  |  |  |  |  |  | -> | -> | IF | ID | EX | ME | WB |  |
| SW $s2, 0($s3) |  |  |  |  |  |  |  |  | -> | -> | IF | ID | EX | ME | WB |

1. **Daca se intrerupe legatura ALUOutM de la intrarea multiplexorului de Forwarding legat la RD2 din schema procesorului MIPS, ce se intampla cu executia de la punctul e)?**

Daca se intrerupe acea legatura, nu mai putem face Forwarding direct din registrul de dupa etapa EX. O metoda de corectia in cazul unui hazard este adaugarea de Stall-uri astfel incat EX urmator sa primeasca prin Forwarding de la registrul de dupa ME, exemplu:

LW $s1, 0($s1) IF ID EX ME WB

AND $s1, $s1, $s2 IF ID -> EX ME WB

SUBIECT MEMORIE CACHE

Memorie cache de capacitate 16 cuvinte si lungimea blocului de 2 cuvinte.

Secventa de acces la memorie cu adresa blocului in ordinea LD 0, LD 8, LD 16, ST 0, LD 12, LD 8, ST 20, LD 8, ST 16, LD 22, LD 24, LD 26, LD 30, LD 32, ST 16, ST 0, LD14.

1. **Daca memoria CACHE are corespondenta directa, sa se calculeze frecventa de esec (Miss Rate) afisand si harta memoriei CACHE si precizand tipul de Miss de acces (de prim acces sau COMPULSORY, de conflict de bloc sau CONFLICT sau de capacitate sau CAPACITY).**

Capacitate = 16 cuvinte.

Lungimea blocului = 2 cuvinte.

* Nr. de blocuri = 8.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Secventa | HIT sau MISS | Setul 0 | Setul 1 | Setul 2 | Setul 3 | Setul 4 | Setul 5 | Setul 6 | Setul 7 |
| LD 0 | MISS (COMPULSORY) | Mem(0) |  |  |  |  |  |  |  |
| LD 8 | MISS (COMPULSORY) | Mem(8) |  |  |  |  |  |  |  |
| LD 16 | MISS (COMPULSORY) | Mem(16) |  |  |  |  |  |  |  |
| ST 0 | MISS (CONFLICT) | Mem(0) |  |  |  |  |  |  |  |
| LD 12 | MISS (COMPULSORY) | Mem(0) |  |  |  | Mem(12) |  |  |  |
| LD 8 | MISS (CONFLICT) | Mem(8) |  |  |  | Mem(12) |  |  |  |
| ST 20 | MISS (COMPULSORY) | Mem(8) |  |  |  | Mem(20) |  |  |  |
| LD 8 | HIT | Mem(8) |  |  |  | Mem(20) |  |  |  |
| ST 16 | MISS (CONFLICT) | Mem(16) |  |  |  | Mem(20) |  |  |  |
| LD 22 | MISS (COMPULSORY) | Mem(16) |  |  |  | Mem(20) |  | Mem(22) |  |
| LD 24 | MISS (COMPULSORY) | Mem(24) |  |  |  | Mem(20) |  | Mem(22) |  |
| LD 26 | MISS (COMPULSORY) | Mem(24) |  | Mem(26) |  | Mem(20) |  | Mem(22) |  |
| LD 30 | MISS (COMPULSORY) | Mem(24) |  | Mem(26) |  | Mem(20) |  | Mem(30) |  |
| LD 32 | MISS (COMPULSORY) | Mem(32) |  | Mem(26) |  | Mem(20) |  | Mem(30) |  |
| ST 16 | MISS (CONFLICT) | Mem(16) |  | Mem(26) |  | Mem(20) |  | Mem(30) |  |
| ST 0 | MISS (CONFLICT) | Mem(0) |  | Mem(26) |  | Mem(20) |  | Mem(30) |  |
| LD 14 | MISS (COMPULSORY) | Mem(0) |  | Mem(26) |  | Mem(20) |  | Mem(14) |  |

COMPULSORY\_MISSES = 4. |

CONFLICT\_MISSES = 12. | => MISSES = 16.

HITS = 1.

Miss Rate = 16/17 = 94,11%.

1. **Daca memoria CACHE are corespondenta set-asociativa pe (1<N<numar blocuri) cai, unde N este gradul de asociativitate si ca algoritmul de inlocuire in caz de Miss al blocului din memoria CACHE este de tip LRU, sa se calculeze frecventa de esec pentru toate cazurile de asociativitate posibile afisand, de asemenea si harta memoriei CACHE si precizand tipul de Miss de acces.**

Capacitate C = 16.

Dimensiunea unui bloc b = 2.

Numarul de blocuri B = C/b = 8.

Gradul de asociativitate N poate avea valorile 2 si 4.

Pentru N = 2 => numarul seturilor S = B/N = 4. Deci avem 4 seturi cu cate 2 blocuri fiecare.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Secventa | HT sau MISS | Setul 0 | | Setul 1 | | Setul 2 | | Setul 3 | |
| LD 0 | MISS (COMPULSORY) | Mem(0) |  |  |  |  |  |  |  |
| LD 8 | MISS (COMPULSORY) | Mem(0) | Mem(8) |  |  |  |  |  |  |
| LD 16 | MISS (COMPULSORY) | Mem(16) | Mem(8) |  |  |  |  |  |  |
| ST 0 | MISS (CONFLICT) | Mem(16) | Mem(0) |  |  |  |  |  |  |
| LD 12 | MISS (COMPULSORY) | Mem(12) | Mem(0) |  |  |  |  |  |  |
| LD 8 | MISS (CONFLICT) | Mem(12) | Mem(8) |  |  |  |  |  |  |
| ST 20 | MISS (COMPULSORY) | Mem(20) | Mem(8) |  |  |  |  |  |  |
| LD 8 | HIT | Mem(20) | Mem(8) |  |  |  |  |  |  |
| ST 16 | MISS (CONFLICT) | Mem(16) | Mem(8) |  |  |  |  |  |  |
| LD 22 | MISS (COMPULSORY) | Mem(16) | Mem(8) |  |  | Mem(22) |  |  |  |
| LD 24 | MISS (COMPULSORY) | Mem(16) | Mem(24) |  |  | Mem(22) |  |  |  |
| LD 26 | MISS (COMPULSORY) | Mem(16) | Mem(24) |  |  | Mem(22) | Mem(26) |  |  |
| LD 30 | MISS (COMPULSORY) | Mem(16) | Mem(24) |  |  | Mem(30) | Mem(26) |  |  |
| LD 32 | MISS (COMPULSORY) | Mem(32) | Mem(24) |  |  | Mem(30) | Mem(26) |  |  |
| ST 16 | MISS (CONFLICT) | Mem(32) | Mem(16) |  |  | Mem(30) | Mem(26) |  |  |
| ST 0 | MISS (CONFLICT) | Mem(0) | Mem(16) |  |  | Mem(30) | Mem(26) |  |  |
| LD 14 | MISS (COMPULSORY) | Mem(0) | Mem(16) |  |  | Mem(30) | Mem(14) |  |  |

MISSES = 17, HITS = 1 => Miss Rate = 16/17 = 94,11%.

Pentru N = 4 => numarul seturilor S = B/N = 2. Deci vom avea 2 seturi cu cate 4 blocuri fiecare.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Secventa | HIT sau MISS | Setul 0 | | | | Setul 1 | | | |
| LD 0 | MISS (COMPULSORY) | Mem(0) |  |  |  |  |  |  |  |
| LD 8 | MISS (COMPULSORY) | Mem(0) | Mem(8) |  |  |  |  |  |  |
| LD 16 | MISS (COMPULSORY) | Mem(0) | Mem(8) | Mem(16) |  |  |  |  |  |
| ST 0 | HIT | Mem(0) | Mem(8) | Mem(16) |  |  |  |  |  |
| LD 12 | MISS (COMPULSORY) | Mem(0) | Mem(8) | Mem(16) | Mem(12) |  |  |  |  |
| LD 8 | HIT | Mem(0) | Mem(8) | Mem(16) | Mem(12) |  |  |  |  |
| ST 20 | MISS (COMPULSORY) | Mem(0) | Mem(8) | Mem(20) | Mem(12) |  |  |  |  |
| LD 8 | HIT | Mem(0) | Mem(8) | Mem(20) | Mem(12) |  |  |  |  |
| ST 16 | MISS (CONFLICT) | Mem(16) | Mem(8) | Mem(20) | Mem(12) |  |  |  |  |
| LD 22 | MISS (COMPULSORY) | Mem(16) | Mem(8) | Mem(20) | Mem(22) |  |  |  |  |
| LD 24 | MISS (COMPULSORY) | Mem(16) | Mem(8) | Mem(24) | Mem(22) |  |  |  |  |
| LD 26 | MISS (COMPULSORY) | Mem(16) | Mem(26) | Mem(24) | Mem(22) |  |  |  |  |
| LD 30 | MISS (COMPULSORY) | Mem(30) | Mem(26) | Mem(24) | Mem(22) |  |  |  |  |
| LD 32 | MISS (COMPULSORY) | Mem(30) | Mem(26) | Mem(24) | Mem(32) |  |  |  |  |
| ST 16 | MISS (CONFLICT) | Mem(30) | Mem(26) | Mem(16) | Mem(32) |  |  |  |  |
| ST 0 | MISS (CONFLICT) | Mem(30) | Mem(0) | Mem(16) | Mem(32) |  |  |  |  |
| LD 14 | MISS (COMPULSORY) | Mem(14) | Mem(0) | Mem(16) | Mem(32) |  |  |  |  |

MISSES = 14, HITS = 3 => Miss Rate = 14/17 = 82,35%.

1. **Daca memoria CACHE are corespondenta complet asociativa si ca algoritmul de inlocuire este LRU, sa se calculeze frecventa de esec afisand si harta memoriei CACHE si precizand tipul de Miss de acces.**

C = 16, b = 2 => B = 8.

N = 8, S = 1.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Secventa | HIT sau MISS | Blocul 0 | Blocul 1 | Blocul 2 | Blocul 3 | Blocul 4 | Blocul 5 | Blocul 6 | Blocul 7 |
| LD 0 | MISS (COMPULSORY) | Mem(0) |  |  |  |  |  |  |  |
| LD 8 | MISS (COMPULSORY) | Mem(0) | Mem(8) |  |  |  |  |  |  |
| LD 16 | MISS (COMPULSORY) | Mem(0) | Mem(8) | Mem(16) |  |  |  |  |  |
| ST 0 | HIT | Mem(0) | Mem(8) | Mem(16) |  |  |  |  |  |
| LD 12 | MISS (COMPULSORY) | Mem(0) | Mem(8) | Mem(16) | Mem(12) |  |  |  |  |
| LD 8 | HIT | Mem(0) | Mem(8) | Mem(16) | Mem(12) |  |  |  |  |
| ST 20 | MISS (COMPULSORY) | Mem(0) | Mem(8) | Mem(16) | Mem(12) | Mem(20) |  |  |  |
| LD 8 | HIT | Mem(0) | Mem(8) | Mem(16) | Mem(12) | Mem(20) |  |  |  |
| ST 16 | HIT | Mem(0) | Mem(8) | Mem(16) | Mem(12) | Mem(20) |  |  |  |
| LD 22 | MISS (COMPULSORY) | Mem(0) | Mem(8) | Mem(16) | Mem(12) | Mem(20) | Mem(22) |  |  |
| LD 24 | MISS (COMPULSORY) | Mem(0) | Mem(8) | Mem(16) | Mem(12) | Mem(20) | Mem(22) | Mem(24) |  |
| LD 26 | MISS (COMPULSORY) | Mem(0) | Mem(8) | Mem(16) | Mem(12) | Mem(20) | Mem(22) | Mem(24) | Mem(26) |
| LD 30 | MISS (COMPULSORY) | Mem(30) | Mem(8) | Mem(16) | Mem(12) | Mem(20) | Mem(22) | Mem(24) | Mem(26) |
| LD 32 | MISS (COMPULSORY) | Mem(30) | Mem(8) | Mem(16) | Mem(32) | Mem(20) | Mem(22) | Mem(24) | Mem(26) |
| ST 16 | HIT | Mem(30) | Mem(8) | Mem(16) | Mem(32) | Mem(20) | Mem(22) | Mem(24) | Mem(26) |
| ST 0 | MISS (CAPACITY) | Mem(30) | Mem(8) | Mem(16) | Mem(32) | Mem(0) | Mem(22) | Mem(24) | Mem(26) |
| LD 14 | MISS (COMPULSORY) | Mem(30) | Mem(14) | Mem(16) | Mem(32) | Mem(0) | Mem(22) | Mem(24) | Mem(26) |

MISSES = 12, HITS = 5 => Miss Rate = 12/17 = 70,58%.

1. **Daca se aplica o strategie de scriere de tip WRITE-THROUGH prin care se scrie in acelasi timp in memoria CACHE dar si in memoria principala MAIN, presupunand ca timpul de acces la memoria CACHE este de 1 ciclu de ceas, iar la memoria principala este de 100 de cicluri de ceas, sa se calculeze timpul mediu de acces AMAT la subsistemul de memorie CACHE-principala, pentru toate cazurile prevazute la punctele anterioare.**

AMAT = tcache + MRcache \* (tMM + MRMM(tVM)).

tcache = 1

tMM = 100

Cazul memorie CACHE cu corespondenta directa: MR = 94,11%.

AMAT = 1 + 94,11 / 100 \* 100 = 95,11.

Cazul memorie CACHE cu corespondenta set-asociativa N = 2: MR = 94,11%.

AMAT = 1 + 94,11 / 100 \* 100 = 95,11.

Cazul memorie CACHE cu corespondenta set-asociativa N = 4: MR = 82,35%.

AMAT = 1 + 82,35 / 100 \* 100 = 83,35.

Cazul memorie CACHE cu corespondenta complet asociativa: MR = 70,58%.

AMAT = 1 + 70,58 / 100 \* 100 = 71,58.

Cu cat MR scade, cu atat si AMAT scade, deci memoria este accesata mai rapid.

1. **Cum influenteaza Hit sau Miss la memoria CACHE accesele la memorie cerute de instructiunile din secventa de instructiuni de la subiectul de examen anterior?**

In secvente de instructiuni de la subiectul de examen anterior, se acceseaza memoria de 4 ori: pentru instructiunea LW $s1, 0($s1) se acceseaza memoria o data, intrustiunea LW $s2, 0($s1) va fi executata de doua ori si acceseaza memoria si instructiunea SW $s2, 0($s3) acceseaza si ea memoria.

Hit sau Miss la memoria CACHE afecteaza aceste accese la memorie. Cu cat avem mai multe Hit-uri si mai putine Miss-uri (deci cu cat Miss Rate este mai mic), cu atat timpul de acces este mai scurt si deci programul este mai rapid.