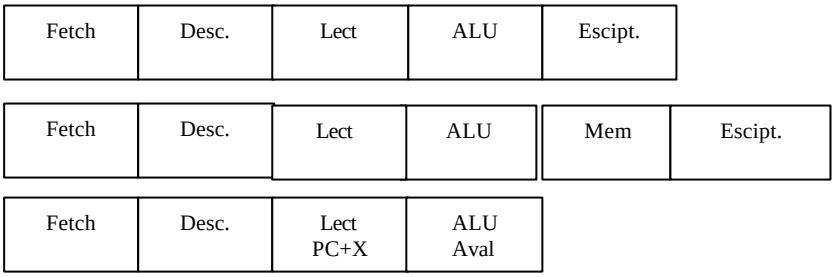
Arquitectures avançades de Computadors
(Enginyeria Tècnica en Informàtica de Sistemes)

Dia : 10 de juny de 2003
Aula : P-II 4B
Hora : 4 de la tarda

Observacions : La solució de l'examen sortirà el dimecres dia 7 a fotocòpies.de P-II
Les notes sortiran el 7.
Els dies 7 ,8 i 9 es podrà sol·licitar revisió per mail o amb una nota al casiller.
La revisió d'examen es farà a porta tancada.

PROBLEMA 1 (2 punts)
En un segmentat multicicle amb tres pipeline (veure les figures) , que utilitza un Delay-load i un Delay-barch s'executa el següent codi:



A on les instruccions: add, sub, mult utilitzen la primera pipeline, les instruccions load i store fan servir la segona i la bne fa servir la tercera.

```
1          load    r1 (r2)          ;
2          add     r4, r2,r3        ;
3          sub     r5, r1, r4       ;
4          add     r3, r2, r1       ;
5          store   r4, (r2)         ;
6          add     r5, r4, r2       ;
7          mult    r8, r5,r7        ;
8          bne     r8,salt          ;
9          sub     r1, r7, r8       ;
10 salt;   fi:
```

Suposant que el salt es produeix contesta:
a) Dibuixa l'execució amb el seu diagrama de temps utilitzant la tècnica de Fordwaring.
b) Enumera les dependència de les dades i els conflictes estructurals amb les solucions donades.
c) Rescriu el codi utilitzant la tècnica del Delay (nop).

PROBLEMA 2 (2,5 punt)

Tenim el següent codi:

```
1          MULT S1, S2, S3
2          MULT S1, S1, S3
3          SUB S2, S4, S5
4          CMP S6, S2 , 0
5          BNE S6, LL1
6          ADD S1, S5, S2
7          SUB S3, S4, S7
8          MULT S2, S4, S3
9     LL1: SUB S3, S2, S3
10         MULT S4, S3, S1
```

Suposa un superescalar de 2 vies amb 3 unitats funcionals: una sencera per operacions suma i resta, una sencera per operacions de salt i comparació, i una de coma flotant per operacions de producte i divisió.

Les unitats tenen les següents característiques:

UF	Estacions de reserva	Operacions	Latència d'entrada	Latència de sortida
Sencera	3	+,-	1	2
Coma Flotant	2	*,/	2	4
Sencera	2	cmp i salt	1	1

Les instruccions de salt fan una predicció estàtica de continuar en seqüència fins a avaluar la condició de salt. La màquina utilitza la tècnica del ***History Buffer*** en la recuperació de les execucions especulatives dels salts. En el cas concret del codi anterior el resultat del salt es saltar.

Contesta:

a) Si tenim Tomasulo com quedaria el processador després d'executar cadascun dels cicles indicats? Utilitza les taules. Suposa les etapes: F, D, X, W. Suposa que l'etapa de W és pot estalviar amb curt-circuits.

b) En quin estat es trobaran els registres abans de la fase d'execució del salt, i després de la recuperació. Justifica la resposta.

PROBLEMA 3 (3 punt)

En un sistema superescalar tenim el següent codi:

```
1          load r1,(r12)0
2          add r0,r1,r2
3          add r1,r2,r1
4          store r1,(r12)0
5          mov r8, var_S3
6          mov r4,0
7     LL2: cmp r5,r4,5
8          ble r5,LL5
9          b LL3
10    LL5: load r13, (r12)0
11         cmp r7,r1,r0
12         beq r7,LL6
13         load r9, (r11)0
14         mult r3,r8,r9
15         add r6,r1,r3
16         store r6, (r12)0
17         b LL7
18    LL6: mult r6,r13,r0
19         add r13,r2,r13
20         store r13, (r12)0
21    LL7: add r4,1,r4
22         b LL2
23    LL3:
```

Aquest codi es pot expressar en pseudocodi amb la següent estructura:

```
S0=S1+S2;
S1=S2+S1;
FOR (I=0;I<=5;I++)
     {      IF S1<>S0
              { S3=S8*S9
                S6=S1+S3 }
            ELSE
              { S4=S3*S0;
                S1=S2+S1; }
     }
```

Si suposem que la disposem de les següents unitats funcionals:

UNITATS	NÚMERO	LATÈNCIA ENTRADA	LATÈNCIA SORTIDA	OPERACIONS
De salt	1	1	1	b, ble
Senceres	1	1	2	add
Multiplicació	1	2	4	mult
Accés memòria	1	1	1	load, store
Complements	1	1	1	mov, cmp

Sabent que la condició del IF es compleix un 70 % de les vegades.

Contesta:

a) Aplica la tècnica de Loop Unrolling a aquest codi.

b) Dibuixa els blocs bàsics que representa el codi **original**.

c) Realitza un trace schedulling (utilitzant l'LST) del codi **original**, i tenint en compte les condicions donades.

PROBLEMA 4 (1,5 punts)

Volem construir un multicomputador amb 16 nodes. Per fer aquest disseny volem comparar dues topologies de xarxa d'interconnexió. Per una banda un hipercub i per l'altra un torus

a) Fer el disseny de la xarxa hipercub i la xarxa torus 2D.

b) Calcular el grau (d), ampla de bisecció (b) i el diàmetre (d) per cadascuna de les xarxes.

c) Executar l'algorisme d'encaminament Dimension Order i XY (hipercub i xarxa torus respectivament) per trobar la ruta entre els nodes 1 i 12. Fer l'etiquetatge dels nodes i calcular la ruta.

d) Si disposem d'enllaços de 2 GB, de quin ampla de banda disposem cada node. (suposem que cada node està connectat a un router)

PROBLEMA 5 (1 punt)

Disposem del següent codi :

```
for(j=1;j<1000;j++)
        {
                for(i=1;i<1000;i++)
                {
                        for (k=1;k<1000;k++)
                        {
                                c[i][j] = ADD (a[i][k],b[k][j]);
                                t = ADD (c[i][k],a[i]);
                        }
                }
        }
```

NOTA : on ADD és una funció que retorna el resultat de la suma dels dos operands

Indicar com afectaria l'aplicació de les següents tècniques d'optimització. Justificar quines es podem aplicar i quines no, i quines podrien arribar a millorar el rendiment. Al efectuar un perfex, tenim el següent perfil en les caches:

% perfex -e 25 multmat <- Per obtenir el número de falles de cahcé de L1

0 Cycles.....59002735555
25 Primary data caché misses.....1338524786

% perfex -e 26 multmat <- Per obtenir les falles de caché de L2

0 Cycles.....59018761168
26 Secondary data caché misses.....54515279

Temps d'execució 351 segons

Tècnica d'optimització	Justificació
SW pipelining	
Cache Blocking	
Loop fusion	
Inlining	
Loop Interchange	

PROBLEMA 6 (1 punt)

Respon, justificant la resposta, les següents preguntes sobre PVM :

a) Quina instrucció fem servir per inicialitzar un buffer (canal) per enviar dades. De forma que les dades es codifiquin en un format estándar a dues arquitectures diferents ?. Com s'anomena el protocol que efectua la conversió de dades per tal de fer-la transparent al programador ?

b) Quin missatge d'error hauria d'aparèixer al 'printf' en el següent fragment de codi PVM:

```
int FILLS[2];
pvm_spawn("TASCAFILLA", NULL, PvmTaskDefault, 0, 2, FILLS);

if (pvm_send(FILLS[1], 2) < 0) {
      printf("????? \n");
      pvm_exit();
      return -1;
}
```