Examen Final Segmentació i Paral.lelisme
21 de gener del 2005

Problema 1

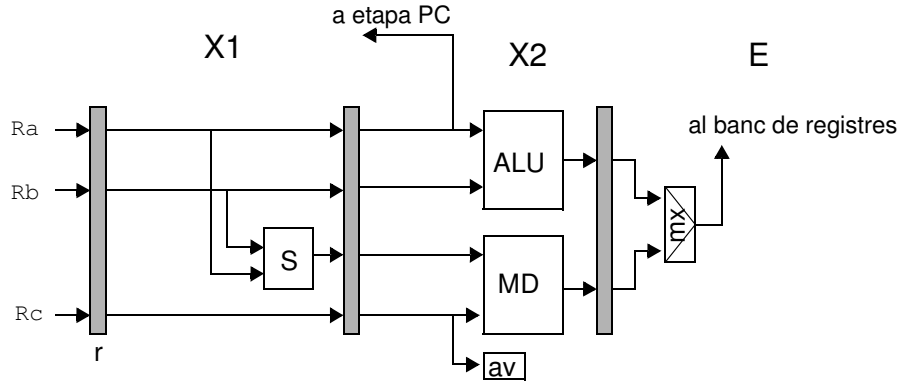
Considereu un processador amb un llenguatge màquina que conté 4 tipus d' instruccions:

load (ld): Rc = M[Ra+Rb]
store (st): M[Ra+Rb] = Rc
aritmètiques: Rc = Ra op Rb
salts condicionals (br): si cond(Rc) llavors PC = Ra

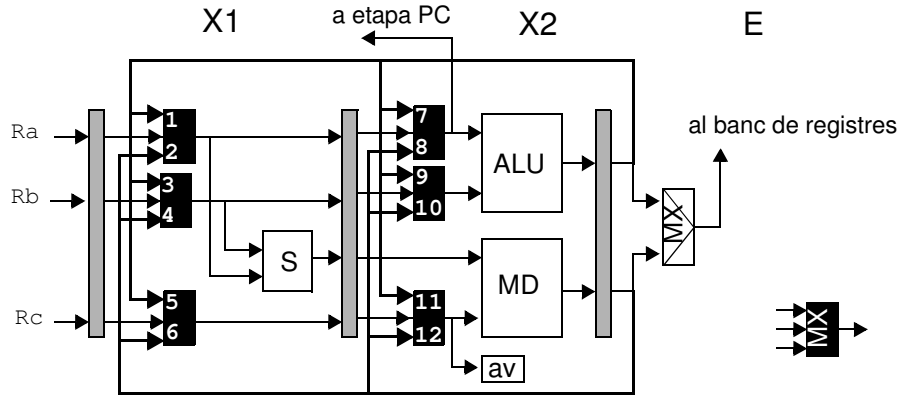
La segmentació del camí de dades del processador es realitza en 6 etapes:

- PC: Càlcul del PC següent instrucció
- F: Cerca de la instrucció
- D: Descodificació, detecció de perills, lectura de registres
- X1: Càlcul adreça efectiva (només per ld/st)
- X2: Operació (op) en les instruccions aritmètiques, accés a memòria en les instruccions ld/st, avaluació condició i modificació del PC amb l'adreça destí de salt si es compleix una condició sobre el contingut del registre Rc (per les instruccions br)
- E: Escriptura resultat en registre, si s'escau

La següent figura mostra part del camí de dades del processador (no inclou les etapes PC, F i D). El banc de registres permet l'escriptura i lectura d'un mateix registre en un cicle de rellotge i ocupen tot el cicle de rellotge. En cas de perill de dades, el processador es bloqueja en l'etapa D fins que desapareix el perill. En el cas de perill de dades, el processador es bloqueja en l'etapa PC fins que la condició està avaluada, de manera que cap al final de l'etapa X2 el nou PC ja està disponible per ser carregat en el registre PC.



Modifiquem el camí de dades afegint els 12 curtcircuits (bypass) tal com mostra la figura següent:



En els dos casos, considerarem que les etapes que determinen el temps de cicle de rellotge són X1 i X2. El temps (en nanosegons) de cada component és:

S (sumador): 2; ALU (unitat aritmètica): 3; MD (memòria de dades):4.5;
AV (circuiteria per l'avaluació de condicions):1; r (registre d'aillament entre etapes): 0.5

Es demana:

Quin hauria de ser el retard màxim (en nanosegons) introduït pels multiplexors (MX) i la lògica que els controla per tal de que el temps d'execució del bucle següent sigui en el disseny amb curtcircuits 1.3 vegades més ràpid que en el processador sense curtcircuits. Per això, hauras de dibuixar el cronograma d'execució d'una iteració, indicant en cada cicle de rellotge els curtcircuits utilitzats (utilitza la numeració indicada a la figura anterior). Per simplificar la resolució del problema, considera que tots els registres ja estan inicialitzats en el moment de començar l'execució del bucle.

```
for (; p!=NULL; p=p->next)
{
      s = s + p->dat
      p->dat = s;
}
```

```
for: ld     r4,(r0+r1)  ;r0 apunta a p, r1 val 0
     add    r3,r4,r3    ;r3 inicialment val 0
     st     r3,(r0+r1)  ;
     ld     r0,(r0+r2)  ;r2 val 8
     bne    r0,r9       ;r9 apunta a for
```

Problema 2

Considereu una implementació superescalar del processador amb el llenguatge màquina bàsic explicat a classe. El processador es capaç d'iniciar l'execució de més d'una instrucció per cicle i incorpora 4 unitats funcionals: entera, memòria, punt flotant i salt. La segmentació de les instruccions es FDIEWC per les enteres i de salt, FDIEEEEEEEEWC (8 cicles d'execució) per les de punt flotant, i FDIEMWC per les d'accés a memòria. Per permetre l'execució fora d'ordre, el processador implementa el mecanisme de Tomasulo explicat a classe, amb un nombre infinit d'estacions de reserva. Seguint l'esquema explicat a classe, els bypass de les estacions de reserva s'activen durant l'etapa W de la instrucció que genera el resultat. En les instruccions d'accés a memòria, el càlcul de l'adreça de memòria es fa tant aviat com possible (Tomasulo), quedant els accessos enquats, si es necessari, en els buffers de load/store a l'espera de la dada del store o resolució de conflictes. En les de salt, la predicció es fa a la fase de F i la validació al final de l'etapa E. El predictor de salts es basa en comptadors saturats de dos bits, inicialment a zero. Si el bit de més pes val zero, la predicció es a saltar; si no, la predicció es a no saltar. El renombrament de registres es fa a l'etapa D i tenim infinits registres per fer renombrament. El processador pot fer el W i el C de 2 instruccions per cicle. L'etapa C es fa en el mateix ordre que D.

Donat el següent fragment de programa:

```
for (i=0; p!=NULL ; i++)
{
      v[i] = v[i] + p->dat;
      p=p->next;
}
```

```
for: ld.f   F1,v(r1)    ;lectura element p
     ld.f   F2,0(r0)    ;lectura element v
     add.f  F2,F2,F1    ;F2 ← F2 + F1
     st.f   F2,v(r1)    ;escriptura element v
     add    r1, r1, 8   ;seguent element de v
     ld     r0,8(r0)    ;r0 ← mem[r0+8]
     bne    r0, for     ;si (r0≠0) salta a for
```

Es demana:

a) Considerant que els vectors p i v ocupen espais de memòria diferents, quin seria el temps d'execució, suposant que en total el bucle realitza 3 iteracions, si el processador es capaç d'iniciar l'execució de 2 o 4 instruccions per cicle? Per això, hauràs de dibuixar el cronograma d'execució indicant que es fa en cada cicle.

b) Que canviaria en el cronograma del apartat a) si els vectors p i v es poden solapar, total o parcialment, en memòria.