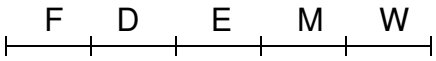
Examen Final Segmentació i Paral.lelisme
10 de Juny del 2004

Problema 1 (no cal fer-lo si el parcial està aprovat)

Tenim un processador segmentat en 5 etapes:



Els riscos s'analitzen a l'etapa D, si no es poden resoldre el processador es bloqueja. L'adreça destí de tots els salts i la condició de les instruccions de salt condicional es calculen a l'etapa E. **El processador té tots els curtcircuits necessaris**. Els salts incondicionals els indiquem amb la instrucció `br`, `beq` indica saltar si el contingut del registre es igual a zero i `bgt` indica saltar si el contingut del registre es mes gran de zero.

Donat el següent codi per calcular el màxim comú divisor:

```
while (a!=b) {
  if (a>b) {
    a= a-b;
  } else {
    b= b-a;
  }
}
mcd= a;
```

```
while:  sub r3, r2, r1
        beq r3, fora        S1
        bgt r3, major       S2
        sub r1, r1, r2
        br while            S3
major:  sub r2, r2, r1
        br while            S4
fora:   ...
```

Inicialment: `r2= a, r1= b`
Resultat: `r2`

Sabem que, per una parella de números *a* i *b,* fan falta **`k`** iteracions del `while` per calcular el resultat i finalment **`S1`** salta a fora del bucle. El salt **`S2`** salta amb una probabilitat **`pt2`**.

(a) Durant les primeres **`k`** iteracions del bucle `while`, donar totes les possibles seqüències d'execució del `while` segons el comportament de **`S2`**. Indicar el codi que s'executa, el cronograma d'execució i els cicles per iteració en cada cas. Per cada possible seqüència de codi donar dos cronogrames: **(1)** el processador es bloqueja fins que els salts es resolen (es coneix la condició i l'adreça destí); i **(2)** predicció de salts estàtica fixa "salta", de manera que després de l'etapa F ja va a cercar la instrucció que hi ha a l'adreça desti del salt (en cas de predicció incorrecte, el processador anul.la les instruccions ja iniciades).

(b) Determinar els cicles necessaris per calcular el *mcd* en funció de **`k`** i les probabilitats adients en cadascun dels dos casos anteriors.

Problema 2

Volem executar el codi del problema 1 en un processador superescalar amb les característiques següents:

- Processador capaç d'iniciar l'execució de dues instruccions noves cada cicle. Unitats funcionals disponibles: 1 entera, 1 d'aritmètica real, 1 de memòria i 1 de salts. Segmentació com la de classe: 7 etapes (FDIEMWC) excepte les d'aritmètica real que en tenen 6 (FDIEWC), on I es l'etapa d'Issue i C es l'etapa de Commit. Considereu que els bypass de les estacions de reserva s'activen durant la fase W.
- Memòria ideal (sempre fa hit amb temps d'accés 1 cicle). Unitat entera i de salts amb temps d'execució 1 cicle; Unitat real segmentada amb temps d'execució 5 cicles.
- Les instruccions son de 4 bytes. Les linies de cache son de 8 bytes i la instrucció que hi ha a l'adreça `while` comença en línia de cache.
- Predictor de salts basat en comptadors saturats de 2 bits, un per cada instrucció de salt condicional; inicialment val zero (si el bit de més pes val zero, llavors la predicció es "salta"). EL comptador s'incrementa si "no salta"; altrament es decrementa. La predicció es fa a l'etapa de fetch. La validació de la predicció es fa a l'etapa W de la instrucció de salt. Els salts incondicionals (**S3** i **S4**) fan predicció de "salta" i per tant sempre, excepte en la darrera iteració, fan una predicció correcte de l'adreça destí del salt.

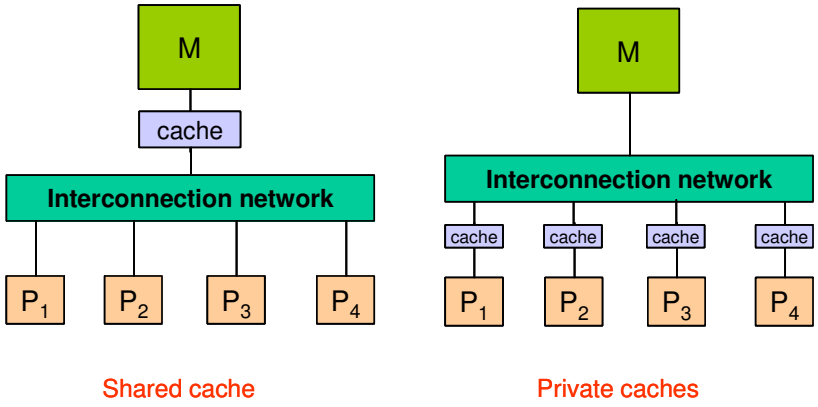
Considereu que el salt **S2** es comporta tal com indiquem a continuació: "no salta", "no salta", "no salta", "salta", "salta", ..., "salta", "no salta" (es a dir, només en les tres primeres iteracions i en la darrera "no salta").

Es demana:
a) Calculeu quin es la probabilitat d'encert del predictor per cadascun dels salts condicionals (**S1** i **S2**) del programa, en funció de k, el nombre d'iteracions per completar l'execució del bucle anterior.

b) Calculeu el número de cicles que trigarà l'execució de les 4 primeres iteracions del bucle. Considereu que el nombre de estacions de reserva i registres de renaming és infinit.

Problema 3

Tenim dos sistemes multiprocessadors alternatius: el primer amb cache compartida i el segon amb caches privades, ambdós amb 4 processadors, tal com mostra la figura següent:



En el sistema anomenat “shared cache”, la cache compartida te una capacitat de 1 Mbyte. En el sistema anomenat “private caches”, les caches locals tenen una capacitat de 256 Kbytes. Les línies son de 128 bytes. El algorisme de reemplaçament es FIFO.

Considereu que cadascun dels 4 processadors executa el programa següent sencer:

```
int a[1024*256]; /* cada int son 4 bytes */

main() {
   int iter, i, var_privada;

   for (iter=0; iter < 2; iter++)
      for (i=0; i < 1024*256; i++) {
         var_privada = var_privada + a[i];
      }
}
```

en el que la variable a es compartida per tots els processadors i les variables i, n i var_privada s’emmagatzemen a registres. Les 4 instàncies del programa s’executen en paral.lel.

Es demana calcular la probabilitat total de hit a memòria cache que s’obté, considerant els accessos a memòria que fan tots els processadors en cadascun dels dos sistemes.