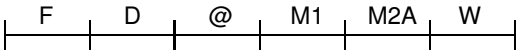
ETSETB
Enginyeries de Telecomunicació i en Electrònica
Segmentació i Paral.lelisme: Disseny dels Processadors Actuals
Curs: Quadrimestre de Primavera 2002/03
Data: 11 de juny 2003

Problema 1

Un processador segmentat lineal amb 7 etapes



- F: búsqueda de la instrucció, càlcul adreca instrucció seqüencial
- D: descodificació, detecció de riscs, lectura operants en registres (segon semicicle)
- @: càlcul de l'adreça efectiva, o adreça destí de salt, avaluació condició de salt
- M1: inici accés a memòria
- M2A: final accés a memòria, operació aritmètico-lògica
- W: escriptura resultat en registre (primer semicicle)

El processador no té curtcircuits. La detecció de riscs de dades es fa en la etapa D, bloquejant la búsqueda i descodificació d'instruccions quan es produeix un risc.
Per l'execució de les instruccions de salt condicional, el processador utilitza el mecanisme de salt amb anulació, continuant la búsqueda d'instruccions considerant que no salta. L'avaluació de la condició i la verificació de la predicció es fa en la etapa @; en cas d'error en la predicció s'anulen les instruccions de la branca seqüencial i s'actualitza el PC amb l'adreça de la instrucció destí de salt.

En el programa de prova el compilador ha desenrotllat dues vegades el cos del bucle, sense eliminar cap instrucció. El número de iteracions és múltiple de 8.

```
for (; p!=NULL; p->next)
{
      ac = ac + p->data;
      p->data = ac;
}
```

Valors inicials:

```
r0 = p (@primer element)
r2 = ac
NULL = 0
```

```
mes:    ld      r1,4(r0)     ;r1 ← mem[r0+4]
        add     r2,r1,r2     ;r2 ← r1 + r2
        st      r2,4(r0)     ;mem[r0+4] ← r2
        ld      r0,0(r0)     ;r0 ← mem[r0+0]
        beq     r0, 0, prou  ;si (r0=0) salta a prou
        ld      r1,4(r0)     ;r1 ← mem[r0+4]
        add     r2,r1,r2     ;r2 ← r1 + r2
        st      r2,4(r0)     ;mem[r0+4] ← r2
        ld      r0,0(r0)     ;r0 ← mem[r0+0]
mes+9:  bneq    r0, 0, mes   ;si (r0≠0) salta a mes
prou:
```

Es demana:

a) El cronograma d'execució de dues iteracions consecutives del bucle en alt nivell (és a dir, la seqüència d'instruccions `mes`, ..., `mes+9`). Indica clarament els cicles de bloqueig. Calcula el número mitjà de Cicles Per Instrucció (CPI) a l'executar el programa de prova.

Afegim al processador els curtcircuits necessaris tractar els riscs de dades.

b) Repeteix l'apartat a), indicant clarament els curtcircuits utilitzats en el programa de prova.

c) Calcula el CPI en dos casos: si el compilador no hagués replicat el cos del bucle (k=1) i en el cas de que hagués desenrotllat el bucle 8 vegades (k=8) sense eliminar cap instrucció. En ambdós casos no s'han de reordenar les instruccions. No cal mostrar cronogrames

Problema 2

Considereu un sistema multiprocessador amb memòria compartida, basat en bus i snoopy (coherencia amb protocol write-invalidate). El sistema inclou 8 processadors, cadascun amb una memoria cache de 64 Kbytes, amb correspondencia directa i linia de 64 bytes. El sistema s'utilitza per executar el següent programa:

```
int*8 a(max)
for (i=0; i<max; i++) a[i]=1.0;
for (i=0; i<max-1; i++) a[i]=a[i+1]*2.0;
for (i=0; i<max-1; i++) a[i-1]=a[i]+a[i+1]*2.0;
```

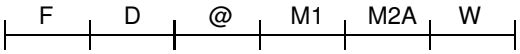
Cada processador executa un bloc consecutiu de max/8 iteracions.

Es demana: Quants bytes es transfereixen pel bus comú, per valors de max=1024 i max=32? Considereu que les invalidacions transfereixen 1 byte.

ETSETB
Enginyeries de Telecomunicació i en Electrònica
Segmentació i Paral.lelisme: Disseny dels Processadors Actuals
Curs: Quadrimestre de Primavera 2002/03
Data: 11 de juny 2003

Problema 1

Un processador segmentat lineal amb 7 etapes



- F: búsqueda de la instrucció, càlcul adreca instrucció seqüencial
- D: descodificació, detecció de riscs, lectura operants en registres (segon semicicle)
- @: càlcul de l'adreça efectiva, o adreça destí de salt, avaluació condició de salt
- M1: inici accés a memòria
- M2A: final accés a memòria, operació aritmètico-lògica
- W: escriptura resultat en registre (primer semicicle)

El processador no té curtcircuits. La detecció de riscs de dades es fa en la etapa D, bloquejant la búsqueda i descodificació d'instruccions quan es produeix un risc.
Per l'execució de les instruccions de salt condicional, el processador utilitza el mecanisme de salt amb anulació, continuant la búsqueda d'instruccions considerant que no salta. L'avaluació de la condició i la verificació de la predicció es fa en la etapa @; en cas d'error en la predicció s'anulen les instruccions de la branca seqüencial i s'actualitza el PC amb l'adreça de la instrucció destí de salt.

En el programa de prova el compilador ha desenrotllat dues vegades el cos del bucle, sense eliminar cap instrucció. El número de iteracions és múltiple de 8.

```
for (; p!=NULL; p->next)
{
      ac = ac + p->data;
      p->data = ac;
}
```

Valors inicials:

```
r0 = p (@primer element)
r2 = ac
NULL = 0
```

```
mes:    ld      r1,4(r0)     ;r1 ← mem[r0+4]
        add     r2,r1,r2     ;r2 ← r1 + r2
        st      r2,4(r0)     ;mem[r0+4] ← r2
        ld      r0,0(r0)     ;r0 ← mem[r0+0]
        beq     r0, 0, prou  ;si (r0=0) salta a prou
        ld      r1,4(r0)     ;r1 ← mem[r0+4]
        add     r2,r1,r2     ;r2 ← r1 + r2
        st      r2,4(r0)     ;mem[r0+4] ← r2
        ld      r0,0(r0)     ;r0 ← mem[r0+0]
mes+9:  bneq    r0, 0, mes   ;si (r0≠0) salta a mes
prou:
```

Es demana:

a) El cronograma d'execució de dues iteracions consecutives del bucle en alt nivell (és a dir, la seqüència d'instruccions `mes`, ..., `mes+9`). Indica clarament els cicles de bloqueig. Calcula el número mitjà de Cicles Per Instrucció (CPI) a l'executar el programa de prova.

Afegim al processador els curtcircuits necessaris tractar els riscs de dades.

b) Repeteix l'apartat a), indicant clarament els curtcircuits utilitzats en el programa de prova.

c) Calcula el CPI en dos casos: si el compilador no hagués replicat el cos del bucle (k=1) i en el cas de que hagués desenrotllat el bucle 8 vegades (k=8) sense eliminar cap instrucció. En ambdós casos no s'han de reordenar les instruccions. No cal mostrar cronogrames

Problema 2

Considereu un sistema multiprocessador amb memòria compartida, basat en bus i snoopy (coherencia amb protocol write-invalidate). El sistema inclou 8 processadors, cadascun amb una memoria cache de 64 Kbytes, amb correspondencia directa i linia de 64 bytes. El sistema s'utilitza per executar el següent programa:

```
int*8 a(max)
for (i=0; i<max; i++) a[i]=1.0;
for (i=0; i<max-1; i++) a[i]=a[i+1]*2.0;
for (i=0; i<max-1; i++) a[i-1]=a[i]+a[i+1]*2.0;
```

Cada processador executa un bloc consecutiu de max/8 iteracions.

Es demana: Quants bytes es transfereixen pel bus comú, per valors de max=1024 i max=32? Considereu que les invalidacions transfereixen 1 byte.