Számítógépek felépítése
5. előadás
a CPU gyorsítása, pipeline, cache

Dr. Istenes Zoltán
ELTE-TTK

5. előadás tartalma

- Processzor gyorsítási lehetőségek
- Pipeline feldolgozás
- Cache gyorsító tár
- CISC vs. RISC...
- Bitszelet processzorok
- Co-processzorok
- Pédák processzorokra...

Órajelfrekvencia, idő, távolság

- 1 „normál” kapu kapcsolási idő = 10ns
- 300 MHz = 0.3*10e9 1/s -> 3 ns
- 300.000 km/s = 0.3*10e9 m/s -> 0.3 m/ns

Processzor gyorsítási lehetőségek

művelet végzés gyorsítása ?

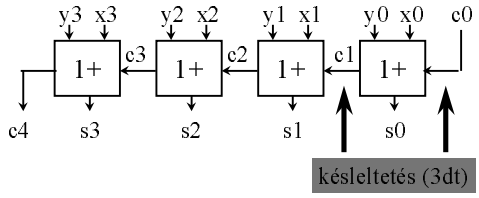
utasítás végrehajtás gyorsítása ?

párhuzamosítás !

Műveletvégzés gyorsítása

Összeadás átvitel-előrelátás
(addition carry-lookahead)

Teljes összeadó „késleltetése”
(ripple carry)



1 dt = 1 kapu
kapcsolási idő (kb. 10ns)

Gyorsított összeadás

- Ci+1=Xi.Yi+Xi.Ci+Yi.Ci
- Ci+1=Gi+Pi.Ci ahol Gi=Xi.Yi és Pi=Xi+Yi
- Gi = generate (Xi=Yi=1)
- Pi = propagate (Xi vagy Yi = 1)

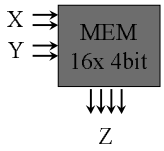
4bites átvitel-előrelátás

(„carry lookahead")

- C1=G0+P0.C0
- C2=G1+P1.C1
  =G1+P1.G0+P1.P0.C0
- C3=G2+P2.C2
  =G2+P2.G1+P2.P1.G0+P2.P1.P0.C0
- C4=G3+P3.C3
  =G3+P3.G2+P3.P2.G1+P3.P2.P1.G0+P3.P2.P1.P0.C0

Szorzás ROM-ból kiolvasással

	X Y	Z
0*0	00 00	0000
0*1	00 01	0000
0*2	00 10	0000
0*3	00 11	0000
1*1	01 01	0001
1*2	01 10	0010
1*3	01 11	0011
2*0	10 00	0000
2*1	10 01	0010
2*2	10 10	0100
2*3	10 11	0110
3*0	11 00	0000
3*1	11 01	0011
3*2	11 10	0110
3*3	11 11	1001



- gyors, egyszerű, olcsó
- tetszőleges függvényre
- csak kisméretben alkalmazható

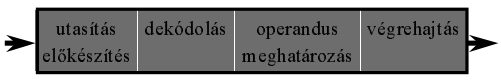
Pipeline feldolgozás

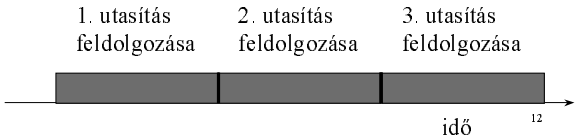
(csővezeték, futószalag feldolgozás)

„Pipelining"

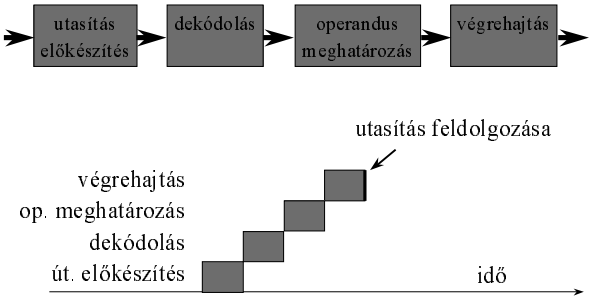
- részfázisokra bontás
- független részfázisok, önálló erőforrásokkal
- az egyik fázis eredménye a következő induló adata

Utasítások egymás utáni végrehajtása

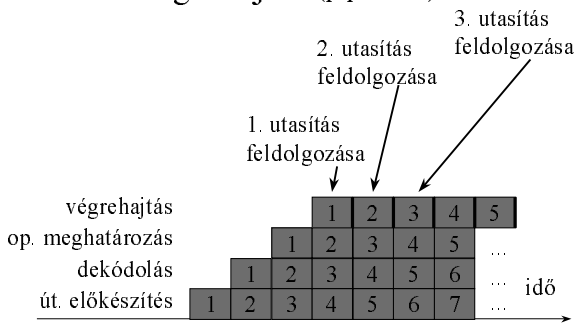




Utasítás végrehajtás fokozatokra bontása



Csővezetékszerű utasítás végrehajtás (pipelined)



Problémák a pipeline-vel

tárolóhivatkozás :

- lassú memória miatt várni kell az operandusokra
- egyszerre kéne ugyanabból a tárból adatot és utasítást elérni

vezérlés átadó utasítások...

megszakítások...

folyamatos pipeline feldolgozás megszakítása, felfüggesztése

Memória utasítások a pipeline-ben

- várakozó ciklusok beiktatása - „lyukak” a pipeline-ben
- késleltetett memória utasítás - utasítások átrendezése, üres utasítás beiktatása

```
load R1, (A)
load R2, (B)
add R3,R1,R2
```


Elágazások kezelése a pipeline-ben

- az ugrási cím csak az utasítás feldolgozása után lesz ismert...
- ha „rossz” ágat utasításait kezdi el feldolgozni akkor a pipeline-t (és az utasítások hatásait) törölni kell...
- delayed branch (NOP utasítással feltöltés)

Adatok felhasználása a pipeline-ben

- adat függőség
```
add R3,R1,R2
sub R4,R3,R0
```

ütközések :

- írás utáni írás : 2. írás után ír az 1.
- írás utáni olvasás : 2. előbb olvas mint az 1. ír
- olvasás utáni írás : 2. előbb ír mint az 1. olvas
- olvasás utáni olvasás :-)

instruction scheduling...

internal forewarding...

scoreboarding...

Pipeline összefoglalás

- egyszerű elv
- bonyolult, „trükkös” megvalósítás
- sokat segíthet a fordítóprogram

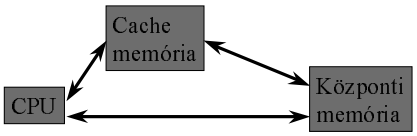
Cache

„gyorsító” tár

Cache memória

Probléma : a központi memória általában sokkal lassabb (5x-10x-...) mint a processzor...

Kicsi, gyors (drága!), puffer memória a központi memória és a processzor közé...



„Tipikus” cache

- 64 Mbyte memóriához 64 Kbyte cache
- 20x gyorsabb
- 1/1000 méret
- 98% cache találat

Cache működés (olvasás)

CPU által generált memóriacím küldése a cache-nek :

1. ha az adat a cache-ben van
   (cache hit, cache találat) :
   - adat elküldése a CPU-nak
2. ha az adat nincs a cache-ben
   (cache miss, cache „nemtalálat”, „találat hiba”) :
   - a generált memóriacím küldése közp. memóriának
   - adat kiolvasása a központi memóriából,
   - adat elküldése a CPU-nak, és a cache-be

Cache

- a memória és a cache között blokkos adatátvitel (4-64 szó)
- teli cache esetén egy cache blokkot fel kell szabadítani (helyettesítési eljárások)

Cache hatása az átlagos elérési időre és a hatékonyságra (példa)

Adatok :

- cache elérési idő tc = 160ns
- memória elérési idő tm = 960ns
- átlagos cache találati arány h = 90%

Kérdés :

- mennyi az átlagos elérési idő ? (ta)
- mekkora a hatékonyság növekedés ? (g)

Cache hatása az átlagos elérési időre és a hatékonyságra (példa)

ta = h*tc + (1-h)*(tc+tm)
= 0.9*160 + 0.1*(960+160)
= 256 ns (átlagos elérési idő)
r = tm/tc = 960/160 = 6
g = 1/(1+r*(1-h)) =1/(1+6*0.1) = 0.625
= 62.5% (hatékonyság növekedés)

Cache hatása (SuperSPARC példa)

- CPU 0.75 CPI (cycles per instruction)
- 3% cache miss
- memória késleltetés 10 ciklus
- 1.33 memória referencia / utasítás
- cache miss „büntetés” 0.4 CPI
  0,75*1,33*3%*10
- 35% teljesítmény csökkenés
  1-1/(1+3%*(10+1))

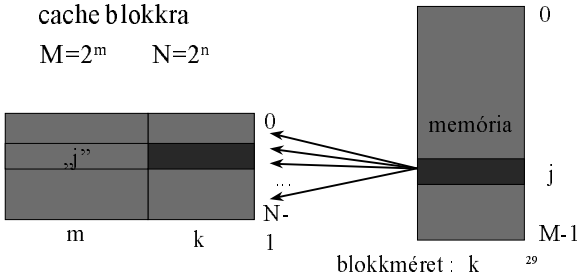
Memória „leképzése” a cache-ra

Teljesen asszociatív cache
Közvetlen leképzésű cache
Csoport asszociatív cache

Teljesen asszociatív cache
(fully associative cache)

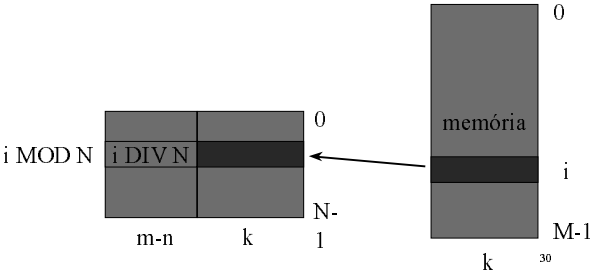
központi memória blokk „i” (0<=i<=M-1)
leképezhető bármelyik „j” (0<=j<=N-1)
cache blokkra

$M=2^m$ $N=2^n$

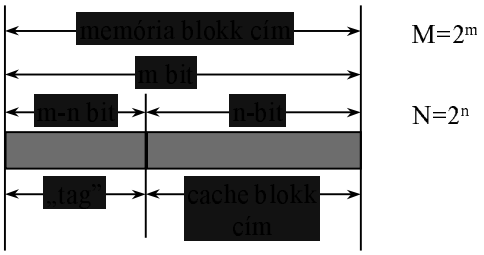


Közvetlen leképzésű cache
(direct mapping cache)

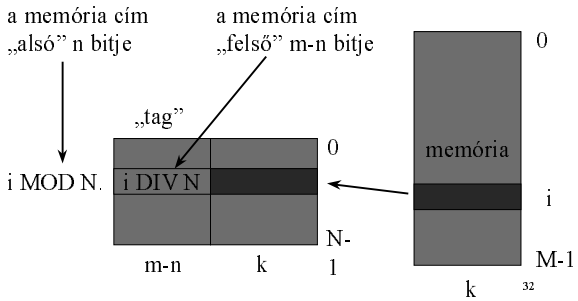
egy-egy memória blokk csak meghatározott helyre kerülhet (I-way set associative cache)



Közvetlen leképzésű cache címformátuma



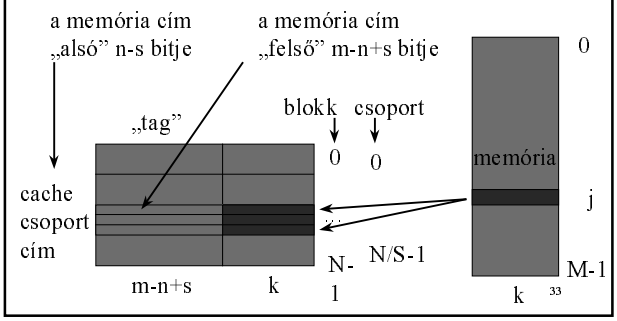
Közvetlen leképzésű cache



Csoport asszociatív cache

(set associative cache)

S csoport blokkonként $S=2^s$



Cache-ben lévő adat megváltoztatása

cache - memória koherencia fenntartása !

- azonnali átírás (write-through) :
  blokk tartalom módosítás esetén
  a blokk visszaírása a memóriába
- visszamásolási eljárás (write back) :
  - a frissen betöltött blokk „dirty” („piszkos”) bitje = 0
  - blokk tartalom módosítás esetén dirty bit = 1
  - blokk cache-ból törlésekor :
    ha dirty bit = 1 akkor vissza kell írni a memóriába
    (különben nem)

CISC vs. RISC processzorok

CISC = Complet Instruction Set Computer
(teljes utasítás készletű számítógép)
RISC = Reduced Instruction Set Computer
(csökkentett utasítás készletű számítógép)

CISC vs. RISC processzorok

CISC

- „könnyebb programozás”
- összetett utasítások, sokféle címzési mód
- utasítás végrehajtás több órajel alatt
- mikroprogramozott
- …

RISC

- egyszerűbb, gyorsabb
- optimalizált fordítók
- kevés utasítás és címzési mód
- rögzített utasítás forma
- utasítás végrehajtás 1 órajel alatt
- kevés memória hozzáférés
- sok regiszter
- huzalozott vezérlő
- pipeline feldolgozás

CISC vs. RISC

CISC (VAX)
- 14 féle címzési mód
- 250 utasítás

RISC
- 1 féle címzési mód
- 32 (50) regiszter...
- 30-40 utasítás

Bit-szelet processzorok

Bit-szelet (bit slice) processzor
- példa : AM 2901
- 4 bites ALU + regisztertömb
- 3 aritmetikai, 5 logikai művelet
- 9 bites vezérlés
- állapot jelek

Segédprocesszorok co-processzorok

- gyorsabb lebegőpontos, mátrix, grafikai műveletek
- új utasítások, adattípusok, regiszterek
- együttműködés a processzorral