

Memória-kezelés a modern operációs rendszerekben

Elméleti alapok

Valós és virtuális címzés

Példák (BSD Unix, Windows NT/2000/XP)



A feladat

- Memóriák típusai és jellemzői

Memória típusa (CPU: Intel P4 Xeon)	Méret (2002)	Sebesség (2002)
Regiszter	~96 B	~24 GB/s
Belső (L1) gyorsítótár	~128 KB	~15 GB/s
Belső (L2) gyorsítótár	~512 KB	~11 GB/s
Külső (L3) gyorsítótár	~1 MB	~8 GB/s
Központi memória	~512 MB	~3 GB/s
Helyi háttértár	~40 GB	~0,05 GB/s
Hálózati háttértár	~512 GB	~0,006 GB/s



2

Alapfogalmak

- Memória: címmel rendelkező adatok sorozata

Cím:	\$0000	\$0001	\$0002	\$0003	\$0004	\$0005	\$0006	\$0007
Adat:	\$46E2	\$F640	\$884A	\$4E75	\$5A5A	\$AA55	\$0000	\$0000

- Memória-kezelés:

- A rendelkezésre álló fizikai memória felosztása a rendszerben létező folyamatok között
- A felosztott memóriaterületek nyilvántartása, foglalások és felszabadítások végrehajtása
- Processzorszintű hardvertámogatás (MMU)

- Memória-kezelő alrendszer:

- Az OR fenti feladatokat ellátó komponense
- Fizikai és stratégiai feladatok



3

Memória-kezelés (összefoglaló)

- Egyszerű memória-kezelés (valós címzés)

- Monoprogramozás
- Multiprogramozás partíciókkal
 - Rögzített partíciók
 - Változó partíciók
- Átfedési technika (*overlay*)

- Virtuális memória-kezelés (virtuális címzés)

- Lapozáson alapuló rendszerek (*paging*)
- Szegmentáláson alapuló rendszerek (*segmenting*)
- Kombinált rendszerek

- Virtuális memória-kezelés ablaktechnikával



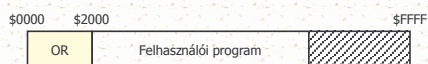
4

Memória-kezelési megoldások

- Egyszerű memória-kezelés (valós címzés)

- Monoprogramozás

- Az OR mellett egy időben egy program van a memóriában
- A programok valós abszolút címeket tartalmaznak
- Példa: MS-DOS (régebbi verziók)



```
...
lea $2000, a0
move (a0), d0
add d0, d0
move d0, $2010
...
```

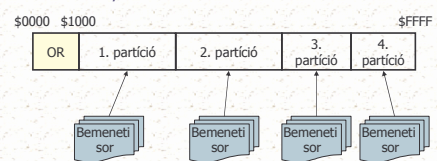


5

Memória-kezelési megoldások

- Egyszerű memória-kezelés (valós címzés)

- Multiprogramozás rögzített partíciókkal
 - Egy időben több program van a memóriában
 - A programok valós címeket tartalmaznak
 - Feladatok: sorkezelés, védelem
 - Példa: IBM System 360

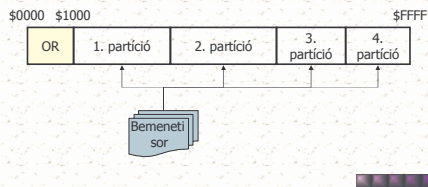


6

Memóriakezelési megoldások

• Egyszerű memóriakezelés (valós címzés)

- Multiprogramozás rögzített partíciókkal
 - Egyidőben több program van a memóriában
 - A programok valós címeket tartalmaznak
 - Feladatok: sorkezelés, védelem
 - Példa: IBM System 360

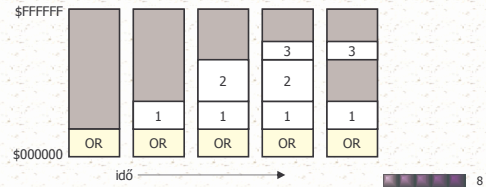


7

Memóriakezelési megoldások

• Egyszerű memóriakezelés (valós címzés)

- Multiprogramozás változó partíciókkal
 - Egyidőben több program van a memóriában
 - A programok valós áthelyezhető címeket tartalmaznak
 - Feladatok: kiosztás, üres területek összevonása, védelem
 - Példa: CDC Cyber



8

Memóriakezelési megoldások

• Egyszerű memóriakezelés (valós címzés)

- Átfedési technika (*overlay*)
 - Valószínű, hogy a teljes programot nem használjuk egyidőben, az egymással egyidőben sosem futó részek használhatnak azonos memóriaterületeket
 - Mára gyakorlatilag kihalt

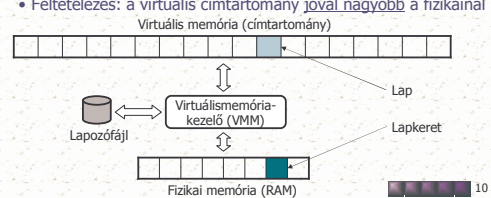


9

Memóriakezelési megoldások

• Virtuális memóriakezelés (virtuális címzés)

- A gépi szintű utasítások virtuális címekkel dolgoznak
- A programok virtuális címeket tartalmaznak, a folyamatoknak saját virtuális címtérük van
- A valós (fizikai) cím meghatározása a processzorban történik hardveres támogatással (architektúrafüggő)
- Feltételezés: a virtuális címtartomány jóval nagyobb a fizikainál



10

Memóriakezelési megoldások

• Virtuális memóriakezelés (virtuális címzés)

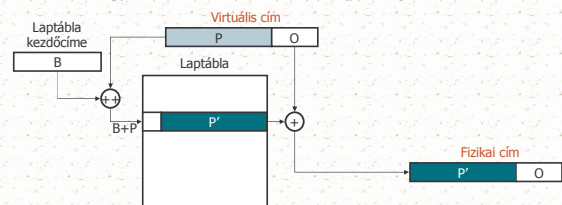
- Lapozáson alapuló rendszerek (*paging*)
 - A virtuális címtartomány egyforma méretű lapokra oszlik
 - Címkepzés (virtuális és valós cím felbontása):
 $V = (P, O)$ P : lap címe (*page*), O : lapon belüli cím (*offset*)
 $R = (P', O)$ P' : lapkeret címe (*page frame*), O : lapon belüli cím (*offset*)
 - A címkepzés a folyamatokhoz rendelt laptáblákkal történik, melyek a P és a P' összerendelését meghatározó táblázatok
 - A címkepzés *dinamikus* (futás közben történik)
 - A laptáblabejegyzések tartalma:
 - Érvényességjelző
 - Védelmi jelzők (olvasható, írható, végrehajtható stb.)
 - Módosítás- és hozzáférési jelző
 - P'

11

Memóriakezelési megoldások

• Virtuális memóriakezelés (virtuális címzés)

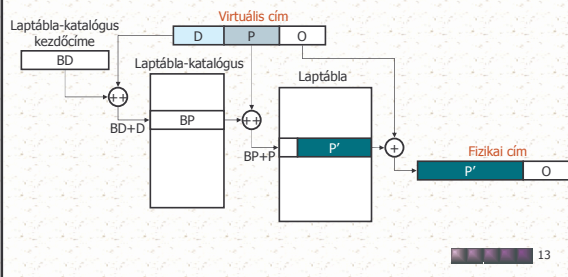
- Lapozáson alapuló rendszerek (*paging*)
 - Egyszintű dinamikus címkepzés (pl. VAX)



12

Memóriakezelési megoldások

- Virtuális memóriakezelés (virtuális címzés)
 - Lapozáson alapuló rendszerek (*paging*)
 - Kétszintű dinamikus címlekképzés (pl. Intel)



13

Memóriakezelési megoldások

- Virtuális memóriakezelés (virtuális címzés)
 - Lapozáson alapuló rendszerek (*paging*)
 - Lapok mozgatása a memória és a háttértár között
 - Bizonyos memóriafoglalási szint felett kilapozás
 - Laphiba esetén belapozás
 - Optimalizálási lehetőségek
 - Laptáblák méretének csökkentése halogató technikával
 - Laptáblák méretének csökkentése invertálással
 - Címfordítás gyorsítása TLB-vel
 - Laphibák gyakoriságának csökkentése különböző ki- és belapozási algoritmusokkal

14

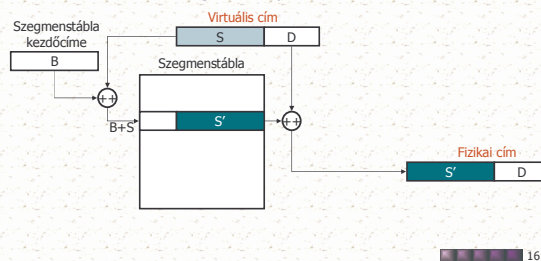
Memóriakezelési megoldások

- Virtuális memóriakezelés (virtuális címzés)
 - Szegmentáláson alapuló rendszerek (*segmenting*)
 - A virtuális címtartomány többdimenziós \mathbb{Z} külön címtartomány használható a kód, az adatok, a verem stb. számára
 - Ezek a címtartományok az ún. szegmensek, melyek önmagukban változó hosszúságú, 0-tól induló lineáris címtartományok
 - A valós (fizikai) cím meghatározása a processzorban történik hardveres támogatással (architektúrafüggő)
 - A szegmenstáblabejegyzések tartalma:
 - Érvényességjelző
 - Védelmi jelzők (olvasható, írható, végrehajtható stb.)
 - Szegmenshossz

15

Memóriakezelési megoldások

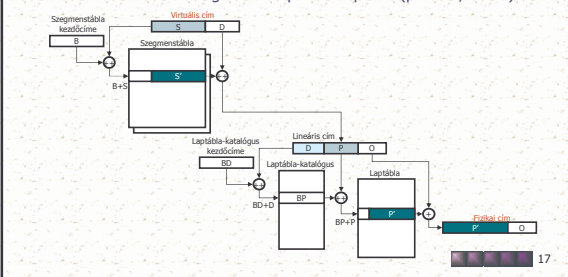
- Virtuális memóriakezelés (virtuális címzés)
 - Szegmentáláson alapuló rendszerek (*segmenting*)
 - Dinamikus szegmenslekképzés



16

Memóriakezelési megoldások

- Virtuális memóriakezelés (virtuális címzés)
 - Kombinált rendszerek
 - Dinamikus szegmenslekképzés + lapozás (pl. i386, 486...)



17

Memóriakezelési megoldások

- Virtuális memóriakezelés ablaktechnikával
 - A fizikai memória meghaladhatja a virtuális címtartomány méretét \mathbb{Z} a címtartományt rá kell „vetíteni” a fizikai memóriára
 - Példa: Intel Pentium Pro/II/III/4, PAE
 - Háromszintű dinamikus címlekképzés
 - 24+12 bit szélességű fizikai címek \mathbb{Z} max. 64 GB memória
 - A folyamatok címtérében egy-egy „ablak” szolgál a fizikai memória egy részének leképezésére
 - Külön programozói felületet igényel \mathbb{Z} csak az erre külön felkészített programok képesek kihasználni
 - Átmeneti megoldás (hosszú távon a 64 bites memóriacímzés oldja meg a problémát)

18

Példák

- UNIX rendszerek
 - Aránylag egyszerű, sok évtizedes hagyományokkal bíró megoldások
 - Az utóbbi évek UNIX-változatai már jóval összetettebbek
- Microsoft Windows NT/2000/XP
 - Rendkívül bonyolult, rafinált trükköket alkalmazó memória-kezelés már a kezdetektől



19

Példa: BSD Unix 4.3

- Lapozáson alapuló virtuális memóriakezelés
 - Memórafoglalási és -felszabadítási funkciók
 - Memóriavédelem több szinten
 - A kernel kódja és adatai felhasználói módban nem érhetők el
 - A folyamatok nem érhetik el egymás memóriáját
 - Hardveres lapszintű védelem
 - „Puha” és „kemény” laphibák: ha a hivatkozott lap még a memóriában van, de érvénytelen, akkor nem a háttértárról tölti be
 - A szabad lapok listája rendszerszinten globális
 - Innen is azonnal, I/O művelet nélkül foglalkozható lapok
 - Folyamat létrehozásakor előre beolvas néhány lapot, és a szabad lapok listájára helyezi

20

Példa: BSD Unix 4.3

- Kilapozás: második esélyes FIFO algoritmus (más néven „óra-algoritmus”)
 - a „*pagedaemon*” nevű folyamat másodpercenként többször lépteti az óramutatókat
 - az első mutató töröl, a második ellenőriz

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
H	0	0	0	0	0	0	0	0	1	1	0	0	1	1	0	0
M	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
Lap																

- Belapozás: igény szerint (*demand paging*)

21

Példa: BSD Unix 4.3

- Memóriabiztosítási stratégia
 - Küszöbértékek: $minfree < desfree < lotsfree$
 - A *pagedaemon* futtatási gyakorisága attól függ, milyen közel van a rendszer a *lotsfree* szinthez
 - A *pagedaemon* áll le utoljára
 - Ha túl kevés a memória (a *minfree* szint alá csökken), és az utóbbi időben folyamatosan a *desfree* szint alatt volt, akkor teljes folyamatokat is kilapoz (*swapping*)
 - Ezt a rendszer igyekszik normál lapozással (*paging*) elkerülni
 - Minél alacsonyabb egy folyamat prioritása, annál valószínűbben válik kilapozás (*paging és swapping*) áldozatává

22

Példa: BSD Unix 4.3

- Fájlok leképzése a memóriába
 - A folyamatok címtérének egy része egy fájlhoz (vagy egy részéhez) rendelhető
 - Ezt követően a fájl tartalma a megadott memóriacímeken is elérhető (írható és olvasható)
 - Fájlok vagy memóriaterületek megosztása
 - Minden résztvevő folyamat kap egy mutatót ugyanarra a (fájl egy részét tartalmazó) memóriaterületre, szinkronizálásra nincs szükség
 - A fájl lezárásakor a memóriába leképzett, időközben módosított adatok a lemezre íródnak

23

Példa: Windows NT/2000/XP

- Lapozáson alapuló virtuális memóriakezelés
 - Kétszintű memórafoglalás
 - *reservation*: virtuális címtartomány lefoglalása
 - *commitment*: fizikai memória lefoglalása (laponként) a fenti címtartományban
 - Memóravédelem több szinten
 - Lásd BSD Unix 4.3; valamint a megosztott memóriaterületek védelem hozzáférési listákkal (ACL)
 - Kupac (*heap*):
 - Segédfunkciók virtuális címtartományok lefoglalásához
 - Saját kupacok is létrehozhatók, ezek egy rendszerhívással felszabadíthatók
 - Egyes lapok a memóriába „zárhatók”
 - Ezek a lapok mindig a fizikai memóriában maradnak
 - Hasznos funkció pl. DMA átvitel megvalósításához

24

Példa: Windows NT/2000/XP

- További részletek a gyakorlaton
 - A 32 bites címtartomány felosztása, címleképzés
 - Ki- és belapozási stratégia
 - Munkakészletek (*working sets*)
 - Memória megosztása
 - A fizikai memória nyilvántartása
 - Lapkeret-adatbázis
 - A memóriakezelést támogató listák