HÁLÓZATI RENDSZEREK ÉS SZOLGÁLTATÁSOK TANSZÉK

MŰEGYETEM 1782

Budapest,
2024.05.22.

SZÁMÍTÓGÉP ARCHITEKTÚRÁK

Védelem

Horváth Gábor, Belső Zoltán

BME Hálózati Rendszerek és Szolgáltatások Tanszék
ghorvath@hit.bme.hu, belso@hit.bme.hu

A VÉDELEM CÉLJA

- Cél:

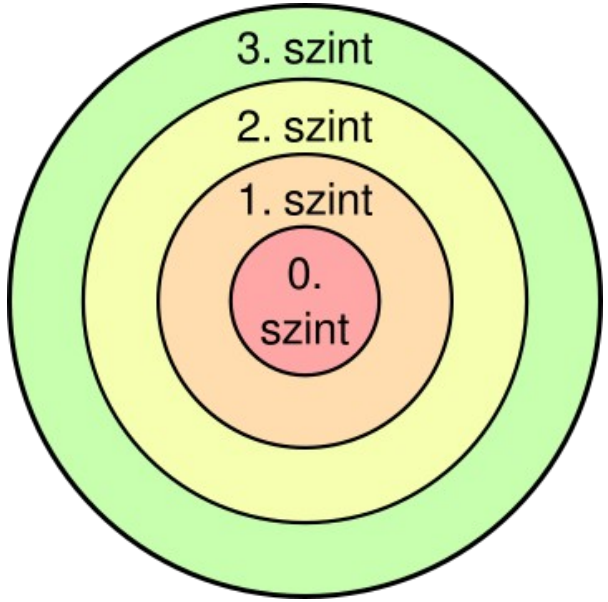
  A rendszer működésének, integritásának garantálása

- Akkor is, ha hibás, vagy rosszindulatú
  - Taszkok,
  - Hardverek

  vannak a rendszerben!
- Védelem = korlátozás
  - Taszkok korlátozása:
    - Más taszk vagy op. rendszer adatait ne érje el
    - Más taszk vagy op. rendszer privát függvényeit ne hívja
    - Ne kommunikáljon perifériával az op. rendszert megkerülve
  - Hardver korlátozása:
    - DMA-val ne firkálja össze a memóriát

PRIVILÉGIUMI SZINTEK

- Privilégiumi szint = mennyi jogosultsága van egy taszknak
  - =0: a lehető legtöbb
  - Minél nagyobb, annál kevesebb
  - Min. 2 kell belőle (pl. x86-nak 4 van)
- Mit befolyásol?
  - A taszk milyen utasításokat használhat
  - Milyen memóriaterületet érhet el
  - Milyen függvényeket hívhat meg
  - Milyen perifériákat használhat



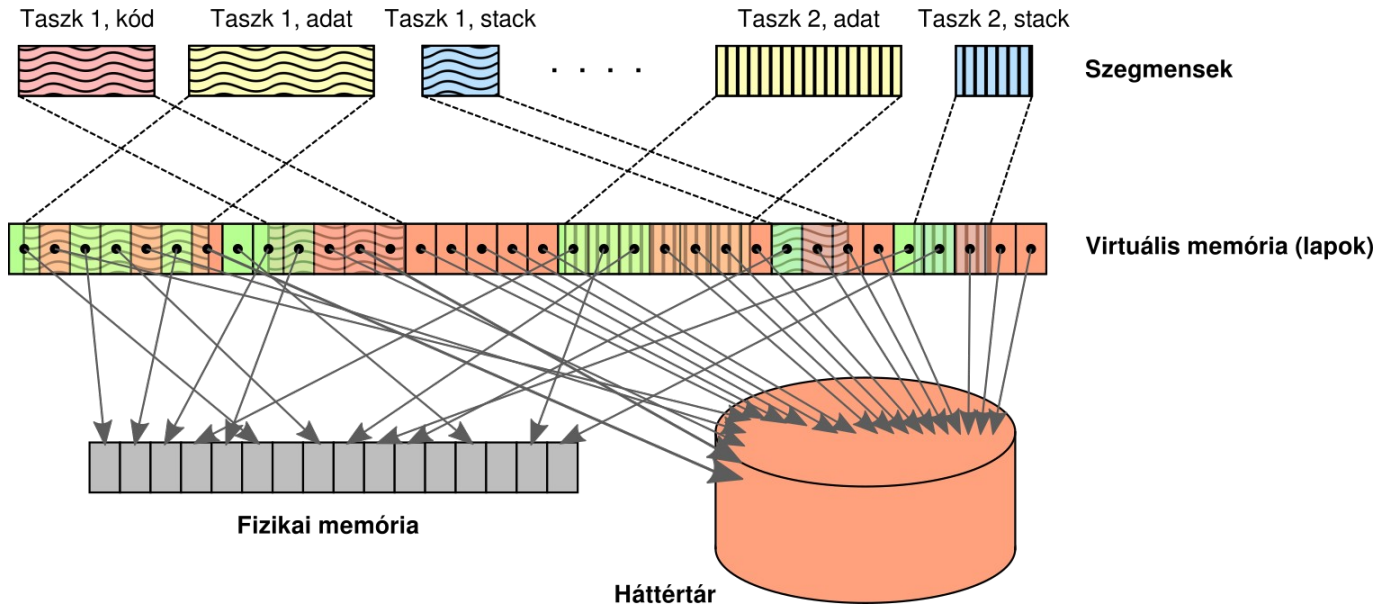
Memóriavédelem

MEMÓRIAVÉDELEM SZEGMENSSZERVEZÉSSEL

- Mi az a szegmensszervezés?
  - Logikailag egybetartozó objektumok
  - A virtuális memóriában
    - bárhol kezdődhetnek
    - folytonosak
    - változó hosszúak
- **Kétlépcsős címfordítás**
  - Szegmens → virtuális memória (szegmensleíró tábla)
  - Virtuális memória → fizikai memória (laptábla)

MEMÓRIAVÉDELEM SZEGMENSSZERVEZÉSSEL

- Mi az a szegmensszervezés?



- Szegmensen belül: közeli címzés / ugrás
- Szegmenseken át: távoli címzés / ugrás
- X86 / 32 bites mód: taszkonként 3 szegmens:
  - Kód / adat / stack
  - Windows/linux nem használja
- X86 / 64 biten: gyakorlatilag megszünt

MEMÓRIAVÉDELEM SZEGMENSSZERVEZÉSSEL

- Memóriavédelem alapja: **címtér elkülönítés**
  - Minden taszk és az op. rendszer külön szegmensben
- Privilégiumi szintek:
  - Minden taszknak van
  - Minden szegmensnek van → szegmensleíróban
- Védelmi szabályok memóriaműveletekre:
  - Taszk PL ↔ elérni kívánt szegmens PL összehasonlítása
  - Egyenlő vagy kevésbé privilegizált szegmens írható/olvasható
  - Magasabban privilegizált szegmens nem érhető el
- Ha sérül a szabály:
  - Operációs rendszer megkapja a vezérlést
  - Terminálja a taszkot

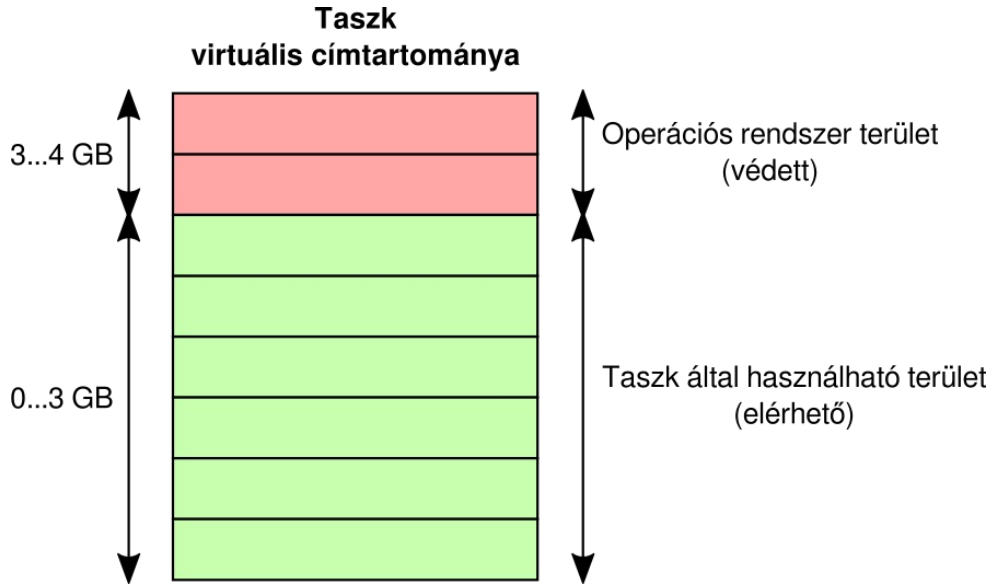
MEMÓRIAVÉDELEM KIZÁRÓLAG LAPOZÁSSAL

- Taszkváltás → laptábla váltás
- Minden taszk külön címteret kap
  → nem tudja elérni mások memóriaterületét!

- Gond: mindig elérhetőnek kell lennie
  - A perifériamegszakításokat lekezelő szubrutinoknak
  - Az operációs rendszer szubrutinjainak (rendszerhívás)
- Megoldás: a címtér kettébontása
  - Alsó rész: a taszké (taszkváltáskor cserélve)
  - Felső rész: az op. rendszeré (állandó)
- Gond: taszk látókörébe került a védendő terület
  → nem firkálhatja össze?
- Megoldás: új bit a laptábla bejegyzésekbe:
  - **User/supervisor bit**
    - Ha =1, a lapot csak az op. rendszer (0. szint) érheti el
    - Ha =0, akkor bárki

MEMÓRIAVÉDELEM KIZÁRÓLAG LAPOZÁSSAL

- Címtér kettéosztása:



- Windows 32 bit: 2 GB – 2 GB (opcionálisan 3 GB – 1 GB)
- Windows 64 bit: user 8 TB-ot kap
- Linux: kernel fordítási paraméter

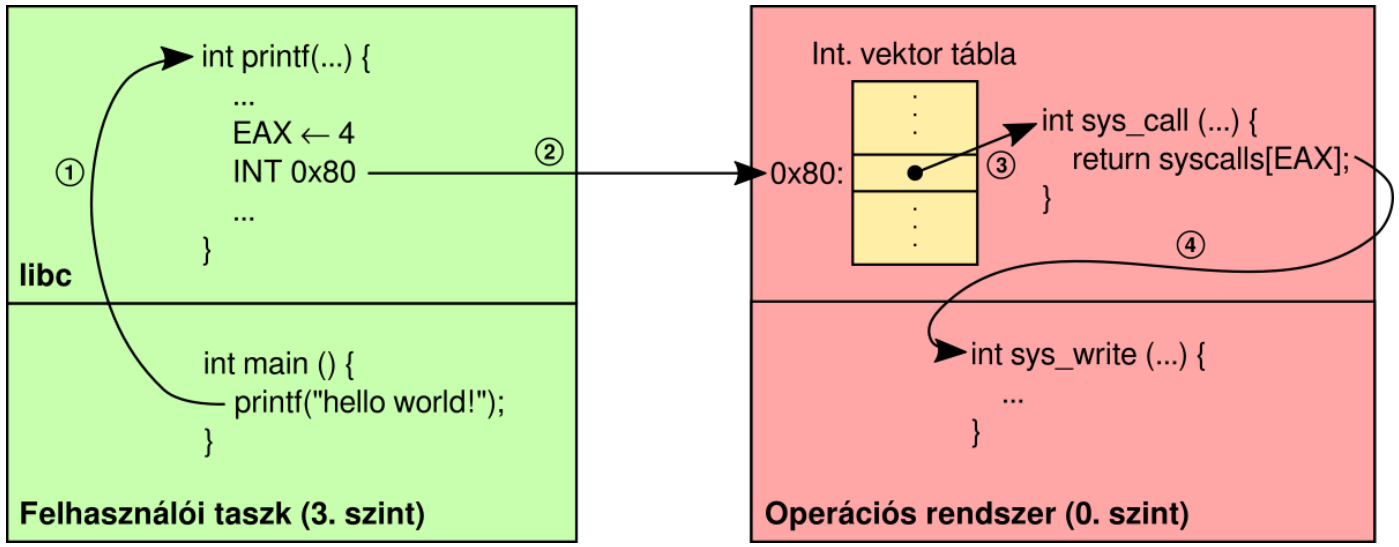
A vezérlésátadások ellenőrzése

VEZÉRLÉSÁTADÁS SZOFTVER INTERRUPTTAL

- Hogy kezeli le az op. rendszer a perifériák eseményeit?
  - **Interruptokkal**
  - A hardver váltja ki, a CPU egy lábán keresztül
- Hogy kezelje le az op. rendszer a taszkok kéréseit?
  - **Interruptokkal**
  - A szoftver váltja ki, egy utasítás meghívásával
- Interrupt vektor tábla:
  - Az interruptokat lekezelő szubrutinok kezdőcímei
  - x86: 256 elemű, ARM: 8 elemű
  - Mindet ki lehet váltani szoftveresen is!
  - x86: `INT` utasítás, ARM: `SWI` utasítás
- SW interrupt vs. függvényhívás:
  - Interruptkor a CPU PL=0-ba kapcsol(hat)!
- Ötlet:
  - Tegyünk félre 1-2 megszakítást az op. rendszer hívásai számára!
  - A taszk hívjon interruptot, ha akar valamit
  - Az interrupt PL=0-ba vált, és el tudja érni az op. rendszer területén a kívánt függvényt

VEZÉRLÉSÁTADÁS SZOFTVER INTERRUPTTAL

- Példa:



- A gyakorlatban:
  - Régen: Windows: 0x2e-s, Linux: 0x80-as megszakítás
  - Most: sysenter/sysexit
    (ugyanez gyorsabban)

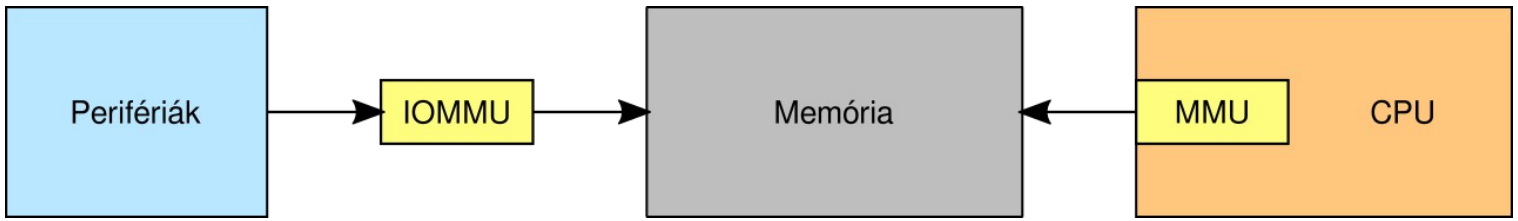
Védelem és a perifériakezelés

PERIFÉRIÁKHOZ VALÓ HOZZÁFÉRÉS

- Korlátozni kell a perifériákhoz való hozzáférést
  - Ha rosszul állítja be a taszk, össze DMA-za a memóriát
- Memóriára leképzett perifériakezelés esetén:
  - Az op. rendszer beállítja a „supervisor” bitet arra a területre
- Szeparált I/O utasítások esetén:
  - Az I/O műveletek privilegizálttá tehetők
    → Durva korlátozás, a taszk ki sem adhat I/O utasítást
  - Az elérhető I/O címek felsorolása
    → Finom felbontás, címenként szabályozható

A PERIFÉRIÁK KORLÁTOZÁSA

- Memória védelme a taszkokkal szemben:
  - MMU feladata
  - Laptábla bejegyzések védelmi információja alapján
  - Ha egy keretet nem fedünk le lappal → nem érhető el a taszknak
- Memória védelme a perifériákkal szemben:
  - Alapesetben semmi!
  - Fejlett rendszerekben **IOMMU**
  - Perifériák is virtuális címtérben dolgoznak
  - A lapokhoz védelmi információk köthetők



HÁLÓZATI RENDSZEREK
ÉS SZOLGÁLTATÁSOK
TANSZÉK

MŰEGYETEM 1782