Operációs rendszerek

Folyamatok kezelése a UNIX-ban

Folyamatok a UNIX-ban

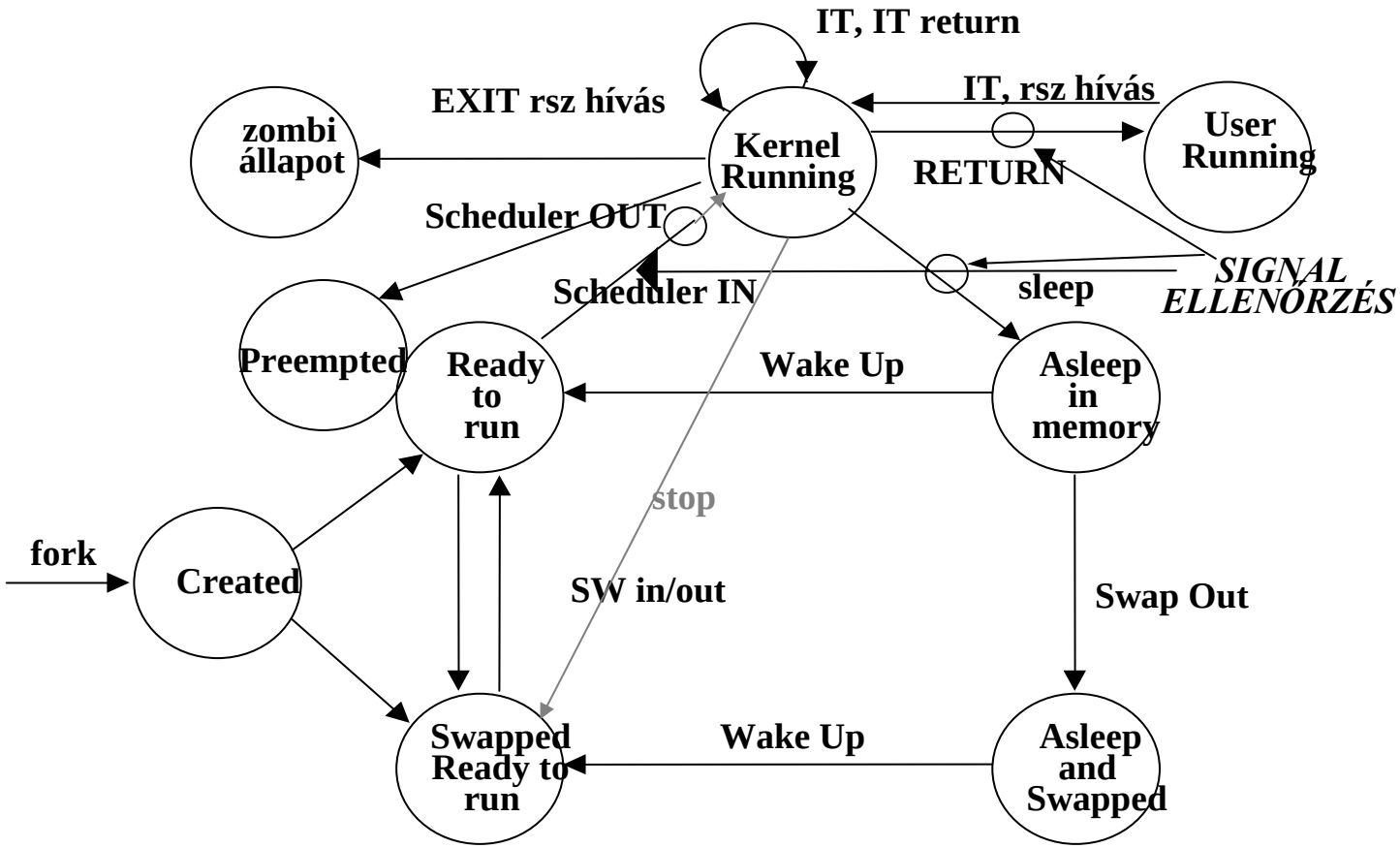
- A folyamat: multiprogramozott operációs rendszer alapfogalma - absztrakt fogalom.
- A gyakorlati kép: egy program végrehajtása és a végrehajtási környezet biztosítása.
- A folyamat virtuálisan egyedül fut a rendelkezésre álló hardveren.
- Hardver erőforrások elérése:
  - az operációs rendszeren keresztül.
- Műveletek:
  - erőforrás igénylés, várakozás.
- Operációs rendszer feladata:
  - összehangolja az egyes folyamatok perifériás műveleteit,
  - elosztja a rendelkezésre álló hardver erőforrásokat.

Felhasználói folyamatok

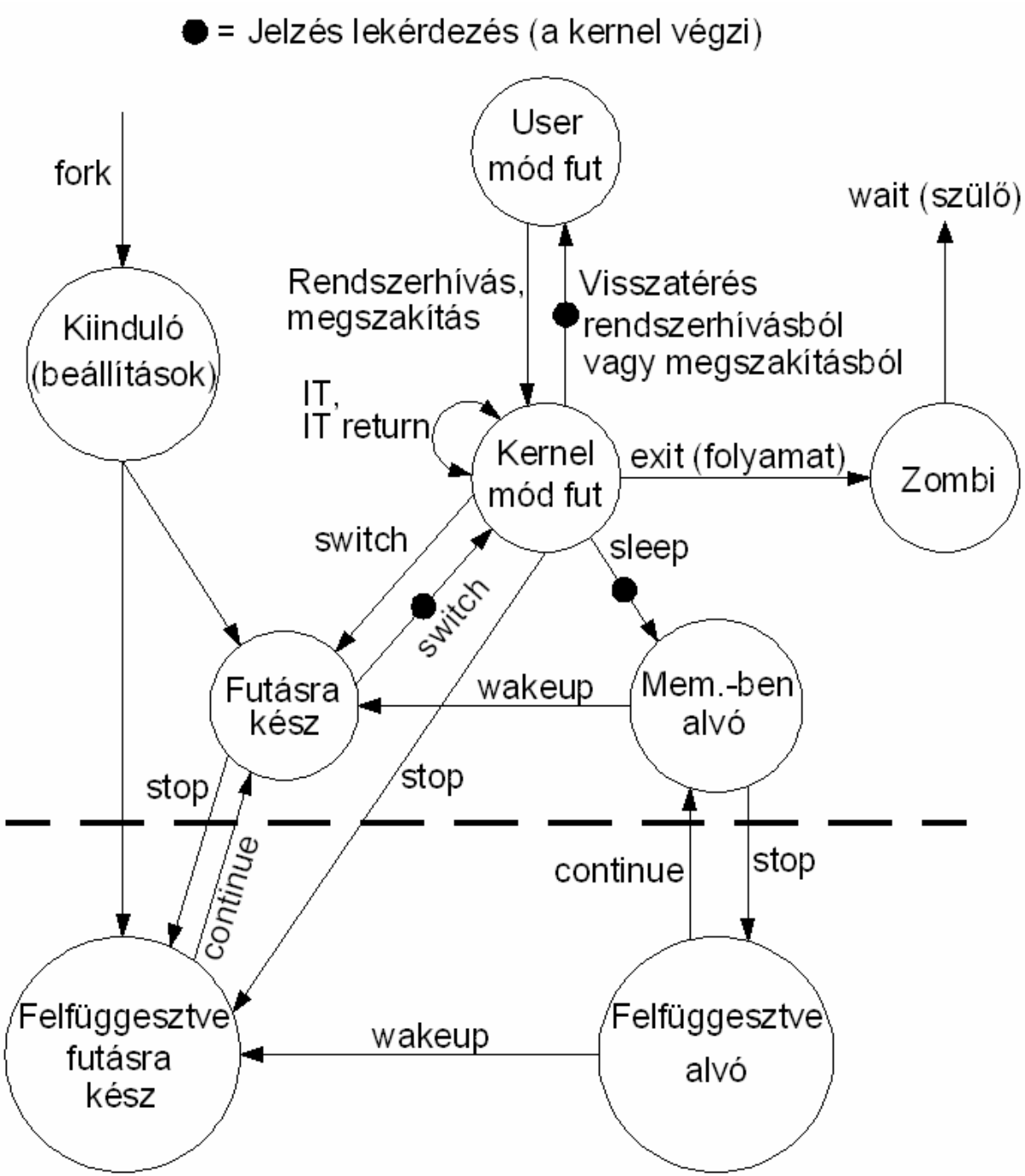
- Folyamatonkénti 1 szülő és 1 vagy több gyerek (*fork*).
- Az ősszülő az *init*, a rendszer indulásakor jön létre.
- Az árva folyamatokat az *init* örökli.
- Könnyűsúlyú folyamatoknál csak részleges környezetváltás van (*stack* terület)!

Folyamatok állapot-átmeneti diagramja

Folyamatok állapot-átmeneti diagramja…



… azaz:


● = Jelzés lekérdezés (a kernel végzi)
User mód fut
fork
wait (szülő)
Rendszerhívás, megszakítás
Visszatérés rendszerhívásból vagy megszakításból
Kiinduló (beállítások)
IT, IT return
Kernel mód fut
exit (folyamat)
Zombi
switch
sleep
switch
Futásra kész
wakeup
Mem.-ben alvó
stop
stop
continue
continue
stop
Felfüggesztve futásra kész
wakeup
Felfüggesztve alvó

Folyamatok állapot-átmeneti diagramja II.

Új folyamat keletkezése:

- *fork()* rendszerhívás:
  - gyermek folyamatnak:
    - *fork()* visszatérési értéke: 0,
  - szülő folyamatnak:
    - *fork()* visszatérési értéke: gyerek PID-je.
- Ősfolyamat: *init* (PID=0).
- *Scheduler* (ütemező, PID=1).

Folyamat indulása

```
int result=fork();
if(result==0)/*visszatérési érték vizsgálata*/
{
  printf("GYERMEK VAGYOK");
  exec("gyermek_kód",...);
  /*a gyermek végrehajtása*/
}
else
{
  if(result<0)
  {
    printf("HIBA");
  }
  else
  {
     printf("Anya proc:gyermekem PID-je:%d",result);
     /*ha a visszatérési érték >0, kiírja */
  }
}
```

Folyamatok állapotai I.

- *Created* (létrehozott) állapot.
- *Ready to run* (futásra kész) állapot.
- *Kernel running.*
- *User running*.
- *Asleep in memory* (memóriában alvó):
  - *sleep* rendszerhívás,
  - *wake up* átmenet.

Folyamatok állapotai II.

- *Swapped out:*
  - *swapped out and asleep,*
  - *swapped out and ready to run.*

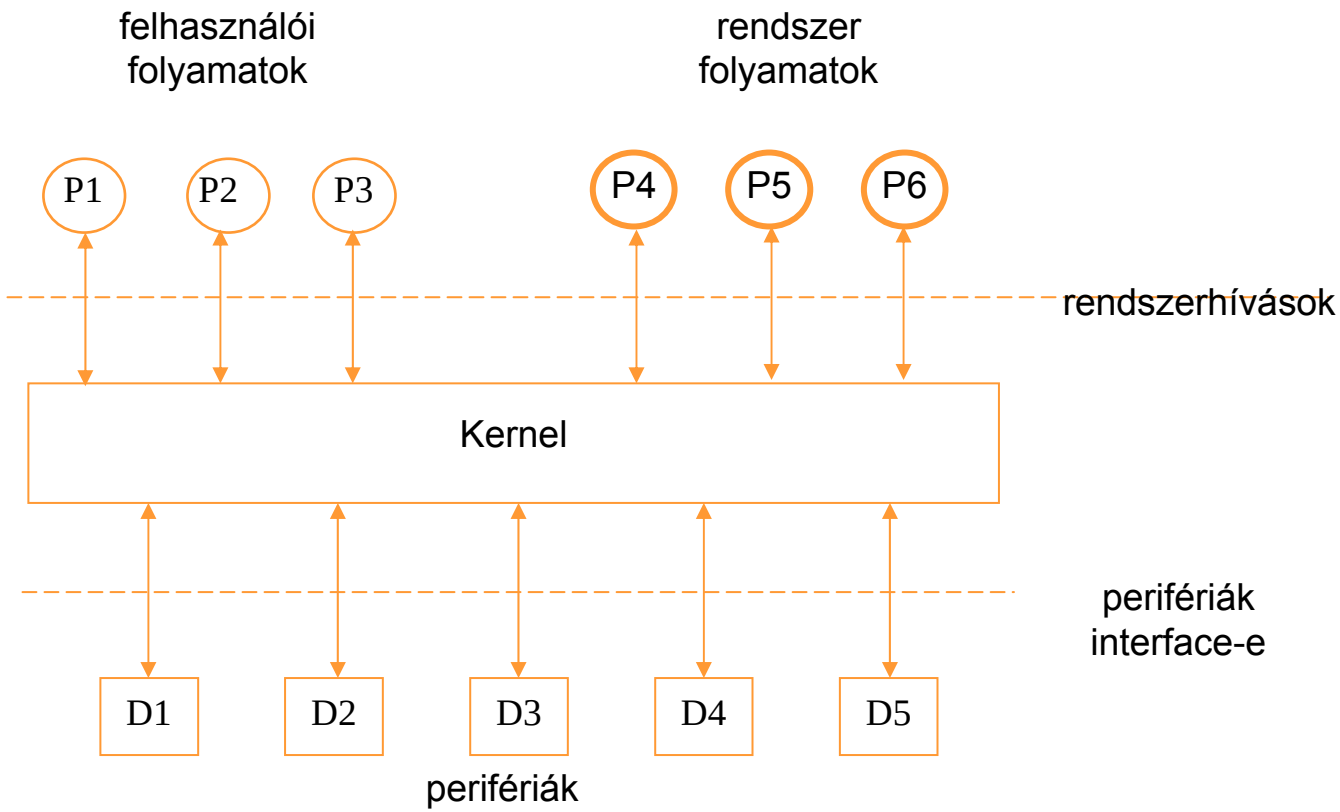
Preemptív ütemezés:

- *kernel running → ready to run,*
- *preemptív* (erőszakosan megszakított, kiszorított) állapot.

Futás befejezése: *zombie* állapot.

Folyamatok és a rendszer kapcsolata

Folyamatok és a rendszer kapcsolata



Interface-ek

OPR interface-ek:

- Programozói interface (rendszerhívások):
  - folyamat → OPR.
- Periféria interface:
  - OPR →I/O eszközök (perifériák).

Folyamatok párhuzamos végrehajtása

- Egy fizikai processzor.
- Virtuális párhuzamos végrehajtás.
- CPU ütemezés:
  gyors váltás, futó folyamatok számára szinte észrevétlen.
- UNIX:
  időszeletes (*time slice* vagy *quantum*),
  prioritásos, preemptív (kiszorításos).

Folyamatok környezete

- Követelmény:
  - folyamatok megszakítása, illetve újraindítása.
- Folyamatok környezete:
  a folyamat és az őt végrehajtó gép állapota (illetve annak leírása). Így például:
  - programszámláló (program counter),
  - regiszterek,
  - Memory Management Unit,
  - használt erőforrások.

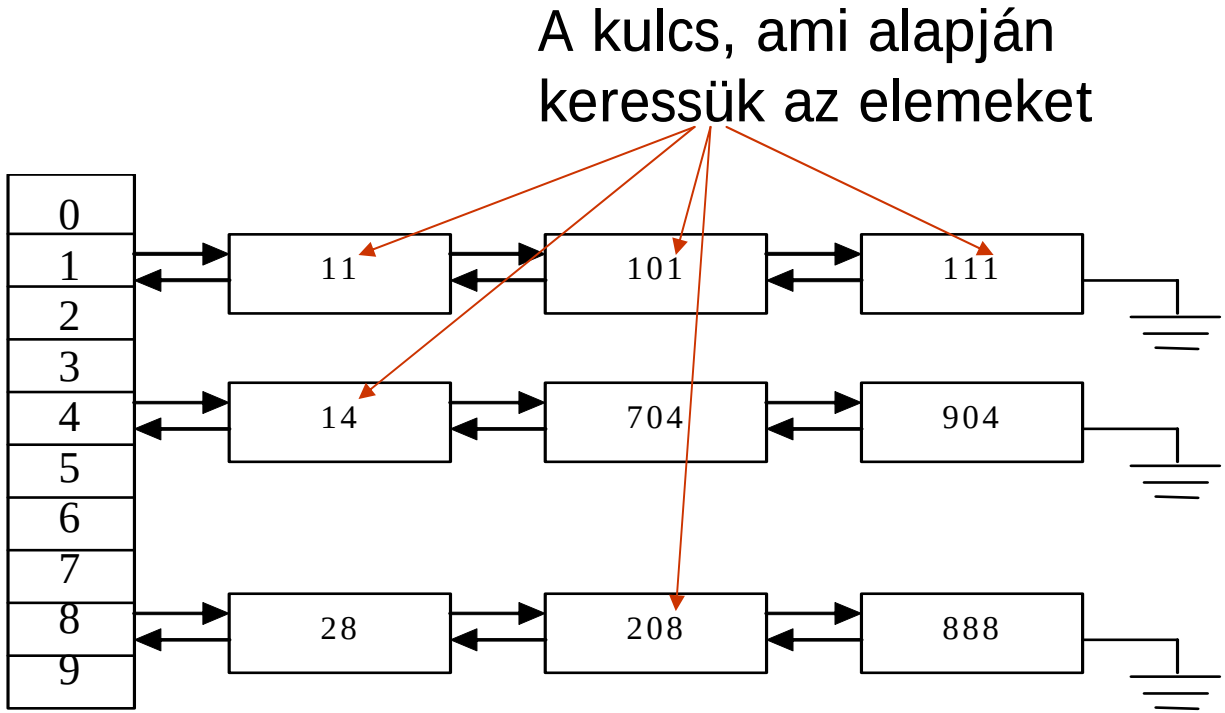
A folyamatok környezetének adatelemei a UNIX-ban

A folyamatok környezetének adatelemei

- Minden adat, ami szükséges a folyamat újraindításához.
- Kettő vezérlési adatstruktúrában tárolódik:
  - *proc* (processz) struktúra a rendszer memóriájában:
    - a kernel az összes folyamat *proc* struktúrájához hozzáfér,
    - tárolás: *hash* tábla,
  - *u area* (*user area,* felhasználói terület):
    a processz memóriájában, de csak (!) kernel módban módosítható:
    - a kernel csak a futó folyamat *u area*-jához fér hozzá közvetlenül,
    - indirekt módon a többit is eléri, de jóval lassabban.

Dinamikus adatszerkezetek

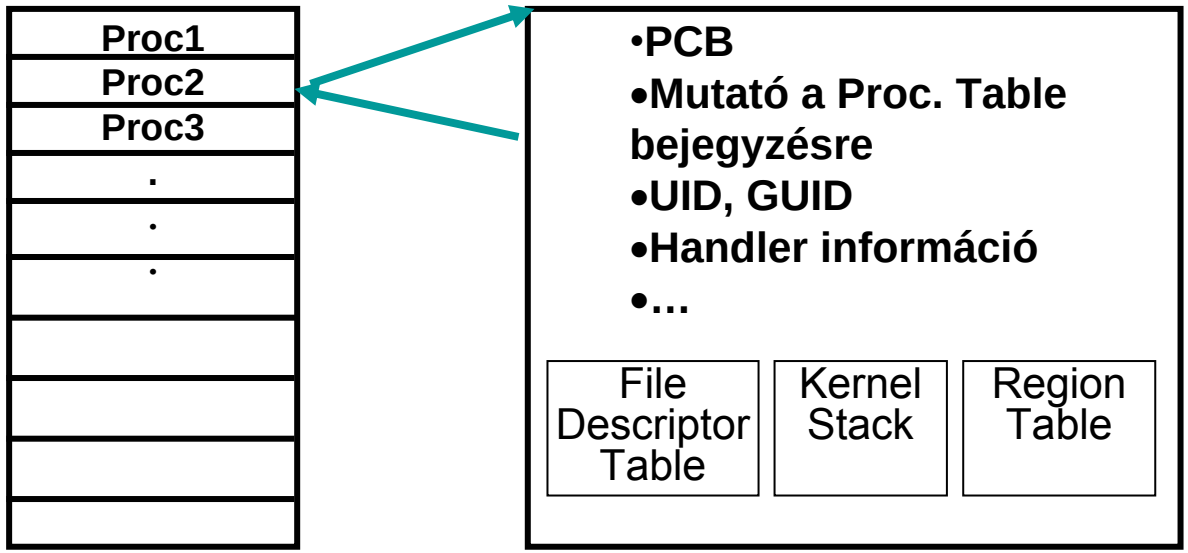
- Láncolt lista.
- Hash tábla.



Folyamat környezet leíró adatok

Processz-táblázat
(bővíthető)

Proc2-höz tartozó u area



A proc struktúra elemei I.

- Processz azonosító (PID).
- User area-ra mutató referencia.
- Processz állapota:
  - pl.: kernel vagy user módban fut, várakozó vagy futásra kész (állapot diagram).
- Ütemezési információk:
  - pl.: *p_cpu*, *p_usrpri.*
- Ütemezésnél használt egyéb mutatók:
  - pl.: milyen prioritási kategóriában van.

A proc struktúra elemei II.

- Signalok kezelésére vonatkozó információ:
  - *ignored-blocked-posted-handled.*
- MMU információ:
  - milyen MMU regisztereket kell frissíteni a folyamat futtatásakor.
- A processzek különböző listáihoz (aktív, zombie, szabad) tartozó link-ek.
- Jelzőbitek.

A proc struktúra elemei III.

- A PID alapján a *hash* táblában történő tároláshoz szükséges mutatók.
- A processzek hierarchiájában elfoglalt helyre vonatkozó információ:
  - pl.: processz szülője, processz gyermekei.

A *user area* elemei I.

Egyszerű adatelemek:

- *PCB* (hardver környezet, regiszterek).
- Mutató a folyamat proc struktúrájára.
- Jogosítványok (valós {signal} és hatásos {file} UID, GID).
- *Signal handler* (kezelő) rutinok címei.
- CPU használati statisztika (ütemezéshez).
- Környezeti változók (pl.: aktuális könyvtár, kontroll terminál, keresési út).

A *user area* elemei II.

Összetett adatelemek:

- A folyamathoz tartozó memória (a felhasználói címtartomány) elérésére szolgáló táblázat (címtranszformációs tábla, per-process ***Region Table***).
- File-leírók táblázata (per process ***File Descriptor Table***).
- Kernel módban használt veremtár (per process ***Kernel Stack***).

Összetett adatelemek

- *Kernel Stack:*
  - verem tár,
  - kernel (privilégizált) módban futáskor,
  - (pl.: rendszerhívások, IT).
- *Region Table:*
  - MMU címtranszformációs táblázat.
- *File Descriptor Table:*
  - a folyamat által megnyitott file-ok eléréséhez szükséges információ tárolása.

A hardver környezet (PCB)

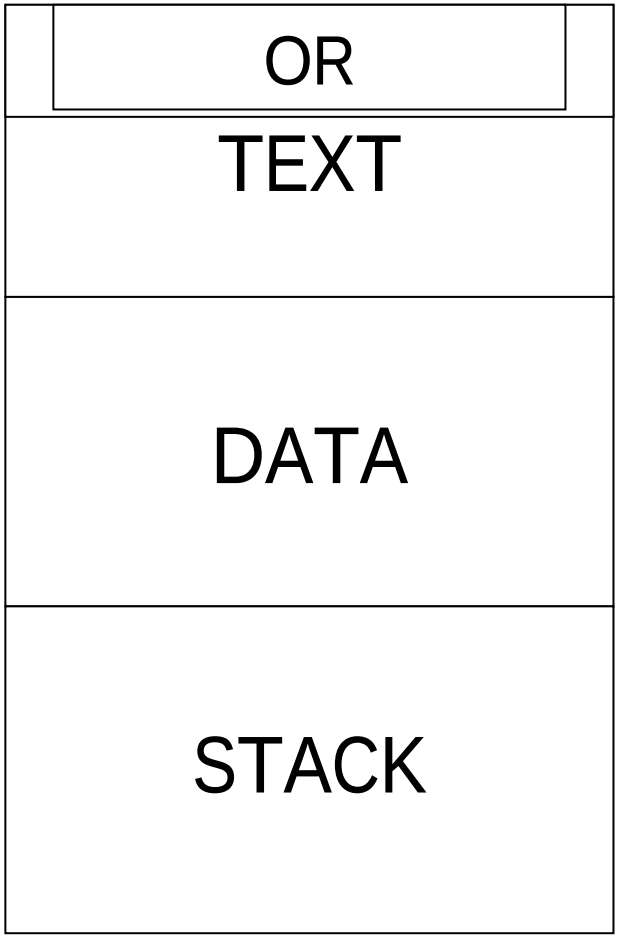
- CPU regiszterei.
- Utasításszámláló (PC).
- Stack pointer (SP).
- Processzor-állapotleíró bit-ek (process status word), pl. carry (átvitel) bit.
- Memóriakezelő egység regiszterei (Memory Management Units, MMU).
- Lebegőpontos egység (Floating Point Units) regiszterei.

Folyamatok memóriája a UNIX-ban

Folyamatok memóriája

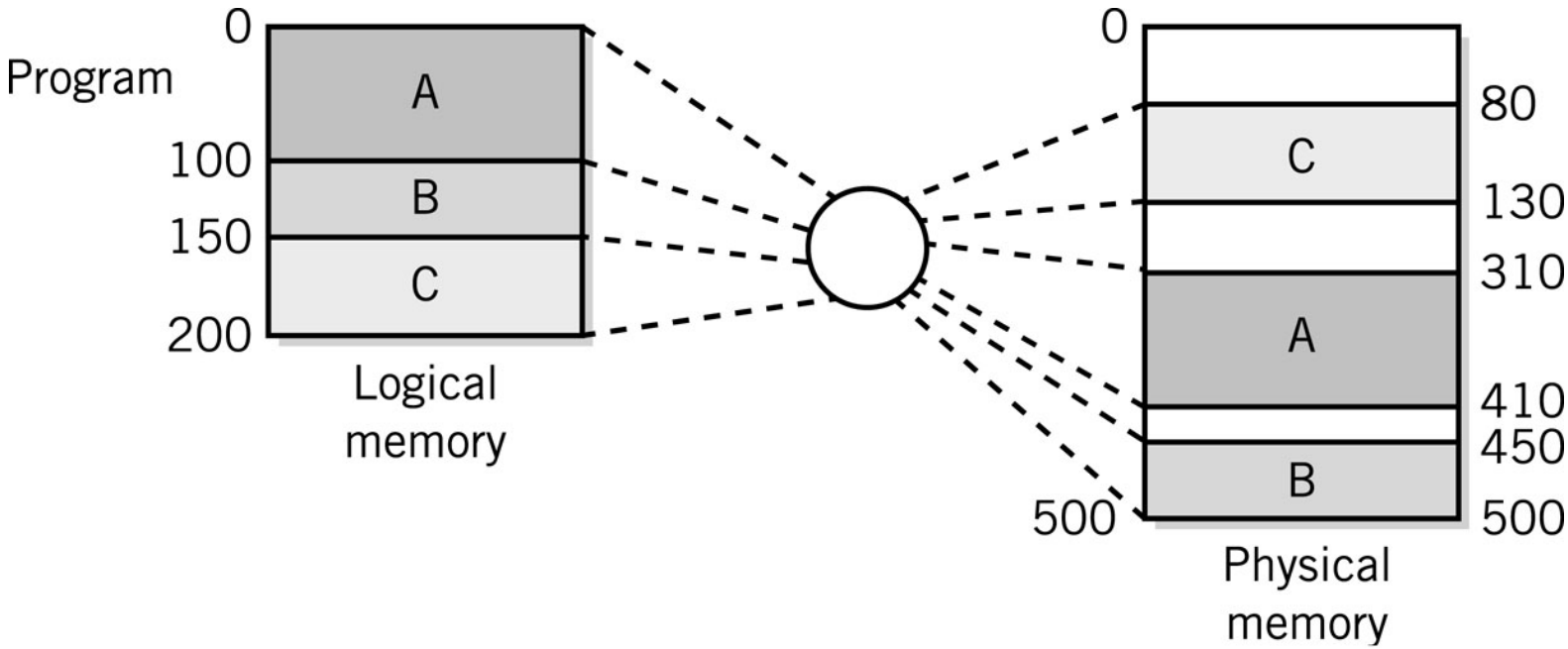
- Egyszerre több folyamat a fizikai memóriában.
- Fogalmak:
  - logikai memóriakép,
  - fizikai memóriakép,
  - virtuális memóriakezelés/tárcsere.
- Az OPR feladata:
  - memória cím–memória tartalom összerendelés,
  - memória tartalom fizikai memóriába mozgatása.

Memória használat

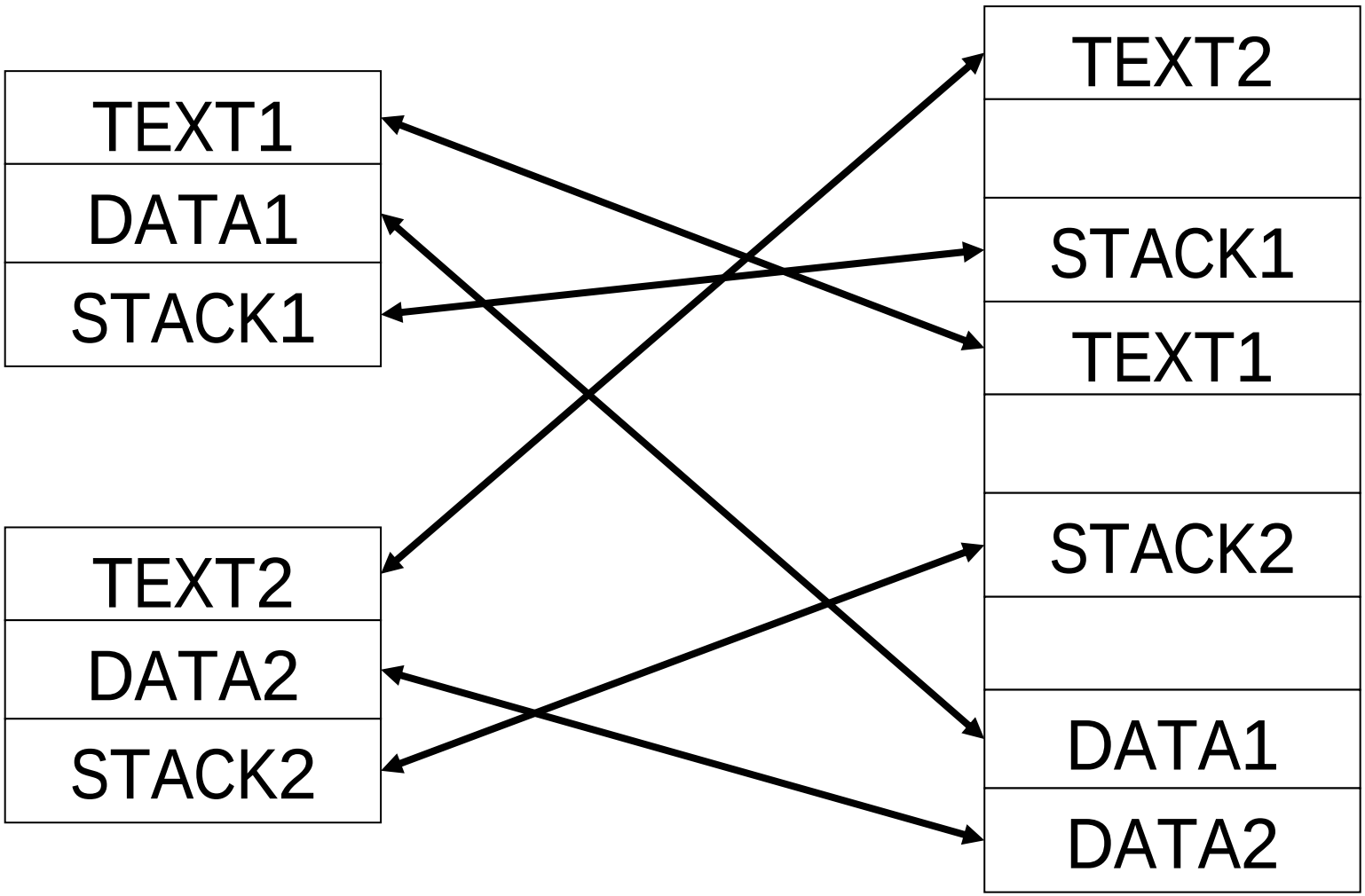


- Program kód:
  - statikus,
  - dinamikus (DLL).
- Adatterületek:
  - inicializált (DATA),
  - nem inicializált (BSS),
  - heap (dinamikus adatok).
- Veremtár:
  - dinamikusan növelhető terület.

Logikai-fizikai címtranszformáció



Logikai-fizikai címtranszformáció



UNIX memória kezelés

- Logikai – fizikai címtranszformáció:
  - folyamat leíró adatokban:
    - Region Table,
    - Memory Management Unit.

A UNIX OPR feladatainak a megvalósítása

Az OPR feladatai

- A UNIX, mint operációs rendszer a feladatait a következő négy formában látja el:
- Kernel:
  - rendszerhívások *(system call*),
  - kivétel *kezelés (exception)*,
  - hardver megszakítások *(hardware interrupt),*
  - *call-out* függvények.
- Önálló folyamat:
  - *daemon* folyamatok.

Rendszerhívások

- UNIX programozói interface-e.
  (*API: Application Programming Interface.)*
- Az OPR szolgáltatásainak elérése:
  - erőforrás foglalás,
  - rendszerinformációk gyűjtése, elszámolás biztosítása,
  - védelmi és biztonsági mechanizmusok biztosítása.
- Privilegizált futási mód.
- Rendszer és folyamat erőforrásainak használata.

Kivétel (*exception*)

- Pl.: a nullával történő osztás, memóriaterületről történő kicímzés.
- Utasítás végrehajtással szinkron események.
- A kivételek kezelése kernel feladat.
- HW eszköz(ök) által generált megszakítás(ok).

Hardver megszakítások (*hardware interrupt*)

- Perifériák (I/O egység) állapotának megváltozása.
- IT kezelés kernel feladat (csak rendszer erőforrásokkal!).
- Aszinkron módon fellépő eseményekre a rendszer reagálni tudjon:
  - interrupt rutin lefuttatása,
  - nincs környezetváltás (*context switch*).

Az interrupt rutin által végzett tipikus műveletek

1. Végrehajtási mód váltása: user→ kernel.
2. Elmenti az aktuális processzor környezetet.
3. IT kiváltó esemény azonosítása.
4. IT kiszolgáló rutin meghatározása.
5. IT rutin hívása.
6. Környezet visszaállítása.
7. Végrehajtási mód váltás: kernel → user.

Óra-interrupt

- Az ütemező hardver-támogatása:
  - az óra-interrupt: a hardver óra adott időközönként megszakítást okoz.
- Gyakorisága: a *tikk* (óra-tikk, CPU-tikk):
  - 10 ms (rendszerfüggő).

Az óramegszakítás-kezelő fő feladatai I.

- A hardver óra újraindítása, amennyiben ez szükséges.
- Az aktuálisan futó folyamat:
  - CPU használati statisztikájának frissítése,
  - a prioritás újraszámítása,
  - az időszelet lejártának kezelése.
- *SIGXCPU signal* küldése az aktuális folyamatnak, amennyiben az túllépte az idő-kvótáját.

Az óramegszakítás-kezelő fő feladatai II.

- A rendszer óráinak frissítése (napi óra).
- A *call-out* függvények ellenőrzése.
- Bizonyos rendszerfolyamatok kezelése (swapper, pagedaemon).
- Riasztások (*alarm*) kezelése.

Daemon folyamatok

- Önálló folyamatokként megvalósított OPR funkciók.
- Nem privilegizált módban futnak.
- Rendszerhívásokon keresztül érik el a kernel szolgáltatásait.

Folyamatok futási környezete I.

- Folyamatok logikai memóriaképe:
  - az operációs rendszer által használt memória,
  - a folyamat által használt memória.
- Memória védelme:
  - a rendszer környezet (*system context*) és
  - a felhasználói környezet (*user context*) megkülönböztetése Java VM (hordozható!), PC emulátorok.

Folyamatok futási környezete II.

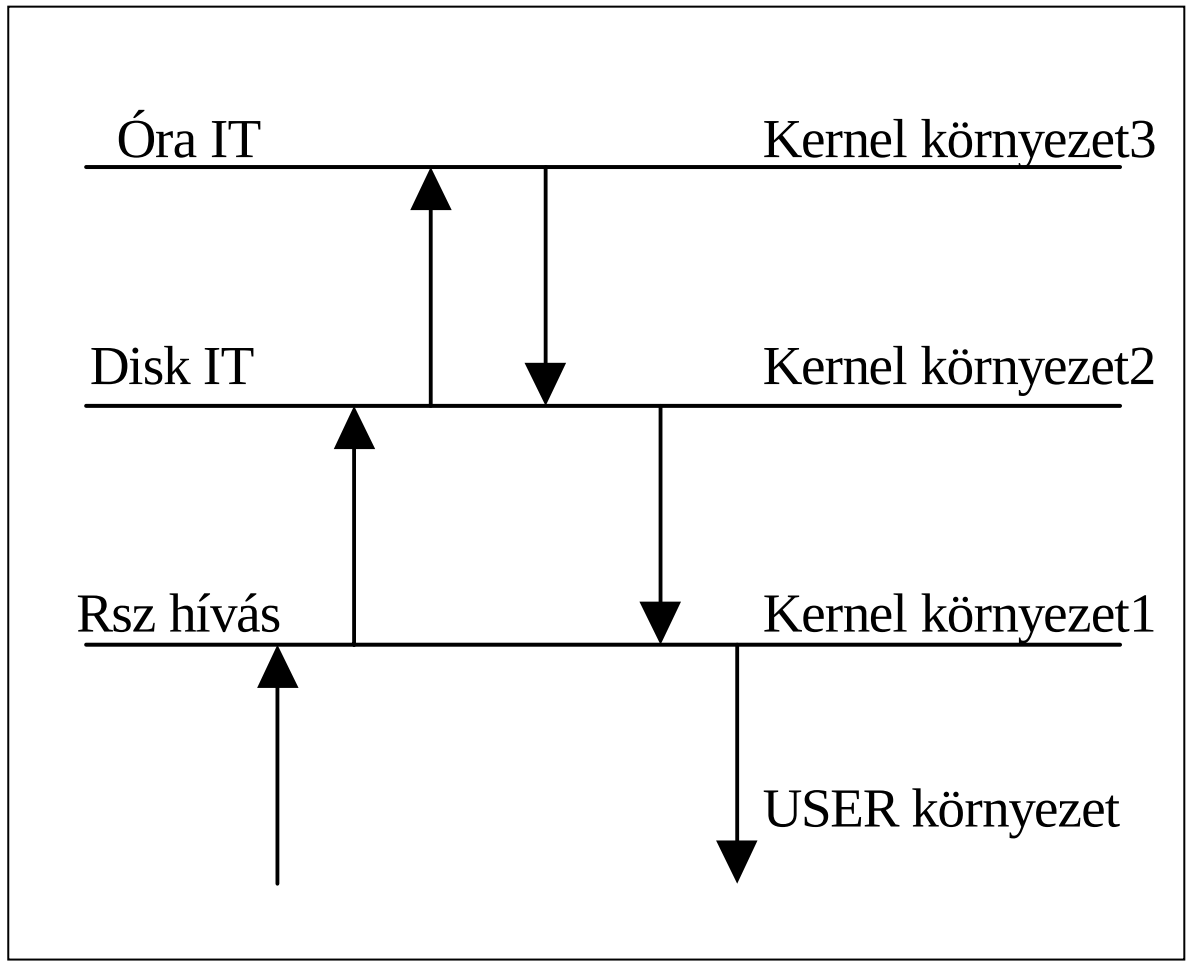
- Rendszer futási környezet:
  - az operációs rendszer memóriaterületei,
  - a kernel erőforrásai,
  - a kernel stack.
- Felhasználói futási környezet:
  - az alkalmazás adatterületei,
  - user stack.

Folyamatok futási környezete III.

- A processzor különböző utasítás végrehajtási módjai:
  - A felhasználói (nem privilegizált) mód (user mode):
    - néhány speciális utasítás végrehajtása és a memória adott részének elérése tiltott.
  - A privilegizált, kernel mód (kernel mode).

Folyamatok futási környezete IV.

Környezetek érvényessége



Folyamatok kezelése
Összefoglalás

Létrehozás

- A *fork* rendszerhívás kiadása a szülő által.
- Területfoglalás a háttértáron.
- PID generálás a gyerekfolyamatnak.
- Proc struktúra foglalás és inicializálás.
- Címleképezési táblák allokálása.
- U area frissítése, - új címleképezési táblák.
- Osztottan használt kódtartomány kezelése.
- A szülő verem- és adattartományainak duplikálása.
- Osztottan használt erőforrások kezelése.
- HW környezet inicializálása, a szülőtől másolva.
- A folyamat ütemezési sorba helyezése.
- A gyereknek 0-t, a szülőnek a gyerek PID-t adja vissza, mint visszatérési értéket.

Befejezés

- Az *exit* rendszerhívás kiadása a gyerek által.
- *Signal*-ok kikapcsolása.
- Nyitott állományok lezárása.
- Erőforrások felszabadítása.
- Statisztika napló kitöltése.
- Erőforrás használati statisztikák és a kilépési kód elmentése a *proc* struktúrába.
- *Zombi* listára kerülés.
- Árvák ”átadása” az *init*-nek.
- A címtér felszabadítása.
- SIGCHLD *signal* küldése a szülőnek.
- A szülő felébresztése amennyiben alszik.
- A *swtch()* rendszerhívással az átütemezés kezdeményezése.