

A számítógépek felépítése 7.b: Input-output

Markó Tamás
PTE TTK, 2003

2003.10.27.

Markó Tamás, PTE TTK

1

A rádiótelefonokat kérem KIKAPCSOLNI!

2003.10.27.

Markó Tamás, PTE TTK

2

Input-output

- Input-output (IO, I/O): adatbevitel-adatkivitel: a processzor és a memória (a gép „belseje”) és a világ többi része között lebonyolított adatforgalom
- Alapvető módszerek:
 - Programozott bevitel-kivitel (programmed I/O)
 - Közvetlen memória-hozzáférés (direct memory access, DMA)
 - I/O-processzorok (input-output processor, IOP)

2003.10.27.

Markó Tamás, PTE TTK

4

Programozott I/O

Az alapelv:

- nincs közvetlen kapcsolat az operatív tár és a periféria között
- az adatátvitel a processzor egyik regisztere (pl. az akkumulátor) és az I/O-eszköz puffer-regisztere között bonyolódik (hasonlóan a processzor és az operatív tár közti adatforgalomhoz)
- az operatív tár, a processzor és a perifériák ugyanazon a buszon vannak
- a busz csatlakozóhelyeit I/O-portnak hívjuk

2003.10.27.

Markó Tamás, PTE TTK

5

Címzési módok

- Az I/O-portok címei a memória címtartományában vannak
 - „**memory mapped I/O**”
 - Példa: PowerPC
- Saját címtartomány
 - Példa: INTEL

2003.10.27.

Markó Tamás, PTE TTK

6

A programozott I/O parancsai

- A legegyszerűbb megoldás két parancs:
IN X és OUT X
 - IN X jelentése: $AC \leftarrow I/O_Port(X)$
 - OUT X jelentése: $I/O_Port(X) \leftarrow AC$
- Figyelem! A programozónak kell megkülönböztetni:
 - az adatokat
 - a vezérlő információkat (pl. az író-olvasó fej pozicionálása)
 - az állapot-információkat (pl. olvasási hiba)

2003.10.27.

Markó Tamás, PTE TTK

7

Ellenőrző (várakozó) ciklus

- Általában a perifériák állapot-információi mindig hozzáférhetők

- Így értelmes a következő ellenőrző ciklus:

```
loop
  begin
    read (status_info);
    ready := status_info = ready_felt;
  end;
until ready;
```

2003.10.27.

Markó Tamás, PTE TTK

8

Példa várakozó ciklusra (Intel)

- Intel 8080-as mikroprocesszor
 - állapot-információk az 1-es portról
 - adatok a 2-es portról
 - a készenléti állapot értéke a READY változóban

```
WAIT: IN 1          A := IO_PORT(1)
                        -- az állapot olvasása
      CPI READY     Z := A = READY
                        -- CPI = ComPare Immediate
                        -- Z = Zero flag
      JNZ WAIT      ugrás WAIT-re, ha nem kész
      IN 2          A <= IO_Port(2)
                        -- az adat olvasása
```

2003.10.27.

9

Példa blokkos adatátvitelre (Intel 8080)

A feladat: 100 bájt beolvasása a 7-es portról

```
LXI  H, 10      H||L := 10 (itt tároljuk az adatokat);
MVI  B, 100     B := 100;
LOOP: IN  7      A := IO_Port(7);
      MOV  M, A   MM[H || L] := A;
      INX  H      H||L := H||L + 1;
      DCR  B      B := B - 1;
      JNZ  LOOP   ugrás LOOP-ra, ha B ≠ 0
```

LXI: Load Extended Immediate

MVI: Move Immediate

2003.10.27.

Markó Tamás, PTE TTK

10

A megszakítás

- Kellemetlen a programozott I/O-ban:
 - az eszköz állapotát folyamatosan figyelni kell (várakozó ciklus)
- Jobb megoldás:
 - az eszköz saját maga jelentkezik, ha változik az állapota
 - a processzornak erre valahogy reagálnia kell
- A megszakítás (interrupt) egy ritkán és váratlanul fellépő esemény, aminek hatására a processzor
 - abbahagyja az aktuális program végrehajtását
 - és helyette egy speciális, a megszakítás lekezelésére szolgáló program végrehajtását kezdi meg

2003.10.27.

Markó Tamás, PTE TTK

11

A megszakítás alapötlete

- A processzorban van egy kívülről beállítható „Interrupt-Flipflop“ (IFF)
- Ha egy eszköz fel akarja hívni magára a figyelmet, beállítja az IFF-et
- A processzor minden utasítás-ciklusban (általában még a betöltési fázis előtt) ellenőrzi az IFF-et
- Ha be van állítva, akkor elmenti az eredeti állapotot és meghívja a kiszolgáló szubrutint (Interrupt Service Routine (ISR))

2003.10.27.

Markó Tamás, PTE TTK

12

A megszakításkezelő szubrutin hívása részletesebben

- A processzor azonosítja a megszakítás okát (de hogyan?)
- A processzor megállapítja a szükséges szubrutin címét (de hogyan?)
- Meghívja a szükséges szubrutint
- A szubrutin egyebek között az IFF-et is alaphelyzetbe állítja
- A megszakított program végrehajtása folytatódik

2003.10.27.

Markó Tamás, PTE TTK

13

Maszkolás

- Előfordulhatnak időintervallumok, amikor nem akarjuk, hogy meg lehessen szakítani a futó programot, pl.
 - az idő szempontjából kritikus szakaszok
 - megszakításkezelő rutinok
- Ezért a megszakítás időlegesen letiltható (maszkolható)
- Tipikus parancs: "disable interrupts"

2003.10.27.

Markó Tamás, PTE TTK

14

A megszakítás okának azonosítása - 1. megoldás

- Feltesszük, hogy 1 közös *interrupt flip-flop* van az összes eszköz számára
 - Nem a flip-flopokkal takarékoskodunk, hanem a processzorba menő vezetékekkel!
- A processzor rögzített sorrendben lekérdezi az összes eszközt, hogy ő kérte-e a megszakítást
- Az elsőként megtalált ilyen eszközt szolgálja ki
- Az eszközök prioritását a lekérdezési sorrend határozza meg

2003.10.27.

Markó Tamás, PTE TTK

15

A megszakítás okának azonosítása - 2. megoldás (daisy chain)

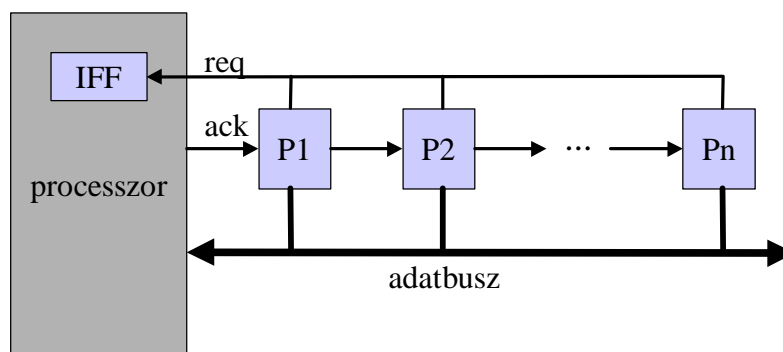
- Feltesszük: egy közös IFF minden eszköz számára
- A processzor „acknowledge” jelet generál, amit minden eszközön keresztülvezetünk
- Az első olyan eszköz, ami megszakítást kért és megkapja ezt a jelet, erre reagál:
 - a saját azonosítóját kiteszi az adatbuszra
 - a processzor „acknowledge” jelét nem adja tovább
- A prioritást az eszközök *daisy chain*ben elfoglalt helye szabja meg

2003.10.27.

Markó Tamás, PTE TTK

16

A *daisy chain* kapcsolás



2003.10.27.

Markó Tamás, PTE TTK

17

A megszakítás okának azonosítása - 3. megoldás (többszintű megszakítás)

- Feltesszük: minden eszköz számára saját IFF (luxusmegoldás)
- A processzor tudja, hogy honnan jön a megszakításkérés
- „Multilevel“ vagy „multiline“ megszakítás
- Keverék megoldás (ma ez az általános):
 - több IFF
 - minden IFF-re több eszköz *daisy chain*nel

2003.10.27.

Markó Tamás, PTE TTK

18

A megszakításkezelő szubrutin azonosítása

- 1. megoldás:
 - egyetlen szubrutin fix címen
 - például Z80, Interrupt Mode 1: a processzor mindig a 38h címre ugrik
- 2. megoldás:
 - az eszköz elküldi az ugrási címet az adatbuszon
 - „vectored interrupt“

2003.10.27.

Markó Tamás, PTE TTK

19

A vektoros megszakítás

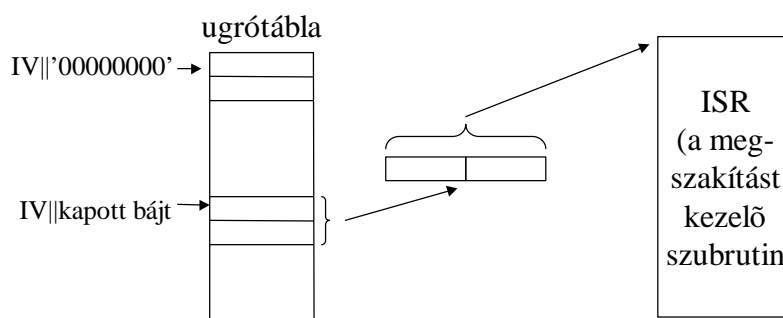
- Z80, Interrupt Mode 2
- Van egy IV (Interrupt Vector) regisztere
- IV || '00000000' egy ugrótábla kezdőcímére mutat az operatív tárban
- A processzorból jövő *Interrupt Acknowledge* jel után az eszköz egy bájtot tesz az adatbuszra
 - páros szám ? legfeljebb 128 különböző érték
- Ez index az ugrótáblába, az itt kezdődő 2 bájtos címre adódik a vezérlés

2003.10.27.

Markó Tamás, PTE TTK

20

A vektoros megszakítás szemléltetése



2003.10.27.

Markó Tamás, PTE TTK

21

Közvetlen memória-hozzáférés

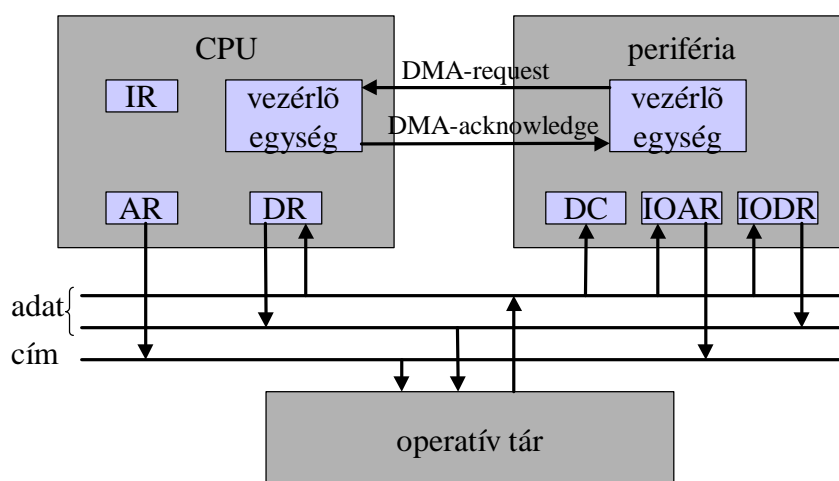
- Direct Memory Access, DMA
- Van egy DMA-egység (hardver), ami önállóan képes adatokat átvinni a perifériák és az operatív tár között
- Az adatátvitel megindításához a processzor közli a DMA-egységgel
 - az adatok kezdőcímét a memóriában
 - az átvendő szavak vagy bájtok számát
- Ezután a DMA-egység önállóan átviszi az összes adatot
- Az átvitel végén visszajelzés (megszakítás)

2003.10.27.

Markó Tamás, PTE TTK

22

Egy DMA-rendszer tipikus felépítése



2003.10.27.

Markó Tamás, PTE TTK

23

A DMA végrehajtásának lépései 1.

1. A CPU két I/O műveletet végez el:
 - IOAR := az adatok kezdőcíme a memóriában (IOAR: I/O Address Register)
 - DC := az átvendő adatok száma (DC: Data Counter)
2. Amikor az eszköz kész az átvitel megkezdésére:
 - DMA_REQUEST := '1'
 - a processzor szabaddá teszi az adat- és címbuszt
 - DMA_ACKNOWLEDGE := '1'
3. Megindul az adatátvitel az eszköz és a memória között. Minden átvitt adat után:
 - DC := DC - 1
 - IOAR := IOAR + 1

24

A DMA végrehajtásának lépései 2.

4. Az eszköz nem képes adatátvitelre, de DC ≠ 0 :
 - a DMA-vezérlő szabaddá teszi a buszokat és DMA_REQUEST := '0'
 - a CPU átveszi a buszok kezelését
 - DMA_ACKNOWLEDGE := '0'
 - ha az eszköz megint kész az átvitelre, folytatás a 2. lépés szerint
5. DC értéke 0 lesz:
 - visszajelzés a CPU-nak (interrupt)
 - a CPU lezárja az I/O-t vagy újabb DMA-átvitelt kezd

2003.10.27.

Markó Tamás, PTE TTK

25

A memória-hozzáférési konfliktusok feloldása

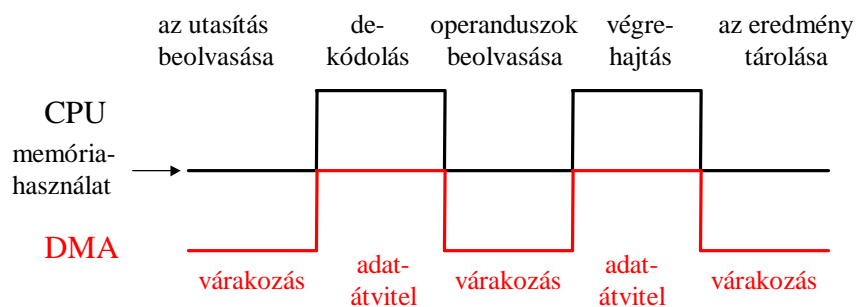
- Blokkos átvitel:
 - a DMA-nak prioritása van a CPU-val szemben
 - a CPU a teljes idő alatt el van tiltva a memóriától
- Cycle stealing:
 - a DMA-nak prioritása van a CPU-val szemben, de csak egy-egy ciklust „lop el” tőle
 - a DMA-vezérlő minden átvitt adat után szabaddá teszi a memória elérését
- Transparens mód:
 - a DMA azokat a ciklusokat használja ki, amikor a CPU nem használja a memóriát

2003.10.27.

Markó Tamás, PTE TTK

26

Példa a transzparens üzemmódra



2003.10.27.

Markó Tamás, PTE TTK

27

I/O-processzorok

- A DMA mentesíti a CPU-t az adatátviteltől, a perifériák vezérlésétől azonban nem
- Az I/O-processzor (csatornának is hívják) átveszi a teljes I/O-felügyeletet
- A CPU már csak kevés I/O-utasítást ismer, pl.:
 - STARTIO
 - STOPIO
 - TESTIO
- Az I/O-processzor is az operatív tárban lévő programot hajt végre (gyakran maga is egy „normális“ processzor)

2003.10.27.

Markó Tamás, PTE TTK

28

Az eszközvezérlő (device driver)

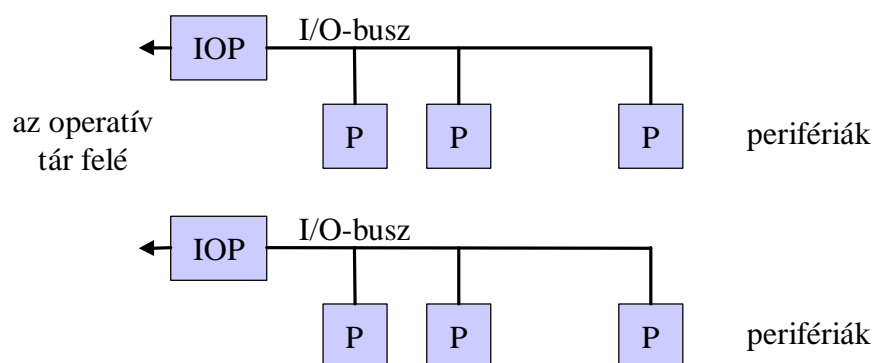
- Az IOP-programok eszközfüggők („device driver“)
- Tipikus IOP-parancsok:
 - átvitel (eszközcím, memória-cím, blokkméret, irány)
 - eszközvezérlés („rewind“, „seek address“, ...)
 - ugrások
 - aritmetikai műveletek

2003.10.27.

Markó Tamás, PTE TTK

29

Az I/O-processzorok és a perifériák kapcsolata 1.: busz

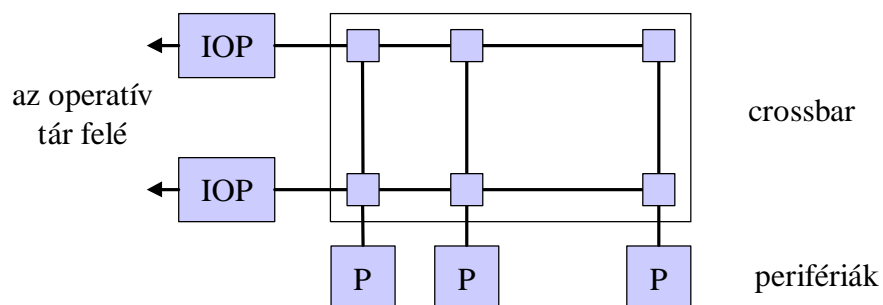


2003.10.27.

Markó Tamás, PTE TTK

30

Az I/O-processzorok és a perifériák kapcsolata 2.: crossbar



2003.10.27.

Markó Tamás, PTE TTK

31

Példa csatornára: az IBM nagygépei

- A CPU-ra 1 - 7 csatorna (IOP) kapcsolható
- A perifériák busszal csatlakoznak a csatornára
- A CPU 3 I/O-műveletet ismer:
 - SIO B1,D1
 - HIO B1,D1
 - TIO B1,D1
 - (TCH B1,D1)

IOP-cím: $(R(B1)+D1).(21:23)$ eszközcím: $(R(B1)+D1).(24:31)$
--

(SIO: Start I/O, HIO: Halt I/O

TIO: Test I/O, TCH: Test Channel)

2003.10.27.

Markó Tamás, PTE TTK

32

A csatornaprogram kezdőcíme

- A 72_{10} -es (**fix !**) memóriacímen található a „Channel Address Word“ (CAW)
- Ide írja a CPU a csatornaprogram kezdőcímét, és csak azután adja ki a SIO utasítást
- Igen rugalmatlan megoldás!!!

2003.10.27.

Markó Tamás, PTE TTK

33

A csatornaparancsok felépítése

- Channel Command Word (CCW)

műveleti kód	memória-cím	...	CDF	CCF
--------------	-------------	-----	-----	-----	------

- CDF (Chain Data Flag): ugyanazt a műveleti kódot kell alkalmazni a következő CCW-re is
- CCF (Chain Command Flag): a csatornaprogramnak még nincs vége további CCW-k következnek
- A csatorna leáll, ha se a CDF, se a CCF nincs 1-re állítva.

2003.10.27.

Markó Tamás, PTE TTK

34

A csatornaprogram befejezése

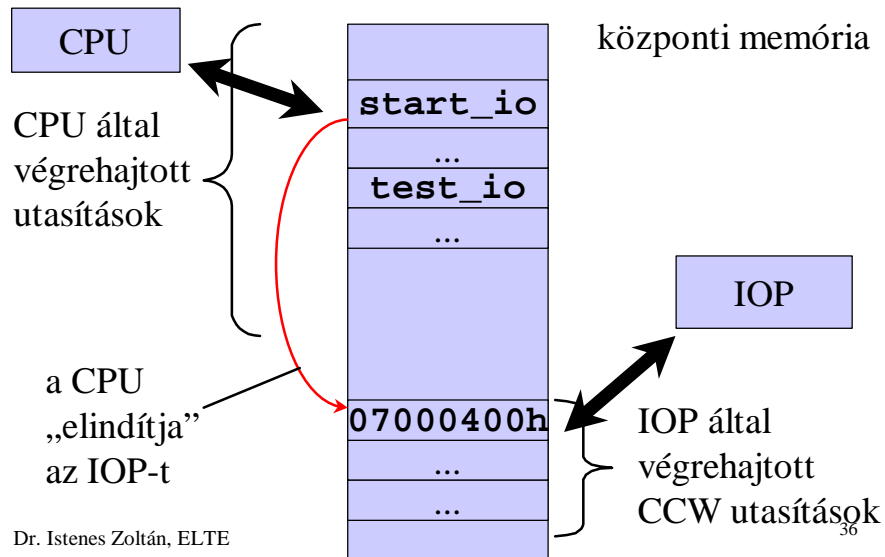
- A csatorna megszakítja a CPU-t, ha
 - a csatornaprogram véget ér (se a CDF, se a CCF nincs beállítva)
 - az eszköz befejezte a működését
 - a CCW-ben lévő *interrupt flag* be van állítva
- Megszakításkor
 - a PSW a memória 56_{10} -os (fix!) címére kerül
 - a CSW a memória 64_{10} -es (fix!) címére kerül(PSW: Program Status Word
CSW: Channel Status Word)

2003.10.27.

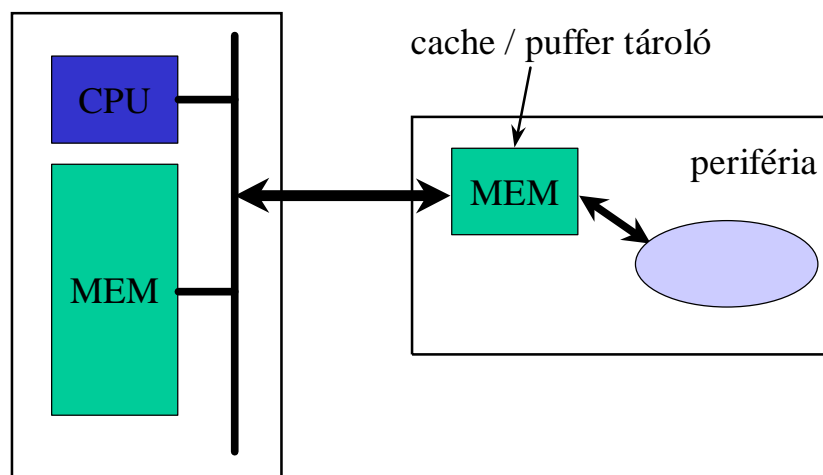
Markó Tamás, PTE TTK

35

CPU és IOP utasítások a memóriában



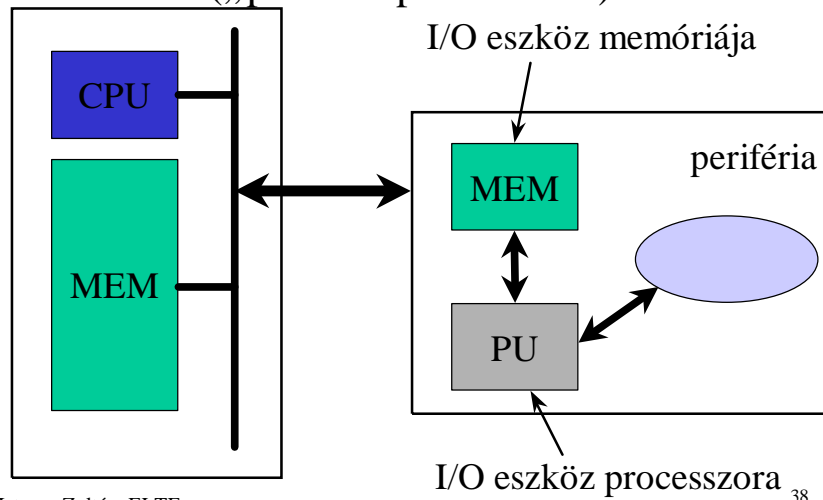
Egyéb lehetőségek 1. (periféria cache/puffer tároló)



Dr. Istenes Zoltán, ELTE

37

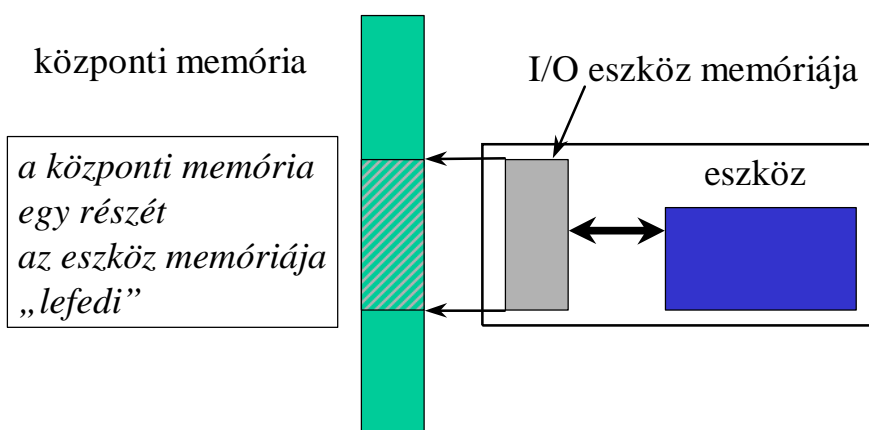
Egyéb lehetőségek 2. („periféria processzor”)



Dr. Istenes Zoltán, ELTE

38

Egyéb lehetőségek 3. („memória lefedés”)



Dr. Istenes Zoltán, ELTE

39