ÓBUDAI EGYETEM

SZÁMÍTÓGÉP ARCHITEKTÚRÁK ALAPJAI

6-7. előadás

Durczy Levente

2024/2025/1

ÓBUDAI EGYETEM

Buszrendszer

A buszrendszer az egységek közötti kommunikációra szolgál (CPU, RAM, perifériák)

A kommunikáció infrastrukturális része és a felhasználó számára transzparens!

Kommunikáció a buszrendszeren szervezett és egységes módon történik.

Történelmi fejlődés eredménye ⟶ ez bizonyult a legjobbnak!

Az eszközök egymással kizárólag a buszrendszeren keresztül tudnak kommunikálni.

Például: SATA, USB, PCIe

Egy rendszerre általában egyszerre több egység kapcsolódik, ezért:

- meg kell oldani adatátvitelben résztvevő eszközök kijelölését
- meg kell határozni az átvitel irányát
- meg kell oldani az adatátvitelben részt vevő eszközök működésének összehangolását (szinkronizálás)

Mindezekhez fontos eszköz a szabványosítás (jelhasználat, vezetékkiosztás), melynek következménye, hogy az eszközök könnyen cserélhetők.

Jelentősége óriási: hiába villámgyors egy eszköz, ha a buszrendszer lassú!

Csoportosítás:

1. Az átvitel iránya szerint:

a) szimplex: egy irányba képes az adatátvitelre (pl.: címbusz, CLK, RST)

b) félduplex: mindkét irányba, de egy időben csak egy irányba közlekedik az adat

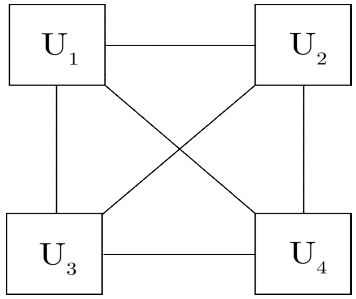
c) full-duplex: egy időben két irányba is közlekedik az adat (pl.: PCIe busz)

2. Átvitel jellege szerint:

a) Dedikált: minden egység mindegyikkel össze van kötve (pl.: Intel QPI)

Előny: gyors, közvetlen kommunikáció

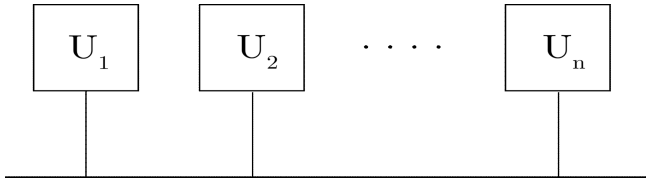
Hátrány: merev és nehezen bővíthető



b) Shared (megosztott): minden egység a közös buszon kommunikál, buszvezérlő utasításokra van szükség vezérlő vonalakon keresztül az ütközések elkerülésére

Előny: olcsó, egyszerű megvalósítás, könnyen bővíthető

Hátrány: viszonylag lassú, vezérlése bonyolult, meghibásodás esetén több eszköz kieshet (pl.: SATA, RS232)



ÓBUDAI EGYETEM

3. Átvitt tartalom szerint:

a. címbusz: eszközök címzésére szolgál

Szélességét (vezetékszám) folyamatosan bővítették

pl.: Intel 8080 CPU 20 bit (1MB)

Intel 80386 CPU 32 bit (4GB)

Manapság jellemzően nem egyesével történik a címzés, egy adatblokk kezdőcímét adjuk meg majd ciklikusan az adatokat, amíg szükséges (a többi cím inkrementálással áll elő)

b. adatbusz: adatokat juttatja el az operatív tárból és perifériákból a processzor felé és vissza

Sávszélessége folyamatosan növekedett (8, 16, 32, 64 bit …)

Közös lehet a vezeték a címbusszal ⟶ időbeli multiplexeléssel oldják meg, amihez vezérlővonalak szükségesek

c. vezérlővonalak: a rendszerben az időzítő jelek és az egység állapotáról szóló információ átvitelére szolgálnak!

<u>4 fő típusú vezérlőjelet különböztetünk meg:</u>

- Adatátvitelt vezérlő jelek
- Megszakítást vezérlő jelek
- Busz vezérlő jelek
- Egyéb vezérlő jelek

Adatátvitelt vezérlő jelek:

M/”I/O”: megmondja, hogy a buszon memória vagy periféria van címezve
R/W: adatáramlás irányát adja meg a CPU felől nézve
WD/B: megadja az adat méretét
D/S (data strobe): az adat felhelyezését jelzi a memória számára
A/S (address strobe): a cím felhelyezését jelzi a memória számára
RDY: az átvitel befejezését / a busz rendelkezésre állását jelzi

Megszakítást vezérlő jelek:

Megszakítást kérő jel
Megszakítást visszaigazoló jel

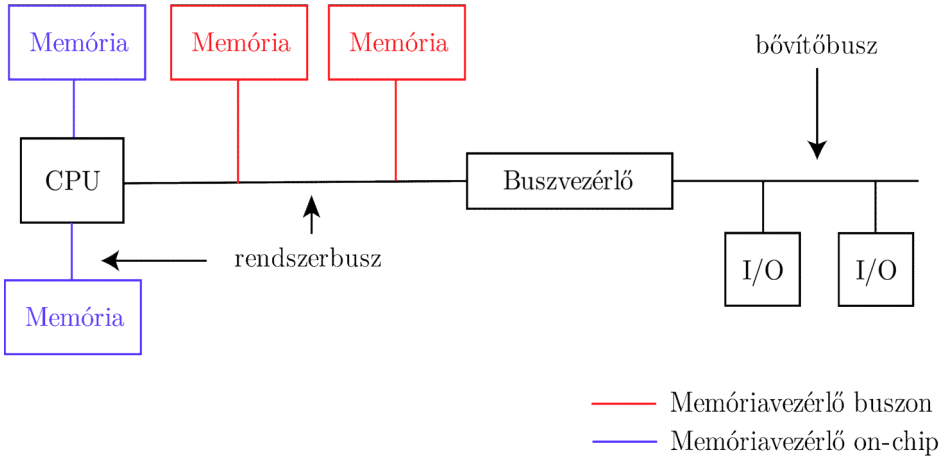
Buszvezérlő jelek: a busz kérésére, foglalásra, visszaigazolásra szolgáló jelek

Egyéb vagy speciális vezérlőjelek:

Reset
CLK

ÓBUDAI EGYETEM

4. Összekapcsolt területek alapján:
   a. rendszerbusz (system bus):
      adatbusz + címbusz
   b. bővítőbusz (expansion bus):
      ide tartozik pl.: a PCI, PCIe, USB



5. Az átvitel módja szerint:
   a. soros
   b. párhuzamos (pl.: PCI, memória)

Az 1981-ben megjelent IBM PC buszrendszere az alapja a mai buszoknak.
Ez volt az első nyitott architektúra! Mit jelent ez?

a) Általános célú expanziós csatlakozók, melyek funkciója nincs előre definiálva
b) Definiálva van a csatlakozón belül az egyes érintkezők feladata és ez nyilvános!
(más cégek is képesek lettek perifériákat gyártani így)

A szabványosítás a buszrendszer esetében azt jelenti, hogy definiálva van egy vezeték köteg és az általános célú csatlakozók kiosztása

Mivel így már bármilyen eszköz csatlakoztathatóvá vált, azért, hogy a processzor tudja milyen eszköz is csatlakozik, megjelentek a driverek. Ennek hatására platformfüggetlen megoldások kezdtek elterjedni (SATA, USB, ...), amik a CPU-tól független szabványos illesztést biztosítanak!

PCI szabvány (párhuzamos busz):

Magában foglalja - a busz fizikai méreteit (aljzat, benne lévő elektromos érintkezők méretei, …),
- az elektormos jellemzőket,
- adat átviteli időzítést
- protokollokat

Annyival nyújtott többet, mint a többi, hogy a PCI buszra csatlakoztatott perifériákat a CPU úgy látja, mintha azok közvetlenül a rendszerbuszra lennének csatlakoztatva, vagyis az általa látott címtérből látja el memóriacímekkel.

Előnye:

- gyorsabb adatátvitel (mondhatni közelebb kerülnek az perifériák a CPU-hoz)
- viszonylag olcsó
- egyszerű és széleskörben elfogadott

A specifikáció magába foglalja a fizikai méreteket, csatlakozó kiosztást, adatátviteli időzítést, protokollokat. Fejlesztése: ⟶ AGP csatlakozó (nagyobb sávszélesség és sebesség)

PCI Express (soros busz):

2004-ben jelent meg, Intel, HP, IBM és Dell közös fejlesztése.

- soros busz ⟶ kevesebb érintkező
- nagy sebességű, 1-32 bit szélesség az első verziónál
- hot-plug funkcióval rendelkezik: menet közben is lekapcsolható
- CPU-tól is függ a teljesítmény!!!
- Ma már a háttértárolók jelentős része is erre a buszra kapcsolódik (M.2-es csatlakozó)
- Miért gyorsabb egy NVMe (Non-Volatile Memory Host Controller Interface Specification) SSD a hagyományos SATA SSD-nél?

Kihasználja a félvezető alapú tároló eszközök

alacsony késleltetését és belső párhuzamosságát!

4 sávos PCIe interface-t biztosítanak

ÓBUDAI EGYETEM

Összehasonlítás:

PCIe	PCI
pont-pont topológia, soros adatátvitel	megosztott párhuzamos architektúrát használ minden eszköz
különálló vonalak kapcsolják az eszközöket a buszvezérlőhöz	minden eszköz közös cím- adat- és vezérlővonalat használ
Full duplex, bármely két végpont között	több master esetén arbitrálás (buszfoglalás) történik
egy időben több végpont párhuzamosan kommunikálhat	egy időben csak egyetlen master működhet egy irányban
többféle szélességű aljzat (1, 4, 8, 16, 32-szeres) → rugalmas	egyetlen nagy teljesítményű, de közös busz
Buszprotokoll: csomagokba ágyazza az adatokat!	

ÓBUDAI EGYETEM

USB szabvány:

- periféria busz
- 1995-ben kezdték el kifejleszteni az akkori nagy IT vállalatok (DEC, Microsoft, IBM, …)

Kiadás éve	Szabvány verzió	Átviteli sebesség
1998	USB 1.1 (első működőképes verzió)	12 Mb/s
2000	USB 2.0	480 Mb/s
2008	USB 3.0	5Gb/s
2013	USB 3.1	10Gb/s
2017	USB 3.2 (USB-C)	20Gb/s
2019	USB 4 (Thunderbolt 3 alapokon)	40Gb/s
2022	USB 4 2.0	80-120Gb/s

Kihívója az Intel által fejlesztett Thunderbolt 3 és 4, ami PCI Express alapokon nyugszik.
Előnye, hogy támogatja az USB, HDMI és Ethernet szabványt.
A Thunderbolt 3 mára már nyílt szabvánnyá vált (USB is tudja)

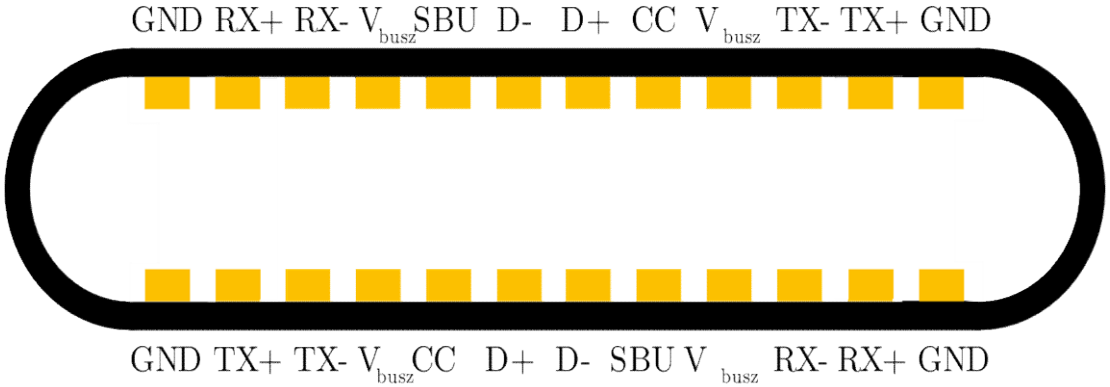
Csatlakozók:

a) hagyományos (USB-A/USB-B)

2 érpárt tartalmaz

teljesítmény maximum 5W

b) USB-C



4 érpárt tartalmaz adatátvitelre

24 pines csatlakozó: 2x12 érintkező

szimmetrikus kiosztásban

teljesítmény 15W (3A, 5V) – 100W

Mind a két végén van egy földcsatlakozó

TX+ TX- nagysebességű adatküldési vezetékpár

Vbusz az áramellátást biztosítja

D+ D- USB 2.0 adatátviteli vezeték

RX- RX+ nagysebességű adatfogadási vezetékpár

CC, SBU alternatív vezetékek, akár videójelek továbbítására, …

számos protokollt támogat

Mind az USB 4.0, mind a Thunderbolt 4.0 támogatja a 100W töltést, és
akár 2 db 4K-s kijelző csatlakoztatható hozzájuk. A Thunderbolt 4.0-hoz Intel tanúsítvány kell (+ költség!)

Az USB 4.0 V2 szabvány aszimmetrikus adatátvitelre is képes, 80-120 Gbit/sec!
Mit jelent ez?

Csomagkapcsolt adatátvitel itt is!
4 típusú csomagot különböztetnek meg:

- Időkritikus csomag: állandó adatsebességet valósítanak meg. Az ilyen adatátvitelnél a potenciális tranziens hibák javítása nem történik meg, azaz az elektromos adatátvitelben fellépő hibákat a hardver nem korrigálja a csomagok ismétlésével. A gyakorlatban az USB bithiba valószínűsége olyan kicsi, hogy ez általában nem okoz semmilyen nehézséget (pl.: audio- és video adatok)
- Nagy adatcsomag: sok adat továbbítására szolgálnak. Alacsony prioritás (pl.: backup, printer,…)
- Megszakítási csomag: ezt használják az egységek kiszolgálás kérésre. Kis adattartalmú csomag, ciklikus lekérdezés
- Vezérlési csomagok: címkiosztáshoz, eszközök azonosításához használatosak, handshake működés

Sérülékenység:

- Thunderspy támadás (Thunderbolt 3.0): A thunderbolt PCI-express részének DMA hozzáférés kihasználásával megkerüli a CPU-t (pl.: jelszó felülírás, …)
- Megoldás: Intel virtualizációs technológia (VT-d): elszigeteli a memória egy részét a csatlakoztatott periféria számára, a többi elérését tiltja!)

Az adatátviteli sebesség folyamatos növelése egyre nagyobb csillapítást, nagyobb „zajt” és több áthallást eredményez!

Adatok továbbítása buszrendszereken:

Jeltípusok:

ÓBUDAI EGYETEM

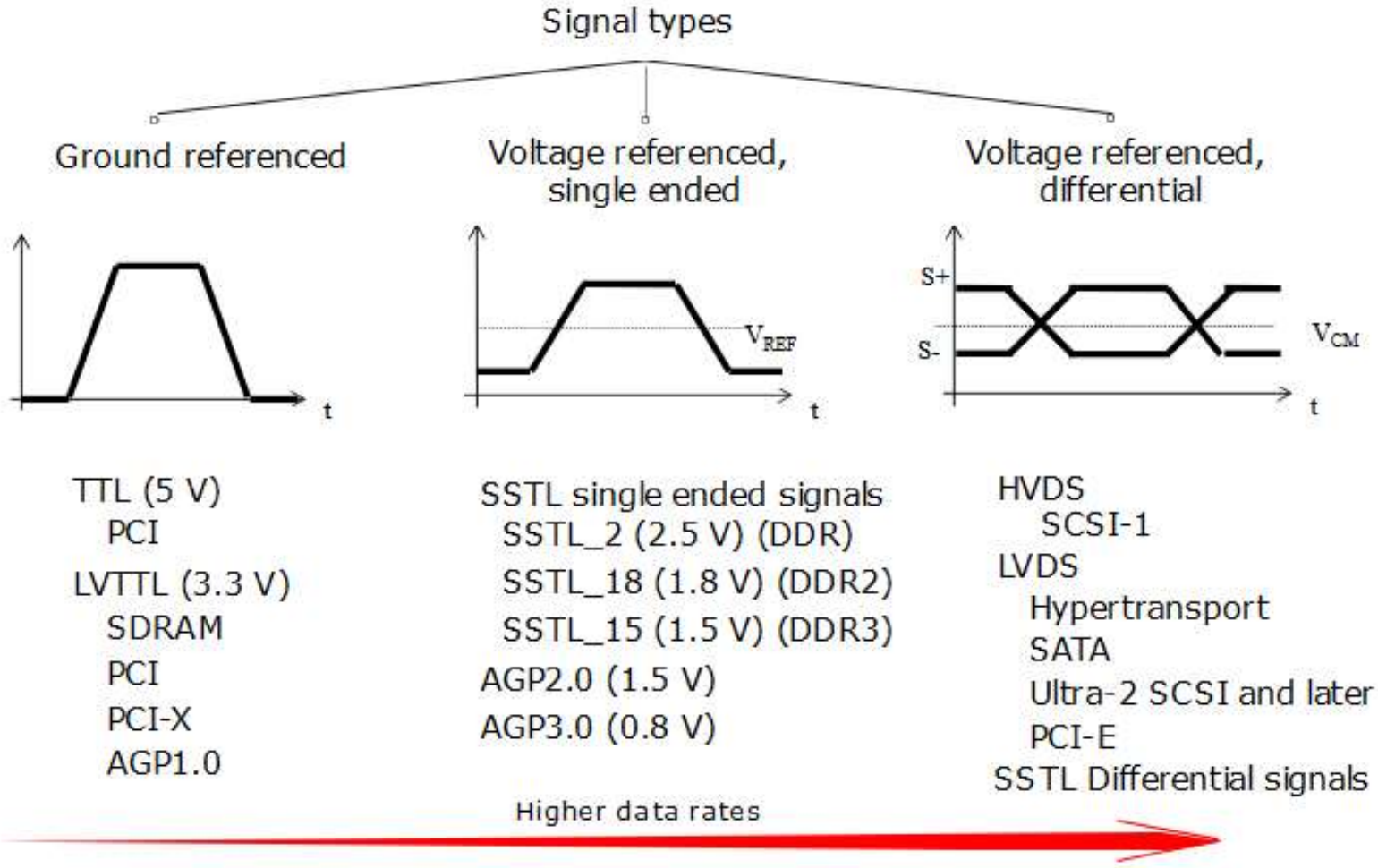


Figure: Overview of signal types

További sebességnövekedéshez új fejlesztésekre volt szükség

Hogyan lehet az adatsűrűséget növelni?

Megoldás: jelkódolás

pl.: PAM3: impulzus amplitúdó modulációs eljárás (USB 4 2.0, GDDR7 memória)

- Két jel van: a digitális jel és egy vivő jel.
- Ezek eredője egy szabálytalan jel lesz, amit 3 jelszinten értelmeznek: -1, 0, 1 (vagy 0, 1, 2)

- PAM3: Impulzusamplitúdó moduláció 3-szintű, hármas kód, általában -1, 0, +1 vagy 0, 1, 2 a hármas értékek megjelenítésére (tritek). Minden PAM3 szimbólum órajelenként 1,58 bitet tud továbbítani ($\log_2 3 = 1{,}58$).

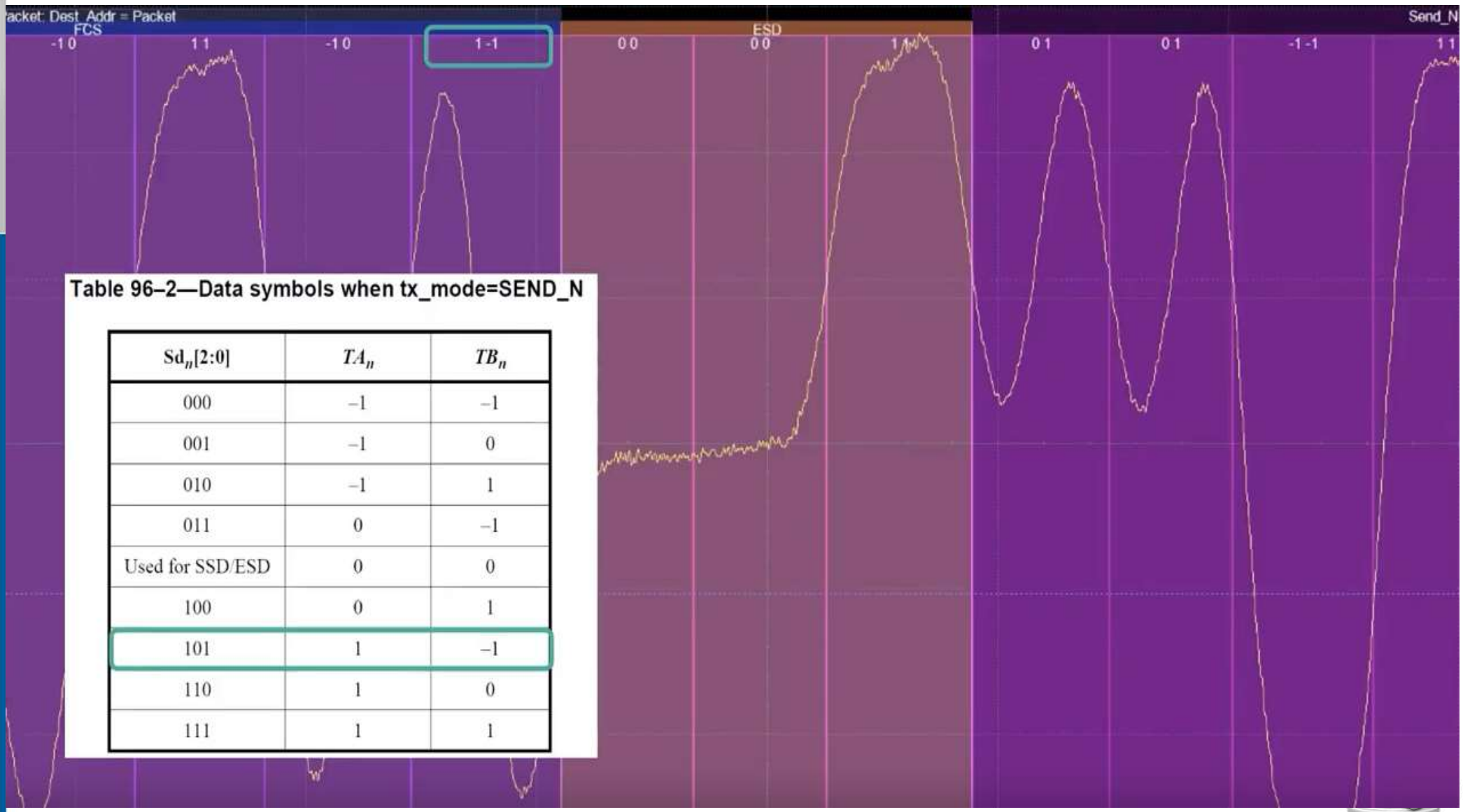
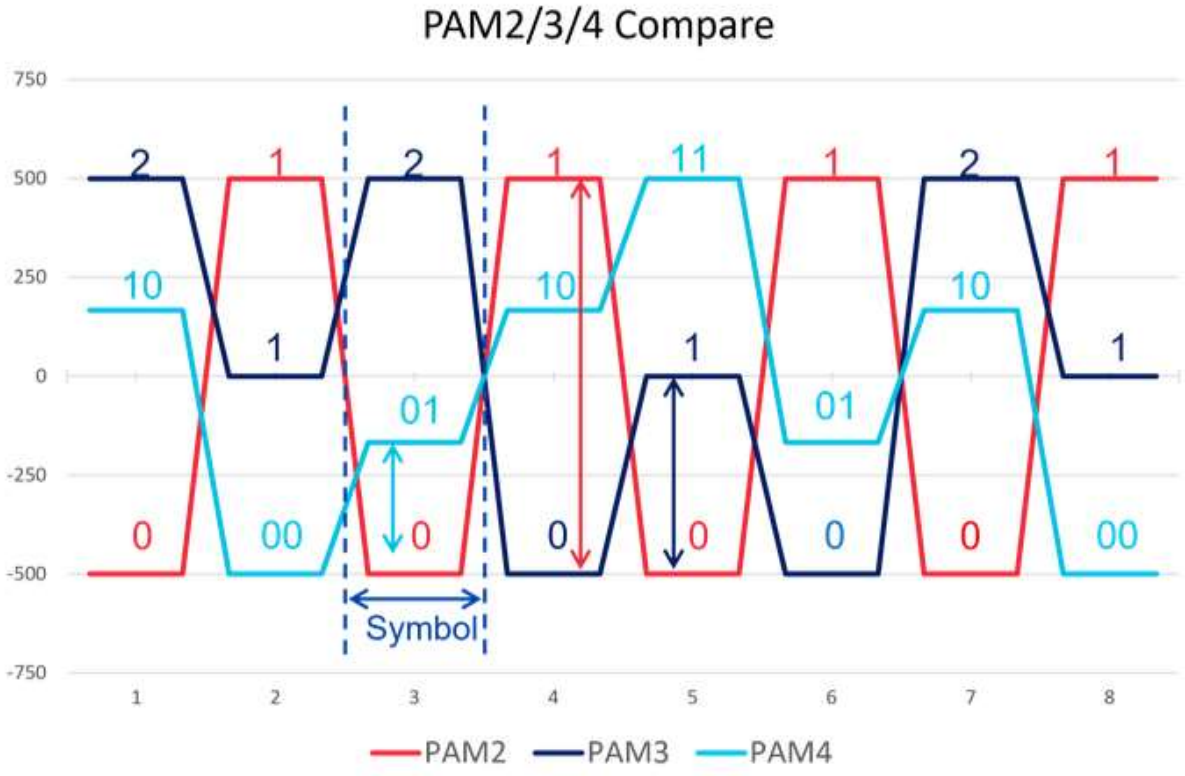


Table 96–2—Data symbols when tx_mode=SEND_N

$Sd_n[2:0]$	TA_n	TB_n
000	–1	–1
001	–1	0
010	–1	1
011	0	–1
Used for SSD/ESD	0	0
100	0	1
101	1	–1
110	1	0
111	1	1

PAM4: Impulzus-amplitúdó moduláció 4-szintű, általában 00, 01, 10, 11-gyel jelölve. Minden PAM4 szimbólum szimbólumonként 2 bitet tud továbbítani ($\log_2 4 = 2$), de a nagyobb bithiba arány miatt még nem gyorsabb (~1,57)! A PAM3 jelmagasság és jel-zaj torzítási aránya (bit error ratio) jobb.

A PCI-express 6.0 már PAM4 jelkódolást használ. (az Nvidia 2024 tavaszán adta ki az első GPU-t)



Párhuzamos és soros buszok összehasonlítása:

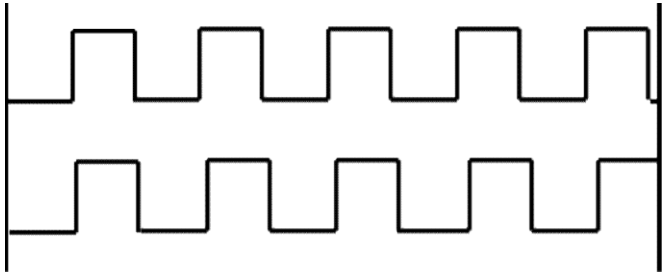
Kezdetben soros buszokat használtak, majd áttértek párhuzamosra. Ennek oka, hogy 32-64 bitet sokkal egyszerűbbé vált már párhuzamosan átküldeni, csak elég vezeték kellett hozzá.

Vezetékek száma:
Nagy előnye viszont egyben a hátránya is a párhuzamos adatátvitelnek. A sok vezeték komplex és drága, ráadásul sok helyet foglal (nagy csatlakozási felület!).

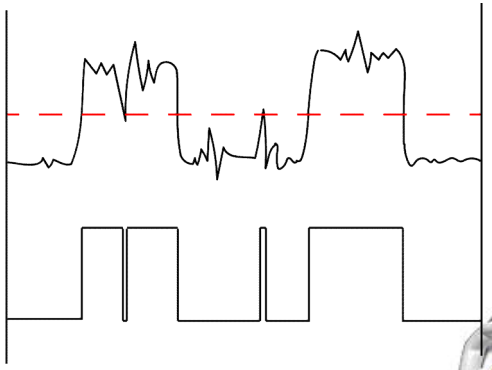
Soros adatátvitel esetén elég egy vezetékpár is, illetve a biteket bitsorozatként kódolva lehet átküldeni, ehhez viszont plusz hardver szükséges!
A megoldás itt is a kódolás! Az egyes adatbiteket (pl.: 64 bit) hosszú sorozatokká alakítják át.
(és vissza is kell konvertálni párhuzamossá!)

Sebesség:
A frekvencia a 2000-es évek elejére jelentősen megnőtt. Kezdetben a párhuzamos adatátvitel alacsony frekvencián könnyen implementálható volt, viszont a magas frekvenciás átvitelnél problémák jelentek meg:

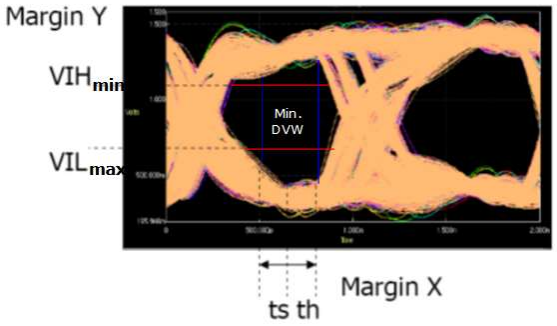
1. Delay Skew (időbeli eltérés): már kis vezeték hossznál is elég magas frekvencia esetén időbeli elcsúszás történhet a jelekben. Nem ugyanabban az időpillanatban fognak megérkezni.

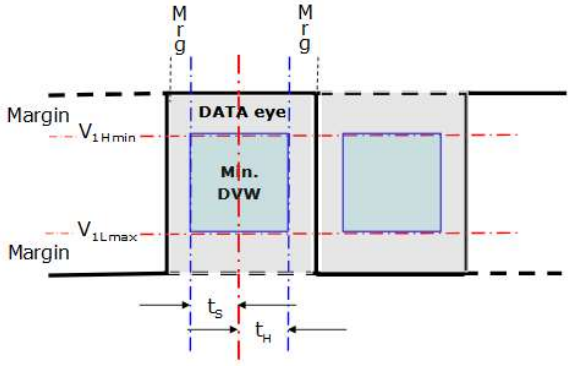


2. Sok párhuzamos vezeték és külső elektromágneses sugárzások elektromágneses interferenciát (EMI) generálhatnak, ez a „zaj”. A zaj annyira torzíthatja a jelet, hogy az kiesve az érzékelési tartományból, hibásan értelmezhető.



Egy logikai érték pontos értelmezéséhez elngedhetetlen, hogy egy bizonyos feszültségszint alatt vagy fölött legyen a jel legalább „Δt” ideig! ⟶ Data valid window:





3. Vezetékek közötti keresztirányú áthallás szintén interferenciát generál.
- Minél hosszabb a vezeték, ez annál nagyobb, ezért a távolság
- limitált párhuzamos rendszereknél. (tekercsként viselkedik!)

Ezek összefoglaló neve a „Jitter”.

Soros busznál ezek a problémák megszűnnek, ezért az nagyobb frekvencián üzemelhet és nagyobb távolságra biztosítja az adatátvitelt, míg a párhuzamos kisebbre.

A párhuzamos adatkapcsolatok hardver vonatkozásban viszonylag könnyen implementálhatók.

Azonban egyetlen gyors érpáron több adat továbbítható, mint egy több vezetékből álló lassún!

A soros interfész rugalmasabban frissíthető, akár szoftveresen is! (pl.: SATA 1-2-3)

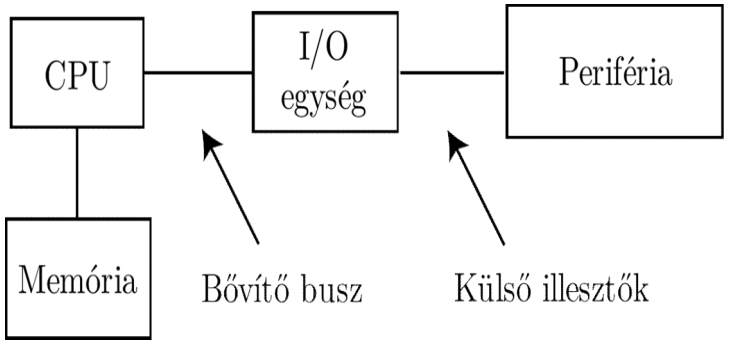
Viszont a CPU és a memória között olyan óriási mennyiségű adatáramlás történik, hogy a soros adatátvitel helyett több száz vezetéken párhuzamos adatátvitel történik.

Ezért vannak a memória csatlakozók a CPU-hoz a lehető legközelebb!

I/O rendszer:

Alapvető képessége a számítógépeknek, hogy más egységekkel adatot cseréljenek.

Hogyan?



Fejlődése:

a) A CPU közvetlenül vezérli a perifériát ⟶ megszakítás nélküli programozott I/O (wait for flag)

b) I/O modul (vezérlő):

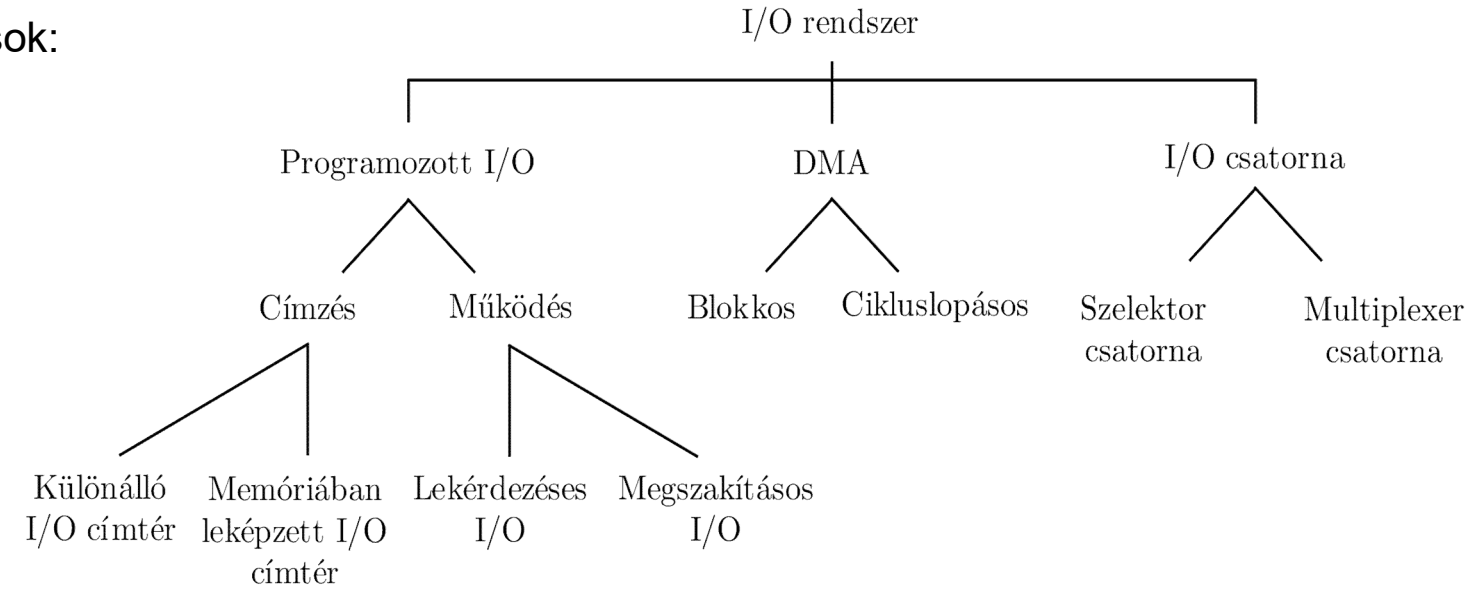
- Megszakítás nélküli programozott I/O
- Megszakításos vezérlés

c) DMA: az I/O modul számára DMA segítségével közvetlen hozzáférés a perifériához!
(lényegesen nagyobb sebesség)

d) I/O csatorna: lassabb perifériák számára. I/O-ra specializált utasításkészlettel rendelkezik

e) I/O processzor: I/O célú utasításokat dolgoz fel, saját operatív tárral rendelkezik

Főbb I/O típusok:



Programozott I/O:

- A CPU által irányítottan történik az adatátvitel
- A processzor vezérli, irányítja és lezárja az I/O műveltet
- Megvalósítása egyszerű, de jelentős processzor időt vesz el!
- Lekérdezéses, vagy megszakításos vezérlés
- A különálló I/O címtér annyit tesz, hogy a CPU két címteret lát, egyet a perifériáknak és egyet a memóriának, ezáltal ugyanaz a cím szerepelhet memóriacímként és I/O címként is.
- Közös buszhasználat van.
- Létezik az M/IO vezérlőjel, mely megmondja, hogy az adott időpillanatban memória- vagy I/O-cím van a címsínen.
- Azon regisztereket, amelyeken keresztül a processzor a perifériákkal kommunikálhat, I/O portnak nevezzük.
- Az I/O port fizikailag a vezérlőkártyán helyezkedik el

Az I/O port részei:

- parancs regiszter (command): a CPU ide írja a „kívánságait”
- adatregiszterek: ez tekinthető a bővítőbusz végállomásának
  - data input regiszter: : ebből olvassa a CPU a perifériáról kapott adatokat
  - data output regiszter: ebbe írja a processzor a periféria számára küldött adatokat
- állapotregiszter: állapotinformációkat közöl a perifériáról. Minden bitje mást jelent.

Legjellemzőbb állapotok például a megszakításkérés, Ready, Busy.

Lehet közös:

- az állapot és parancs regiszter, mert a CPU felől az egyik írva, másik olvasva van
- adatregisztereknél pedig a data input és output

Egyéb regiszterek:

- jelenlét ellenőrző regiszter: megmondja van-e eszköz kapcsolva a I/O portra
- eszköz tulajdonságait tartalmazó regiszter (plug and play funkciót teszi lehetővé)
- Bonyolultabb periféria esetén egyes funkciókhoz akár több regiszter is tartozhat

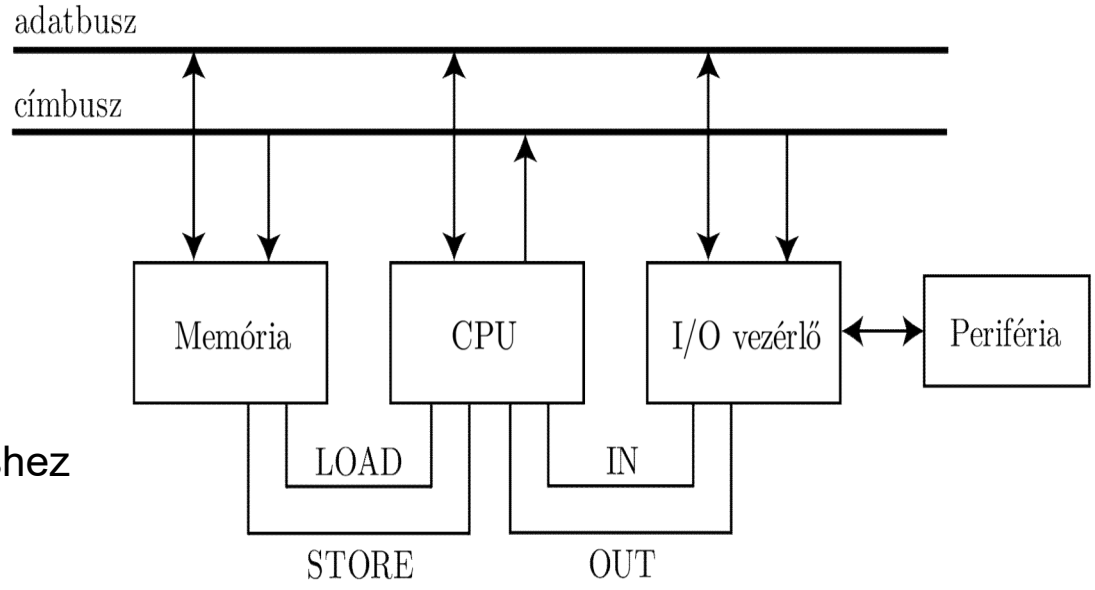
Különálló I/O címtér:

A CPU a memóriával a kommunikációt LOAD/STORE utasításokkal végzi.
Megcímzi a memóriát, és adatátvitellel kommunikál.
Ugyanígy címzi a I/O vezérlőt, viszont külön I/O utasításokkal (pl.: IN és OUT) biztosítja az adatátvitelt.
Az I/O vezérlőn megtalálhatóak a portok, melyek biztosítják a periféria felé az adatkommunikációt.
A memória és a periféria címek lehetnek azonosak!!!

Előnye:
- egyszerű
- olcsó megvalósítás

Hátránya:
- terheli a CPU-t
- plusz utasítások az I/O kezeléshez

Pl.: PS2, RS232

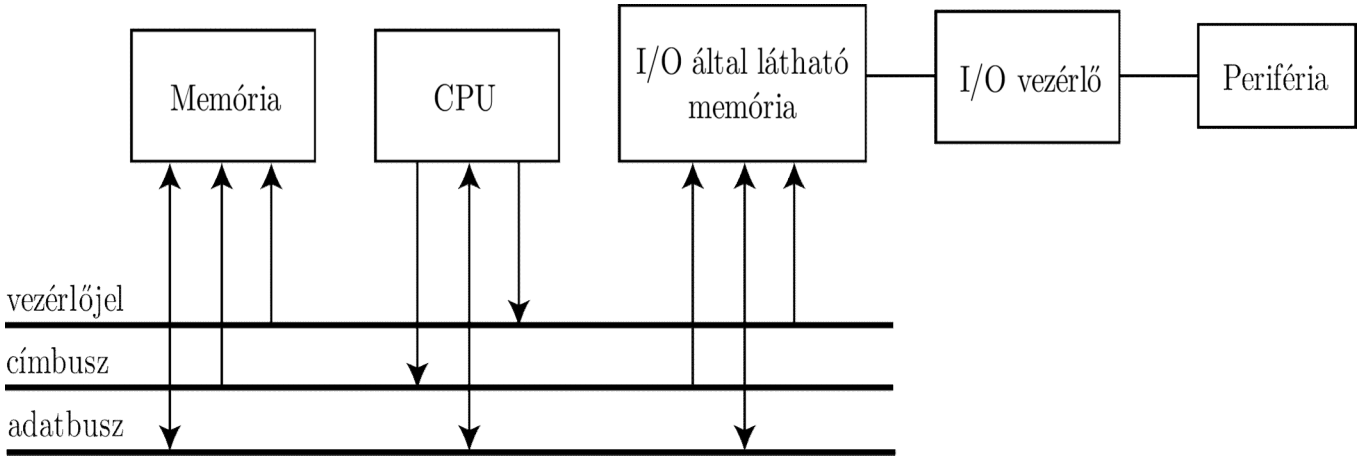


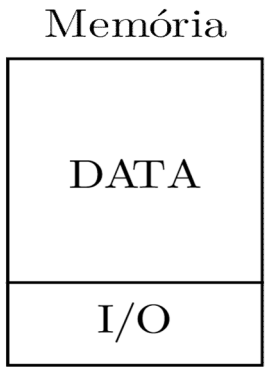
Memóriában leképzett I/O:

Annyiban különbözik a különálló I/O címtértől, hogy a perifériák számára a memóriában lefoglal a CPU egy területet.

A vezérlőjelen biztosítja a processzor a vezérlő információkat (írás, olvasás, M/IO).

Egy memória hivatkozás akkor válik I/O utasítássá, hogyha a címbe az I/O által látható memóriacím kerül!





Előny:

- LOAD/STORE utasítással történik az adatok mozgatása, nem kell külön utasítás
- I/O vezérlő hozzáfér a rendszerbuszhoz ⟶ gyorsabb átvitel

Hátrány: Továbbra is terheli a CPU-t!

Példa:
Kijelző: a legtöbb grafikus rendszer megengedi a CPU számára, hogy közvetlenül címezze azt a frame buffert, amely a képernyőn az adott pillanatban megjelenő képet tartalmazza.

Működés:

Átviteli módszerek alapján:
- feltétel nélküli adatátvitel (direkt programozott)
- feltételes adatátvitel

Feltétel nélküli adatátvitel előfeltételei:

- a perifériának mindig adatátvitelre alkalmas állapotban kell lennie
- ellenőrzésre sem az adatátvitel előtt, sem utána nincs szükség
- semmilyen szinkronizálás nincs a CPU és a periféria között

Például a kijelző.
Hátránya: nincs visszacsatolás (adatvesztés előfordulhat!)

Feltételes adatátvitel:

valamilyen feltétel teljesülésétől tehető függővé az adatátvitel

Típusai:

- Lekérdezéses adatátvitel: a processzor beírja a kívánságát az I/O vezérlőbe és várja a választ oly módon, hogy folyamatosan lekérdezi az állapotregiszter tartalmát (nem tudja, meddig tart!).
  („készen vagy már? készen vagy már? készen vagy már?”)
  nagyon pazarló működés, mert a periféria nagyságrendekkel lassabb, ezért találták fel a megszakításos adatátvitelt.

- Megszakításos adatátvitel: a processzor beírja a kívánságát az I/O vezérlőbe és elkezd mással foglalkozni. A periféria vezérlő rendelkezik egy megszakítás rendszerrel. Ha megtörténik az adat beolvasás, akkor megszakításkérést küld a processzornak és az a következő utasítás töréspontban lekérdezi az állapotregisztert (és ha „Ready”, akkor megtörténik az adatátvitel).
  Tehát a megszakítás kiszolgálása eredményezi az adatátvitelt!

  pl.: nyomtató

Előnye:

- A CPU lényegesen kevesebb időt tölt el az adatátvitel irányításával
- sokkal gyorsabb

Hátrány:

- nagy mennyiségű adat esetén lassú és igen sok megszakítással jár !
  (pl.: HDD és SSD esetén nem elég!)
- még mindig a CPU kezdeményezi és vezérli az átvitelt

Tehát a programozott I/O nem túl előnyös: lassú és leköti a CPU-t

DMA:

Direct Memory Access: tovább gyorsítja az adatátvitelt a perifériákról és még inkább leveszi a terhet a processzor válláról.

Fontos tulajdonságai:

- nagy mennyiségű adat esetén (általában blokkos adatátvitellel)
- gyors perifériáknál
- a CPU közreműködése nélkül végzi az adatátvitelt

A DMA vezérlő a memórián keresztül a nagy mennyiségű adatátvitelt biztosítja a gyors periféria számára, a CPU csak a vezérlést biztosítja. A hagyományos I/O vezérlő a lassú periféria számára biztosítja az adatátvitelt a processzor közreműködésével.

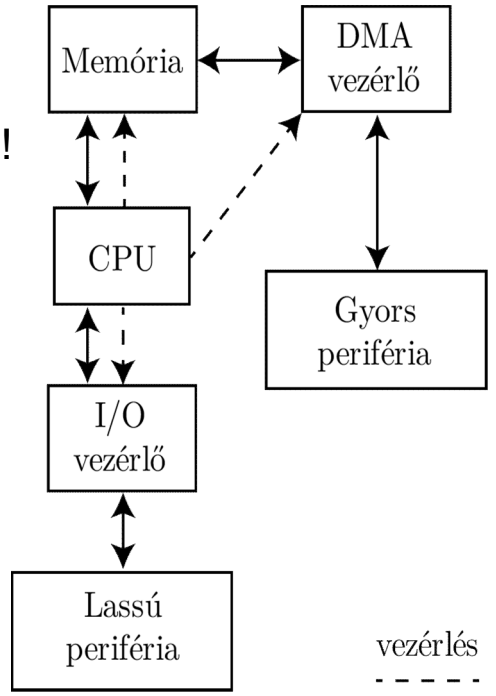
Viszonylag szerény mértékű komplexitás növekedés segítségével valósítható meg a közvetlen adatátvitel.

ÓBUDAI EGYETEM

Feltétele, hogy címet kell tudni generálni a közvetlen tárhozzáféréshez, valamit busz vezérlési funkciókkal is el kell látni (busz igénylési és kiválasztási mechanizmus).

A blokk átvitelek kezdeményezéséért továbbra is a CPU a felelős!

Azonban a szükséges megszakítások száma nagyságrendekkel csökken!

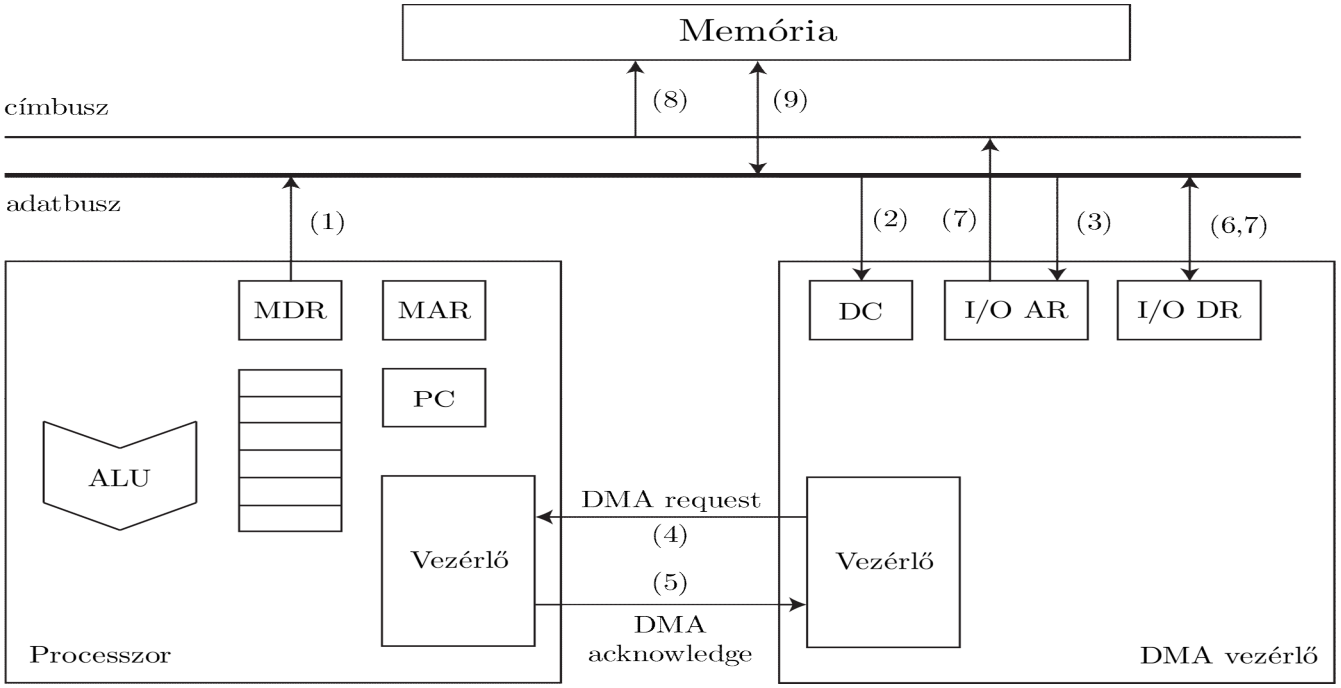


Működés:
A DMA el van látva regiszterekkel:
DC, I/O címregiszter, I/O adatregiszter, belső transzparens regiszterek.
A működés mind blokkos, mind ciklusloppásos esetén
felparaméterezéssel kezdődik:

Programozott I/O-val átvisszük a processzorból a DMA vezérlőbe
az átvitelhez szükséges információkat:

- írás- vagy olvasási művelet (tulajdonképpen az átvitel iránya)
- I/O egység címe

- memória cím kezdőértéke I/O AR-be (ahonnan olvasunk, vagy ahová írunk)
- átvivendő adat jellege (byte, félszó, szó)
- olvasandó/írandó egységek száma (DC-be)
- átvitel módja (blokkos vagy ciklusloppásos)
- a DMA csatornáihoz prioritási értékeket rendelhetünk
- résztvevő egységek (memória-memória, I/O-memória, I/O-I/O)



Blokkos átvitel:

1-3. A CPU felparaméterezi a DMA vezérlőt és elindítja azt.

4. A DMA vezérlő „DMA request” jellel kéri a rendszerbusz használati jogát (megszakítás)

5. A processzor „DMA acknowledge” jelzéssel lemond a rendszerbusz használati jogáról

6. A DMA vezérlő a kapott adatok alapján a perifériától (I/O port) bekéri az első átvinni kívánt adatot az I/O DR-be

7-9. A DMA vezérlő az I/O DR-ben lévő adatot a rendszerbuszon keresztül beírja az I/O AR által meghatározott memóriacímre

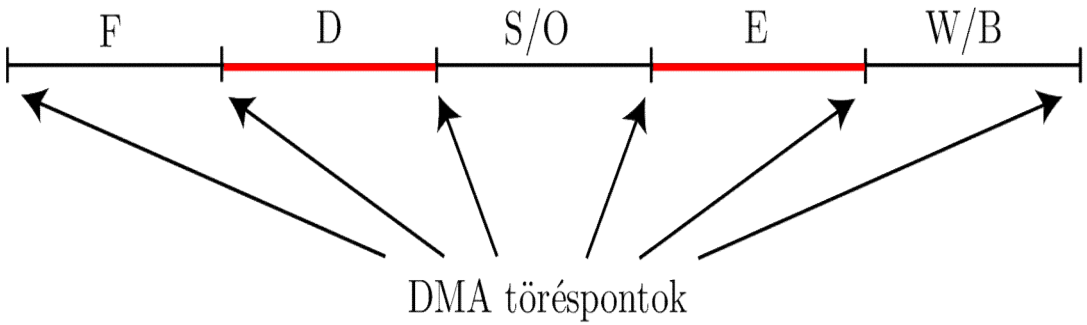
10. A DMA vezérlő dekrementálja a DC-ben tárolt értéket, és inkrementálja az I/O AR -ben tárolt értéket (egy adategységgel növel – byte, félszó, szó)

11. A DMA ellenőrzi a DC tartalmát. Ha nem 0, vissza a (6) -ra. Ha igen, megszakításkéréssel jelzi a CPU felé, hogy befejeződött egy blokk átvitele. A CPU ellenőrzi az adatvitelt, megszünteti a buszhasználat engedélyezést és visszaveszi a vezérlést.

Ciklusloposás átvitel:

Ciklusloposás esetén nem a teljes időtartamra foglalja le a DMA vezérlő a busz használatát, hanem csak 1-1 adat átvitelének időtartamára. Ezt kevesebb adat esetén használják (például gyors nyomtatók).

Egy utasítás végrehajtásának felbontása részfeladatokra:



F - fetch
D - decode
S/O - forrás operandus beolvasás
E - execute
W/B - writeback

ÓBUDAI EGYETEM

Ezek közül a processzor „FETCH”,az „S/O” és a „W/B” esetén használja a rendszerbuszt.

„DECODE” és „EXECUTE” esetén a CPU le tud mondani a rendszerbusz használatának jogáról és a DMA 1-1 adatot át tud irányítani a perifériáról a DMA szubrutin szerű működést végez.

Itt a megszakítás kérés nem egy buszvezérlési igény, hanem egy megszakítási program végrehajtásának a kérése

Működése:

1. DMA vezérlő felprogramozása (felparaméterezés)
2. I/O egység aktiválja a DMA request vezérlővonalat
3. A következő DMA töréspontban a CPU lemond a buszhasználatról és ezt a DMA acknowledge jel aktiválásával jelzi
4. Adatátvitel I/O és memória között
5. Ha a DC nem 0, de a periféria nincs Ready állapotban, akkor a DMA visszaadja a vezérlést a CPU-nak. (leveszi a DMA request jelet).
6. Ha DC 0, a DMA megszakításjelzést küld és visszaadja a vezérlést a CPU-nak
7. A CPU vagy leállítja az I/O egységet, vagy új I/O adatátvitelt kezdeményezhet.

Hátránya: manapság a párhuzamos működés során a CPU közel minden óraciklusban használja a rendszerbuszt.

I/O csatorna:

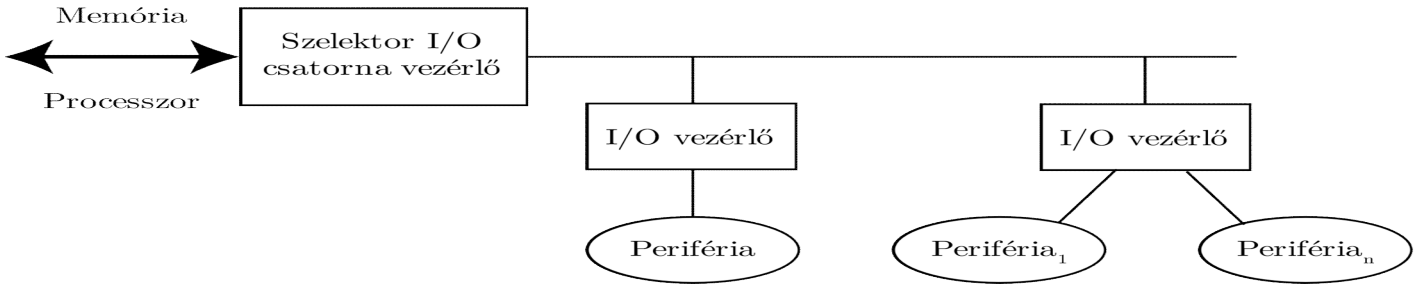
A DMA kiterjesztése a lassabb perifériákra. Fejlettebb, mint a programozott I/O, de egyszerűbb, mint a DMA. Ilyenkor a CPU nem hajt végre I/O műveletet. Az utasítások az I/O csatorna vezérlő számára a memóriában vannak tárolva. A CPU kezdeményezi az átvitelt, viszont az I/O csatorna hajtja végre a memóriában tárolt programot, ami specifikálja az átvitelt (nincs felparaméterezés).

Típusai:

- szelektor csatorna
- multiplexer csatorna

Szelektor csatorna:

A lassabb eszközök közül is a nagyobb sebességű I/O eszközök számára. A szelektor I/O csatorna vezérlő egyetlen I/O vonallal rendelkezik, amire több I/O vezérlő is rácsatlakozhat különböző perifériákkal. A szelektor I/O csatorna vezérlő kommunikál a CPU-val és memóriával. Egyszerre csak egy egység kommunikálhat a vezérlővel és azon keresztül a memóriával. Azért szelektor, mert az adatátvitel viszonylag gyorsan történik, de egyszerre csak egy eszköz.



Multiplexer csatorna:

Kifejezetten lassú perifériákhoz találták ki, amik folyamatosan küldik az információt, de viszonylag nagy az időkülönbség az adatok között. A csatorna vezérlőre több I/O vezérlő csatlakozhat párhuzamosan. Elég lassúak a perifériák ahhoz, hogy a vezérlő egy időben több perifériával kommunikálhat.

Két típusa van:

- byte multiplexer: bytonként küldi az adatokat
- blokk multiplexer: kisebb-nagyobb blokkonként küldi az adatoka

Cél az átvitel maximalizálása.