

Pannon Egyetem

Villamosmérnöki és Információs Tanszék



Digitális Áramkörök

(Villamosmérnök BSc /
Mechatronikai mérnök MSc)

2. hét - Boole algebra (függvény, igazságtábla, kanonikus alak). Kombinációs Hálózatok tervezése

Előadó: Dr. Vörösházi Zsolt

voroshazi.zsolt@virt.uni-pannon.hu

Kapcsolódó jegyzet, segédanyag:

- <http://www.virt.uni-pannon.hu>
→ Oktatás → Tantárgyak → Digitális
Áramkörök (Villamosmérnöki BSc /
Mechatronikai mérnöki BSc/MSc).
- Fóliák, óravázlatok (.ppt)
- Frissítésük folyamatosan



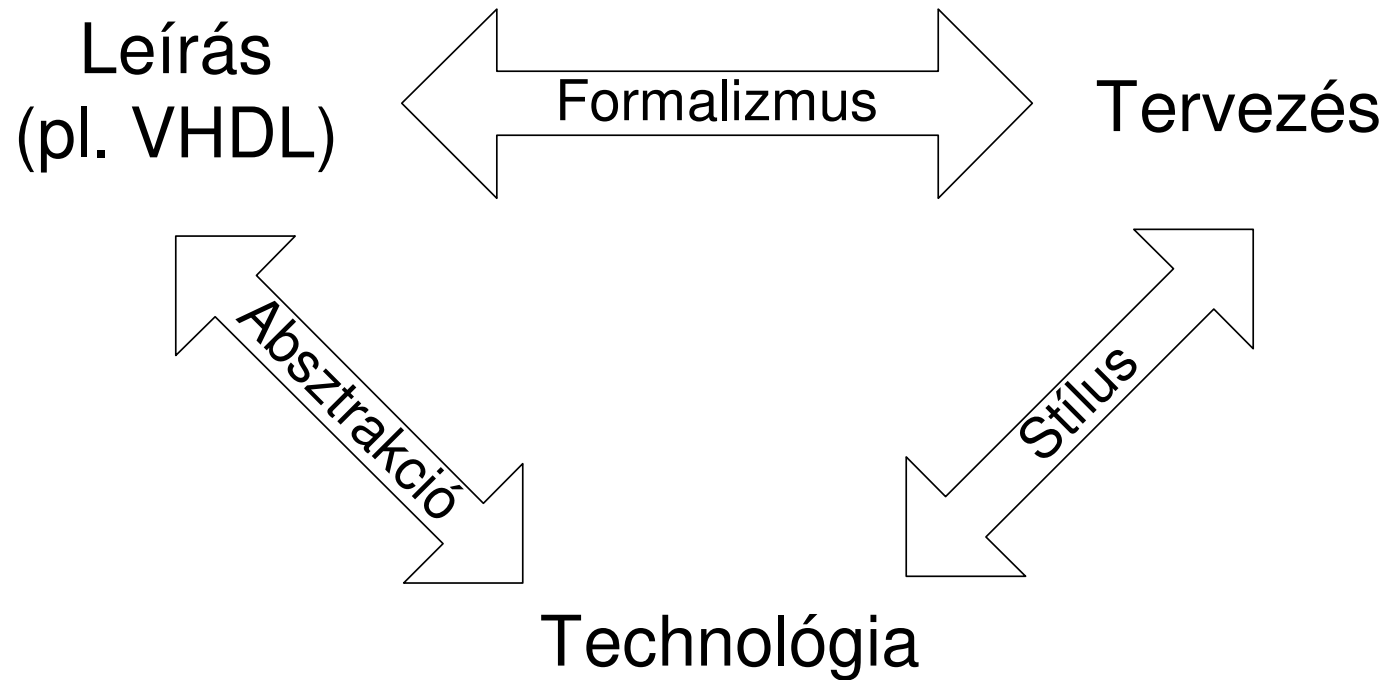
Logikai tervezés alapjai

A logikai tervezés kritériumai

- Tervdokumentációt (követelmény / tervezési specifikációt) kell készíteni a hálózatról
 - Egységes, szabványos jelölésrendszer szükséges
- A berendezésnek könnyen vizsgálhatónak, megbízhatónak kell lennie
 - Tagolt legyen (könnyen átlátható, javítható)
 - Moduláris, strukturált felépítés: hiba esetén csak adott modult kelljen cserélni
- Gyárthatónak kell lennie
 - Mérőpontokat, tesztelő helyeket kell kialakítani (Test pins, JTAG)
 - Építőelem készletet figyelembe kell venni
- Gazdaságosság: lehetőleg minimális építőelem felhasználás
- Működési sebesség: elérhető teljesítmény

Logikai hálózatok/rendszerek tervezése

- Stílus – Absztrakció – Formalizmus

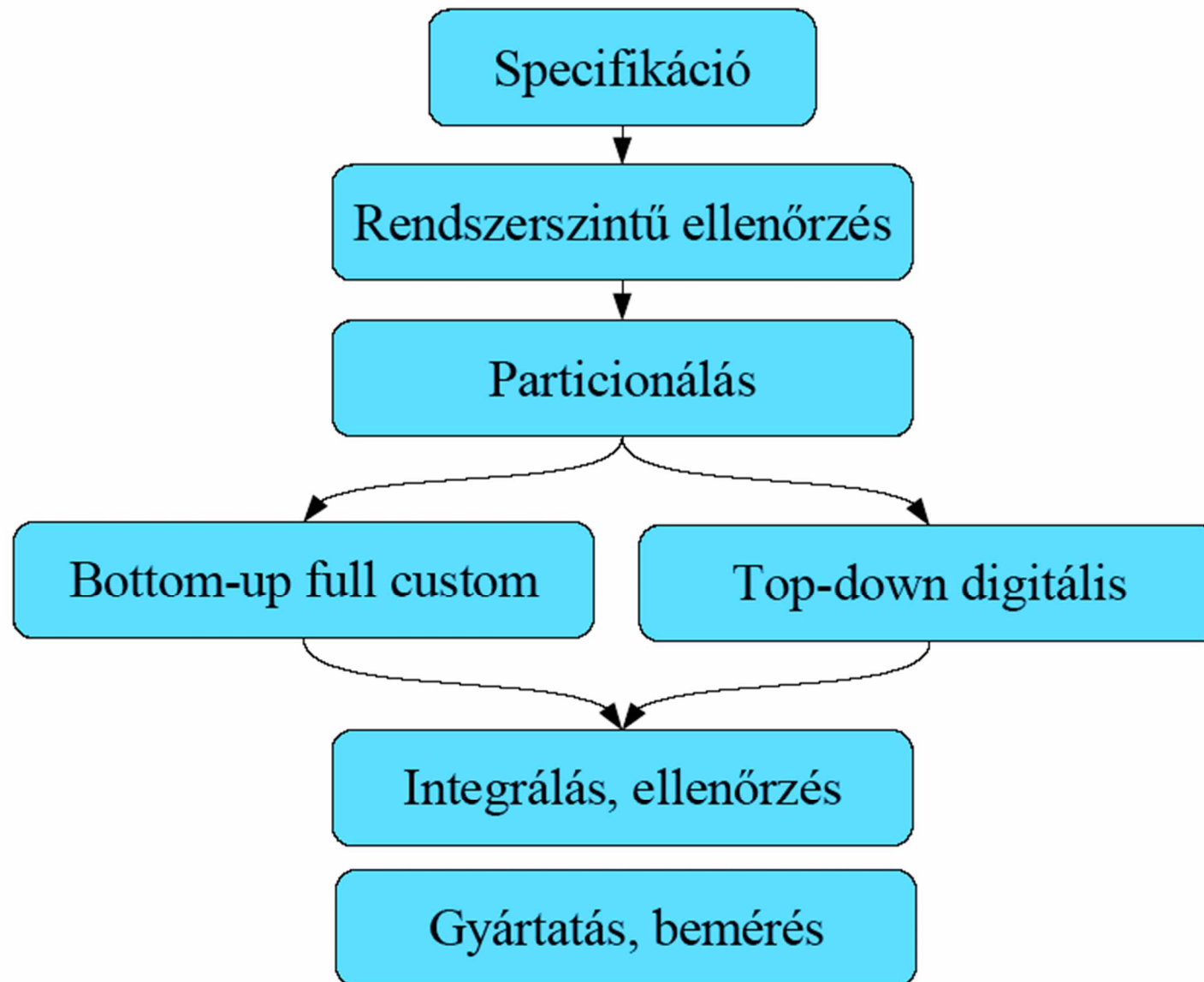


1. Stílus

- Komplex feladat \Rightarrow egyszerűbb, kezelhető részfeladatokra bontása
 - Szisztematikus: módszer
 - Érthető módszerek kellene

Pl: programozási/tervezési stílusok (Top-down, Bottom-up, strukturált stb.)
- Jó stílus kialakításának szabályai:
 - „Top-down”/Bottom-up módszer szerinti tervezés
 - Csak kiforrott, biztos technikákat szabad alkalmazni
 - Fontos a dokumentálás!

Általános Design Flow

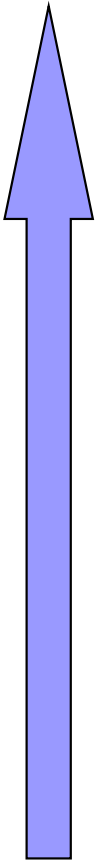


2. Absztrakció

- Digitális tervezés „elvi-fogalmi” szintje
 - Kezdeti absztrakció a tervezés során meghatározó, kritikus pont!
 - 1. koncepcionális modell (elvi elgondolás)
 - 2. megvalósítható, realizálható modell (HW)
 - Magas-szintű absztrakció \Rightarrow elvi modell szintenkénti finomítása és felépítése

Absztrakciós szintek

- Rendszer szint
- Algoritmikus szint
- Funkcionális szint (pl. multiplexer, dekóder, ALU, stb.)
 - RTL szint: regiszter-transzfer leírás (pl. VHDL, Verilog)
- **Logikai szint (kapuk – Boole algebra)**
- Fizikai áramkör szint (tranzisztor - erősen gyártási-technológia függő – pl. deep submicron technology)



3. Formalizmus

- A rendszer viselkedésének leírására szolgál
 - Szisztematikus szabályok és eljárások
 - Minden absztrakciós szinten fontos a használatuk
 - PI: alapvető formalizmus a **Boole-algebra** (bináris logika elmélete) – de csak alsóbb szinteken használható

(felsőbb-, rendszer-szint)

Absztrakció

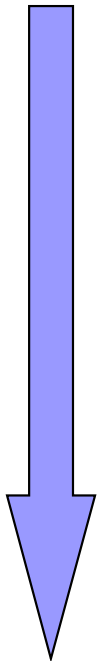
↔

(alsóbb-, áramköri szint)

Boole algebra
(konkretizálás)

Integrált áramkörök osztályozása komplexitás (integráltsági fok) szerint:

- SSI (Small-Scale Integration): ~10 alacsony-szintű elemet (kaput/tranz.) tartalmaz
- MSI (Medium-Scale Integration): 10-1000
- LSI (Large-Scale Integration): 1000-10000
- **VLSI (Very-Large-Scale Int.): >100000**
- **ULSI (Ultra-Large Scaling Int.) > 1 millió**
 - Mikroprocesszorok (2 milliárd tr. – '2010-től)
 - Memóriatömbök (256/512Gbit Toshiba - 2017)



Fogalmak: ASIC, VLSI

- **VLSI:** Nagyon-nagy integráltsági fokú áramkörök – elektromosan programozható
- **ASIC:** Application Specific IC (BOÁK: Alkalmazás v. Berendezés Orientált Áramkörök) – maszk-programozható eszközök
 - Nagy integráltság - Csökkenő komponens költség
 - Növekvő teljesítmény
 - Növekvő „csomag” sűrűség – lábszám (package)
 - Olcsó összeszerelés (assembly)
 - Kis disszipáció, nagy kapcsolási sebesség

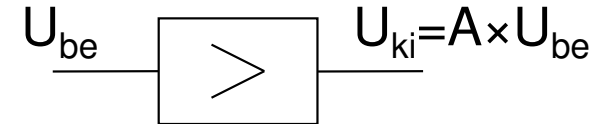
Logikai feladat definiálása:

- A *logikai* feladat definiálását a könnyebb megérthetőség miatt tegyük egy **ellenpéldával**
 - Pl: egy *analóg* feszültségerősítő esetén

U_{be} = bemenő jel (feszültség)

U_{ki} = kimenő jel (feszültség)

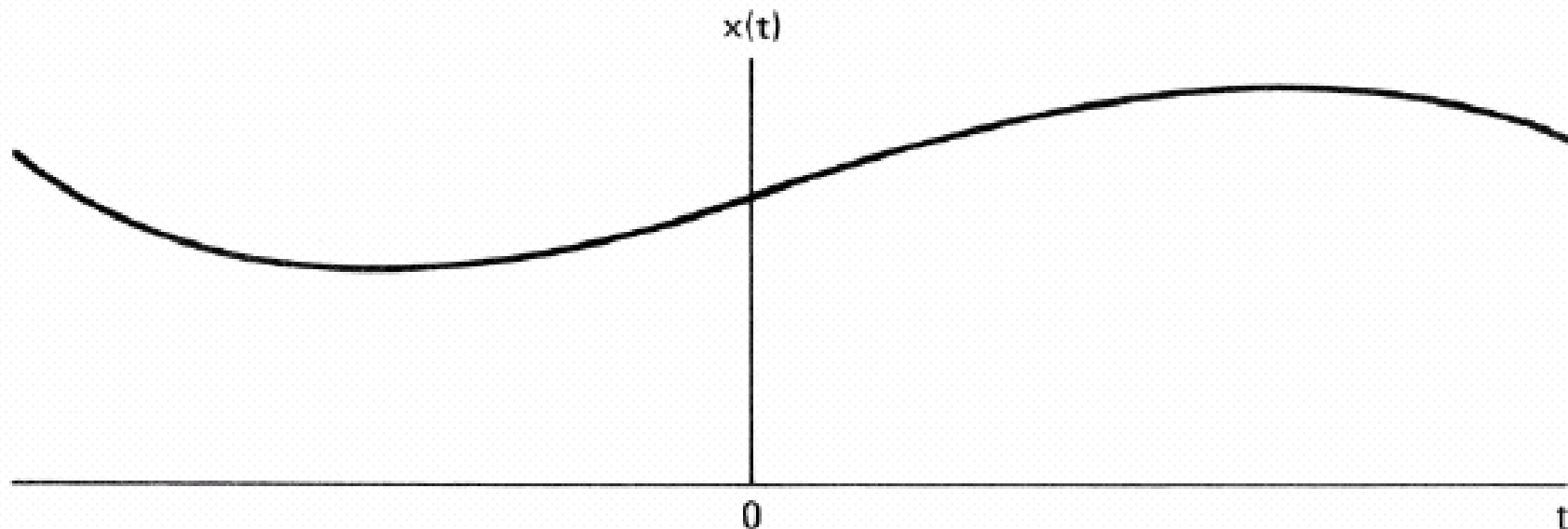
A = erősítési tényező



- Az *analóg* rendszer esetén, csak az van definiálva, hogy a be-, ill. kimeneti jel között mi a kapcsolat (jelen esetben: $U_{ki} = A \times U_{be}$), míg a bemeneti érték egy bizonyos tartományon belül végtelen számú lehet, és ehhez végtelen számú kimeneti érték tartozhat.

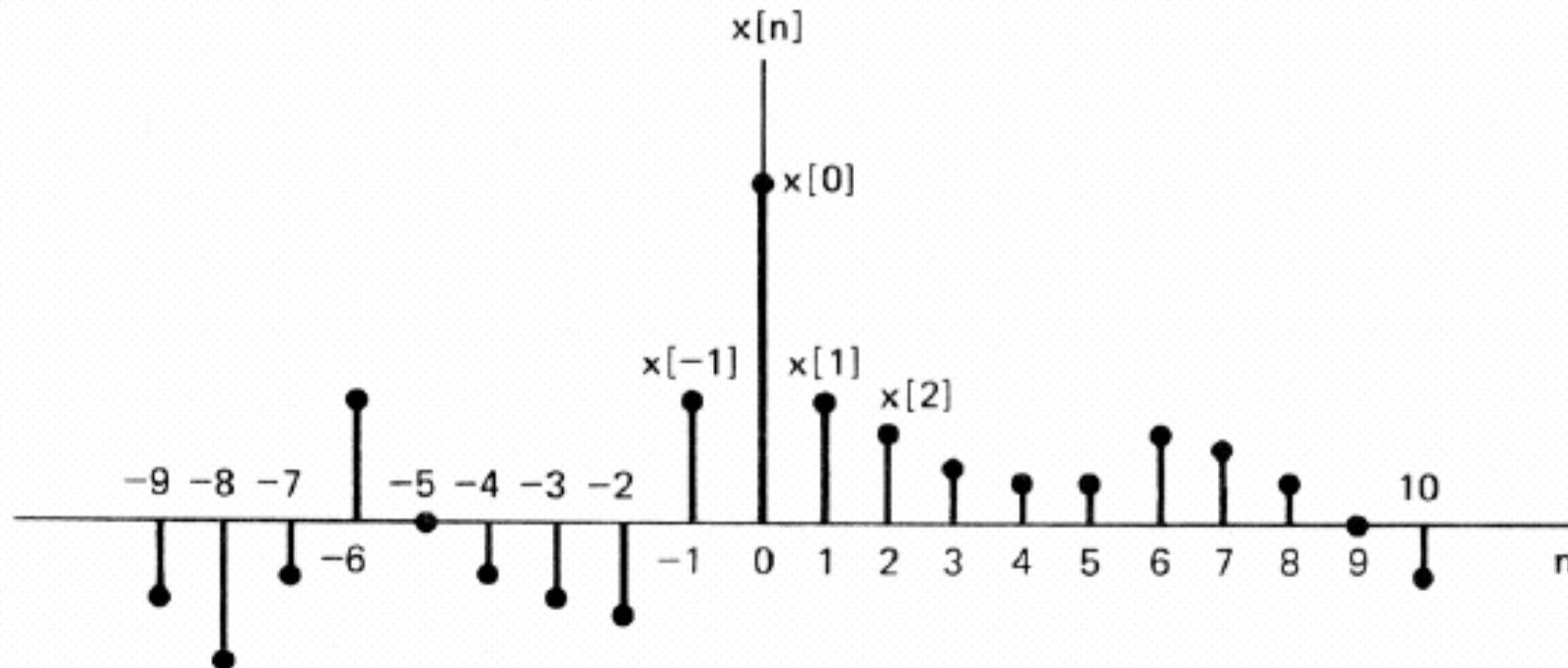
Def: Folytonos jelek

- **Def: Folytonos (analóg) változójú jelek:** Olyan jelek amelyeknek minden független változója folytonos. A mi vizsgálataink szerint az ilyen csupán az *időtől folytonosan függő jeleket* folytonos idejű jeleknek (Continuous Time Signals) azaz **CT** (jelölés) nevezzük. A folytonos idejű jelek két időpont között *végtelen* sok értékre vannak definiálva, ezért *bármely* időpontra felvesznek értéket.

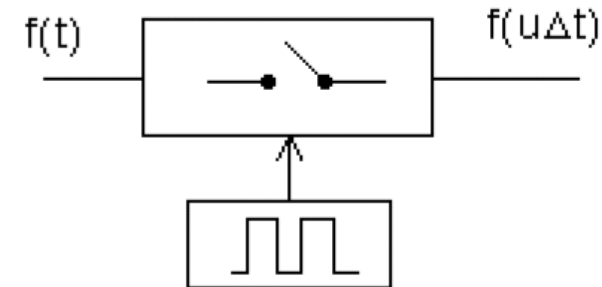
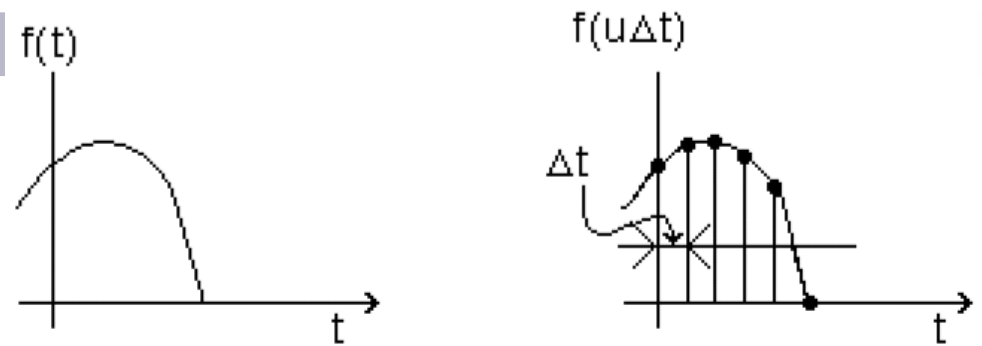


Def: Digitális jelek

- **Def: Diszkrét változójú jelek.** Olyan jelek amelyeknek minden független változója diszkrét értékű. A mi vizsgálataink szerint az ilyen, csupán az *időtől diszkrétén függő jeleket* diszkrét idejű jeleknek (Discrete Time Signals) azaz **DT** (jelölés) nevezzük. A diszkrét idejű jelek csak diszkrét időpontokra vannak definiálva, ezért mindig csak meghatározott időközökben vesznek fel értékeket, és az időközök között nem definiáltak.



Mintavételezés csoportosítása:



- A **jeltől független** mintavételezés, **ekvidisztáns** időközönként. Ezeket lineáris, rögzített lefolyású mintavevő rendszereknek nevezzük.
- A **jeltől függő** mintavételezés, amikor a változás sebességének növekedése pontosabb ábrázolást igényel, de gazdasági okokból a mintavételezés gyakoriságát valamilyen jellemzőnek a változásához kötjük. Ezeket **nem lineáris**, jeltől függő mintavevő rendszereknek nevezzük.
- **Statisztikai** mintavételezés, általában a manuális mintavételezés tartozik ide.

ADC: analóg-digitális konverzió

- Digitalizálás két lépésben történik: (folyt. (A) → diszkrét → digitális (D))
 - 1. Az analóg jelből (A) *mintavételezéssel* (sampling) előállított diszkrét, majd
 - 2. *kvantálással* (quantization) előállított digitális (D) / bináris jel.
- Mintavételezési gyakoriság problémája: „aliasing” effektus (gyors jelváltozásból). Mintavétel: konstans számú minta / sec. Lehet
 - Időbeli: ált. pl. hang, video esetén
 - Térbeli: valamilyen minta szerinti (pl. Moire pattern)
- Kvantálási pontosság problémája: Kvantálási zaj $e_q(t)$
- Ezek a hibák kezelhetők, szabályozhatók megfelelően tervezett szűrők segítségével: quantizer, anti-aliasing filter stb. és betartva a következőt:
 - **Nyquist-Shannon mintavételi elv alkalmazása:** $f > 2f_0$: tehát a mintavételezési frekvenciának (f) nagyobbnak kell lennie, minimálisan a jel sávszélességének kétszeresénél

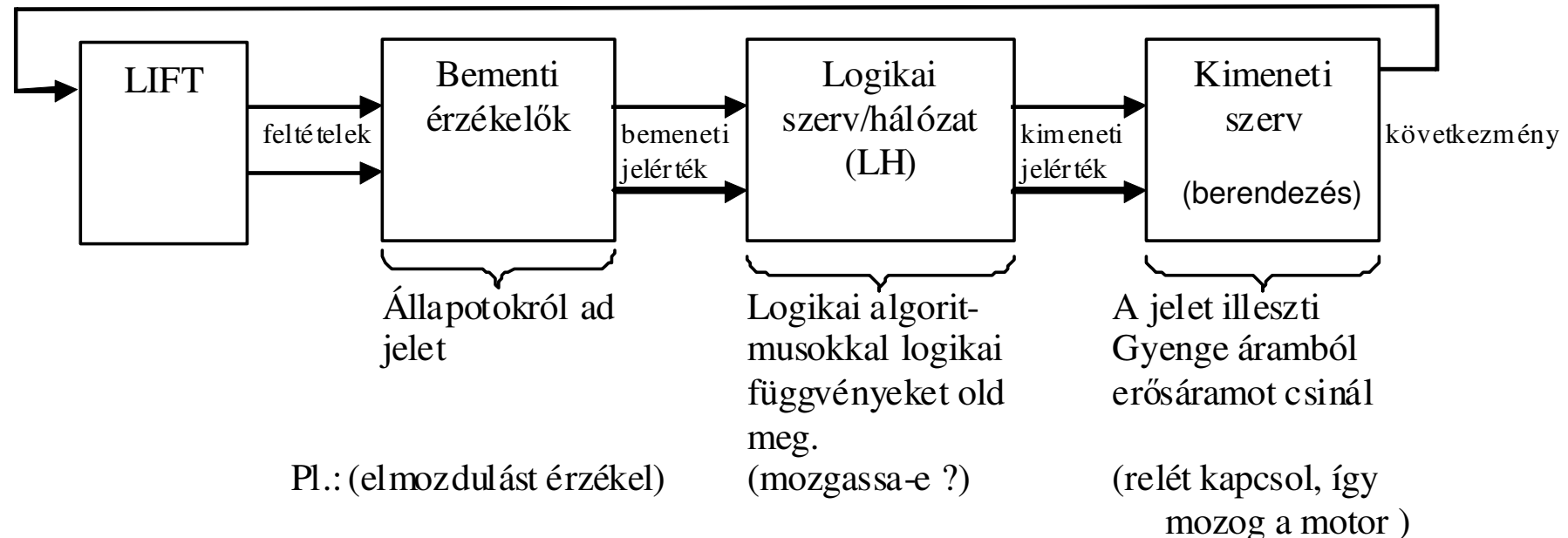
Definíciók-Alapfogalmak:

- *Hogyan ábrázolják ezeket a jelértékeket a fizikai hardver eszközök? Egy hardver-tervező az implementáció során saját maga szabhatja meg, hogy mit-mivel jelöl.*
- **Digitális változó:** eldöntendő kérdésre adott egyértelmű válasszal adható meg.
- **Logikai érték:** a kérdésre adott válasz igen vagy nem.
 - Logikai konstansok:
 - Logikai állítás: Igaz / Hamis, True / False, 1 / 0, Asserted/Non-Asserted
 - **Logikai (bináris) változók:**
 - Pl: 'A' logikai változó esetén legyen, A:=fotódióda hiba
'A' lehet: T / F (A=F nincs hiba; A=T hiba)
 - Logikai változó neve lehetőleg utaljon a funkciójára
 - Logikai operátorok:
 - Felírásuk igazság táblázattal (Truth Table)

Neumanni alapelvek

- **Neumann János** (1903-1957)
 - használjon kettes (**bináris**) számrendszert
 - a számítógép legyen teljesen elektronikus
 - legyen külön vezérlő (CU) és végrehajtó (ALU) egysége
 - Memória struktúra: adat-, és program egyazon memóriában helyezkedjen el, azaz a belső tárban (Tárolt programozás elve)
 - A számítógép legyen egy univerziális Turing-gép (digitális számítás elmélet)

Példa: LIFT vezérlése



Liftmotor mozgatás

Ebben a feladatban azt definiáljuk (az úgynevezett igazságtábla segítségével), hogy adott bemeneti értékek mellett mikor induljon el a lift. Jelen esetben az igazságtáblánknak $2^4=16$ sora kell, hogy legyen, mert 4 bemeneti változónk van.

- Bemeneti változó:
 - nyomtak-e gombot (GNY)
 - liftajtó be van-e csukva (LZ)
 - van-e valaki a liftben (VL)
 - túlterhelt-e (TT)
- Kimeneti változó:
 - Bekapcsolja-e a motort (0 v. 1)

Igazságtábla: Lift vezérlése

Automatikus feltételek				következmények	
GNY	LZ	VL	TT	Liftmotor	
0	0	0	0	0	nem indul
0	0	1	0	0	nem indul
1	0	1	0	0	nem indul
1	1	1	0	1	indul
1	1	1	1	0	nem indul
1	1	0	1	-	NTSH*

NTSH*: Nem Teljesen Specifikált Hálózat, don't care „közömbös ‘állapotokkal’

Logikai hálózatok csoportosítása a megoldandó logikai feladatok szerint

- **Logikai hálózat:** ált. készen kapható logikai építőelemekből állítható össze
- **Logikai rendszer:** logikai hálózatoknak egy adott feladat megoldása céljából együttműködő összességét logikai rendszernek nevezzük.
- Mivel a logikai feladat megoldása szempontjából, nem mindig elég a mindenkor fennálló pillanatnyi bemeneti kombinációk figyelembevétele a kimeneti kombinációk előállításához, szükségünk lehet **másodlagos** (ún. **szekunder**) **feltételeknek** megfelelő szekunder kombinációkra is.
 - **PI: a lift vezérlése:** nem elég, ha a liftmotor a feltételek teljesülése után, csak úgy elindul, mert nem mindegy *melyik irányba* forog (fel v. le). Hiszen, ha az 1.-ről akarunk eljutni a 3.-ra, akkor más irányba kell forognia, mintha az 5.-ről akarnánk eljutni a 3.-ra. Tehát a szekunder változó ebben az esetben a *felvonó mindenkor helyzetéről* ad felvilágosítást érzékelők segítségével. (A szekunder változó feladata a hálózat belső állapotának figyelése.)

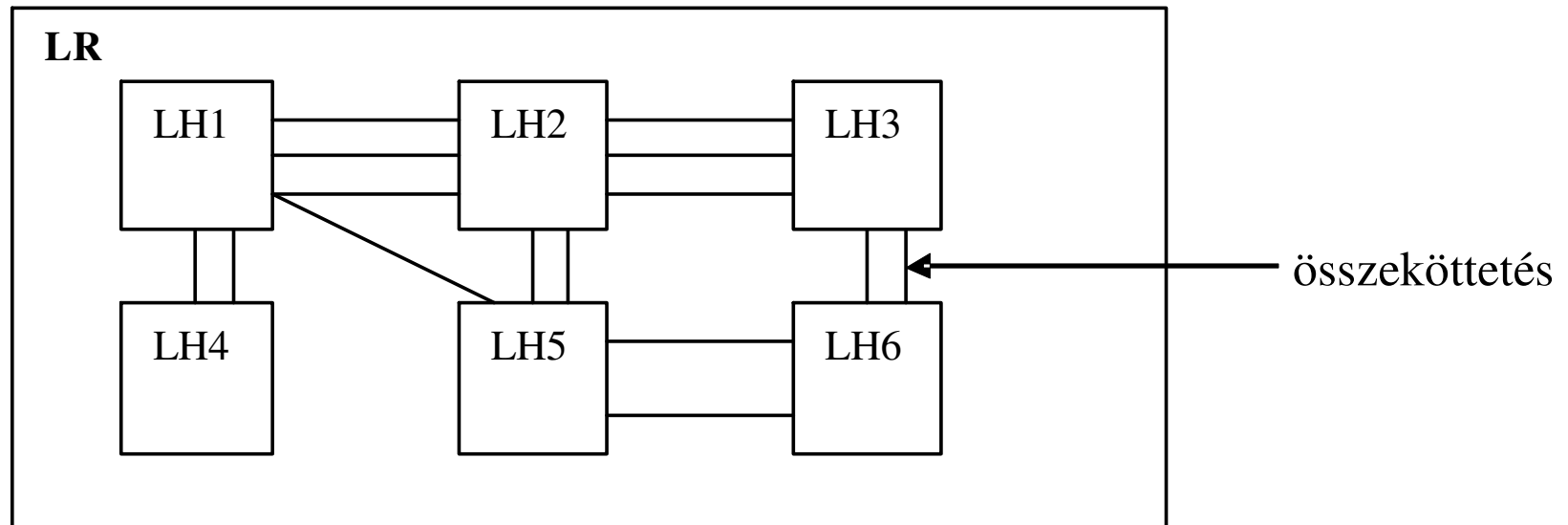
Logikai hálózatok csoportosítása

Ezek alapján kétféle hálózatot különböztetünk meg

- **(K.H.) Kombinációs logikai hálózat**ról beszélünk: ha a mindenkori kimeneti kombinációk létrehozásához *elég a bemeneti kombinációk pillanatnyi értéke*
- **(S.H.) Sorrendi (szekvenciális) logikai hálózat**ról beszélünk: ha a mindenkori kimeneti kombinációt, nemcsak *a pillanatnyi bemeneti kombináció*, hanem *a korábban fennállt bemeneti kombinációk és azok sorrendje* is befolyásolja. (A *szekunder kombinációk* segítségével az ilyen hálózatok képessé válnak arra, hogy az ugyanolyan bemeneti kombinációkhoz **más-más kimeneti kombinációt** szolgáltatassanak, attól függően, hogy a bemeneti kombináció fellépésekor, milyen értékű a szekunder kombináció)

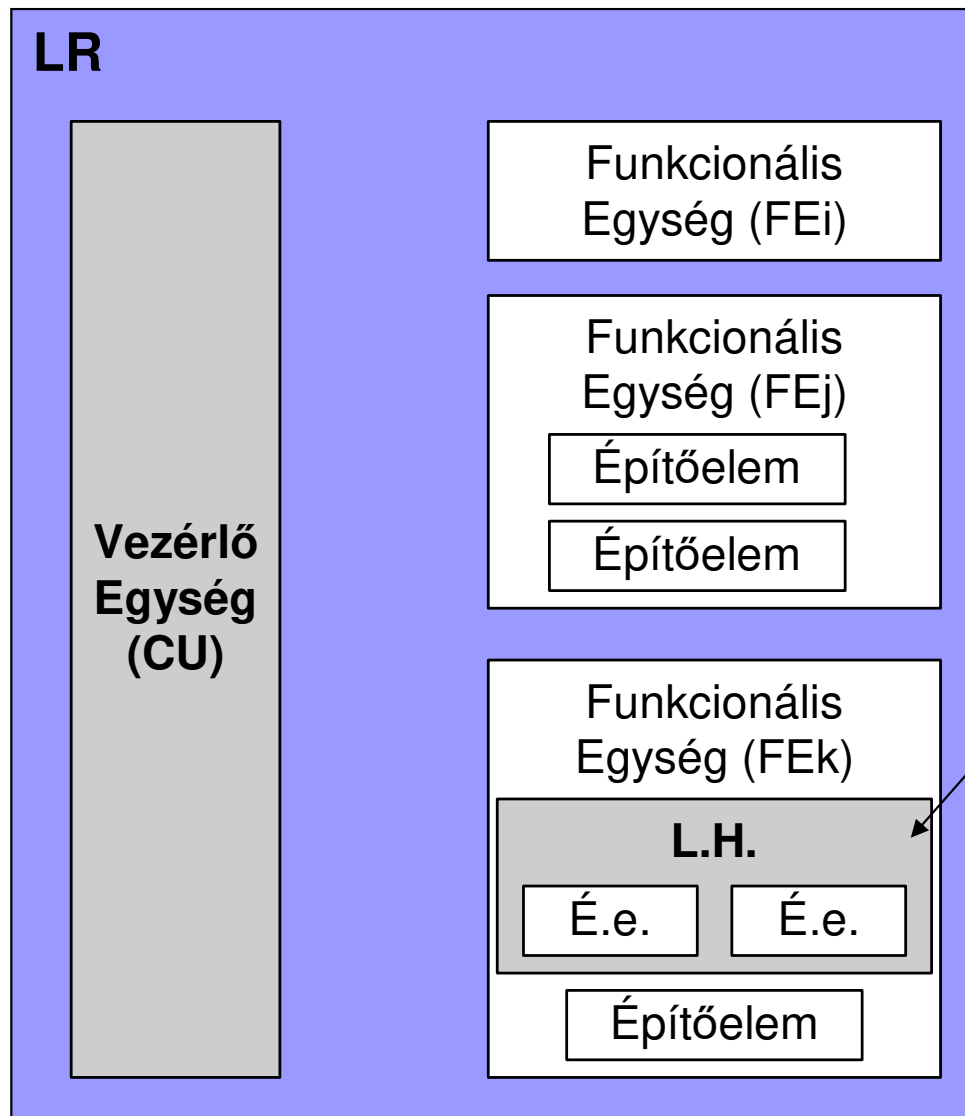
A logikai hálózat és a logikai rendszer kapcsolata

- **Logikai rendszer (LR):** logikai hálózatokból (LH) épül fel (be-, és ki-menetekkel)



- Egy bonyolult logikai feladat megoldásához nem célszerű egy bonyolult logikai hálózatot (LH) tervezni; sokkal célszerűbb egyszerű LH-k összehangolt működésével megoldanunk a problémát. Az LH-k összehangolt működésű rendszerét logikai rendszernek (LR) nevezzük.

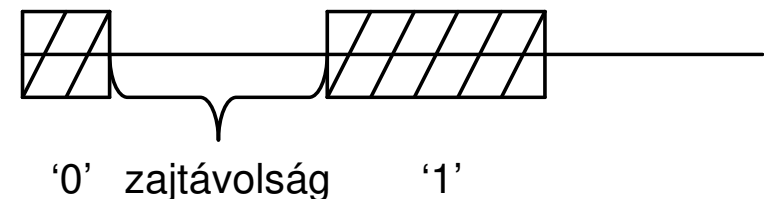
Logikai rendszer hierarchikus felépítése



Látszólagos építőelem,
ami ha valódi
építőelemmé válik
generációváltásról
beszélünk.

Digitális jelértékek ábrázolása fizikai eszközöknél:

- Lyukkártya: lyuk=1, nincs lyuk=0
- Mágneses szalag / lemez: a hordozó felület adott oldalának felmágnesezése
- Kapcsolók: két állapota: nyitott=0, zárt=1
- Feszültség logikai szintek: pl: TTL 74LS esetén:
 - $0.0 - 0.5V$ = alacsony szint ('0') - L
 - $2.7 - 5V$ = magas szint ('1') - H



Zajtávolság: standard 0 – 5 V feszültség tartomány esetén

Feszültség

0-5 V-os tartományban

„0” = 0-0,8V-ig

„1” = 2-5 V-ig

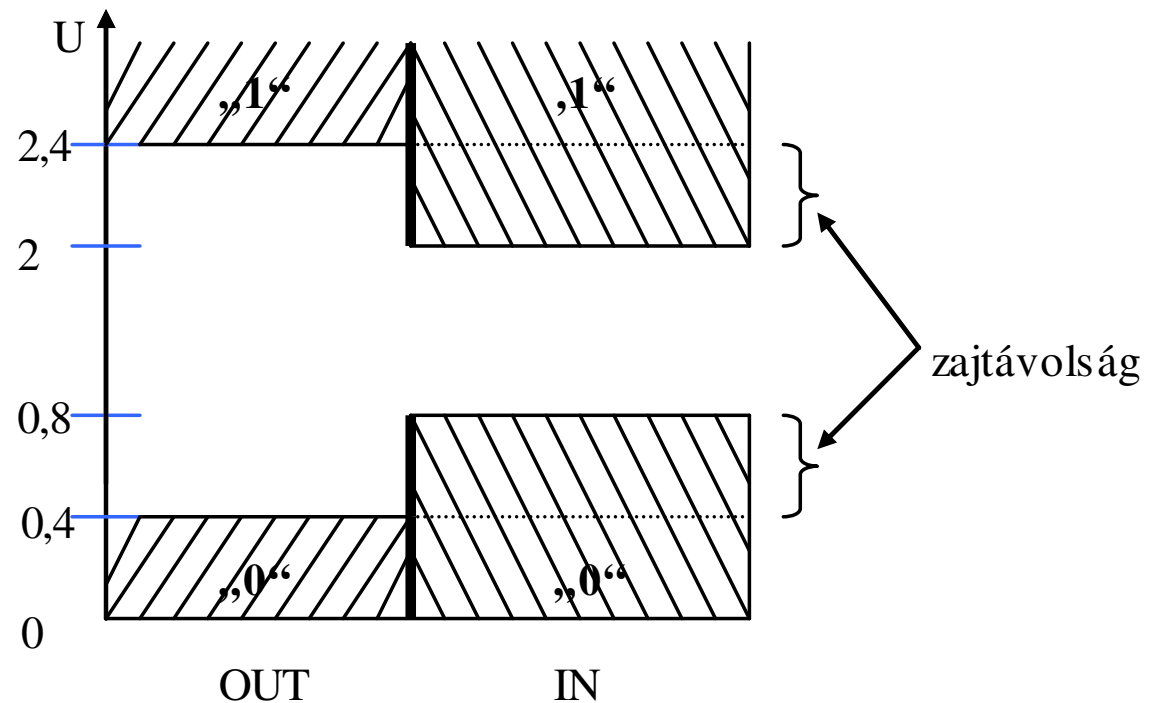
0,8-2-ig nem értelmezzük

A zajtávolság azért kell, mert ha a feszültség pont a határon van akkor a különféle külső környezeti hatások (zajok) miatt rossz értéket kaphatunk.

Ilyen zajok Pl.:

a termikus zajok -- a vezeték hőmozgása;

az induktív zajok -- külső elektromágneses tér, aminek a hatása akár pár V is lehet az eltérés;



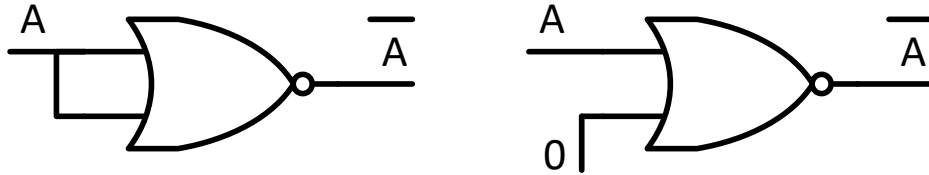


Logikai operátorok és igazságtáblájuk

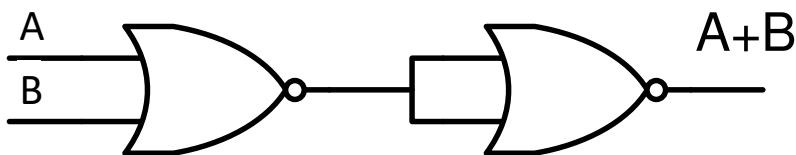
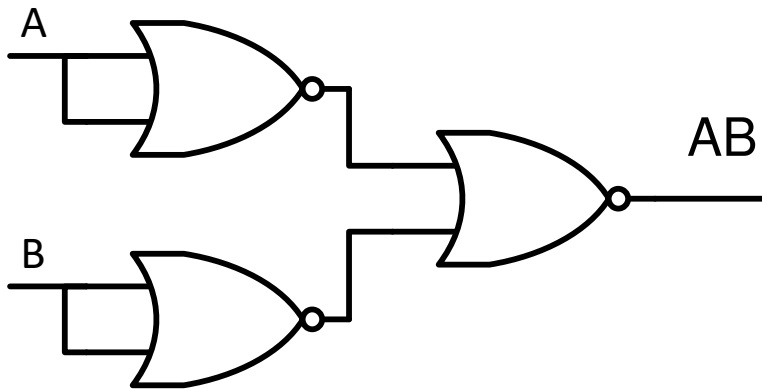
Logikai operátorok

- Fajtáik:
 - Egy-változós
 - Kettő-, vagy több-változós
- Három alapművelet:
 - NOT (NEM)
 - AND (ÉS)
 - OR (VAGY)
- **Funkcionális / Univerzális teljesség:** bizonyos logikai műveletek használatával bármely tetszőleges más logikai függvény megadható.
 - CMOS technológiában ilyenek a NAND, NOR kapuk!

Funkcionális teljesség: példák



Funkcionálisan teljes vagy univerzális áramköri alapelemek: Logikai hálózatok esetén a CMOS NAND, illetve NOR kapu.



(Aritmetikai egységek esetében ilyen univerzális építőelem az 'összeadó'.)

Igazságtábla: logikai operátorok felírása

- 'F' logikai függvény megadása az 'A,C,B' logikai változók összes lehetséges értékétől függően

Jel: $F(A,C,B)$ //3 változó $\rightarrow 2^3 = 8$ sor//

A	C	B	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0



A	C	B	Fgv
F	F	F	F
F	F	T	T
F	T	F	T
F	T	T	F
T	F	F	T
T	F	T	F
T	T	F	F
T	T	T	F

sor
0.
1.
2.
3.
4.
5.
6.
7.

- Kanonikus „standard” igazság tábla:
000 – 111 -ig (3 változó esetén)

Logikai operátorok és igazság táblázatuk (NOT)

- Jel: $\text{NOT } A = \bar{A}$
- Formális definíció igazságtáblával:

A	NOT A
0	1
1	0

- Def:
 - ha 'A' hamis, 'NOT A' igaz
 - ha 'A' igaz, 'NOT A' hamis

Logikai operátorok és igazság táblázatuk (AND)

- Jel: B **AND** C = B·C
- Formális definíció igazságtáblával:

B	C	B·C
0	0	0
0	1	0
1	0	0
1	1	1

- Def: B·C értéke pontosan akkor 'igaz' ha 'B' és 'C' is egyszerre 'igaz', különben hamis

Logikai operátorok és igazság táblázatuk (OR)


- Jel: $B \text{ OR } C = B+C$
- Formális definíció igazságtáblával:

B	C	B+C
0	0	0
0	1	1
1	0	1
1	1	1

- Def: $B+C$ értéke pontosan akkor 'igaz', ha 'B' és 'C' közül legalább az egyik 'igaz', különben hamis

Smart áramkörök 😊



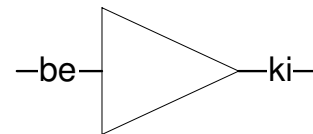


Egy- ill. két-változós logikai függvények bemutatása és szabványos jelöléseik

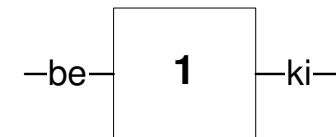
Egyváltozós logikai függvények:

- Jelmásoló („buffer” - jel-erősítő):

be	ki
0	0
1	1



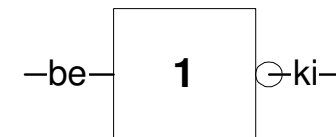
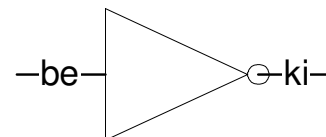
Nemzetközi
szabvány



Magyar
szabvány

- Negálás - Inverter (NOT): \bar{A}

be	ki
0	1
1	0

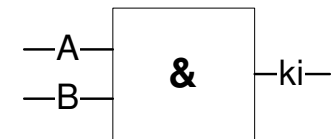
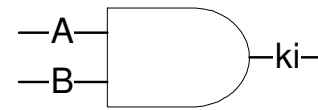


Kétváltozós logikai függvények:

- ÉS (AND):

$$A \cdot B$$

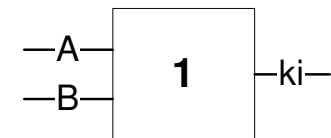
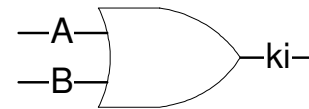
A	B	ki
0	0	0
0	1	0
1	0	0
1	1	1



- VAGY (OR):

$$A + B$$

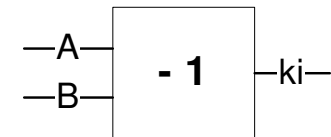
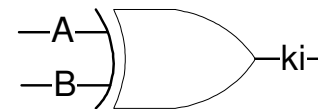
A	B	ki
0	0	0
0	1	1
1	0	1
1	1	1



- Antivalencia (XOR):

$$A \oplus B$$

A	B	ki
0	0	0
0	1	1
1	0	1
1	1	0

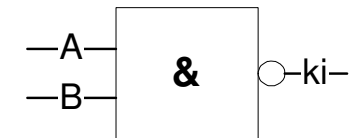
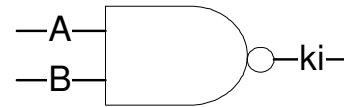


Kétváltozós log.függv. (folyt.):

■ NEM-ÉS (NAND):

$$\overline{A \cdot B}$$

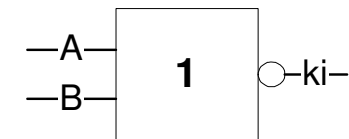
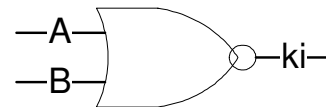
A	B	ki
0	0	1
0	1	1
1	0	1
1	1	0



■ NEM-VAGY (NOR):

$$\overline{A + B}$$

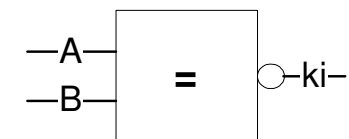
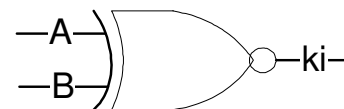
A	B	ki
0	0	1
0	1	0
1	0	0
1	1	0



■ Ekvivalencia (NXOR):

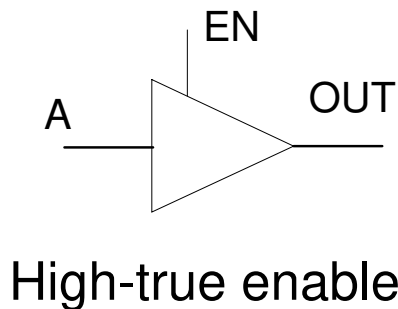
$$A \odot B$$

A	B	ki
0	0	1
0	1	0
1	0	0
1	1	1

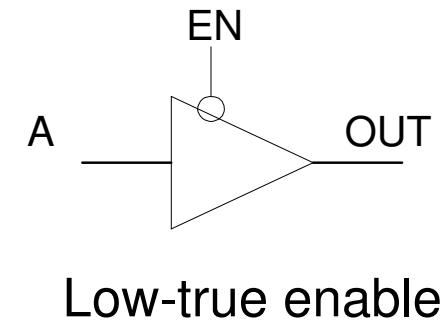


Tri-State Buffer:

- buszok esetén használatos: kommunikációs irány változhat
 - Driver: egyirányú kommunikációra
 - Transceiver: kétirányú kommunikációra
- 3 állapota lehet:
 - magas: '1'
 - alacsony: '0' (normál TTL szintek)
 - nagy impedanciás áll: 'Z' – mindkét kimeneti tranzisztor zár



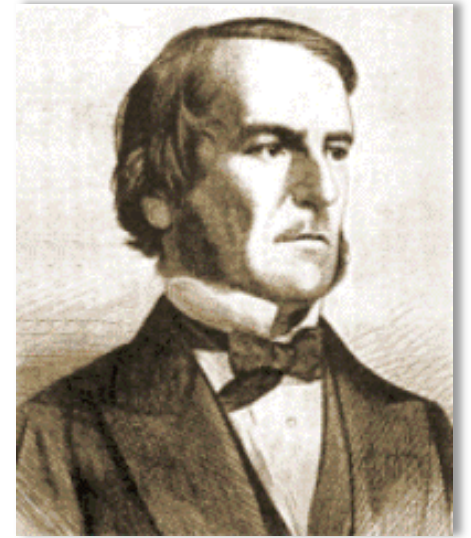
A	EN	OUT
0	1	0
1	1	1
X	0	Z





Boole algebra

Boole-algebra



(1815-1864)

- Logikai operátorok algebrája
- George Boole: először mutatott **hasonlóságot** az általa vizsgált **logikai operátorok** és a már jól ismert **aritmetikai operátorok között**.
- HW tervezés alacsonyabb absztrakciós szintjén rendkívül fontos szerepe van.
(Specifikáció + egyszerűsítések)

Boole algebra elemei:

- 3 alapl művelet: AND, OR, NOT
 - Tulajdonságaik (AND, OR esetén):
 - Kommutatív: $A+B=B+A$, $A \cdot B=B \cdot A$
 - Asszociatív: $A+(B+C)=(A+B)+C=A+B+C$
 $A \cdot (B \cdot C)=(A \cdot B) \cdot C=A \cdot B \cdot C$
 - Disztributív: $A \cdot (B+C) = A \cdot B + A \cdot C$,
 $A+(B \cdot C)=(A+B) \cdot (A+C)$
 - Operátor **precedencia** (csökkenő sorrendben):
 - NOT
 - AND
 - OR
- átzárójelezhetőség!

Boole algebrai azonosságok!

$$1.) \overline{\overline{A}} = A$$

NOT

$$2.) A + 0 = A$$

$$3.) A + 1 = 1$$

$$4.) A + A = A$$

$$5.) A + \overline{A} = 1$$

OR

$$6.) A \cdot 1 = A$$

$$7.) A \cdot 0 = 0$$

$$8.) A \cdot A = A$$

$$9.) A \cdot \overline{A} = 0$$

AND

$$10.) A + A \cdot B = A \quad *$$

$$11.) A \cdot (A + B) = A \quad *$$

Elnyelési
tul.

$$12.) A \cdot B + A \cdot \overline{B} = A$$

$$13.) (A + B) \cdot (A + \overline{B}) = A$$

$$14.) A + \overline{A} \cdot B = A + B \quad *$$

$$15.) A \cdot (\overline{A} + B) = A \cdot B$$

De-Morgan azonosságok:

$$18.) \overline{\overline{A + B}} = \overline{\overline{A}} \cdot \overline{\overline{B}}$$

$$19.) \overline{\overline{A \cdot B}} = \overline{\overline{A}} + \overline{\overline{B}}$$

DUAL
ITÁS

Boole-algebrai azonosság igazolása igazságtáblával

■ Pl: De Morgan

$$\overline{A \cdot B} = \overline{A} + \overline{B}$$

Dualitás elve

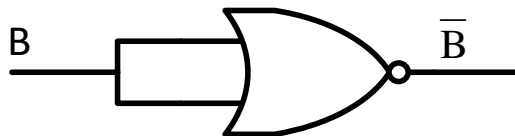
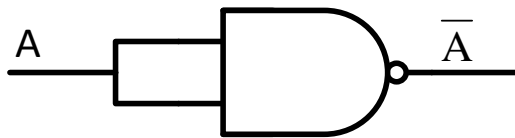
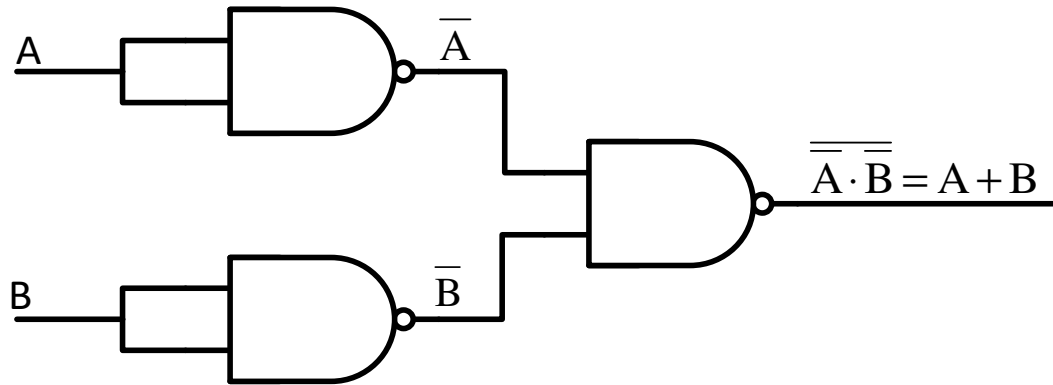
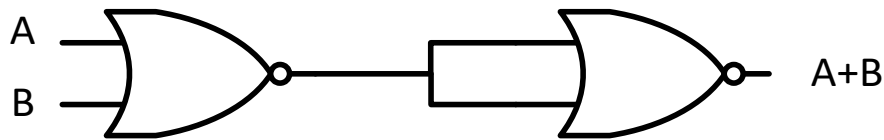
A	B	A·B	NOT (A·B)
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

A	B	NOT A	NOT B	NOT(A) + NOT(B)
0	0	1	1	1
0	1	1	0	1
1	0	0	1	1
1	1	0	0	0

■ Példa: egyszerűsítésre

$$\overline{A \cdot (B + C \cdot (B + \overline{A}))} = \overline{A} + \overline{B}$$

Funkcionális teljesség: példák



Funkcionálisan teljes, vagy univerzális áramköri alapelemek: VLSI CMOS logikai hálózatok esetén a **NAND**, illetve **NOR** kapu.

(Aritmetikai egységek esetén ilyen univerzális építőelem az „**összeadó**”.)

Logikai egyenletek megadása igazság-táblázat alapján

■ P|:

sor	A	B	F
0	0	0	0
1	0	1	0
2	1	0	1
3	1	1	0

F pontosan akkor lesz **igaz**, ha A igaz és B hamis, egyébként F **hamis** lesz. Vagyis egyenletként kifejezve:

$$F = A \cdot \bar{B}$$

■ P|:

sor	A	B	F
0	0	0	1
1	0	1	0
2	1	0	1
3	1	1	1

F pontosan akkor lesz **igaz**, ha A és B is hamis, vagy A igaz és B hamis, vagy A és B is igaz, egyébként F **hamis** lesz. Vagyis egyenletként kifejezve:

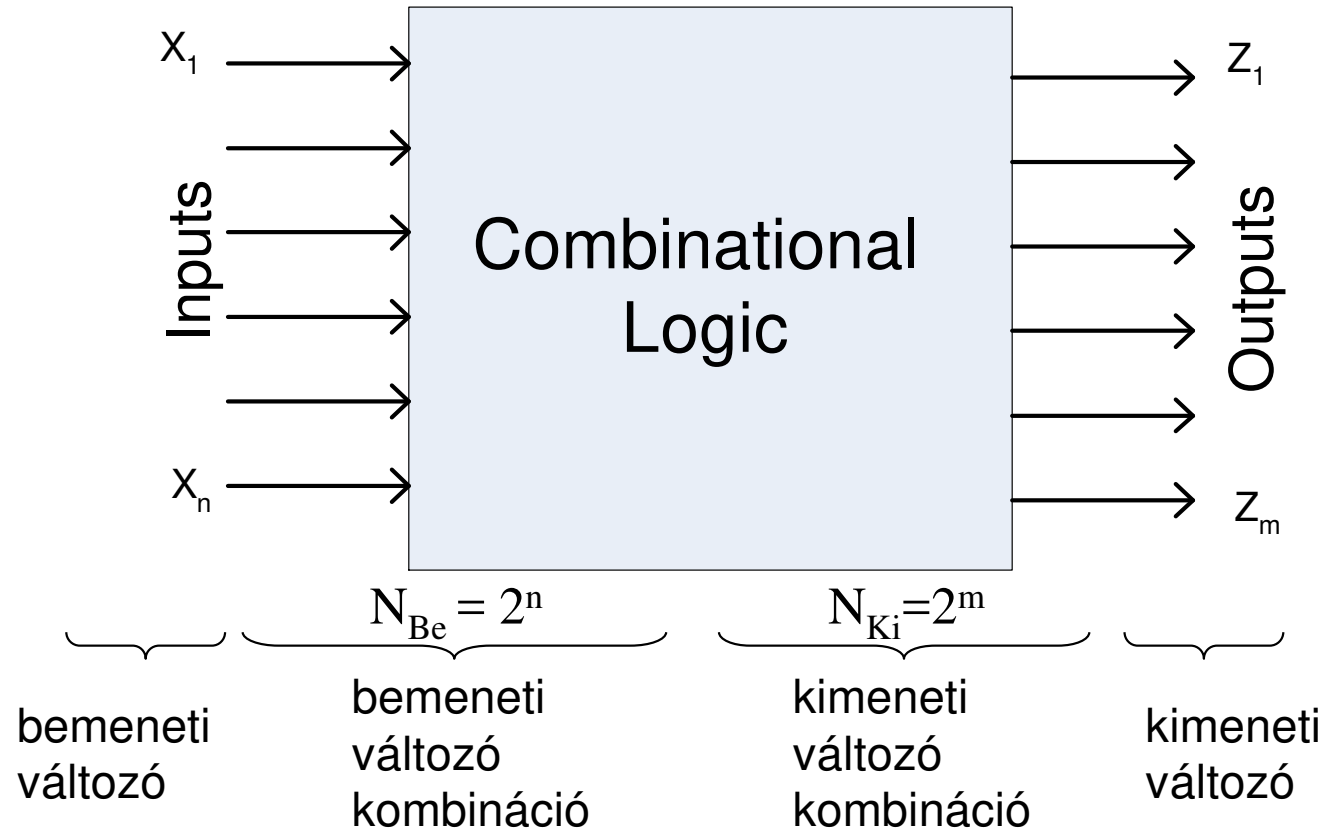
$$1 = F = \bar{A} \cdot \bar{B} + A \cdot \bar{B} + A \cdot B \Rightarrow \bar{B} + A \cdot B \Rightarrow \bar{B} + A$$

$$0 = \bar{F} = \bar{A} \cdot B$$

Def: Logikai függvény

- Független változó: bemeneti változó(k)
- Függő változó: kimeneti változó(k)
- **Logikai függvény:**
 - minden kimeneti változó értéke a függvénykapcsolat alapján határozható meg (minden egyes bemeneti kombinációhoz meghatározható a kimeneti változó).
 - Egyértelmű hozzárendelés (de nem kölcsönösen egyértelmű!)

Teljesen határozott logikai hálózat hány különböző függvénnel írható le összesen?



$$N_f = N_K^{N_B} = (2^m)^{2^n}$$

PÉLDA: 4 bemenet, 2 kimenet esetén a függvények lehetséges száma:
 $N_f = (2^2)^{(2^4)} = 4^{16}$ logikai függvényt lehet definiálni (teljesen határozott LH)

Nem teljesen határozott logikai hálózat hány különböző függvénnel írható le összesen?

- d : közömbös bemeneti kombinációk száma ($0 \leq d < 2^n$)

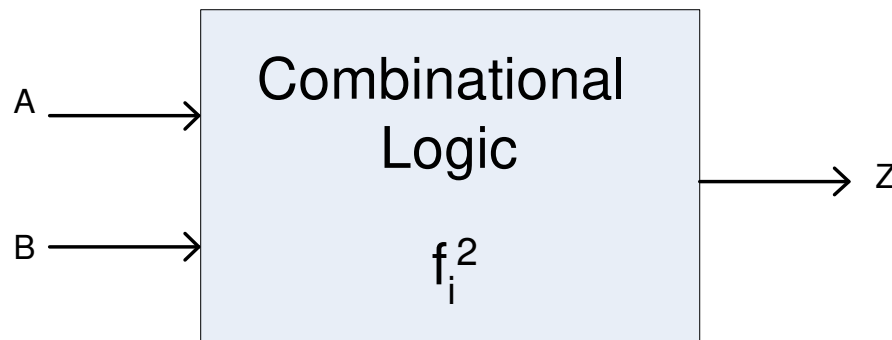
- Egy teljesen határozott log. feladatból \rightarrow annyi nem teljesen határozott feladatot tudunk képezni, ahányféleképpen 2^n bemeneti kombinációból d számút közömbössé tudunk tenni.

- Ekkor a *nem teljesen határozott* logikai feladat

$$\binom{2^n}{d} \text{-szereése a teljesen határozott logikai}$$

feladatok számának.

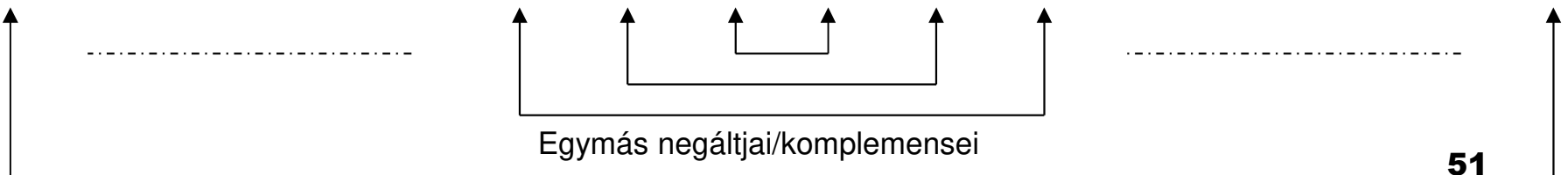
Pl. Összes lehetséges két-bemenetű / egy-kimenetű logikai függvény igazságtáblája



$$N_f = N_K^{N_B} = (2^m)^{2^n}$$

$$= (2^1)^{2^2} = 2^4 = 16$$

A	B	f_0^2	f_1^2	f_2^2	f_3^2	f_4^2	f_5^2	f_6^2	f_7^2	f_8^2	f_9^2	f_{10}^2	f_{11}^2	f_{12}^2	f_{13}^2	f_{14}^2	f_{15}^2
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1



$$Z = f_0^2(A, B) \equiv 0$$

$$Z = f_1^2(A, B) = A \cdot B$$

$$Z = f_2^2(A, B) = A \cdot \bar{B}$$

$$Z = f_3^2(A, B) = A \cdot \bar{B} + A \cdot B \equiv A$$

$$Z = f_4^2(A, B) = \bar{A} \cdot B$$

$$Z = f_5^2(A, B) = \bar{A} \cdot B + A \cdot B \equiv B$$

$$Z = f_6^2(A, B) = \bar{A} \cdot B + A \cdot \bar{B} \equiv A \oplus B$$

$$Z = f_7^2(A, B) = \bar{A} \cdot B + A \cdot \bar{B} + A \cdot B = \\ = \bar{A} \cdot B + A \cdot (\bar{B} + B) = \bar{A} \cdot B + A \equiv A + B$$

$$Z = f_8^2(A, B) = \bar{A} \cdot \bar{B} \Leftrightarrow \overline{f_7^2(A, B)}$$

$$Z = f_9^2(A, B) = \bar{A} \cdot \bar{B} + A \cdot B \equiv A \odot B \Leftrightarrow \overline{f_6^2(A, B)}$$

$$Z = f_{10}^2(A, B) = \bar{A} \cdot \bar{B} + A \cdot \bar{B} \equiv \bar{B} \Leftrightarrow \overline{f_5^2(A, B)}$$

$$Z = f_{11}^2(A, B) = \bar{A} \cdot \bar{B} + A \cdot \bar{B} + A \cdot B = \\ = \bar{B} \cdot (\bar{A} + A) + A \cdot B = A + \bar{B} \Leftrightarrow \overline{f_4^2(A, B)}$$

$$Z = f_{12}^2(A, B) = \bar{A} \cdot \bar{B} + \bar{A} \cdot B = \bar{A} \cdot (\bar{B} + B) = \bar{A} \\ \Leftrightarrow \overline{f_3^2(A, B)}$$

$$Z = f_{13}^2(A, B) = \bar{A} \cdot \bar{B} + \bar{A} \cdot B + A \cdot B = \\ = \bar{A} \cdot (\bar{B} + B) + A \cdot B = \bar{A} + B \Leftrightarrow \overline{f_2^2(A, B)}$$

$$Z = f_{14}^2(A, B) = \bar{A} \cdot \bar{B} + \bar{A} \cdot B + A \cdot \bar{B} = \\ = \bar{A} \cdot (\bar{B} + B) + A \cdot \bar{B} = \bar{A} + \bar{B} \Leftrightarrow \overline{f_1^2(A, B)}$$

$$Z = f_{15}^2(A, B) \equiv 1 \Leftrightarrow \overline{f_0^2(A, B)}$$



Logikai függvények kanonikus (normál) alakjai

1.) Sum-of-Products (szorzat, „termek” összege)

- **Szorzat (AND) termék összege (OR kapcsolata)**
- Emberi szemléletmódhoz közelebb áll: a táblázat soraiból azokat a függvényértékeket (F) vesszük amelyek ‘1’-esek
- Def: Triviális forma: ha egy változó egy adott szorzat termében vagy pozitívan, vagy negatíván legfeljebb egyszer szerepel.
Ezt hívják még **mintermnek** (m_i) vagy **kanonikus szorzat termék** is. Tegyük fel hogy $F(A,B,C)$

□ Pl: valós / triviális / kanonikus formulák: $A, \bar{A}, A \cdot B, \bar{A} \cdot B \cdot \bar{C}$

□ Pl: érvénytelen formulák (de ettől még Boole kifejezés), ami jelenti azt is, hogy tovább egyszerűsíthetők:

$$A \cdot \bar{A}, \bar{A} \cdot B \cdot B \cdot \bar{C}$$

Diszjunktív Normál Forma:

- Jel:
$$F(DNF) : \sum_{i=0}^{2^n-1} m_i$$

- n változó esetén 2^n lehetséges minterm van.
- Képzésük: az igazságtáblázatból azoknak a **mintermeknek a VAGY kapcsolatát** vesszük, ahol függvényértékek sorában (F) '1' -es szerepel, vagy ahol a függvény komplementésének (\bar{F}) értéke '0'.
- minterm: m_i (i . sora a kanonikus táblának, ahol F értéke '1').

Példa: DNF felírása

- Igazságtábla:

sor	A	B	F
0	0	0	1
1	0	1	0
2	1	0	1
3	1	1	1

- Kapott egyenlet: $1 = F = \overline{A} \cdot \overline{B} + A \cdot \overline{B} + A \cdot B = m_0 + m_2 + m_3$
[0 0] [1 0] [1 1]

- Komplementens: $'0' = \overline{F} = \overline{A} \cdot B = m_1 = \overline{(m_0 + m_2 + m_3)}$
[0 1]

2.) Product-of-Sums: összeg„termek” szorzata

- **összeg (OR) termék szorzata (AND kapcsolata)**
- **Maxterm (M_i):** olyan kanonikus összeg term, amelyben **mindegyik** logikai változó **pontosan egyszer** fordul elő, ponált, vagy negált alakban.
 - Tegyük fel hogy $F(P,Q,R)$
 - Valós maxterm: $P + \bar{Q} + \bar{R}$, de nem valós: $P + Q$
 - Kanonikus forma: $F = (P + Q + R)(P + \bar{Q} + \bar{R})$
 - Nem kanonikus forma: $F = (P + Q)(P + \bar{Q} + \bar{R})$
- Gyakorlatban kevésbé használt forma.

KNF: Konjunktív Normál Forma

■ Jel:
$$F(KNF) = \prod_{i=0}^{2^n - 1} M_i$$

■ Képzésük: a kanonikus igazságtábla azon **maxtermjeinek ÉS kapcsolatát** vesszük, ahol a függvény (F) értéke '0', vagy a komplementens függvény (\overline{F}) értéke '1'.

■ Pl: $F = \overline{A} + \overline{B}$ vagy

$$\overline{F} = (A + B) \cdot (\overline{A} + \overline{B}) \cdot (A + \overline{B}) \stackrel{\text{disztributív}}{=} \overline{A \cdot B} = \overline{A \cdot B}$$

■ Maxterm (M_i): az igazságtáblázat i . sora, ahol az F kimeneti függvényérték '0'.

Példák: KNF

- Legyen: $M_i = \bar{A} + B + \bar{C}$ ahol az F kimeneti függvényérték *hamis* volt. Ez az M_i maxterm igaz A,B,C változók értékének kombinációjára, kivéve egyet, ahol $A=1, B=0, C=1$. Tehát $i=[101]=5$. $\rightarrow M_5$ (táblázat 5.sora)
- Legyen: $M_i = A + B + C$ ahol az F kimeneti függvényérték *hamis* volt. Ez az M_i maxterm igaz A,B,C változók értékének kombinációjára, kivéve egyet, ahol $A=0, B=0, C=0$. Tehát $i=[000]=0$. \rightarrow Így M_0 (táblázat 0.sora)

Példa: KNF/DNF felírása

■ Igazságtábla

sor	J	K	L	F
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	0
7	1	1	1	0

■ Igazságtáblából kapjuk, hogy:

$$0 = F(KNF) = (J + K + L) \cdot (\bar{J} + K + L) \cdot (\bar{J} + K + \bar{L}) \cdot (\bar{J} + \bar{K} + L) \cdot (\bar{J} + \bar{K} + \bar{L})$$

$$F(KNF) = [000] \cdot [100] \cdot [101] \cdot [110] \cdot [111] = M_0 \cdot M_4 \cdot M_5 \cdot M_6 \cdot M_7$$

$$1 = F(DNF) = (\bar{J} \cdot \bar{K} \cdot L) + (\bar{J} \cdot K \cdot \bar{L}) + (\bar{J} \cdot K \cdot L)$$

$$F(DNF) = [001] + [010] + [011] = m_1 + m_2 + m_3$$

Igazságtábla felírása logikai kifejezésekből I.

- a.) **DNF-ből**: felírás egyszerű
 - **Kanonikus** mintermből: egy sor képződik (ahol F igaz),
 - **Nem kanonikus**, kevesebb változót tartalmazó termből: *több sor is képződhet*, mivel egy ilyen term egy adott logikai változó ponált és negált értékére is igaz kimeneti eredményt ($F=1$) ad,
 - Egy sorhoz **több term** is tartozhat!

Példa: DNF \rightarrow Igazságtábla

■ Eredeti egyenlet:

$$'1' = F(DNF) = J \cdot \bar{K} + \bar{J} \cdot K \cdot L + J \cdot K \cdot \bar{L} + K \cdot L$$

term1

term2

term3

term4

kanonikus (minterm)

■ Kapott igazságtábla:

sor	J	K	L	F
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

term2 és term4

term1

term1

term3

term4

Igazságtábla felírása logikai kifejezésekből II.

- b.) **KNF-ből**: felírás „nehezebb” (az egyes logikai változók negált értékeit kell venni)
 - **Kanonikus maxtermből**: egy sor képződik (ahol F hamis),
 - **Nem kanonikus**, kevesebb változót tartalmazó termből: több sor is képződhet, mivel egy ilyen term egy adott logikai változó ponált és negált értékére is '*hamis*' kimeneti eredményt ($F=0$) ad,
 - Egy sorhoz **több term** is tartozhat!

Példa: KNF \rightarrow Igazságtábla

- Eredeti egyenlet:

$$'0' = F(KNF) = (\bar{A} + B + C) \cdot (\bar{A} + B) \cdot (\bar{A} + \bar{B} + \bar{C})$$

term1 term2 term3
 ↑ ↑
 kanonikus (maxterm)

- Kapott igazságtábla:

sor	A	B	C	F
0	0	0	0	1
1	0	0	1	1
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	0

term1 és term2

term2

term3

Igazság táblák tömörebb felírási formája

- Eml: Kanonikus ig. táblánál: n változó $\rightarrow 2^n$ sor (összes lehetséges változó kombináció felírásával)

- Egyszerűsített / tömörebb felírás:

□ „X”: Don't Care változó két értéke: 0 és 1 is lehet.

- Pl: $'1' = F(\text{DNF}) = \underbrace{J \cdot \bar{K}}_{\text{term1}} + \underbrace{\bar{J} \cdot K \cdot L}_{\text{term2}} + \underbrace{J \cdot K \cdot \bar{L}}_{\text{term3}} + \underbrace{K \cdot L}_{\text{term4}}$

kanonikus (minterm)

J	K	L	F
0	0	0	0
0	0	1	0
0	1	0	0
X	1	1	1
1	0	X	1
1	1	0	1

term2 és term4

term1

term3

Term1: L don't care (0 v. 1)

Term4: J don't care (1 v. 0)

NTSH: Nem Teljesen Specifikált Hálózat (Don't Care kimenet)

- Bizonyos bemeneti kombinációkra ugyanazt a kimeneti eredményt kapjuk (irreleváns)
- Jele: „-” Don't care kimeneti értéke F-nek

■ Pl.

A	B	F
0	X	1
1	0	-
1	1	0

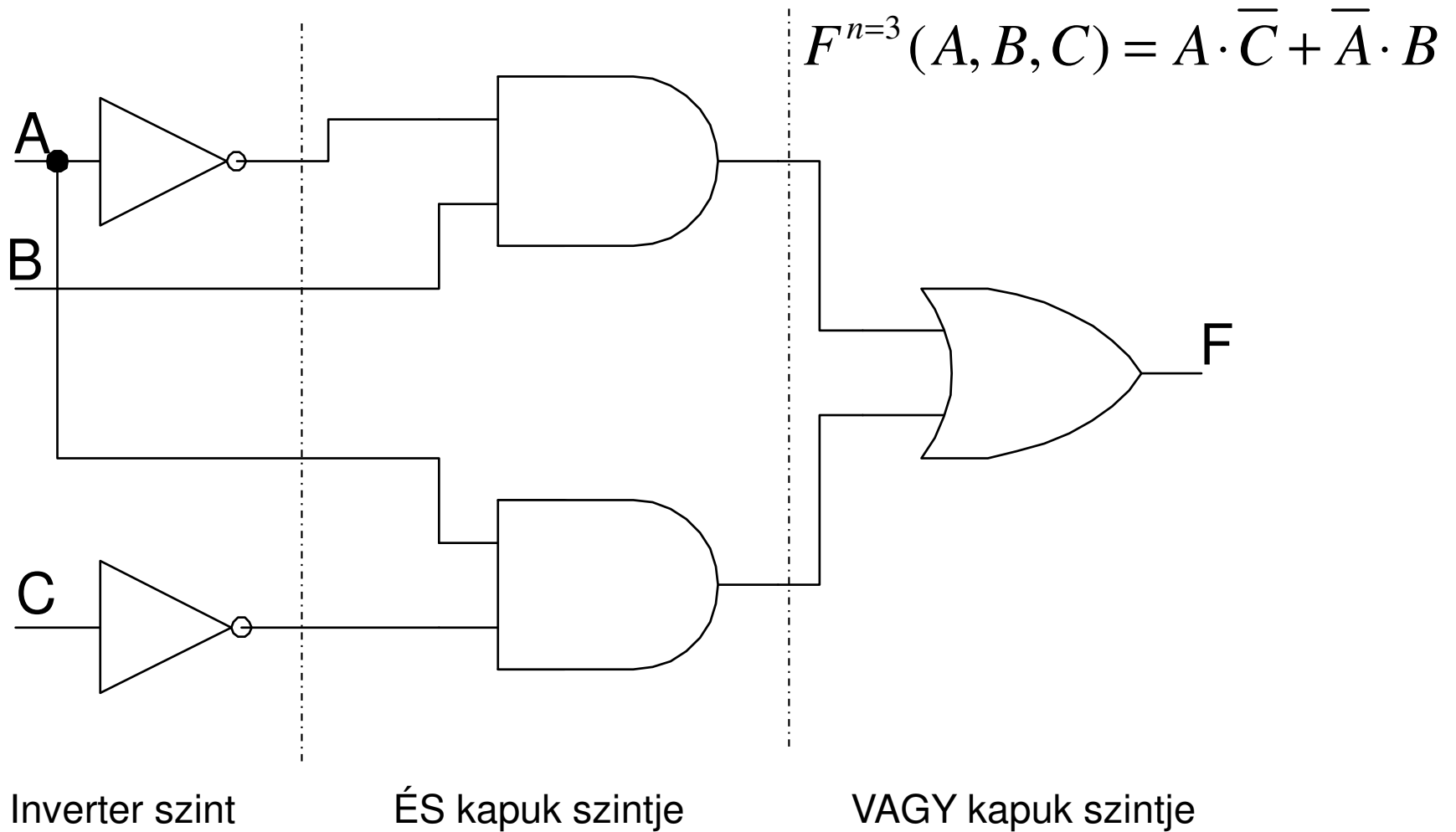
ha '–'=1, $F = \bar{A} + A \cdot \bar{B} = \bar{A} + \bar{B}$

ha '–'=0, $F = \bar{A}$




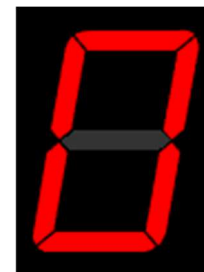
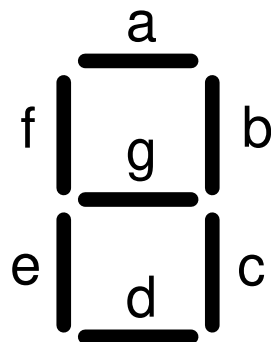
Elvi logikai rajz

Az egyszerűsített függvény logikai áramköri realizációja



Példa 1: 7-szegmenses dekóder áramkör tervezése

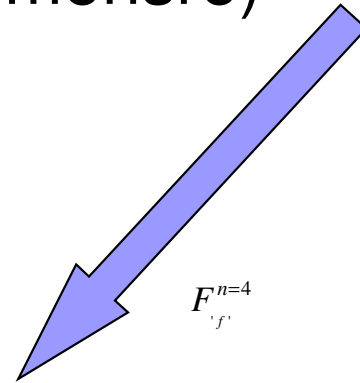
- **Számjegyek** (0-9) és spec. **hexadecimális karakterek** megjelenítésére ()
- nemzetközi elnevezései a szegmenseknek: (a, b, c, d, e, f, g)
 - 16 érték (4 biten ábrázolható): F(X,Y,Z,W)



Példa: 7-szegmenses dekóder tervezése (folyt)

- Igazságtábla (**f** szegmensre)

$$F_{'f'}^{n=4} = ?$$



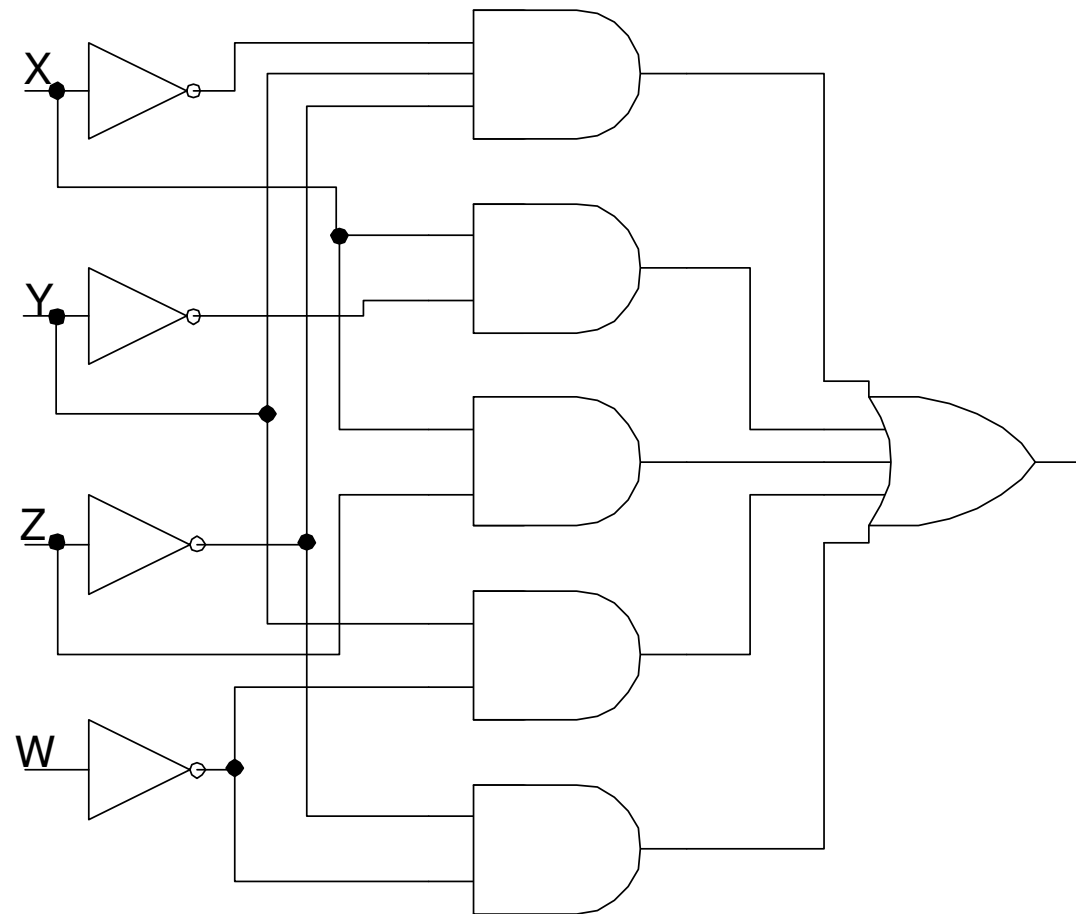
- Kapott optimalizált F kimeneti függvény **f** szegmensre:

$$F_{'f'}^{n=4}(X, Y, Z, W) = \bar{Z} \cdot \bar{W} + X \cdot \bar{Y} + Y \cdot \bar{W} + X \cdot Z + \bar{X} \cdot Y \cdot \bar{Z}$$

sor	X	Y	Z	W	F _f
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	1
9	1	0	0	1	1
10	1	0	1	0	1
11	1	0	1	1	1
12	1	1	0	0	1
13	1	1	0	1	0
14	1	1	1	0	1
15	1	1	1	1	1

Példa 1: 7-szegmenses dekóder logikai áramköri realizációja

(**f** szegmensre)



$$F_{f'}^{n=4}(X, Y, Z, W) = \bar{Z} \cdot \bar{W} + X \cdot \bar{Y} + Y \cdot \bar{W} + X \cdot Z + \bar{X} \cdot Y \cdot \bar{Z}$$