PANNON EGYETEM, Veszprém
Villamosmérnöki és Információs Rendszerek Tanszék

Digitális Rendszerek és Számítógép Architektúrák

1. előadás: Boole-algebra, logikai függvények

Előadó: Dr. Vörösházi Zsolt
voroshazi.zsolt@mik.uni-pannon.hu

Kapcsolódó jegyzet, segédanyag:

- Angol nyelvű könyv: http://www.virt.uni-pannon.hu → Oktatás → Tantárgyak → Digitális Rendszerek és Számítógép Architektúrák (nappali)

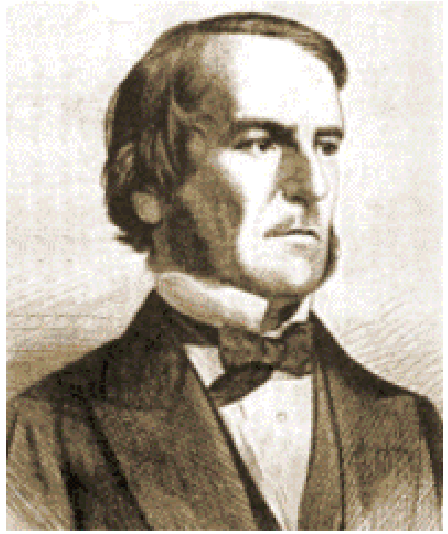
  Bevezetés: Számítógép Generációk (chapter01.pdf)

- Fóliák, óravázlatok .ppt (.pdf)
- Feltöltésük folyamatosan

További ajánlott irodalom

- 🕮 Dr. Holczinger, Dr. Göllei. Dr. Vörösházi: Digitális Technika I. (TAMOP 4.1.2A - 2012) : Digitalis_technika_I_TAMOP
- 🕮 Dr. Holczinger, Dr. Göllei. Dr. Vörösházi: Digitális Technika II. (TAMOP 4.1.2A - 2013) : Digitalis_technika_II_TAMOP

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 + egyszerűsítés)

Boole algebra elemei:

- 3 alapművelet: 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!

NOT

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

OR

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

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

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

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

AND

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

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

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

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

Elnyelési tul.

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

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

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

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

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

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

De-Morgan azonosságok:

DUAL ITÁS

$$16.)\ \overline{A + B} = \overline{A} \cdot \overline{B}$$

$$17.)\ \overline{A \cdot B} = \overline{A} + \overline{B}$$

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

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

Logikai hálózatok csoportosítása

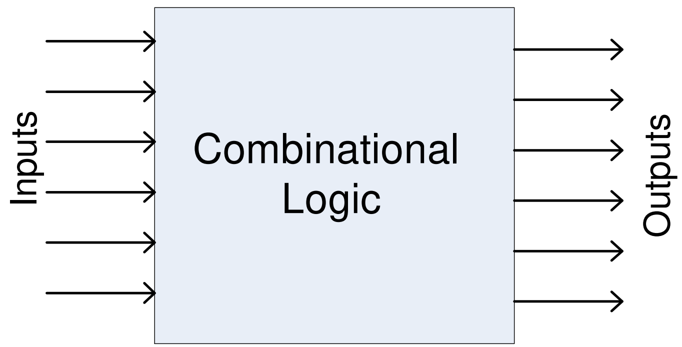
Ismétlés: Ezek alapján kétféle hálózatot különböztetünk meg:

- **(K.H.) Kombinációs logikai hálózat**ról beszélünk: ha a mindenkori kimeneti kombinációk létrehozásához *elég a bemeneti kombinációk* pillanatnyi értéke.

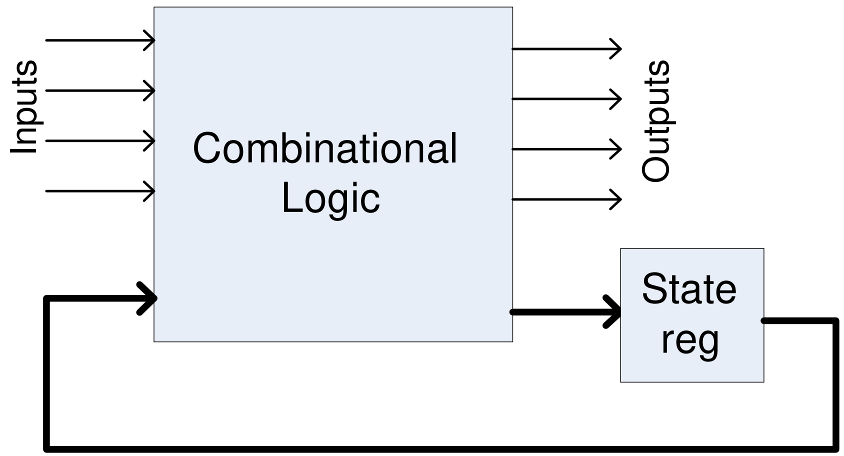
- **(S.H.) Sorrendi (szekvenciális) logikai hálózat**ról beszélünk**:** ha a mindenkori kimeneti kombinációt, nemcsak *a pillanatnyi* bemeneti kombináció, hanem *a korábban fennállt bementi kombinációk* és azok *sorrendje* is befolyásolja. (Ezen *szekunder kombinációk* segítségével az ilyen hálózatok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

Kombinációs vs. sorrendi hálózatok:

- Kombinációs hálózat:



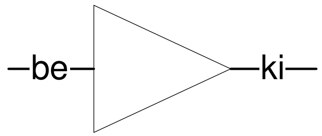
- Sorrendi hálózat:

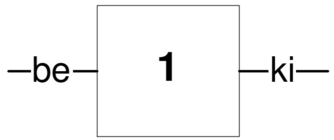


Egyváltozós logikai függvények:

- Jelmásoló („buffer” - jel-erősítő):

be	ki
0	0
1	1





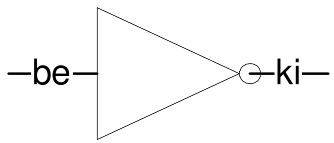
Nemzetközi szabvány

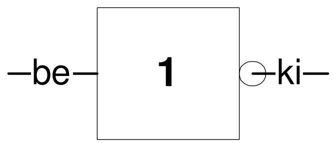
Magyar szabvány

- Negálás - Inverter (NOT):

$$\overline{A}$$

be	ki
0	1
1	0



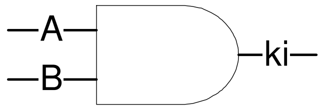


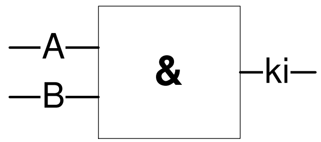
Kétváltozós logikai függvények:

- ÉS (AND):

$$A \cdot B$$

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

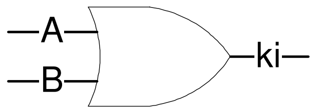


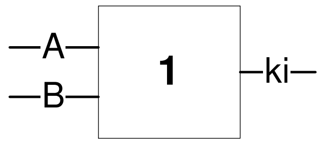


- VAGY (OR):

$$A + B$$

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

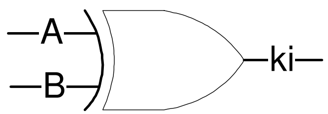


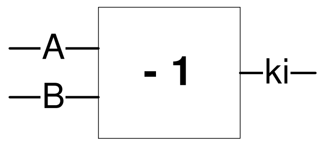


- Antivalencia (XOR):

$$A \oplus B$$

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





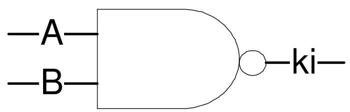
Kétváltozós log.függv. (folyt.):

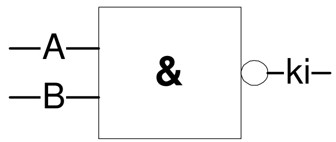
Univerzálisan teljes rendszert a NAND illetve NOR függvény alkot!

- NEM-ÉS (NAND):

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

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

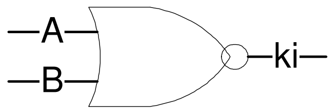


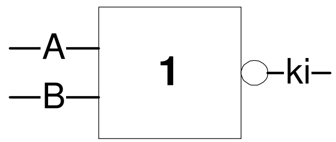


- NEM-VAGY (NOR):

$$\overline{A + B}$$

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

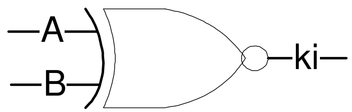


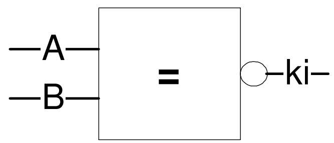


- Ekvivalencia (NXOR):

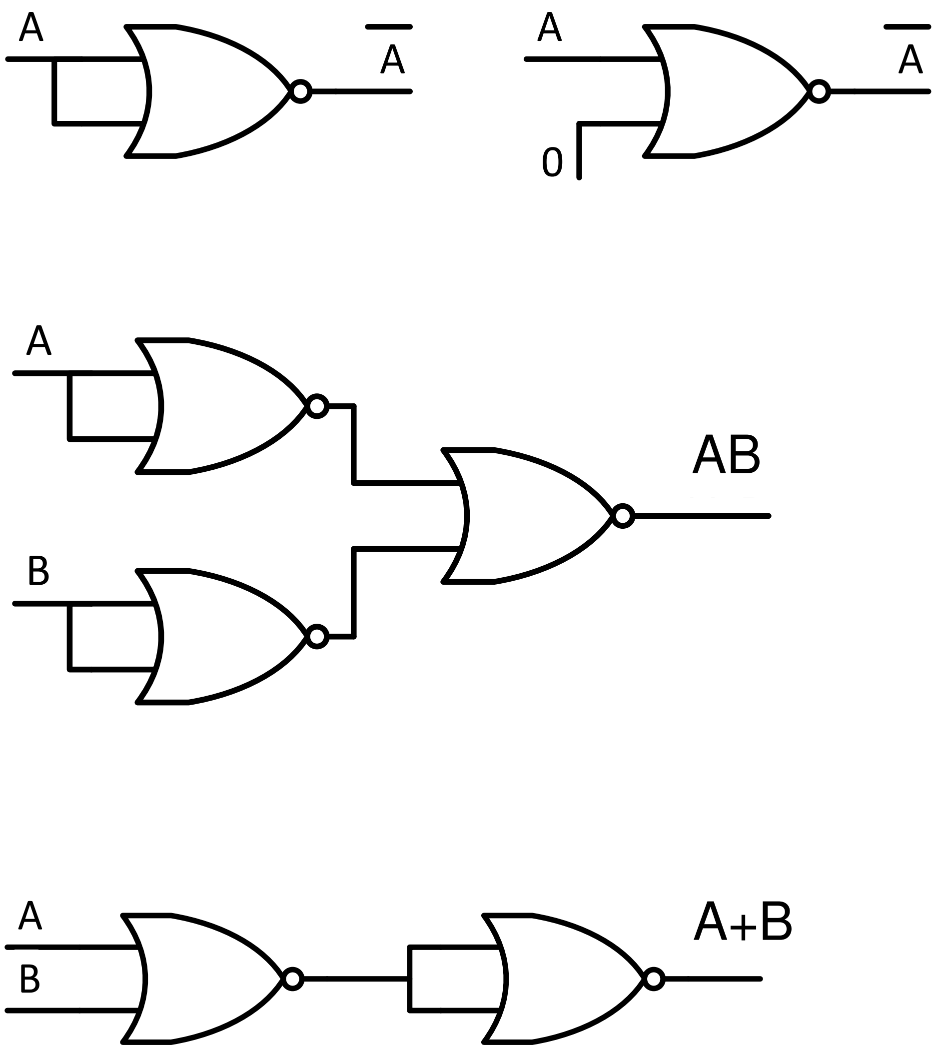
$$A \odot B$$

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





Funkcionális teljesség: példák

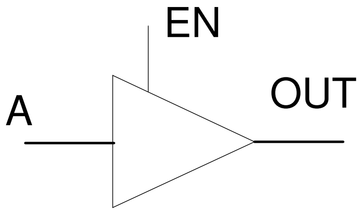


Funkcionálisan teljes vagy univerzális áramköri alapkapuk: Logikai hálózatok esetén a CMOS NAND, illetve NOR kapu.

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

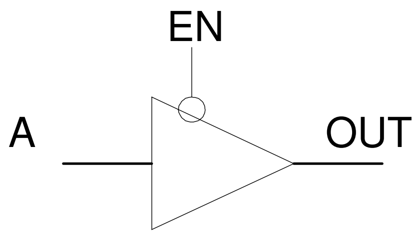
Tri-State Buffer:

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



High-true enable

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



Low-true enable

Smart áramkörök ☺