Hobbi Elektronika

x \ y	∧ 0	∧ 1
0	0	0
1	0	1

x \ y	∨ 0	∨ 1
0	0	1
1	1	1

x \ y	→ 0	→ 1
0	1	1
1	0	1

x \ y	⊕ 0	⊕ 1
0	0	1
1	1	0

Figure 1. Truth tables

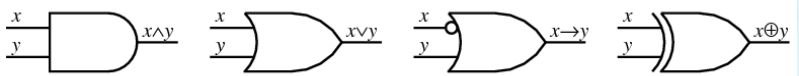


Figure 2. Logic gates

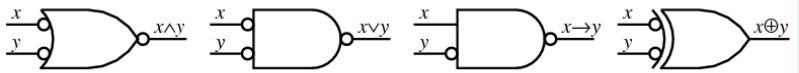
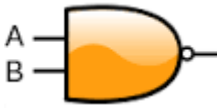


Figure 3. De Morgan equivalents

Figure 4. Venn diagrams

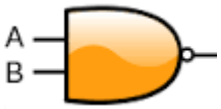
A digitális elektronika alapjai:

Kombinációs logikai hálózatok – 1. rész



Felhasznált anyagok

- M. Morris Mano and Michael D. Ciletti: Digital Design - With an Introduction to the Verilog HDL, 5th. Edition
- **Electronics-course.com** (Digital Electronics Course I. – II.)
  - Boolean Algebra Calculator
  - Karnaugh Map Calculator
- Végh János: Ismerkedés a digitális elektronikával
- Mészáros Miklós: Logikai algebra alapjai, logikai függvények I.
- BME FKE: Logikai áramkörök
- Logisim szimulátor: www.cburch.com/logisim/
- Falstad.com: Circuit simulator
- F-alpha.net: Boolean Algebra
- Neuproductions-be: The Logic Lab



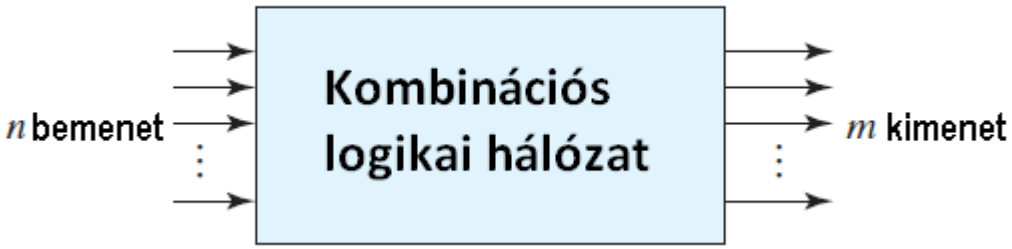
Kombinációs logikai hálózat

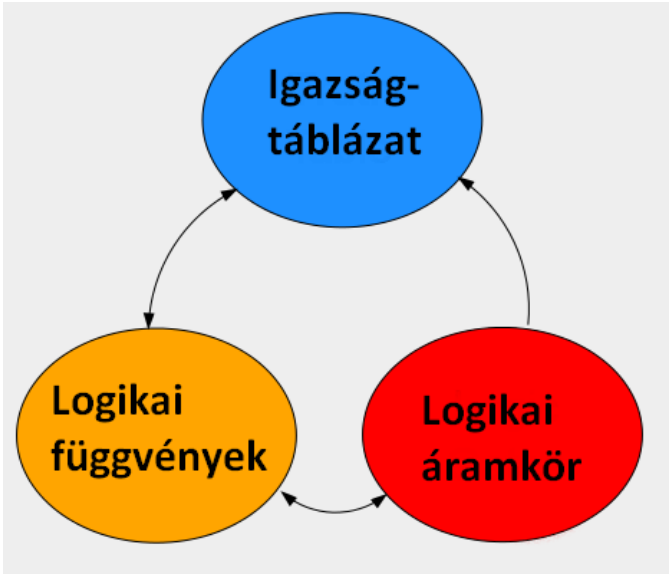
A logikai áramköröket két nagy csoportba sorolhatjuk: **kombinációs** és **sorrendi** hálózatok.

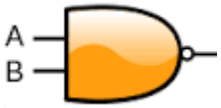
- **A kombinációs logikai hálózat** viselkedése logikai függvényekkel leírható. Kimeneteinek állapota csupán a bemenetek pillanatnyi állapotának függvénye.
- **A sorrendi hálózatok** tárolóelemeket is tartalmaznak. Ezek kimeneteinek állapota emiatt nem csupán a bemenetek, hanem a tárolók állapotától is függ, vagyis a bemenetek korábbi állapotainak sorrendjétől.

A kombinációs logikai hálózat összekapcsolt logikai kapukból áll. Az n bemenetnek összesen 2^n állapota lehetséges, amelyek mindegyikéhez a kimenetek egy-egy állapota rendelhető. Így a kombinációs logikai hálózat egyértelműen megadható az **igazságtáblázatával,** amely leírja a „viselkedését”.

A kombinációs logikai hálózat kimenetei egy-egy n változós **logikai függvénnyel** is megadhatók.







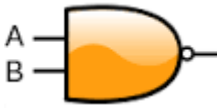
Logikai elemzés

A kombinációs logikai áramkörök elemzésének az a célja, hogy meghatározzuk azokat a logikai összefüggéseket (függvényeket), amelyeket az adott áramkör megvalósít. Az elemzésnél egy áramkörből indulunk ki, s a végeredmény néhány függvény, egy igazságtáblázat, vagy az áramkör működésének leírása.

Az elemzés első lépése annak megállapítása, hogy az áramkör valóban kombinációs hálózat, nem pedig sorrendi hálózat. **A kombinációs logikai hálózat nem tartalmaz visszacsatolást, sem tárolóelemeket.**

A logikai áramkör rajzából a logikai függvényeket az alábbi lépésekben kaphatjuk meg:

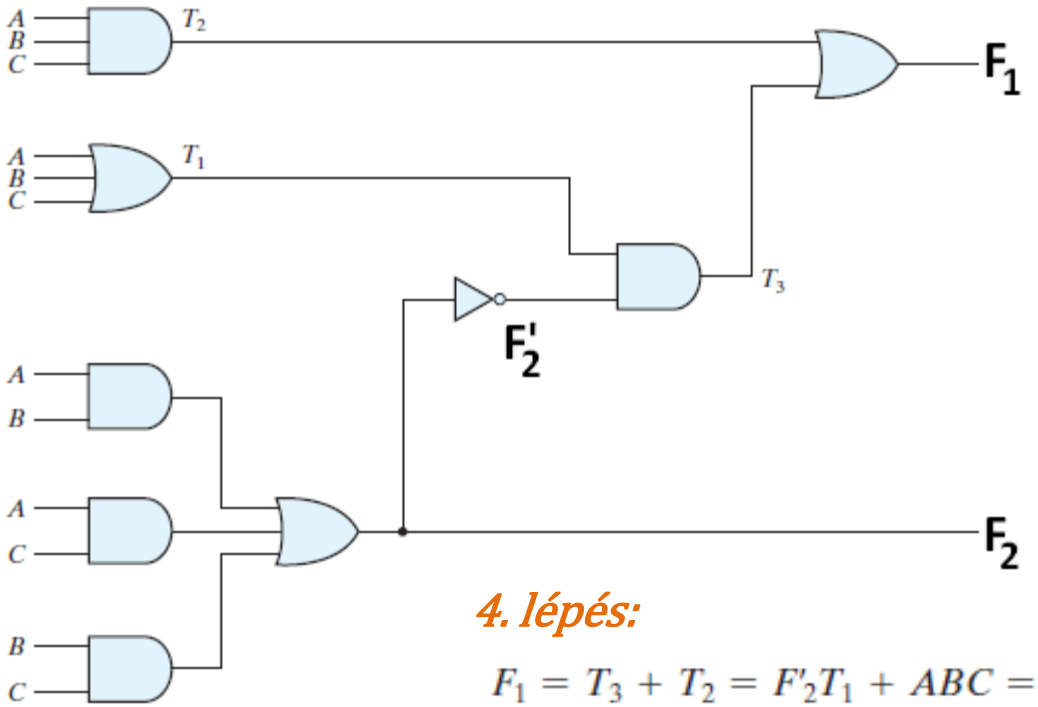
1. Címkézzünk fel minden kapu kimenetet, ami a bemeneti változók függvénye! Határozzuk meg a kapuk által megvalósított logikai függvényeket!
2. Címkézzünk fel minden kapu kimenetet, ami a bemeneti változók és az előző pontban felcímkézett kapu kimenetek függvénye! Határozzuk meg az ezen kapuk által megvalósított logikai függvényeket is!
3. Ismételjük meg a 2. lépést mindaddig, amíg a kimenetek leírása elő nem áll!
4. Az előzőekben meghatározott függvényekbe ismételt behelyettesítésekkel állítsuk elő a kimeneteket a bemeneti változók logikai függvényeként!



Példa az elemzésre

Az előző oldalon leírt lépéseket az alábbi példával szemléltetjük:

- Az áramkör három bemenettel (A, B, C) és két kimenettel rendelkezik (F1 és F2).



1. lépés:

$$F_2 = AB + AC + BC$$
$$T_1 = A + B + C$$
$$T_2 = ABC$$

2. lépés:

$$T_3 = T1 \cdot F_2{}'$$

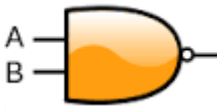
3. lépés:

$$F_1 = T_2 + T_3$$

4. lépés:

$$\begin{aligned} F_1 = T_3 + T_2 &= F'_2T_1 + ABC = (AB + AC + BC)'(A + B + C) + ABC \\ &= (A' + B')(A' + C')(B' + C')(A + B + C) + ABC \\ &= (A' + B'C')(AB' + AC' + BC' + B'C) + ABC \\ &= A'BC' + A'B'C + AB'C' + ABC \end{aligned}$$

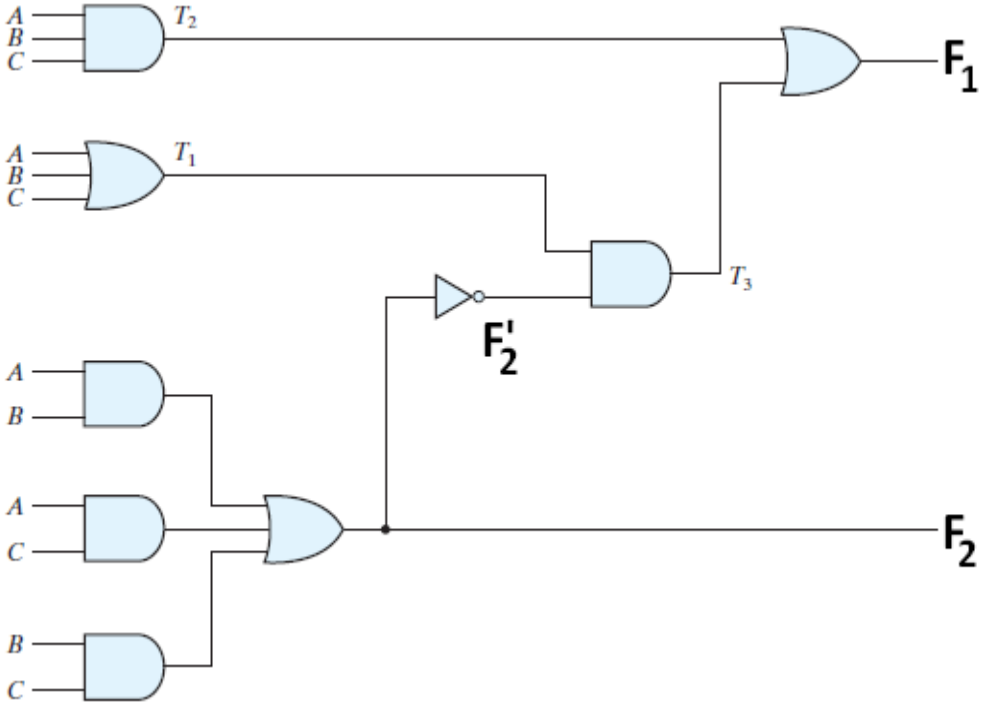
A logikai függvények meghatározása önmagában nem elegendő az áramkör működésének megértéséhez. Később majd látni fogjuk, hogy ez az áramkör a teljes összeadót valósítja meg.



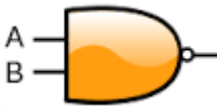
Az igazságtáblázat felírása

Az igazságtáblázatot az előző oldalon meghatározott logikai függvények segítségével is felírhatjuk, de magából az áramkörből is kiindulhatunk:

1. Állítsuk elő az *n* bemenet összes lehetséges (2^n darab) kombinációját!
2. Minden bemeneti kombinációhoz határozzuk meg azon kapuk kimeneti állapotát, amelyek csak a bemeneti változóktól függenek!
3. Minden bemeneti kombinációhoz határozzuk meg azon kapuk kimeneti állapotát is, amelyek a bemeneti változók mellett az előzőleg meghatározott kapu kimenetektől is függenek!



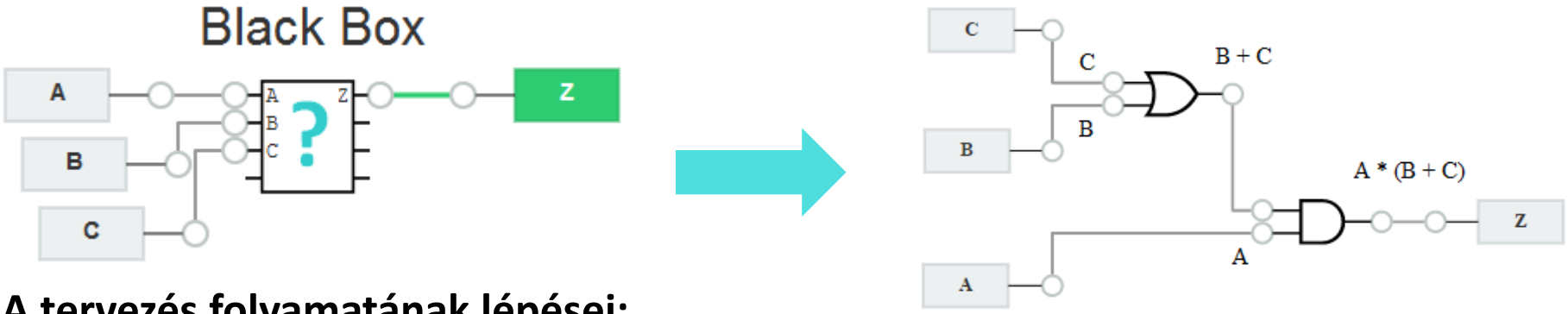
A	B	C	F_2	F'_2	T_1	T_2	T_3	F_1
0	0	0	0	1	0	0	0	0
0	0	1	0	1	1	0	1	1
0	1	0	0	1	1	0	1	1
0	1	1	1	0	1	0	0	0
1	0	0	0	1	1	0	1	1
1	0	1	1	0	1	0	0	0
1	1	0	1	0	1	0	0	0
1	1	1	1	0	1	1	0	1



A tervezés folyamata

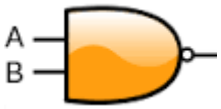
A tervezés folyamata az előzőeknek az ellentettje:

- A kiindulási alap általában a feladat specifikációja.
- A cél egy áramkör megtervezése, vagy az ehhez szükséges logikai függvények előállítása.



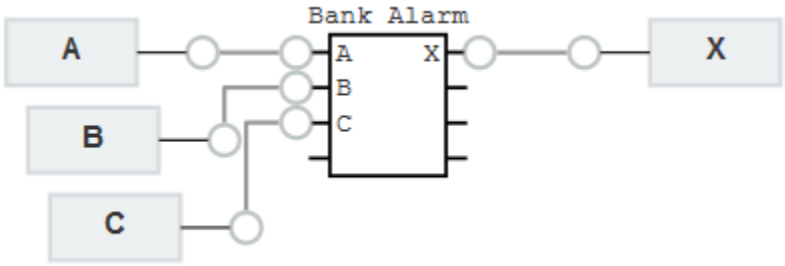
A tervezés folyamatának lépései:

1. A feladat specifikációjából határozzuk meg a szükséges bemenetek és kimenet számát, s nevezzük el azokat!
2. A bemeneti és a kimeneti állapotok viszonyából állítsuk elő az igazságtáblázatot!
3. Az igazságtáblázat alapján állítsuk elő a kimeneteket leíró logikai függvényeket és egyszerűsítsük azokat!
4. Rajzoljuk meg a logikai áramkört és ellenőrizzük a helyességét (kézzel vagy szimulációs programmal)!



Tervezési példa: banki riasztó

Tegyük fel, hogy egy bank riasztórendszert akar telepíteni, amely három mozgásérzékelőn (A, B, C) alapul. A hamis riasztások kiszűrésére (pl. egy pók éppen az egyik érzékelő előtt ereszkedik le…) a riasztás csak akkor indul, ha a három érzékelő közül legalább kettő egyidejűleg jelez. (Forrás: learn.electronics-course.com/#0)



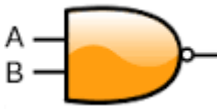
C	B	A	X
0	0	0	?
0	0	1	?
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	?
1	1	1	?

0
0
1
1

Logikai függvény:

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

Hmm, kicsit komplikált! Egyszerűbben, ha kérhetném!



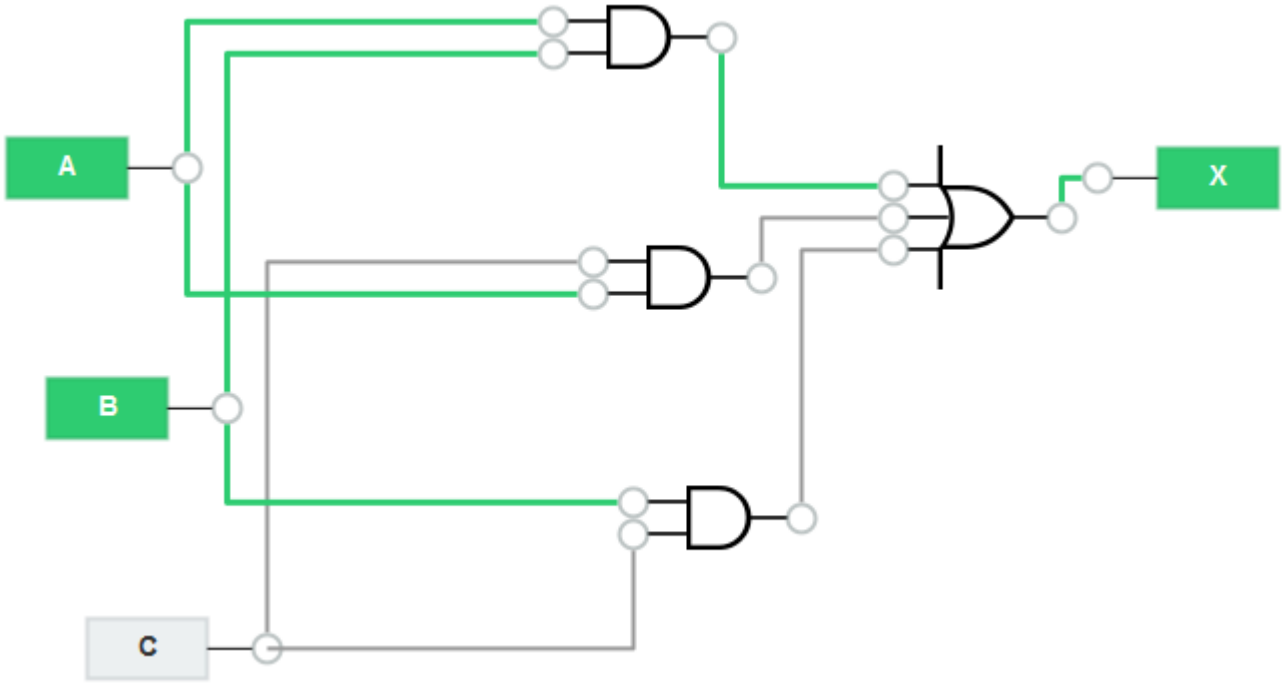
Egyszerűsítés Karnaugh-tábla alapján

C	B	A	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

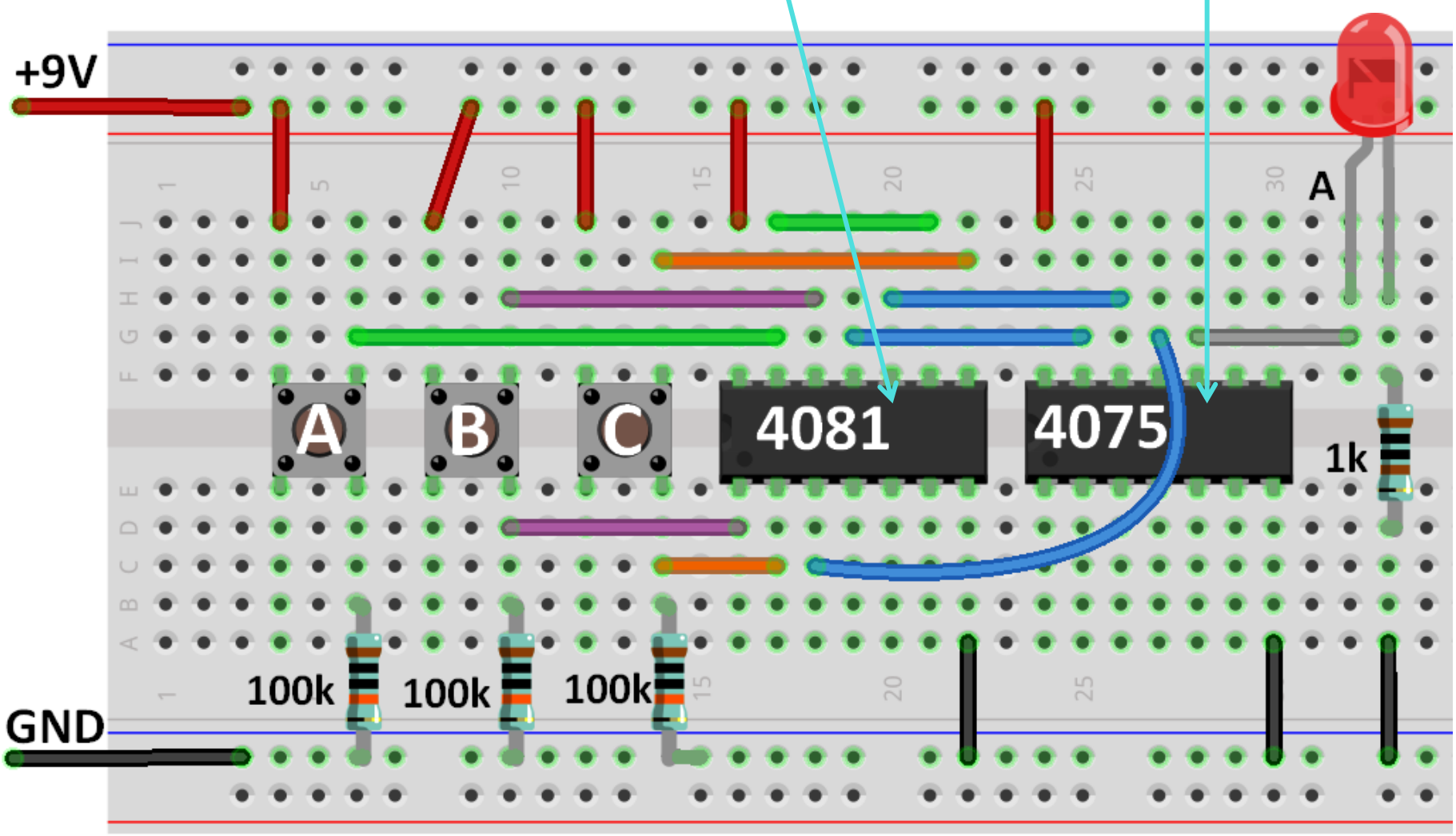
A **Karnaugh-tábla** az igazságtáblázat egy másik ábrázolási formája. Az adatok úgy vannak csoportosítva, hogy az 1-bites változások szomszédos cellákba essenek, így az F = ABC' + ABC = AB típusú egyszerűsítések könnyen felismerhetők benne.

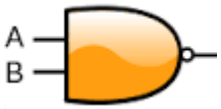
		C B			
		00	01	11	10
A	0	0	0	1	0
	1	0	1	1	1

$$X = \overline{C}\bullet B\bullet A + C\bullet\overline{B}\bullet A + C\bullet B\bullet\overline{A} + C\bullet B\bullet A \Rightarrow X = B\bullet A + C\bullet A + C\bullet B$$



A „banki riasztó” megépítése





Dobókocka kijelző vezérlése

A kijelző LED-eket négy csoportba sorolhatjuk. Az ábra színezése szerint:
R = piros, **Y** = sárga, **B** = kék, **G** = zöld.

A számunkra közömbös állapotok értékét úgy célszerű megválasztani, hogy ezáltal egyszerűbb függvényt kapjunk!

Vesd össze: learn.electronics-course.com/#12

Szám	B2 B1 B0	R	Y	B	G
1	001	1	0	0	0
2	010	0	1	0	0
3	011	1	1	0	0
4	100	0	1	1	0
5	101	1	1	1	0
6	110	0	1	1	1

R = B0

B2	B1	B0	R	
0	0	0	X	0
0	0	1	1	
0	1	0	0	
0	1	1	1	
1	0	0	0	
1	0	1	1	
1	1	0	0	
1	1	1	X	1

Y = B2 + B1

B2	B1	B0	Y	
0	0	0	X	0
0	0	1	0	
0	1	0	1	
0	1	1	1	
1	0	0	1	
1	0	1	1	
1	1	0	1	
1	1	1	X	1

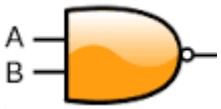
B = B2

B2	B1	B0	B	
0	0	0	X	0
0	0	1	0	
0	1	0	0	
0	1	1	0	
1	0	0	1	
1	0	1	1	
1	1	0	1	
1	1	1	X	1

G = B2•B1

B2	B1	B0	G	
0	0	0	X	0
0	0	1	0	
0	1	0	0	
0	1	1	0	
1	0	0	0	
1	0	1	0	
1	1	0	1	
1	1	1	X	1

A kapcsolás megépítésére majd akkor térünk vissza, ha a sorrendi hálózatoknál eljutunk a számlálókhoz!



Bináris összeadás

0	1	0	1
+ 0	+ 0	+ 1	+ 1
0 0	0 1	0 1	10

x

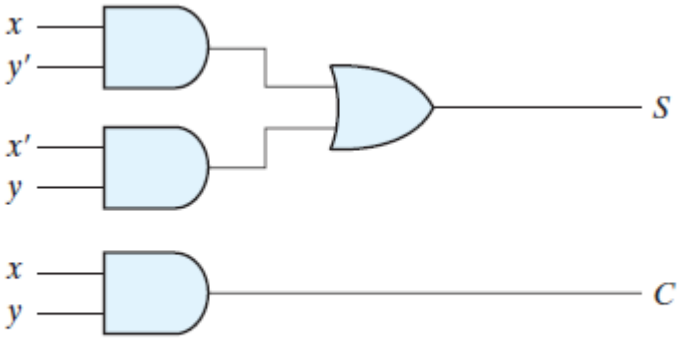
y

C átvitel

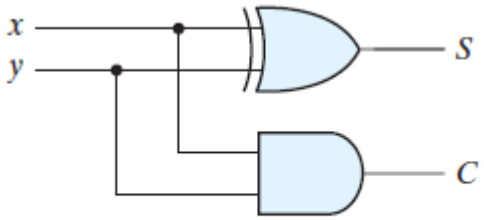
S összeg

A bináris összeadás szabályai

x	y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



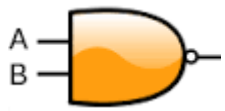
(a) $S = xy' + x'y$
$C = xy$



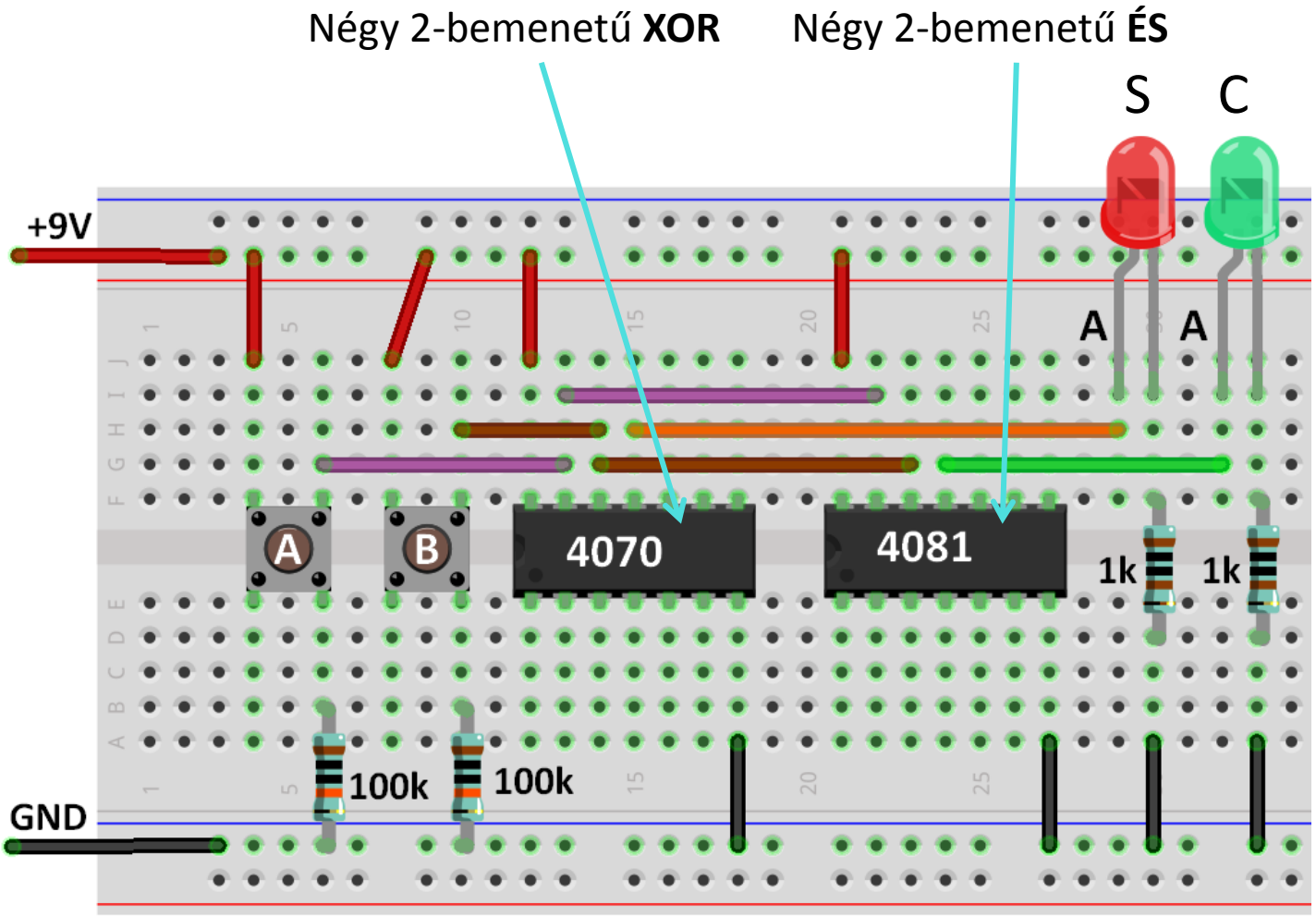
(b) $S = x \oplus y$
$C = xy$

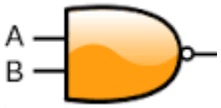
Az XOR kapu alkalmazása egyszerűbb megépítést tesz lehetővé.

Ezt a kapcsolást fél-összeadónak hívják.
A teljes összeadáshoz az áthozatot is kezelni kell…



A fél-összeadó megépítése





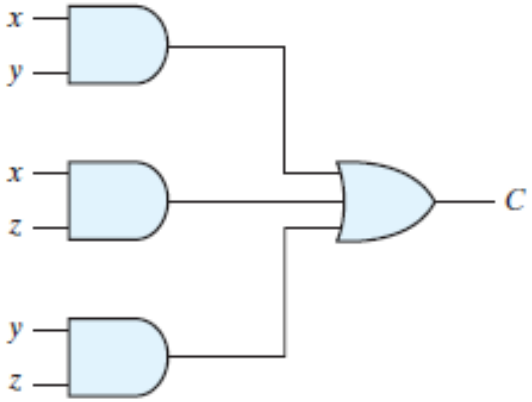
A teljes összeadó

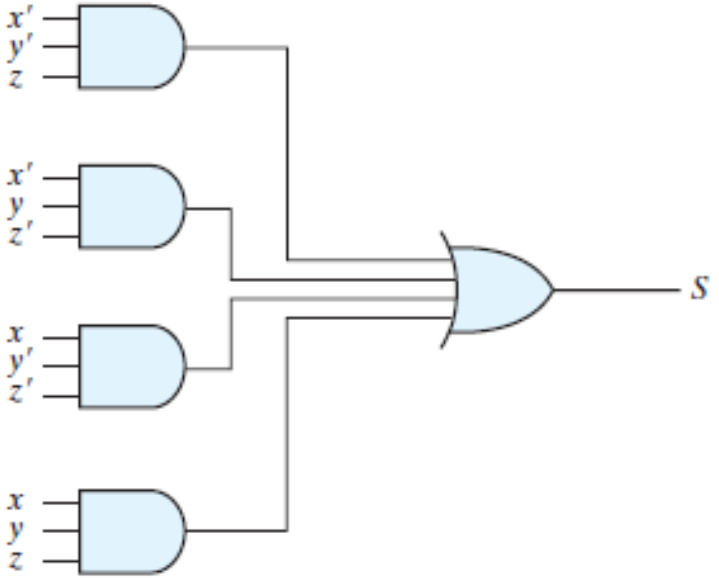
A teljes összeadó a két összeadandón (*x, y*) kívül az áthozatot is fogadja. Az igazságtáblázatot az összeadás szabályai szerint könnyen felírhatjuk. Az ebből kapott függvények:

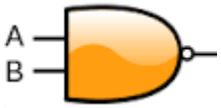
x	*y*	*z*	*C*	*S*
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$$S = x'y'z + x'yz' + xy'z' + xyz = x \oplus y \oplus z$$

$$C = xy + xz + yz \quad \text{(egyszerűsített alak)}$$

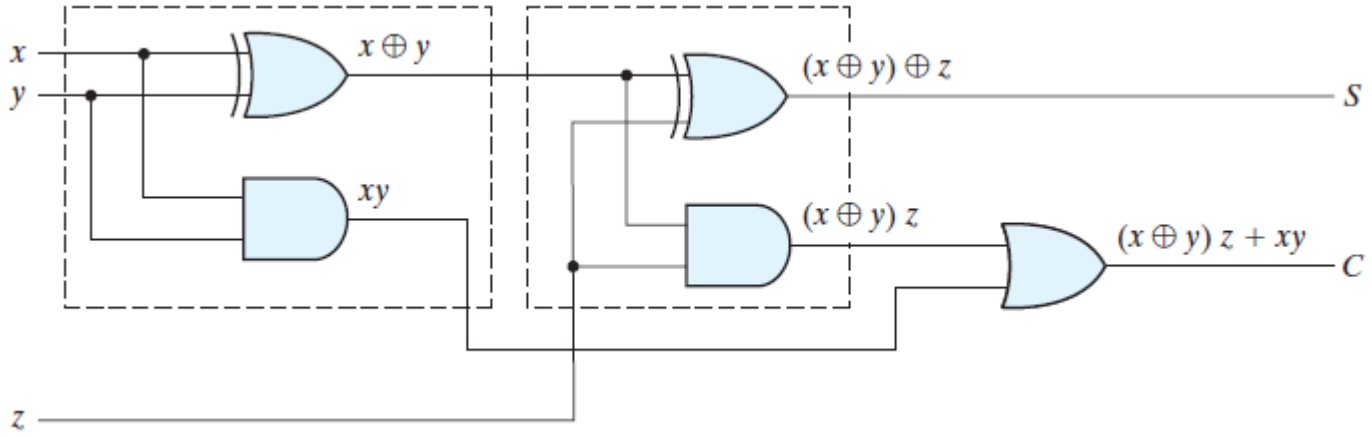






A teljes összeadó – másképp

Két fél-összeadó összekapcsolásával és egy VAGY kapuval is megépíthetjük a teljes összeadót.



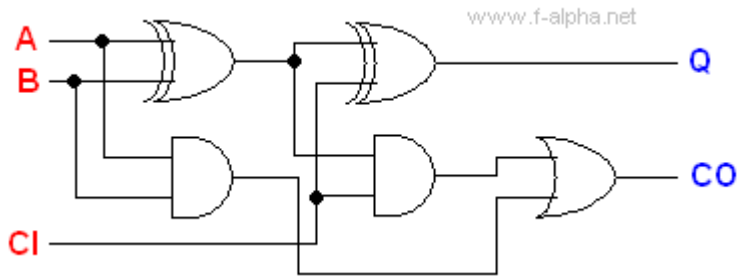
Ellenőrizzük a kimeneteket előállító logikai függvényeket!

$$
\begin{aligned}
S &= z \oplus (x \oplus y) \\
&= z'(xy' + x'y) + z(xy' + x'y)' \\
&= z'(xy' + x'y) + z(xy + x'y') \\
&= xy'z' + x'yz' + xyz + x'y'z
\end{aligned}
$$

$$C = z(xy' + x'y) + xy = xy'z + x'yz + xy$$ = xy + xz + yz

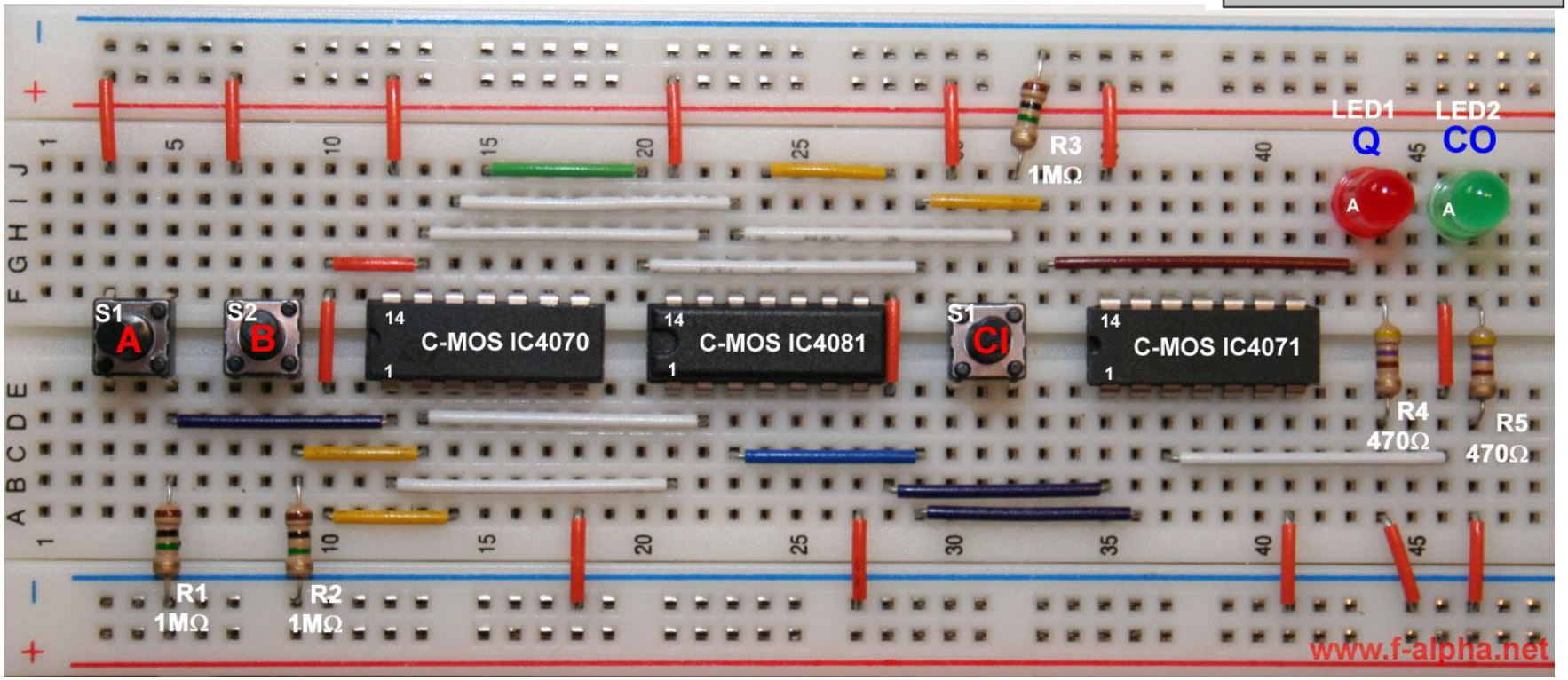
(lásd pl.: Boolean Algebra Calculator)

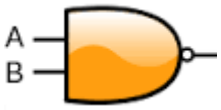
A teljes összeadó megépítése



CI	A	B	Q	CO
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

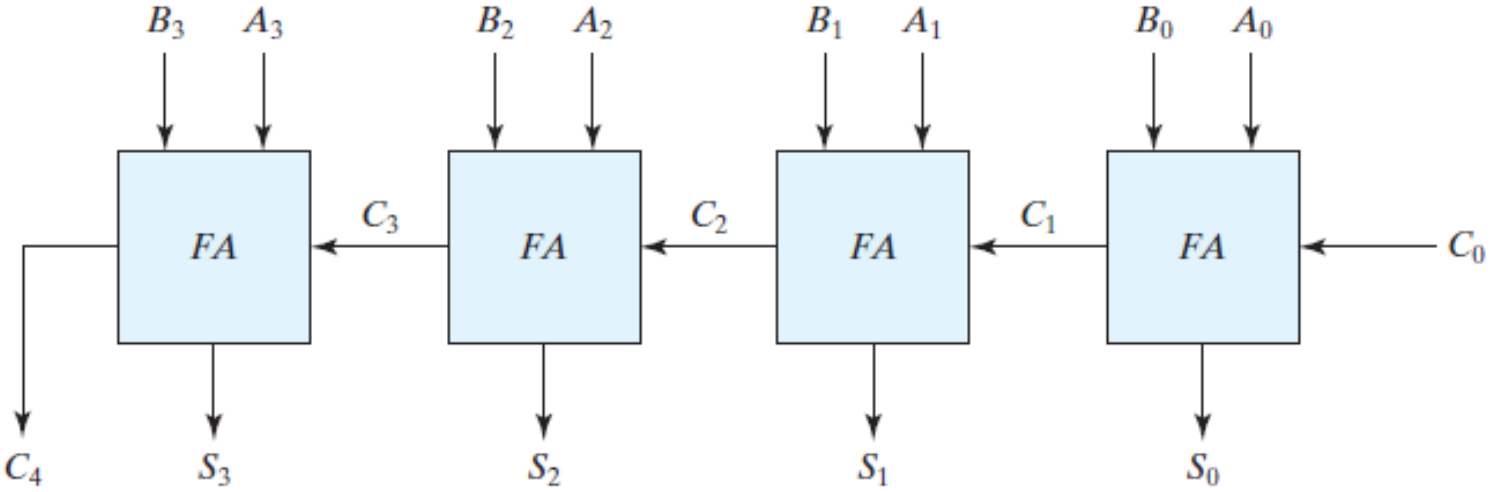
Forrás: en.f-alpha.net/electronics/digital-electronics/boolean-logic/go-on/experiment-12-the-full-adder/



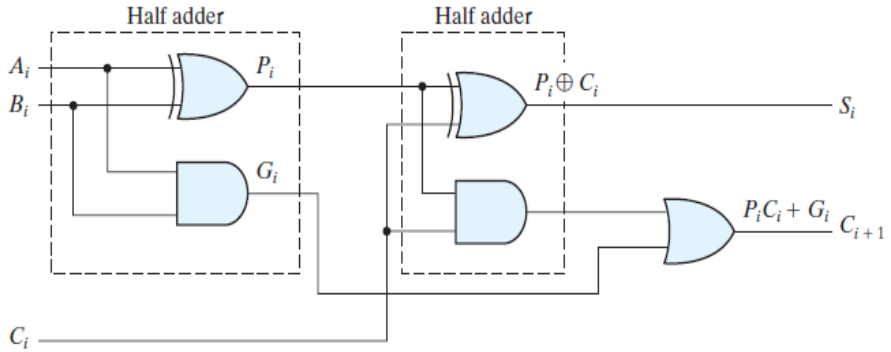


Többjegyű összeadó

Az előzőekben bemutatott teljes összeadót (full adder) láncbafűzhetjük egy többjegyű bináris összeadó megvalósítására. Az ábrán egy négybites összeadó látható.



Ennek a kapcsolásnak a hátrány az átvitel lassú terjedése, ami a működési sebességet korlátozza. A probléma megoldására speciális kapcsolást dolgoztak ki, az átvitel előre történő kiszámításához (look ahead carry).



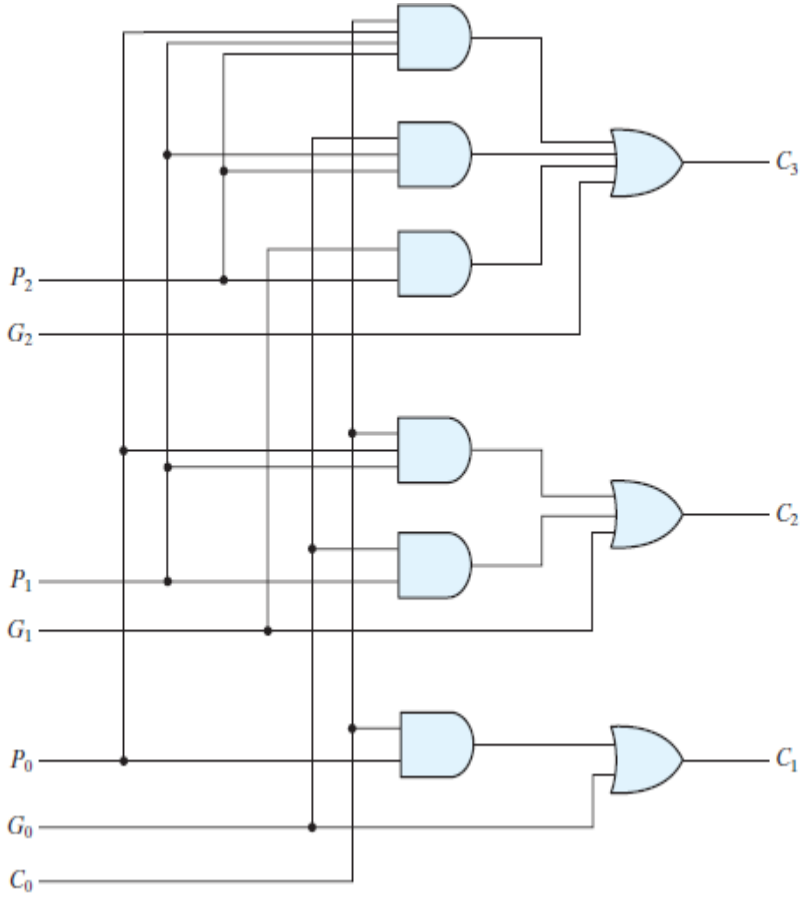
$C_0 =$ áthozat

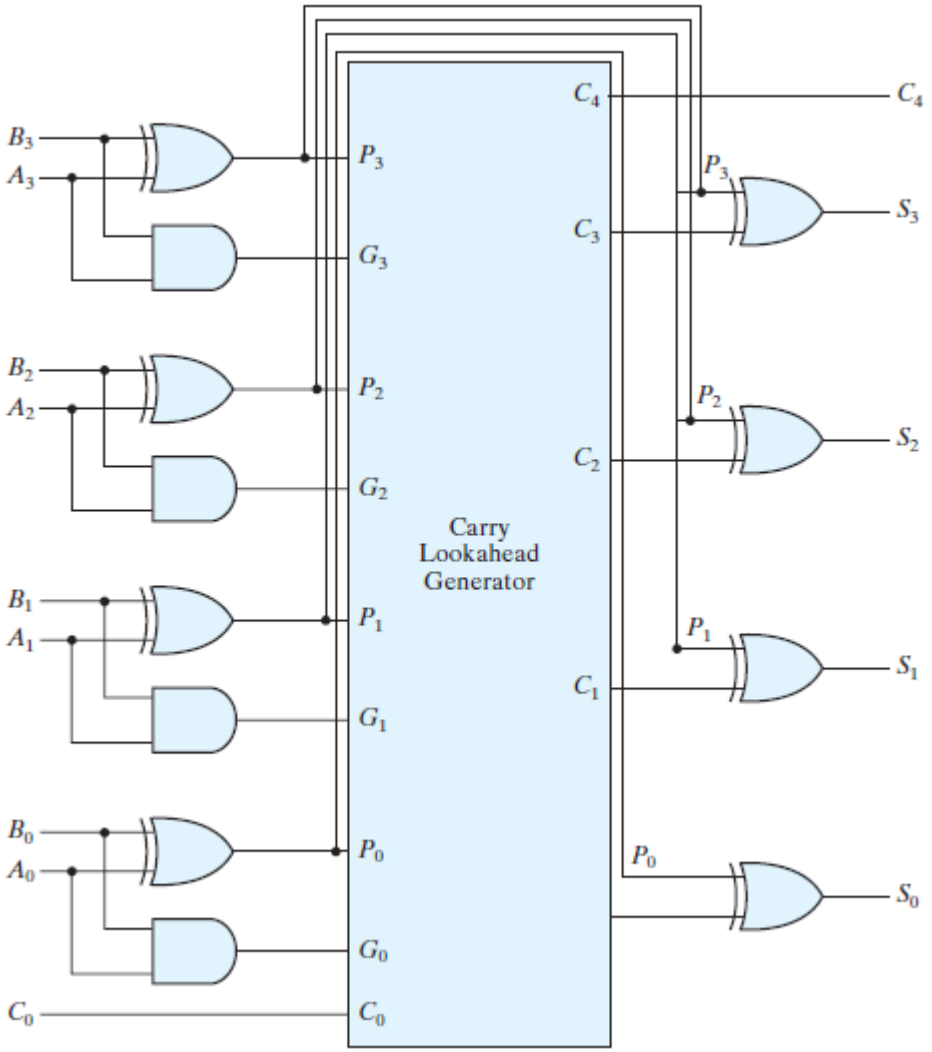
$$C_1 = G_0 + P_0C_0$$

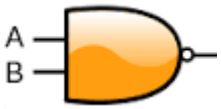
$$C_2 = G_1 + P_1C_1 = G_1 + P_1(G_0 + P_0C_0) = G_1 + P_1G_0 + P_1P_0C_0$$

$$C_3 = G_2 + P_2C_2 = G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C_0$$

4-bites összeadó az átvitel előre történő kiszámításával

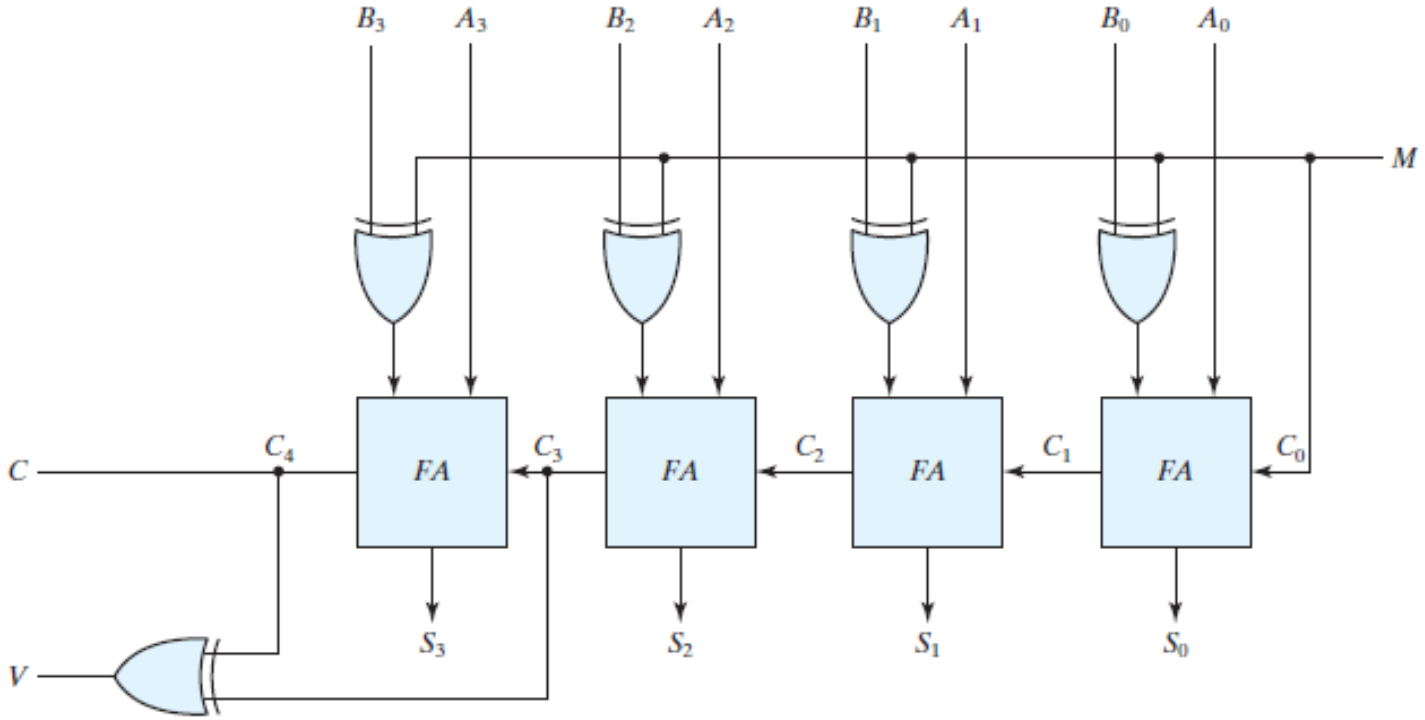






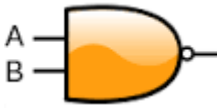
Bináris összeadó/kivonó

Ha M = 0, akkor az alábbi áramkör összeadóként működik.
Ha M = 1, akkor a kapcsolás kivonó áramkörként működik.



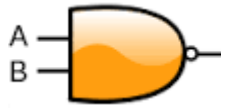
Kivonásnál a **XOR** kapuk segítségével a ***B*** szám egyes komplemensét vesszük, s a C_0 bemenetet is ’1’-be állítjuk, így az ***S*** eredmény ***A*** + ***B*** egyes komplemense + 1 lesz.

A ***B*** szám „egyes komplemense + 1” mennyiséget a ***B*** szám kettes komplemensének hívjuk.



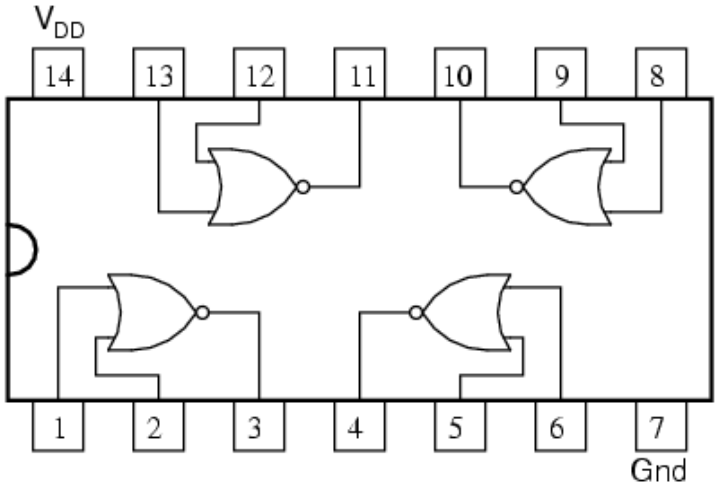
A 4000-es sorozat tipikus tagjai

4001 CMOS Quad 2-Input NOR Gate
4011 CMOS Quad 2-Input NAND Gate
4013 CMOS Dual D-Type Flip Flop
4017 CMOS Decade Counter with 10 Decoded Outputs
4021 CMOS 8-Stage Static Shift Register
4022 CMOS Octal Counter with 8 Decoded Outputs
4023 CMOS Triple 3-Input NAND Gate
4025 CMOS Triple 3-Input NOR Gate
4026 CMOS Decade Counter/Divider with Decoded 7-Segment Display Outputs and Display Enable
4027 CMOS Dual J-K Master-Slave Flip-Flop
4028 CMOS BCD-to-Decimal or Binary-to-Octal Decoders/Drivers
4043 CMOS Quad NOR R/S Latch with 3-State Outputs
4046 CMOS Micropower Phase-Locked Loop
4049 CMOS Hex Inverting Buffer/Converter
4050 CMOS Hex Non-Inverting Buffer/Converter
4051 CMOS Single 8-Channel Analog Multiplexer/Demultiplexer with Logic-Level Conversion
4052 CMOS Differential 4-Channel Analog Multiplexer/Demultiplexer with Logic-Level Conversion
4053 CMOS Triple 2-Channel Analog Multiplexer/Demultiplexer with Logic-Level Conversion
4060 CMOS 14-Stage Ripple-Carry Binary Counter/Divider and Oscillator
4066 CMOS Quad Bilateral Switch
4069 CMOS Hex Inverter
4070 CMOS Quad Exclusive-OR Gate
4071 CMOS Quad 2-Input OR Gate
4072 CMOS Dual 4-Input OR Gate
4073 CMOS Triple 3-Input AND Gate
4075 CMOS Triple 3-Input OR Gate
4081 CMOS Quad 2-Input AND Gate
4082 CMOS Dual 4-Input AND Gate
4093 CMOS Quad 2-Input NAND Schmitt Triggers
4094 CMOS 8-Stage Shift-and-Store Bus Register

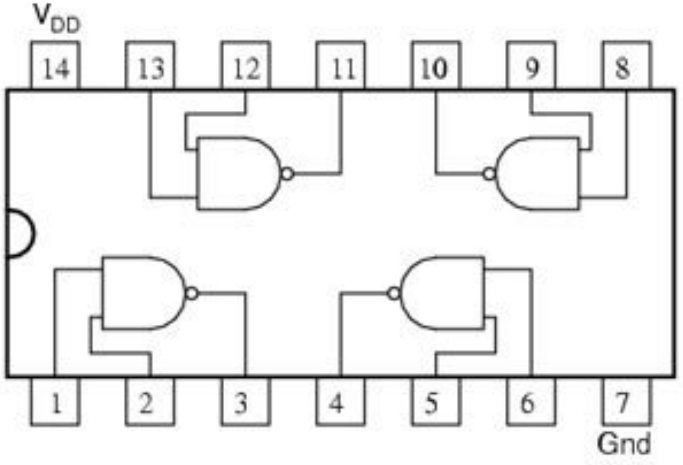


A 4000-es sorozat tipikus tagjai

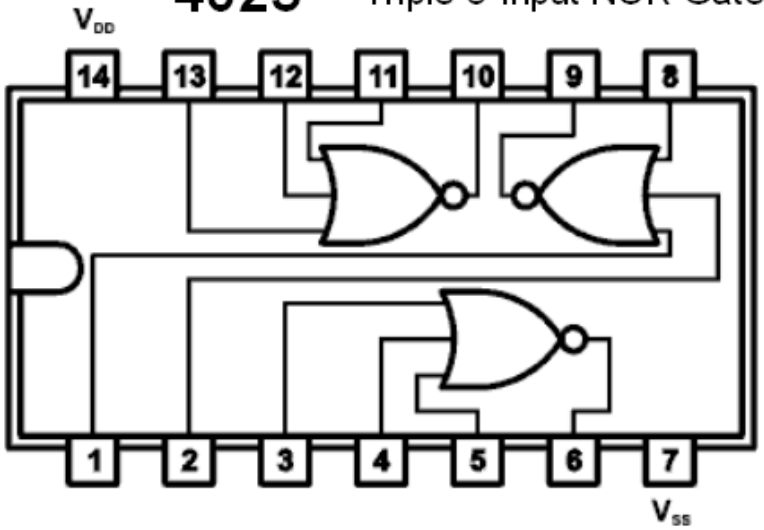
4001 Quad 2-Input NOR Gate



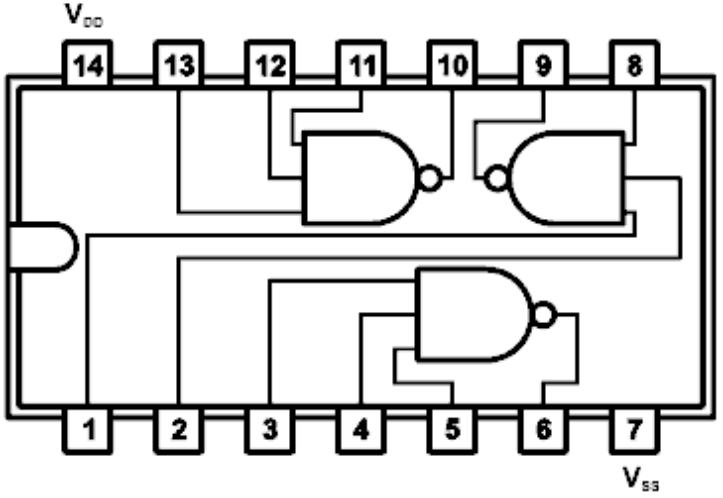
4011 Quad 2-input NAND

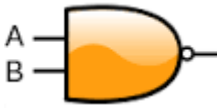


4025 Triple 3-Input NOR Gate



4023 Triple 3-Input NAND Gate

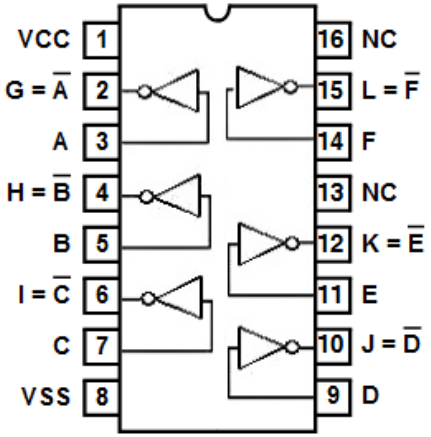




A 4000-es sorozat tipikus tagjai

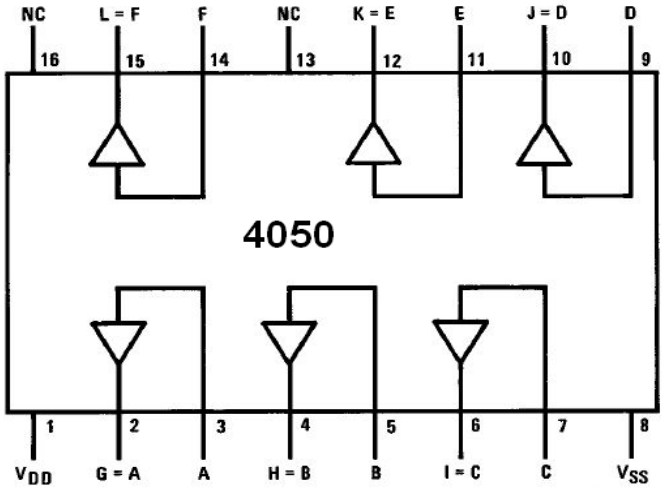
4049

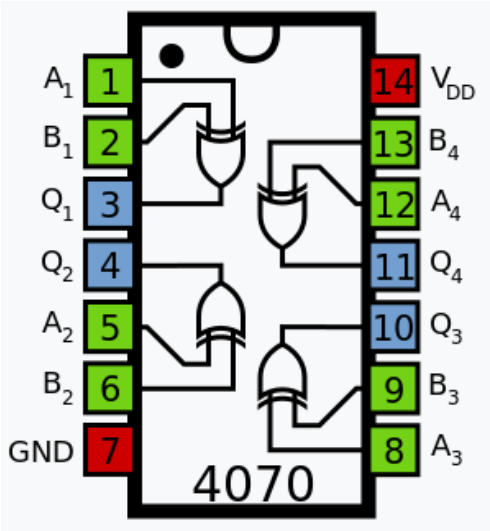
Hex Inverting Buffer/Converter

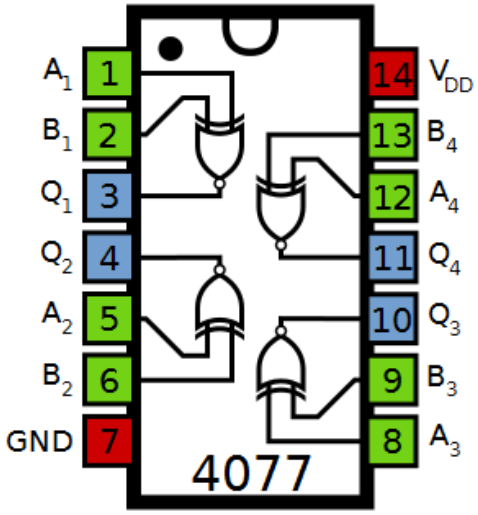


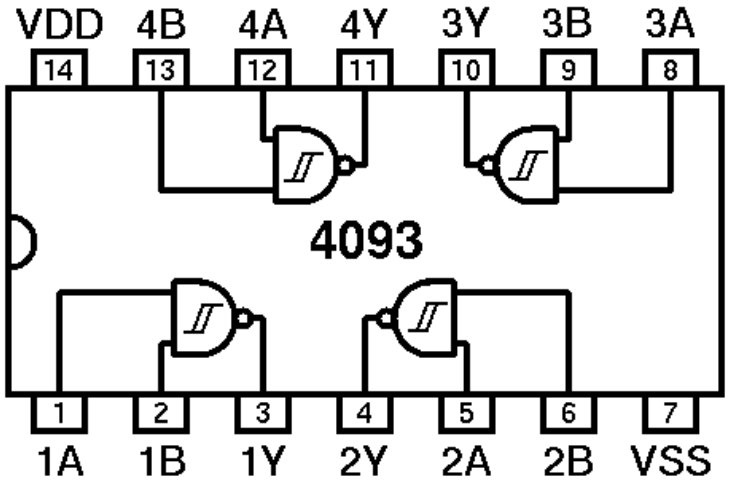
4050

Hex Non-Inverting Buffer/Converter



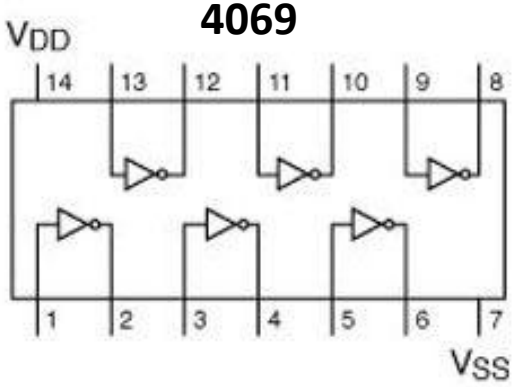




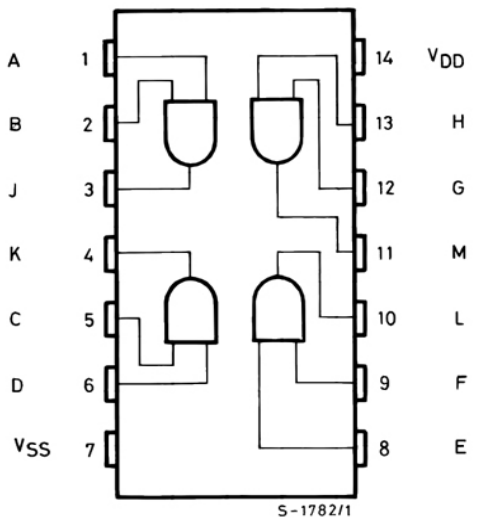


A 4000-es sorozat tipikus tagjai

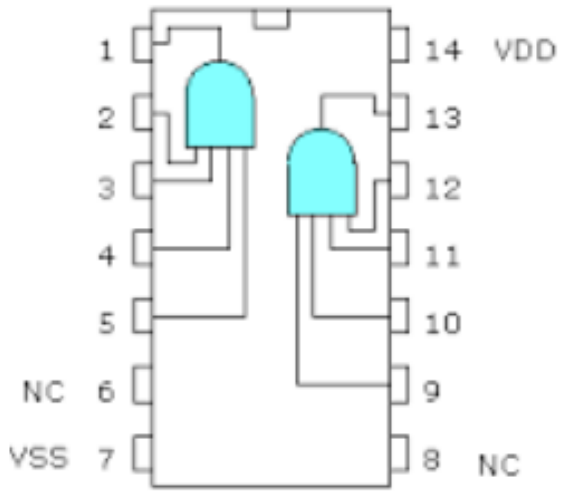
4069

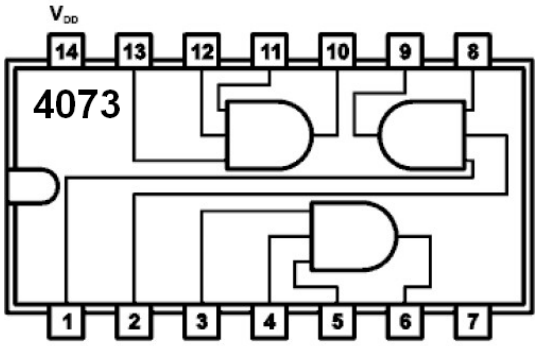


4081



4082





4075