Hobbi Elektronika

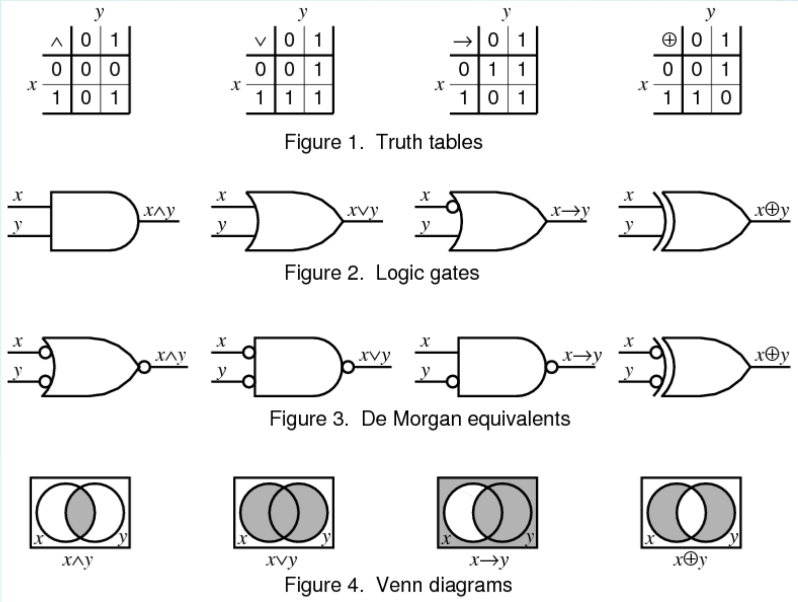


Figure 1. Truth tables

Figure 2. Logic gates

Figure 3. De Morgan equivalents

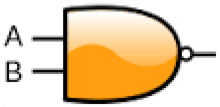
Figure 4. Venn diagrams

Bevezetés az elektronikába:

Boole algebra, logikai kapuáramkörök

Felhasznált anyagok

- M. Morris Mano and Michael D. Ciletti: Digital Design - With an Introduction to the Verilog HDL, 5th. Edition
- Végh János: Ismerkedés a digitális elektronikával
- Mészáros Miklós: Logikai algebra alapjai, logikai függvények I.
- BME FKE: Logikai áramkörök
- Electronics-course.com: Boolean Expression Calculator
- Falstad.com: Circuit simulator
- F-alpha.net: Boolean Algebra
- Neuproductions-be: The Logic Lab
- Breadboard.electronics-course.com: SR-NAND

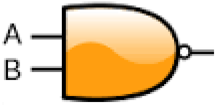


Logikai áramkörök

Számítógépekben, műszerekben, vezérlő automatákban alapvető szerep jut az olyan **áramköröknek,** melyek valamilyen **logikai összefüggést** fejeznek ki. Ezeknek a logikai áramköröknek az építőkövei az úgynevezett **kapuáramkörök,** amelyek egy-egy elemi logikai összefüggés (NEM, ÉS, VAGY kapcsolat) kiértékelésére képesek.

A logikai áramkörök tervezésénél az a cél, hogy **bizonyos események bekövetke-zésénél** az áramkör meghatározott módon **vezéreljen valamilyen eszközt.**
Például a lift induljon felfelé, ha a liftben megnyomtak egy magasabb emeletnek megfelelő gombot, vagy ha egy felsőbb emeleten megnyomták a hívógombot, de ne induljon, ha az ajtó nyitva van, stb.

Az eseményeket, melyek bekövetkeznek vagy nem, **a bekövetkezésükre utaló állításokat,** melyek **igaznak** vagy **hamisnak** bizonyulnak, *logikai változóknak* tekinthetjük, melyeknek két lehetséges értéke 1 és 0.
A logikai változó értéke 1, ha az esemény bekövetkezik, ha az állítás igaz, és 0 az ellenkező esetben. A **logikai áramkörökben** többnyire a feszültség magas, illetve alacsony szintje képviseli az 1 vagy 0 állapotot.



Boole-algebra

A matematikában a **Boole-algebra** (vagy Boole-háló) egy olyan kétműveletes algebrai struktúra amely a **halmazműveletek,** a **logikai műveletek** és az **eseményalgebra** műveleteinek közös tulajdonságaival rendelkezik. **George Boole** angol matematikus mutatott rá először arra (1854), hogy a fenti három terület közötti szoros algebrai jellegű kapcsolat áll fenn.

A **Boole-algebra** megadható közös tulajdonságokkal bíró elemek egy halmazával, a halmazelemeken végezhető műveletekkel és néhány axiómával vagy posztulátummal, amely bizonyos követelményeket támaszt a halmazra és a halmazelemeken végezhető műveletekre nézve. A **posztulátum** szó jelentése: követelmény, kívánalom, saroktétel, alapigazság, elmélet kiindulási pontja.

Többféle **Boole-algebra** definiálható. Bennünket csak az ún. kétértékű Boole-algebra érdekel, amelyet **kapcsolási algebrának** is neveznek. A **kapcsolási algebra** azt vizsgálja, hogy a *logikai kapuáramkörökből* összeállított hálózat kimenetén a lehetséges két állapot melyike valósul meg, ha a bemenetek az egyik vagy másik lehetséges állapotban vannak. Ezért a Boole-algebra az *elektronikus digitális számítógép konstruálásának* nélkülözhetetlen elméleti alapja.

A Boole-algebra axiomatikus definiálása

A kétértékű **Boole-algebra** egy olyan algebrai struktúra, amelyet megadhatunk **B = {0, 1}** halmazával és a halmaz elemein végezhető **+** (szóban: VAGY) és · (szóban: ÉS) bináris műveletekkel, s ezekre teljesülnek **E. V. Huntington** 1904-ben megfogalmazott **posztulátumai:**

1. (a) A **B** halmaz zárt a + bináris műveletre nézve, azaz bármely **x,y** $\in$ **B** esetén a **z** = **x** +**y** művelet eredménye is eleme **B-nek (z** $\in$ **B).**
   (b) A **B** halmaz zárt a · bináris műveletre nézve, azaz bármely **x,y** $\in$ **B** esetén a **z** = **x** · **y** művelet eredménye is eleme **B-nek (z** $\in$ **B).**

2. (a) A 0 zéruselem a + műveletre nézve, vagyis **x + 0** = **0 + x** = **x.**
   (b) Az 1 egységelem a · bináris műveletre nézve, vagyis **x · 1** = **1 · x** = **x.**

3. (a) A struktúra kommutatív a + bináris műveletre nézve, azaz **x + y** = **y + x.**
   (b) A struktúra kommutatív a · bináris műveletre nézve, azaz **x · y** = **y · x.**

4. (a) A · művelet disztributív a + műveletre nézve, azaz **x · (y + z)** = **(x · y) + (x · z).**
   (b) A **+** művelet disztributív a · műveletre nézve, azaz **x + (y · z)** = **(x · y) + (x · z).**

5. bármely **x** $\in$ **B** esetén létezik olyan **x'** $\in$ **B** elem (amelyet x komplemensének nevezünk), amelyre teljesül, hogy: (a) **x + x'** = **1** és (b) **x · x'** = **0.**

6. A **B** halmaznak létezeik legalább két különböző **x,y** $\in$ **B** eleme, amelyre teljesül, hogy **x** $\neq$ **y.**

Megjegyzés: a ***B = {0, 1}*** halmaz elemei nem számok, hanem szimbólumok! Az *x, y, z* logikai változók pedig olyan szimbólumok, melyek értéke vagy a 0 vagy az 1 halmazelem.

A Boole-algebra műveletei

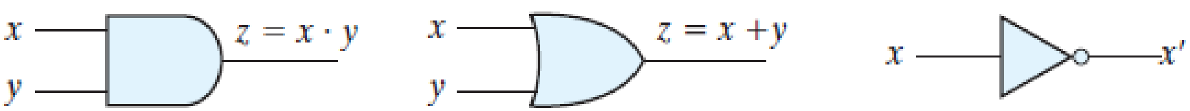
A kétértékű **Boole-algebra** bináris (kétváltozós) műveleteinek eredménye a **B** halmaz valamelyik eleme lesz (0 vagy 1), a műveletben szereplő két logikai változó értékétől függően. A műveletvégzés szabályait táblázatos formában is megadhatjuk.

Az ábrán szereplő harmadik művelet az 5. posztulátumban definiált *komplemens* előállítására szolgál.

x	y	$x \cdot y$
0	0	0
0	1	0
1	0	0
1	1	1

x	y	$x + y$
0	0	0
0	1	1
1	0	1
1	1	1

x	x'
0	1
1	0



Kétbemenetű ÉS kapu Kétbemenetű VAGY kapu NEM kapu (inverter)

A logikai áramkörök építőkövei az úgynevezett **kapuáramkörök,** amelyek egy-egy elemi logikai művelet (NEM, ÉS, VAGY kapcsolat) elvégzésére képesek. A logikai áramköröknél a „0” pl. alacsony jelszinttel, az „1” pedig magas jelszinttel reprezentálható.

Egy logikai kifejezés több, különböző műveletet is tartalmazhat, ekkor a műveleti sorrend:
1. Zárójel, 2. Negáció (**NEM**), 3. **ÉS** művelet, 4. **VAGY** művelet.

Tételek és tulajdonságok

A posztulátumokból és műveleti táblázatokból újabb összefüggéseket vezethetünk le:

Dualitás elve: A posztulátumokat (a) és (b) párok formájában adtuk meg a **+** és · műveletekre. Egyikből megkaphatjuk a másikat, ha a **+** és · műveleteket felcseréljük, valamint a 0 helyébe 1-et, illetve az 1 helyébe 0-t írunk. Ez egy általános tulajdonsága a **Boole-algebrának,** a levezetett tételekre és a levezetés lépéseire is igaz.

Alaptételek

1. **Idempotencia:** (a) $x + x = x$ és (b) $x \cdot x = x$
2. **Korlátosság:** (a) $x + 1 = 1$ és (b) $x \cdot 0 = 0$
3. **Involúció:** $(x')' = x$
4. **Asszociativitás:** (a) $x + (y + z) = (x + y) + z$ és (b) $x \cdot (y \cdot z) = (x \cdot y) \cdot z$
5. **De Morgan azonosságok:** (a) $(x + y)' = x' \cdot y'$ és (b) $(x \cdot y)' = x' + y$,
6. **Abszorpció:** (a) $x + (x \cdot y) = x$ és (b) $x \cdot (x + y) = x$

A Boole-algebra azonosságai

A Boole-algebrában az azonos átalakításokat az alábbi táblázatban összefoglalt azonosságok szerint végezhetjük.

Postulates and Theorems of Boolean Algebra

Postulate 2	(a)	$x + 0 = x$	(b)	$x \cdot 1 = x$
Postulate 5	(a)	$x + x' = 1$	(b)	$x \cdot x' = 0$
Theorem 1	(a)	$x + x = x$	(b)	$x \cdot x = x$
Theorem 2	(a)	$x + 1 = 1$	(b)	$x \cdot 0 = 0$
Theorem 3, involution		$(x')' = x$		
Postulate 3, commutative	(a)	$x + y = y + x$	(b)	$xy = yx$
Theorem 4, associative	(a)	$x + (y + z) = (x + y) + z$	(b)	$x(yz) = (xy)z$
Postulate 4, distributive	(a)	$x(y + z) = xy + xz$	(b)	$x + yz = (x + y)(x + z)$
Theorem 5, DeMorgan	(a)	$(x + y)' = x'y'$	(b)	$(xy)' = x' + y'$
Theorem 6, absorption	(a)	$x + xy = x$	(b)	$x(x + y) = x$

Megjegyzés: Ahol nem zavarja a megértést, a szorzásjelet elhagytuk.

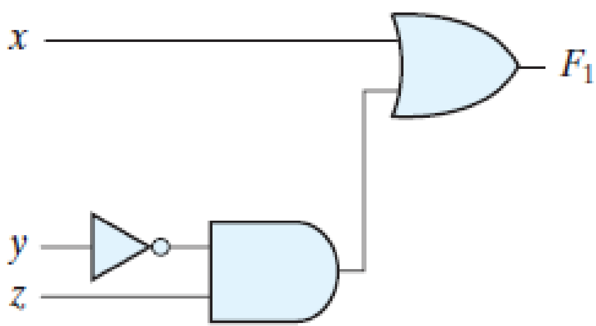
Logikai függvények

Logikai függvényen olyan algebrai kifejezést értünk, amelyek bináris (kétértékű) változókból, a 0 és 1 konstansokból és logikai műveleti jelekből állnak. A bináris változók adott értékénél a függvény értéke 0 vagy 1 lesz.

Például : $\mathbf{F_1 = x + y'z}$

Az $\mathbf{F_1}$ értéke 1 lesz, ha x = 1, vagy ha y' és z egyaránt 1. Minden más esetben $\mathbf{F_1}$ értéke 0 lesz. A komplemensképzés szabályai szerint akkor lesz y' = 1, amikor y = 0. Tehát $\mathbf{F_1}$ értéke akkor lesz 1, ha x = 1 vagy ha y = 0 és z = 1.

Egy logikai függvény megadható az igazságtáblázatával is, illetve az algebrai kifejezés megfelelően összekapcsolt logikai kapuk hálózatával is reprezentálható (minden tagnak egy kapu felel meg, a változók kapubemenetet jelentenek, a kifejezés értéke pedig a kimeneten jelenik meg).



x	*y*	*z*	F_1
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

A logikai áramköröknél a függvény változói bemenetként szerepelnek, az **F1** értékét pedig az áramkör kimenete jelenti.

Logikai függvények egyszerűsítése

Az **igazságtáblázattal** egy logikai függvényt csak **egyféleképpen** fejezhetünk ki. **Algebrai formulával** azonban **többféle módon** is megadhatjuk a függvényt úgy, hogy minden alak ugyanarra az eredményre vezessen. Minden egyes formula megfelel egy-egy logikai kapukból kialakított kapcsolásnak.

Ez az, ami a **Boole-algebra** használatára ösztönöz bennünket. A logikai függvényeknek a **Boole-algebra** szabályait követő átalakítása sok esetben hozzásegít bennünket, hogy egyszerűbb kifejezéssel írjuk fel ugyanazt a függvényt, ezáltal kevesebb kapuval vagy kevesebb bemenetű kapuáramkörökkel valósíthatjuk meg az áramkört.

Tekintsük például a következő logikai függvényt: $\mathbf{F_2 = x'y'z + x'yz + xy'}$

Most egyszerűsítsük a kifejezést a **Boole-algebra** azonosságainak felhasználásával:

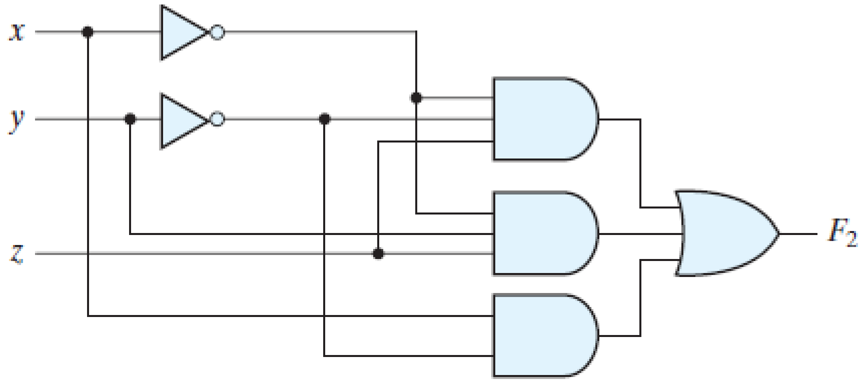
$$F_2 = x'y'z + x'yz + xy' = x'z(y' + y) + xy' = x'z + xy'$$

A függvényt leíró kifejezés két tagra egyszerűsödött, ami egyszerűbb áramköri megvalósítást is jelent.

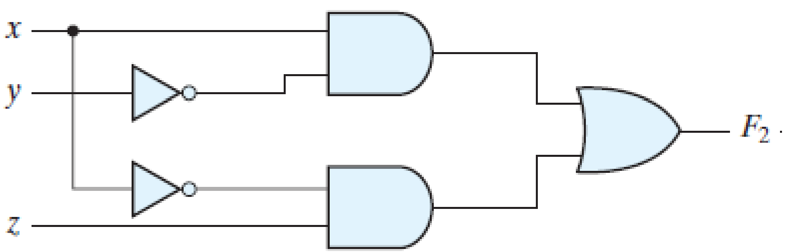
Logikai hálózat egyszerűsítése

Az előző oldalon bemutatott függvény eredeti és egyszerűsített alakjának áramköri megvalósítása az alábbi ábrákon látható.

Nyilvánvalóan az a kedvezőbb megvalósítás, amelyik kevesebb kaput és kevesebb kapubemenetet igényel, mivel az kevesebb alkatrészből és kevesebb vezetékezéssel kivitelezhető. Általában egy-egy logikai függvénynek sokféle megvalósítása lehet. A leggazdaságosabb megvalósítás megtalálása fontos része a tervezésnek.



(a) $F_2 = x'y'z + x'yz + xy'$



(b) $F_2 = xy' + x'z$

Algebrai átalakítások

Egyszerűsítsük az alábbi logikai függvényeket!

1. $x(x' + y) = xx' + xy = 0 + xy = xy.$

2. $x + x'y = (x + x')(x + y) = 1(x + y) = x + y.$

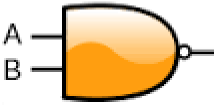
3. $(x + y)(x + y') = x + xy + xy' + yy' = x(1 + y + y') = x.$

4.
$$\begin{aligned} xy + x'z + yz &= xy + x'z + yz(x + x') \\ &= xy + x'z + xyz + x'yz \\ &= xy(1 + z) + x'z(1 + y) \\ &= xy + x'z. \end{aligned}$$

5. $(x + y)(x' + z)(y + z) = (x + y)(x' + z),$ **a dualitás elve alapján 4.-ből.**

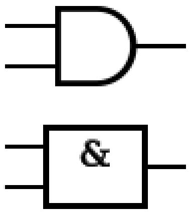
Megjegyzések:

- Az 1. és 2. függvény egymás duális párjai, hasonlóan a levezetés megfelelő lépései is.
- A 3. függvény egyszerűsítésére könnyebb utat is találhatunk a 2.1 táblázatban szereplő **4.b** posztulátum felhasználásával: (x + y)(x + y') = x + yy' = x.
- Az 5. függvény minimalizálását nem vezettük le, hanem a dualitás elve alapján a 4. feladatból származtattuk.
- A 4. és 5. azonosságot együttesen egyébként a **Boole-algebra „Konszenzus-tétel”**-ének nevezzük.



Logikai kapuk

A logikai kapuk jelölése kiviteltől és felépítéstől független. A hagyományos és a „szögletes” jelek tehát az absztrakt tervezésnél is használhatók.

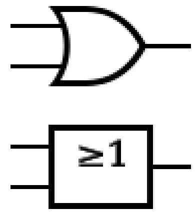


Y = A · B

a kimenet csak akkor 1, ha mindkét bemenet 1 állapotban van.

ÉS (AND)
Logikai szorzás

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

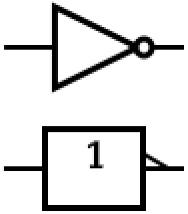


Y = A + B

a kimenet akkor 1, ha bármelyik bemenet 1 állapotban van.

VAGY (OR)
Logikai összeadás

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1



Y = $\bar{A}$

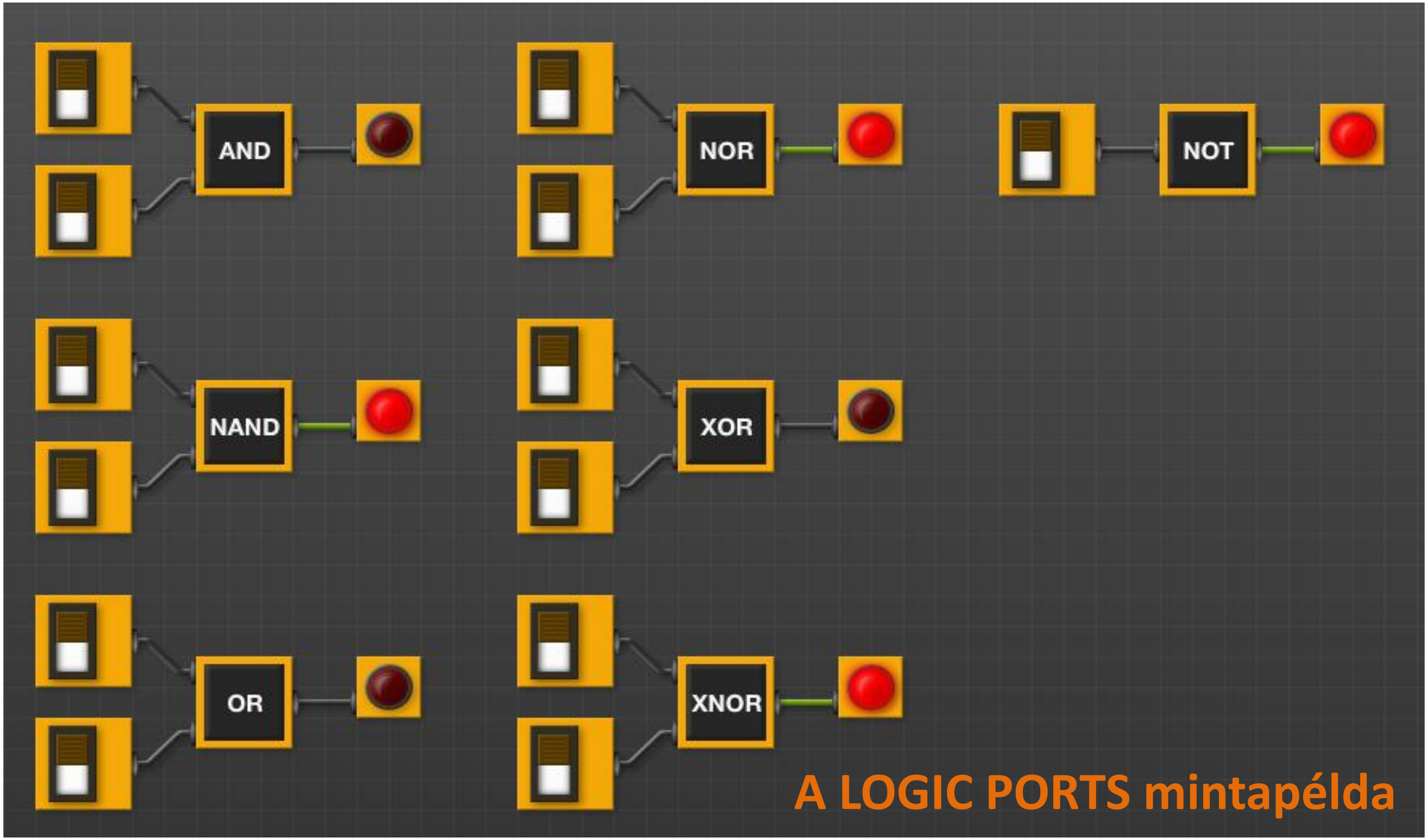
A kimenet akkor 1, ha a bemenet 0 állapotban van.

NEM (NOT, INV)
Logikai tagadás, invertálás

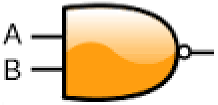
A	Y
1	0
0	1

Új játék: The Logic Lab

A logikai áramkörök szimulációját a THE LOGIC LAB segítségével is végezhetjük!
Link: http://www.neuroproductions.be/logic-lab/



A LOGIC PORTS mintapélda

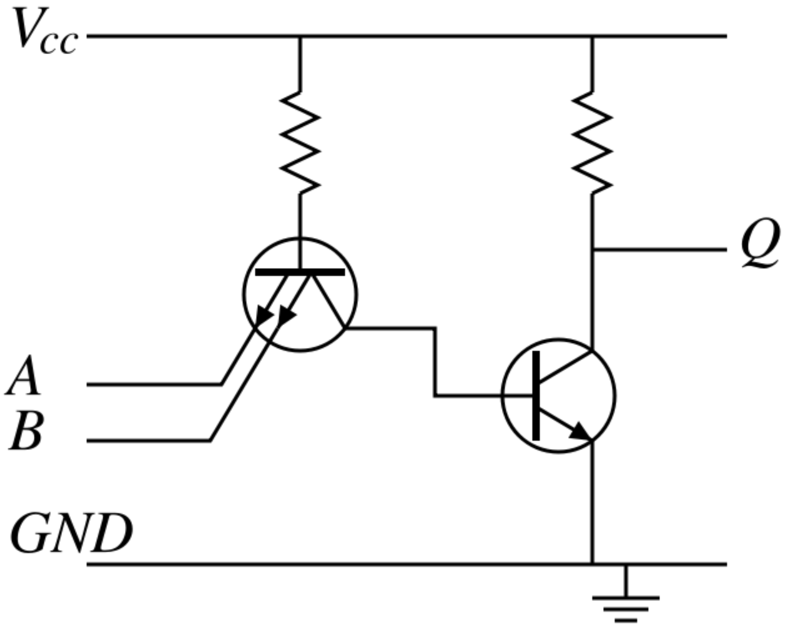


TTL Logikai áramkörök

A TTL (Transistor-Transistor Logic) kapcsolásban bipoláris tranzisztorok vannak, s a logikai bemenetek többemitteres tranzisztorok emitter kivezetései.

1. A több emitterrel rendelkező tranzisztorok működése egyenértékű azzal, ha több tranzisztor bázisát és kollektorát összekötjük.
2. Az ábrán látható kétbemenetű NEM-ÉS (NAND) áramkör ebben a leegyszerűsített formában nem mentes az RTL áramkörök ismert problémáitól.

A probléma a következő oldalon bemutatott totem-pole kimenettel orvosolható.



Két bemenetű NAND áramkör

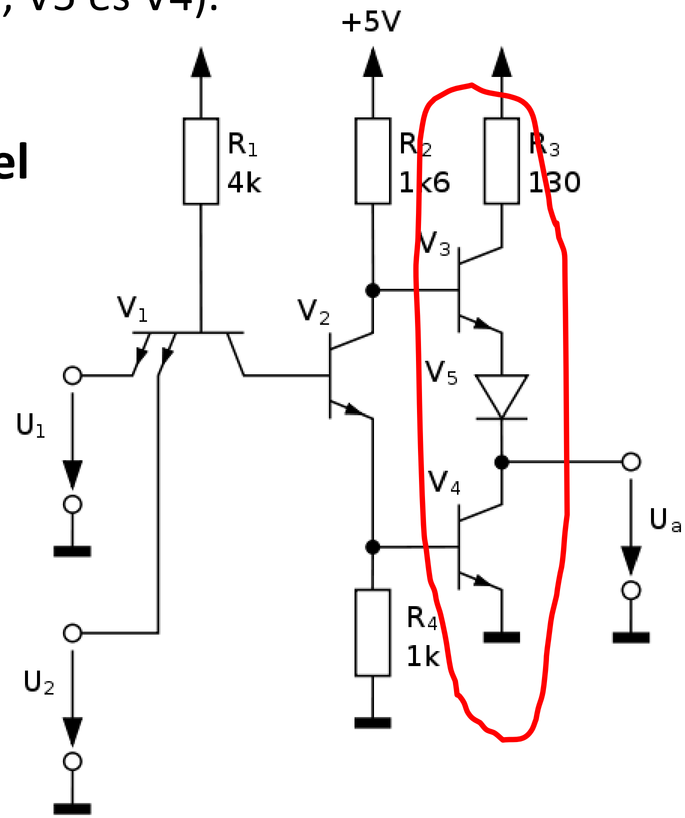
TTL kapu totem-pole kimenettel

Totem-pole = totem oszlop. Fából faragott oszlop, figurái egymás fölött helyezkednek el, „egymás fején ülnek”… A TTL kimenet elnevezése onnan ered, hogy itt is egymás tetejére ültetett alkatrészeket látunk (az ábrán V3, V5 és V4).

Kétbemenetű TTL NAND áramkör totem-pole kimenettel

1. Alaphelyzetben V2 és V4 vezet, a kimenet alacsony szintű.
2. Ha valamelyik bemenetet lehúzzuk, V2 és V4 nem vezet, V3 vezet: magas kimeneti szint.

Hátrány: viszonylag alacsony kimeneti magas szint (~3,5 V)



TTL Inverter szimulációja

A http://www.falstad.com/circuit/ címen elérhető áramkör szimulátor segítségével vizsgáljuk a kapcsolás működését!
A Circuits/Logic Families/TTL Inverter mintapéldát nézzük meg!

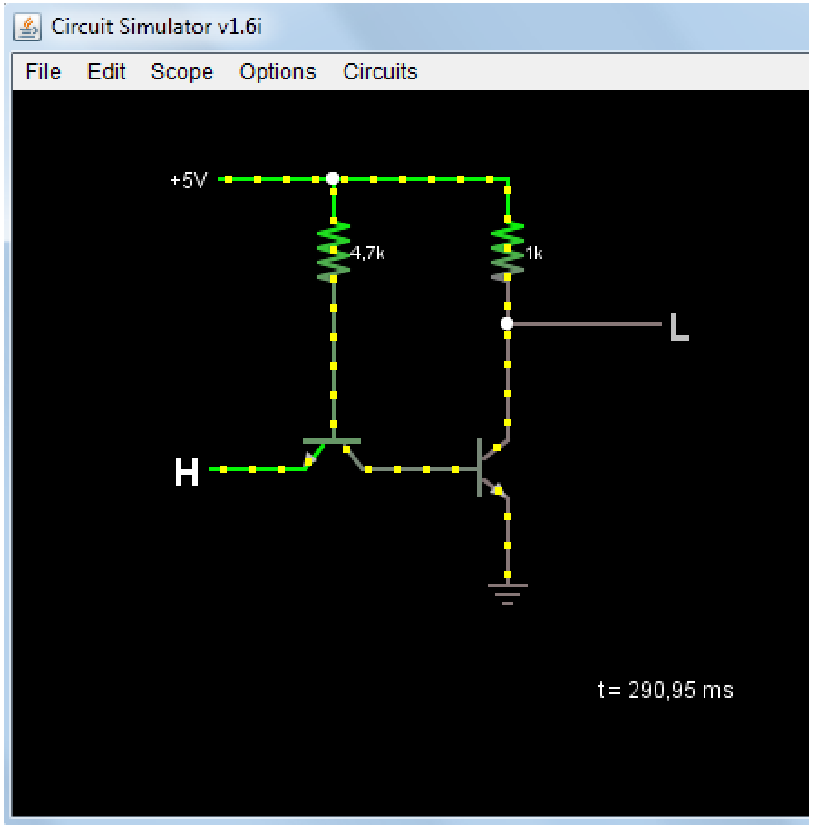
Baloldalt a bemenet állapota egérkattintással váltogatható:

H = magas szint (logikai 1)

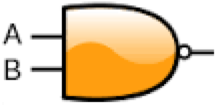
L = alacsony szint (logikai 0)

Jobboldalt a kimenet állapota látható.

Len a be- és kimeneti feszültségek oszcilloszkópos megjelenítése látható.

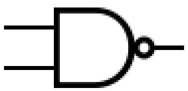


TTL Inverter (NEM áramkör)

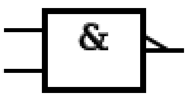


TTL NAND (NEM-ÉS)

A http://www.falstad.com/circuit/ címen elérhető áramkör szimulátor segítségével vizsgáljuk a kapcsolás működését!
A Circuits/Logic Families/TTL NAND mintapéldát nézzük meg!

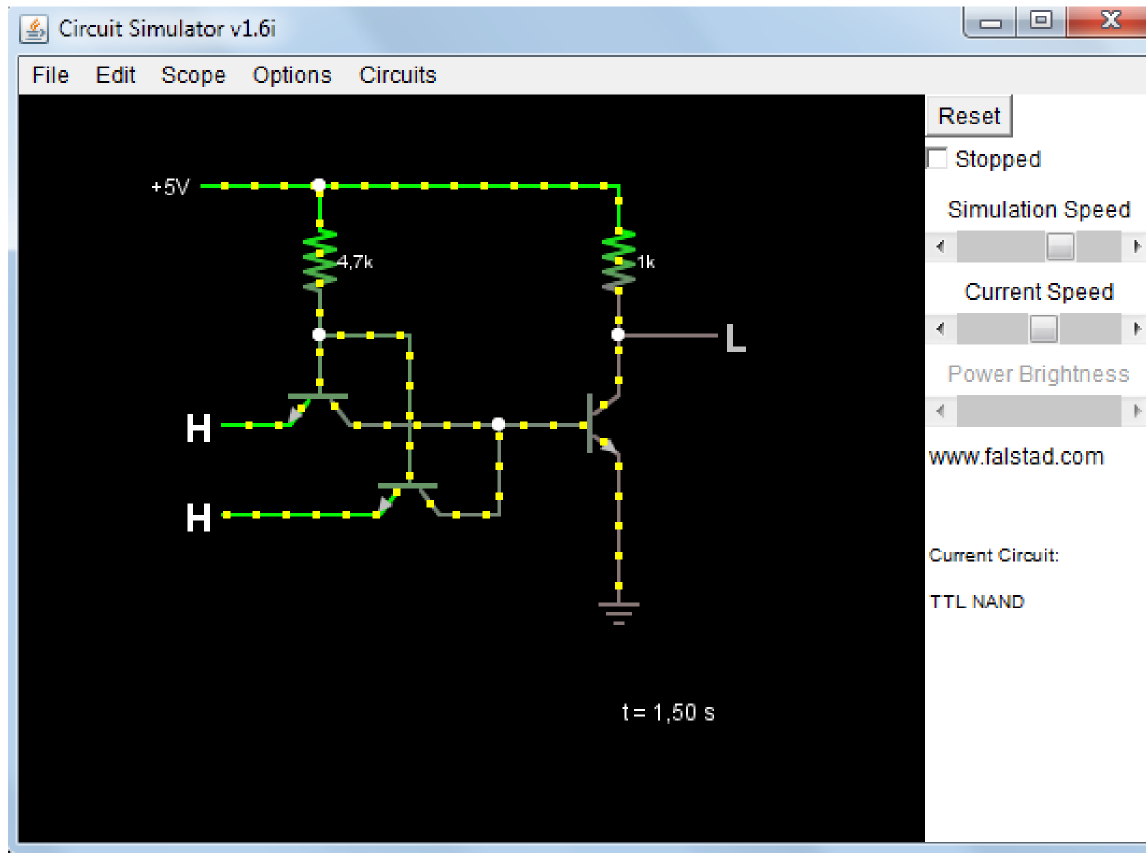


$$Y = \overline{A \cdot B}$$

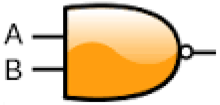


A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

A kimenet csak akkor alacsony, ha minden bemenet magas szinten van



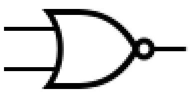
2 bemenetű TTL NAND (NEM-ÉS)



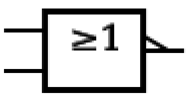
TTL NOR (NEM-VAGY)

A http://www.falstad.com/circuit/ címen elérhető áramkör szimulátor segítségével vizsgáljuk a kapcsolás működését!
A Circuits/Logic Families/TTL NOR mintapéldát nézzük meg!

Rajzjele:

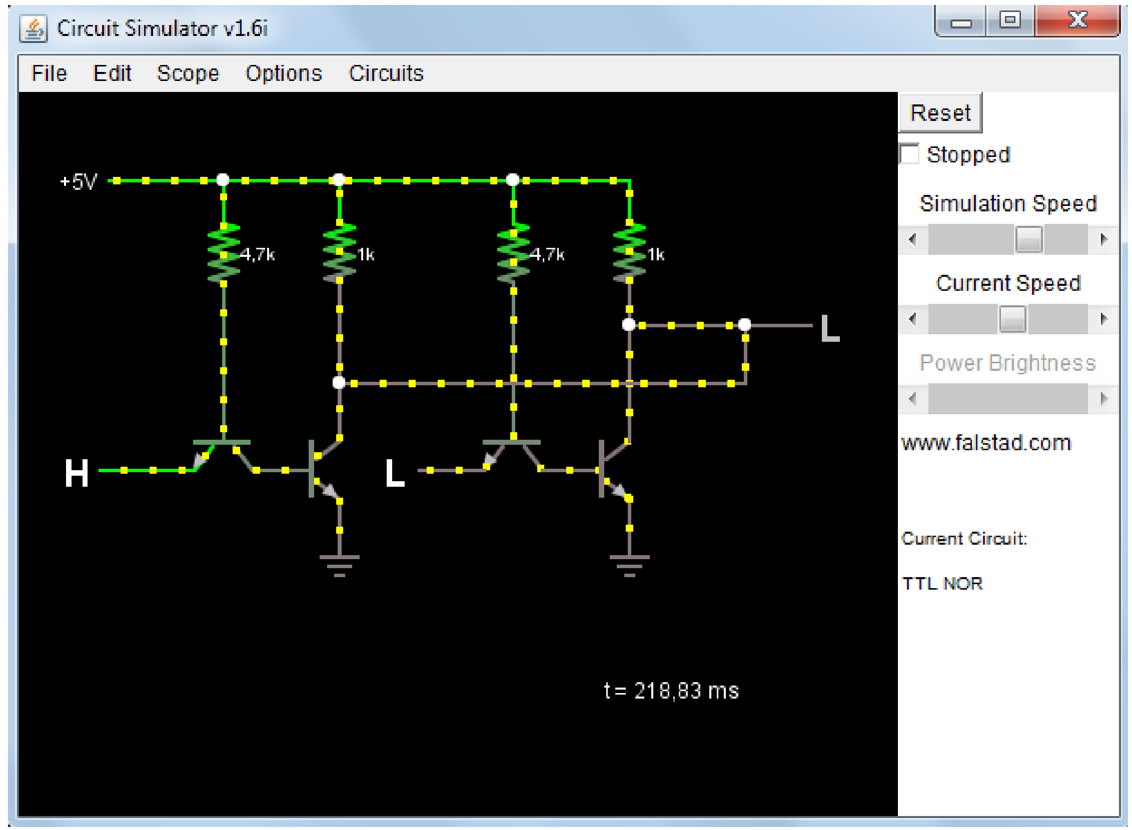


$$Y = \overline{A + B}$$

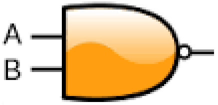


A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

A kimenet csak akkor magas, ha minden bemenet alacsony szinten van



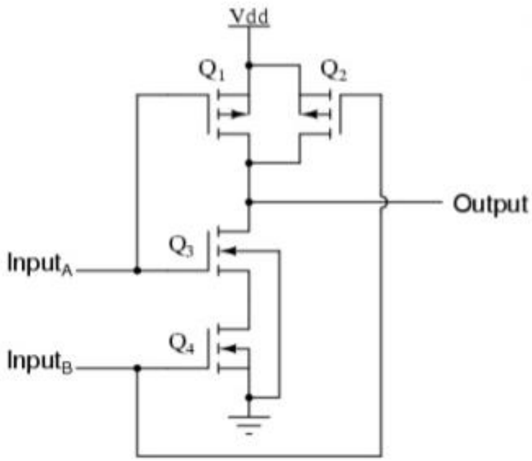
2 bemenetű TTL NOR (NEM-VAGY)



CMOS kapuk

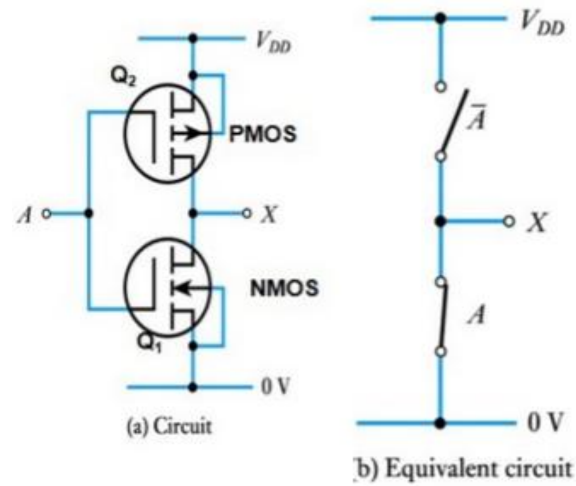
CMOS = Complementary MOS, p- és n-csatornás FET-ekből épülnek fel.

CMOS NAND

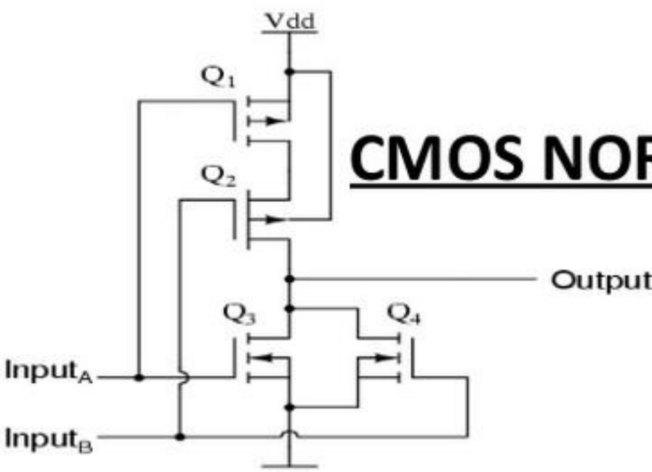


Input		Transistor				O/P
A	B	Q1	Q2	Q3	Q4	Y
0	0	ON	ON	OFF	OFF	1
0	1	ON	OFF	OFF	ON	1
1	0	OFF	ON	ON	OFF	1
1	1	OFF	OFF	ON	ON	0

CMOS inverter



CMOS NOR

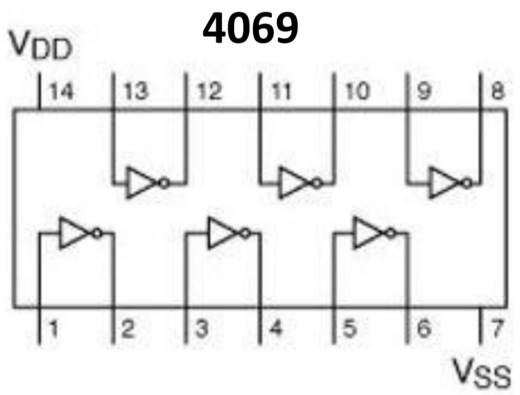


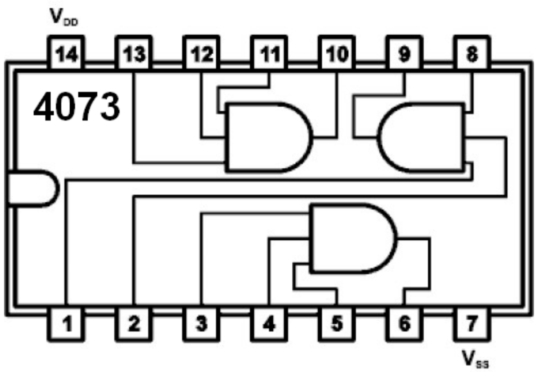
Input		Transistor	Output
A	B	Q1 Q2 Q3 Q4	Y
0	0	ON ON OFF OFF	1
0	1	ON OFF OFF ON	0
1	0	OFF ON ON OFF	0
1	1	OFF OFF ON ON	0

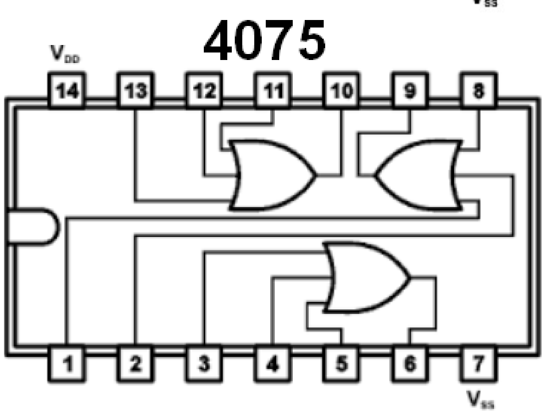
A 4000-es sorozat tipikus tagjai

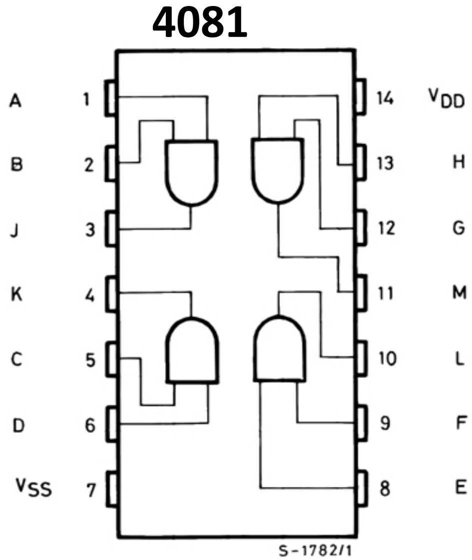
4001	CMOS Quad 2-Input NOR Gate
4011	CMOS Quad 2-Input NAND Gate
4013	CMOS Dual D-Type Flip Flop
4017	CMOS Decade Counter with 10 Decoded Outputs
4021	CMOS 8-Stage Static Shift Register
4022	CMOS Octal Counter with 8 Decoded Outputs
4023	CMOS Triple 3-Input NAND Gate
4025	CMOS Triple 3-Input NOR Gate
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

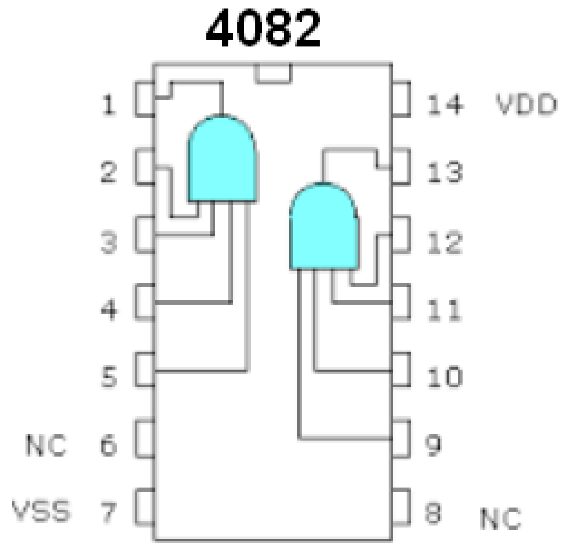
A 4000-es sorozat tipikus tagjai

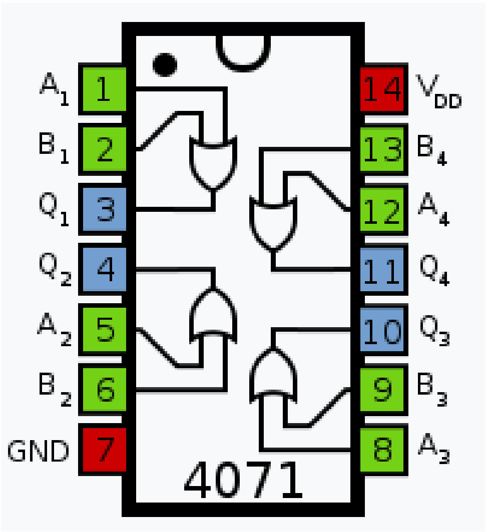


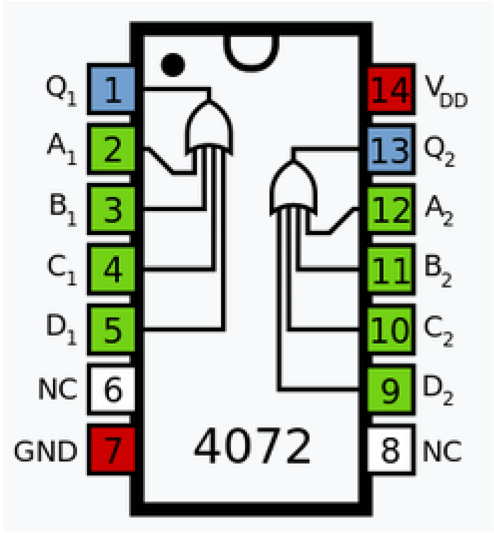


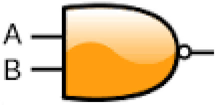






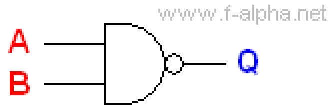






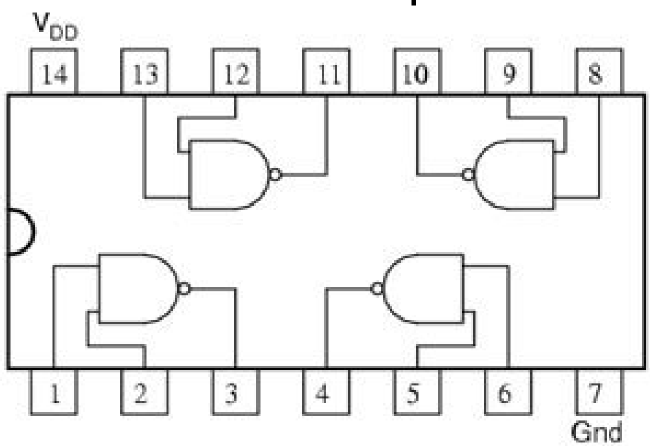
NAND kapu

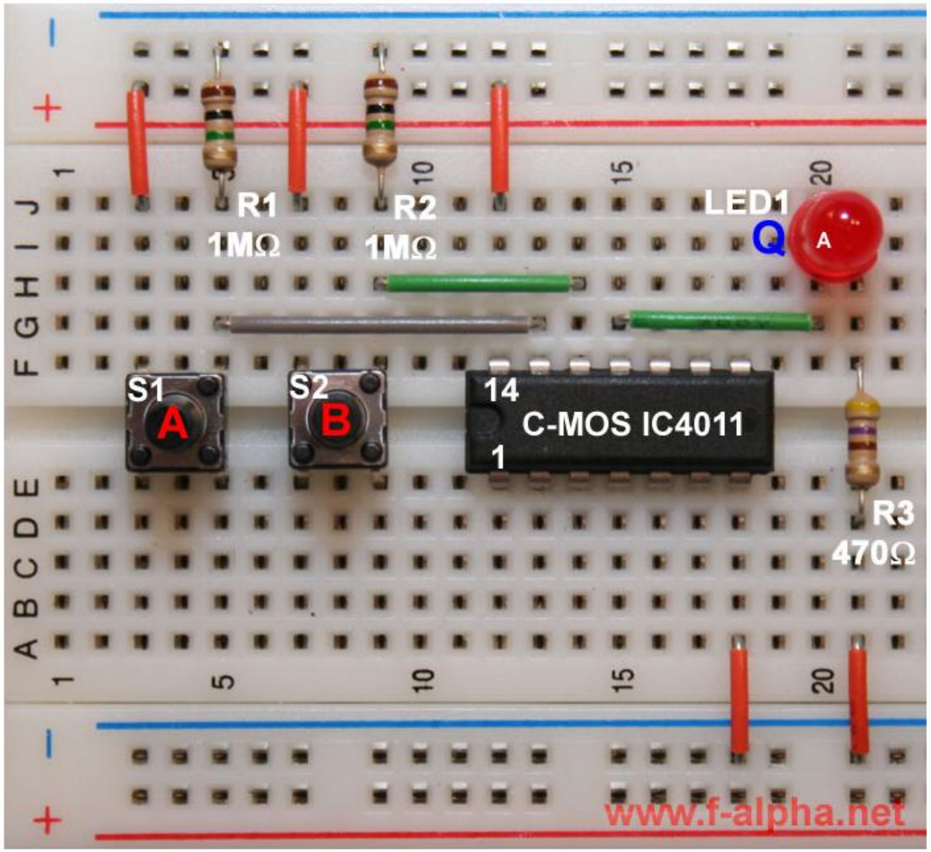
NAND = nem-ÉS



A	B	Q
0	0	1
0	1	1
1	0	1
1	1	0

4011 Quad 2-input NAND





A LED akkor világít, ha a Q kimenet (itt a 11-es láb) jelszintja magas (Q = 1).

A nyomógombok elengedve alacsony, lenyomva magas szintet biztosítanak a megfelelő bemeneten (A: 13. láb, B 12. láb).

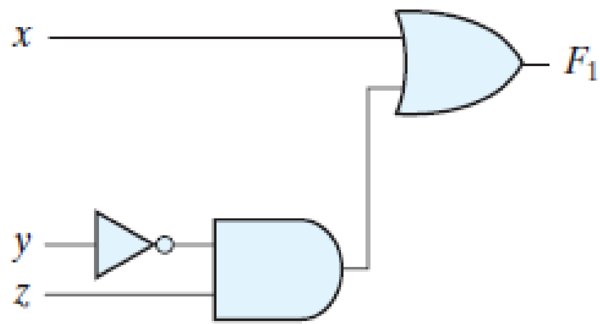
Ellenőrizzük az igazságtábla teljesülését!

Forrás: en.f-alpha.net

Átalakítás NAND kapukra

A kétértékű Boole-algebra tulajdonságaiból következik, hogy bármely logikai függvény megvalósítható csupán NAND (nem-ÉS), vagy hasonlóan csupán NOR (nem-VAGY) áramkörökből.

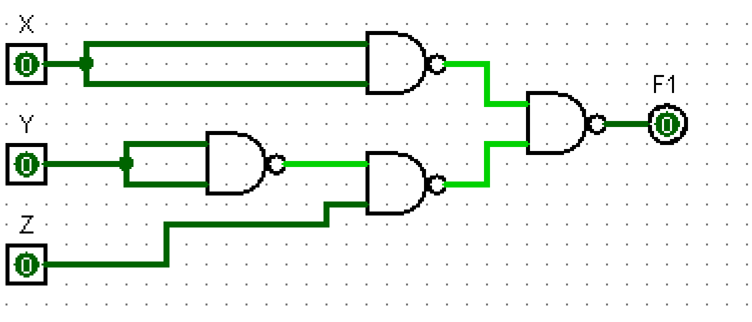
Például : **$F_1 = x + y'z$**



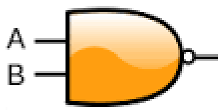
x	y	z	F1
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Alakítsuk át a kapcsolást csupa NAND kapura a **Logisim** Program segítségével! Az eredeti kapcsolás megrajzolása után a **Project/Analyze Circuit** menüben kattintsunk a **Build Circuit** gombra és tegyünk pipát a Use NAND Gates sor elé!

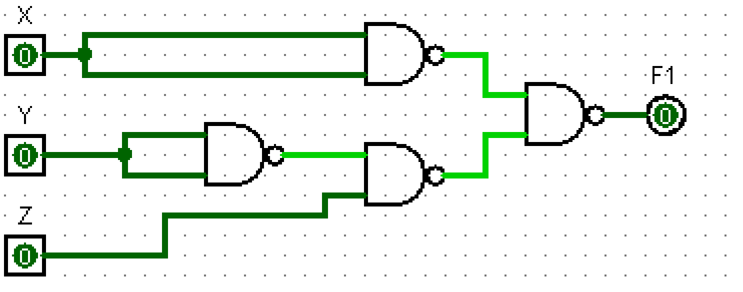
Az így átalakított kapcsolás egyetlen **CD4011** IC-vel megépíthető!



A Logisim program honlapja:
www.cburch.com/logisim/

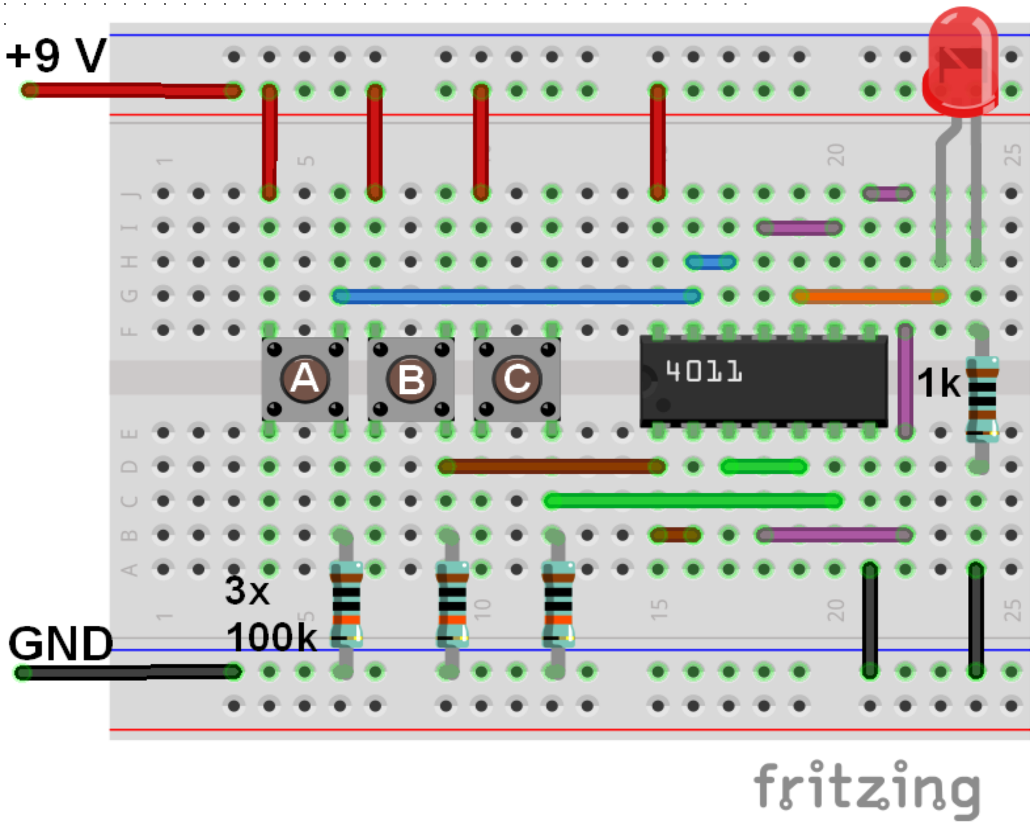


$F_1 = x + y'z$ megvalósítása NAND kapukkal



Építsük meg NAND kapukkal az áramkört és ellenőrizzük a működését!

x	y	z	F_1
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1



4011 Quad 2-input NAND