Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Méréstechnika és Információs Rendszerek Tanszék

Digitális technika
VIMIAA03
1. EA

Fehér Béla, Benesóczky Zoltán
BME MIT

Digitális Technika

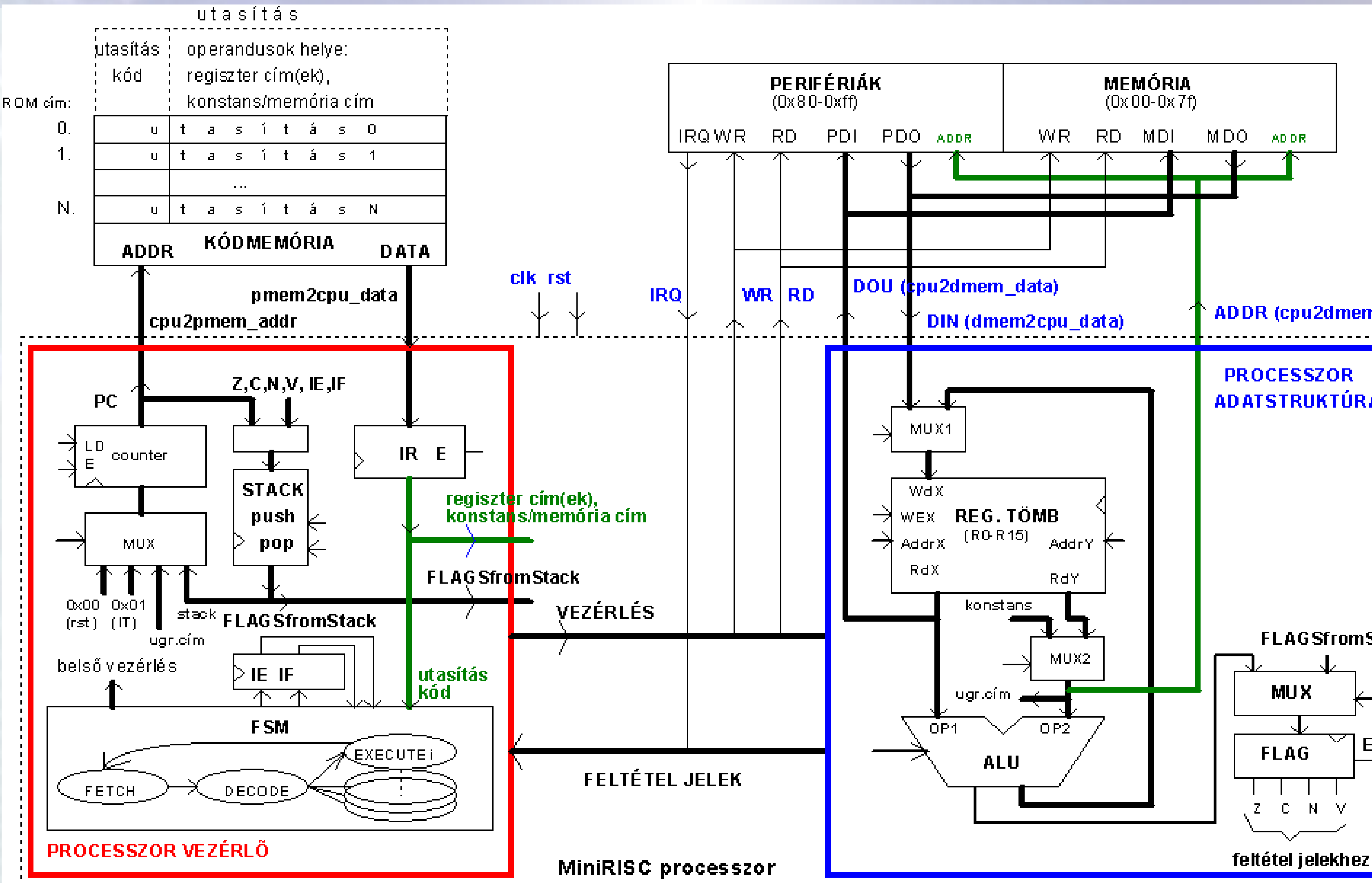
Az előadás témái:

- **Elméleti alapok**
  - Digitális információfeldolgozó gépek, logikai jelek
  - Boole algebra, logikai kapuk
  - Logikai függvények és specifikálásuk
  - Kombinációs hálózatok tervezése

Digitális Technika

- A tárgy célja azon digitális technikai ismeretek átadása, amelyek elsajátításával *megérthető egy egyszerű processzor ill. mikrokontroller felépítése, működése és a programozásának alapjai.*
- A következő oldali ábrán a tárgyban ismertett MiniRISC mikrokonroller funkcionális blokkvázlata látható. Elég bonyolult...
- A teljes rendszer megértéséhez először el kell sajátítani az egyes részegségek működésének alapjait.
- Az alapoktól indulunk, a feladat nagy. A ***folyamatos*** (hétről hétre) ***tanulás szükséges***!

Digitális Technika



Digitális Technika

- A digitális technika tárgyban *digitális információfeldolgozó gépek*kel foglalkozunk. (A MiniRISC mikrokontroller egy bonyolult információfeldolgozó gép, de a legegyszerűbb alapelemek, a logikai kapuk is annak tekinthetők.)

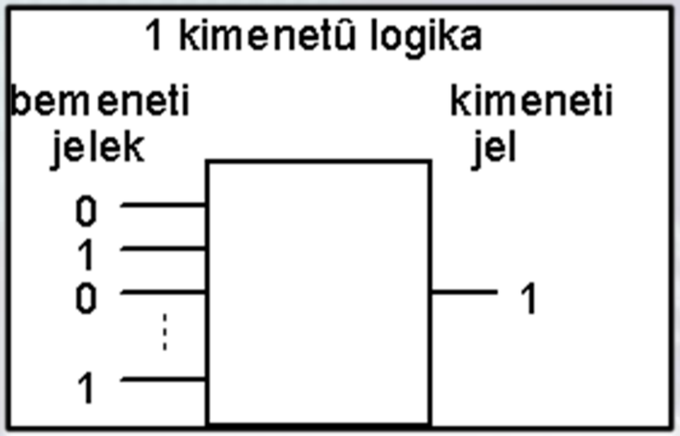
- Ezekre, illetve ezek részeire sokszor a *logikai áramkör*, *logikai hálózat*, *logikai modul*, *logikai egység*, *logikai elem* vagy egyszerűen a *logika* elnevezésekkel hívatkozunk.

Digitális Technika

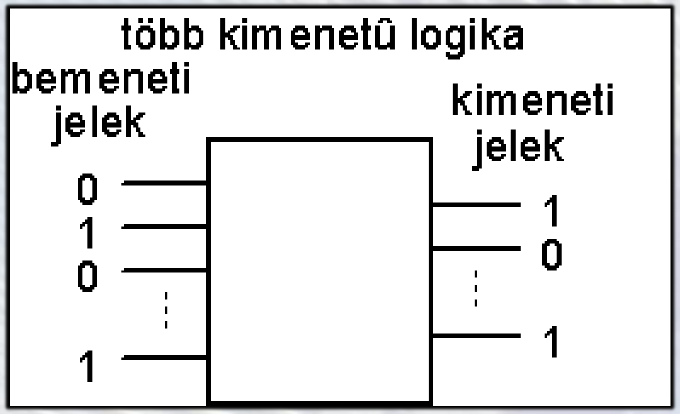
- A logikai egységek bemeneti és kimeneti jelei ***két értéket vehetnek fel:*** **0, 1**
- Ezeket az értékeket másképpen is szokták jelölni:
  A **0** jelölései: ***hamis, false, low, L, alacsony szint***
  Az **1** jelölései: ***igaz, thru, high, H, magas szint***
- A digitális információfeldolgozó gépek a jel bemeneteire érkező 0-ákból és 1-ekből álló ***bemeneti kombinációt*** (és a belső tárolóikból származót is, ha van ilyen) felhasznál-va állítják elő a kimeneti jeleiket.

Digitális Technika

- A kimeneti jelek száma egyszerűbb esetekben 1 (*1 bites*).



- Azonban a kimenet száma az esetek többségében *több mint 1*.



Digitális Technika

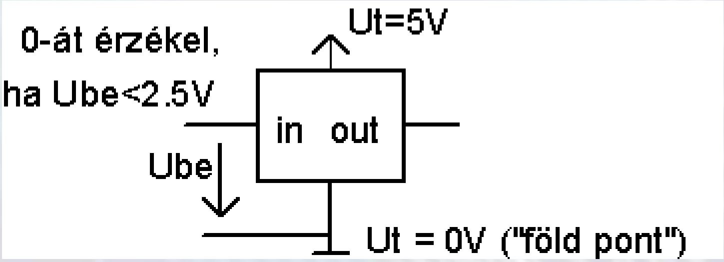
- A logikai áramkörök esetében az információt, általában a feszültség értéke hordozza (de hordozhatja az áram értéke is).

- Sokféle szabvány létezik, itt példaképpen a legelterjettebb alapján mutatjuk az elvet, de nem mutatunk be konkrét szabványt és nem megyünk részletekbe.

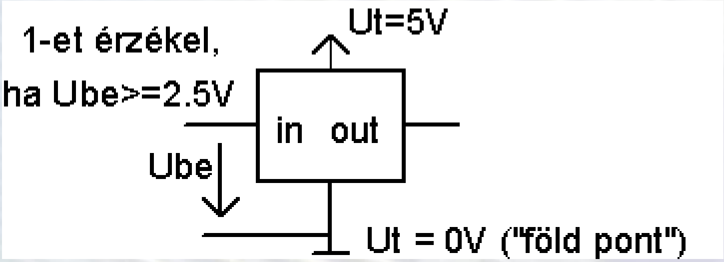
- A logikai áramkörök tápfeszültsége többféle lehet, elterjedt az 5V (Volt) és a 3.3V.

Digitális Technika

- A áramkör a bemeneti logikai a jelet 0-nak érzékeli, ha a feszültség kisebb egy előre meghatározott értéknél (U_c). (Pl. A tápfeszültség (U_t) felénél.)

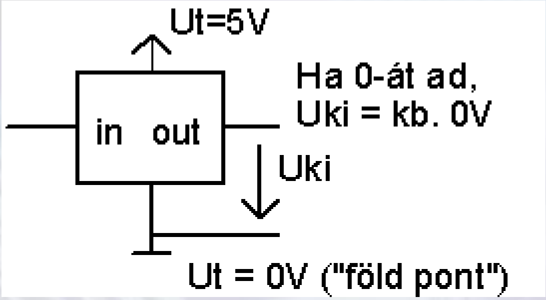


- Az áramkör bemeneti logikai jelet 1-nek érzékeli, ha a feszültség nagyobb vagy egyenlő egy előre meghatározott értéknél.

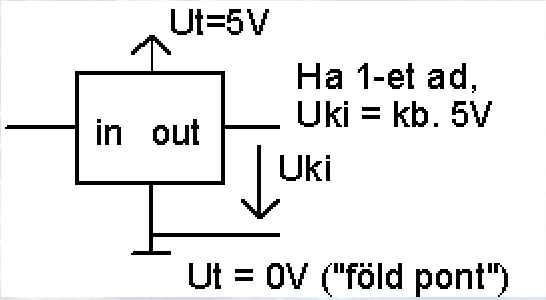


Digitális Technika

- A logikai áramkör kimenete 0 logikai érték esetén közel 0 V (Volt) feszültséget ad.



- A logikai áramkör kimenete 1 logikai érték esetén közel a tápfeszültség (U_t) értékét adja.



- Az egymáshoz kapcsolódó logikai elemek a jelekre rakódódó, a tápfeszültség felével összemérhető zaj (zavaró jel) esetén is jól működnek.

Boole algebra

Ahhoz, hogy megértsük akár a legegyszerűbb logikai áramkör működését, szükségünk lesz a logikai áramkörök tervezésének matematika alapjára, ez a

Boole algebra

A {*B*, +, *, ¯} négyes a Boole algebra, ahol

- A ***B*** Boole halmaz a logikai konstansok: (0, 1) és logikai változók (A, B, C, ...) halmaza.

- A logikai változók a konstansok (0, 1) értékét vehetik fel.

A műveletek a ***B*** elemei között :

- **+** logikai **összeadá**s, **VAGY, OR**
- ***** logikai **szorzás**, **ÉS, AND**
- **¯** **negálás**, (invertálás, ellentett képzés, /-el is jelölhetjük)

Boole algebra

- Az Boole algebra különböző alkalmazási területeken jelenik meg pl:
  - **Logikai algebra** ÉS (*), VAGY (+), INV (¯)
  - Halmazalgebra ∩, U, ¯ (metszet, únió,komplemens.)
  - Kapcsolási algebra (kapcsolókkal megvalósított logikai műveletek)
- A műveletek jelölései adott területeken eltérőek lehetnek. Mi a **logikai algebra** jelöléseit használjuk, de később **Verilog** hardver leíró nyelvvel is foglalkozunk, ezért az ottani jelöléseket is megadjuk:

  VAGY: + **(pl. A+B)** (Verilog: A | B)

  ÉS: *** (pl. A*B)** (Verilog: A & B)

  Ellentett (inverz, invertálás): ¯, /, **n**

  (pl. $\overline{A}$ **, /A, nA)** (Verilog: ~A)

Boole algebra

- A Boole algebrát **axiómák**kal (igaznak elfogadott állításokkal) adjuk meg.
- **Az axiómákból levezethetők a tételek** (a Boole algebrában igaz, fontos állítások).

A Boole algebra axiómái a *B* (Boole halmaz) B elemeire

- A1: B=0, ha B≠1 A1d: B=1, ha B≠0 (kétértékű)
- A2: $\overline{0} = 1$ A2d: $\overline{1} = 0$ (invertálás)
- A konstansokkal végzett műveletek:
- A3: 0 * 0 = 0 A3d: **1 + 1 = 1**
- A4: 1 * 1 = 1 A4d: 0 + 0 = 0
- A5: 0*1=1*0=0 A5d: 1+0=0+1=1

(Más axióma rendszerek is léteznek.)

Boole algebra

- **Műveletvégzési sorrend** (prioritás): ( ) → ¯ → * → +

  pl. /A*B+C → (/A) → (/A)*B, → ((/A)*B) + C

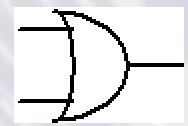
Dualitás elve

- **Ha a Boole** ***azonosságokban a 0 és 1 szimbólumokat és a * és + műveleteket egyidejűleg felcseréljük, az azonosságok érvényesek maradnak*** (Előbbiekben Aid jelöli az Ai axióma duálisát, i=1..5)

  Pl. A3: 0 * 0 = 0 duálisa: A3d: 1 + 1 = 1

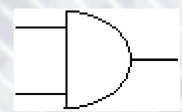
Boole algebra

- Az alapműveleteket megvalósító elemi logika áramköröket **logikai kapuknak** nevezzük.
- A három alapműveletnek megfelelő logikai kapu **grafikai** (kapcsolási rajz) **szimbóluma** a következő:
  - **’+’ művelet neve: VAGY, OR**



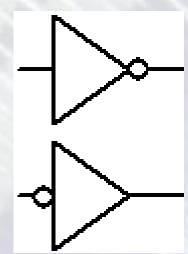
OR kapu

  - **’*’ művelet neve: ÉS, AND**



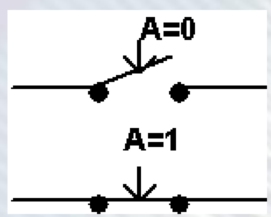
AND kapu

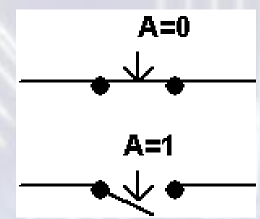
  - **’¯’ művelet neve: negálás INVERTÁLÁS (Önmagában a kör is invertálást jelöl.)**

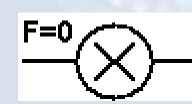


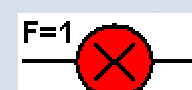
INVERTER

Boole algebra

- **Kapcsolás algebrával** a működés könnyebben érthető. (*Csak a megértés segítésére használjuk. ZH-n és vizsgán kapcsolás algebrai rajzokat nem kérünk vissza*.)
- Kapcsolás algebrában a kapcsolók a logikai áramkör bemenetei.
- A kapcsolókhoz logikai változókat rendelünk.
- Egy kapcsolóhoz rendelt logikai változó 0 értékű, ha a kapcsoló nincs lenyomva, 1 értékű, ha le van nyomva.
- Normál kapcsoló: 

Invertáló kapcsoló: 

- A kapcsolás kimenetét egy lámpa jelképezi.
A kimenet 0, ha a lámpa nem világít (nem folyik áram): 

A kimenet 1, ha világít (folyik áram): 

Boole algebra

ÉS (AND)

Hagyományos leírás: **F = A * B** (Verilog: assign F=A & B;)

A * jelet nem kötelező kitenni. (pl: F=AB)

Kapcsolás algebrai ÉS:

Sorosan kapcsolt kapcsolók. áram csak akkor folyik, ha **minden** kapcsoló le van nyomva.

ÉS (AND) kapu:

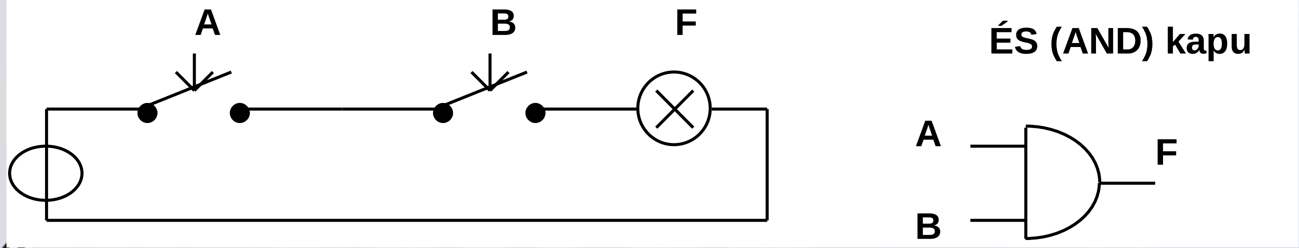
A kapu kimenete csak akkor ad a logikai 1-et **minden bemenetén logikai 1** van. (Valós áramkör esetén a fenti logikai jeleknek megfelelő feszültségszint.)

Tetszőleges számú változó AND kapcsolata **csak akkor 1, ha mindegyik 1.**

F = A*B*C*…= 1, ha A,B,C... = 1

Tetszőleges számú változó AND kapcsolata **0, ha bármelyik 0.**

Feszültségfor-rás

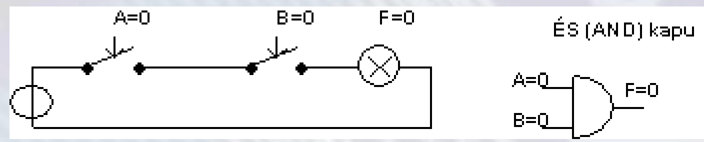


Boole algebra

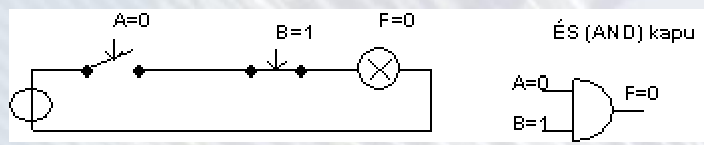
ÉS (AND) művelet eredménye 2 változóval táblázatosan (igazság tábla):

F = A*B

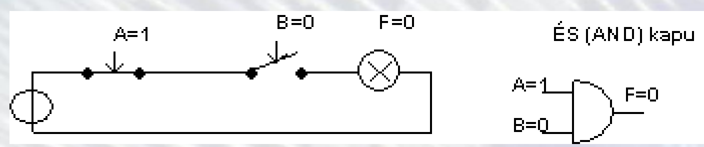
A	B	F
0	0	0
0	1	0
1	0	0
1	1	1



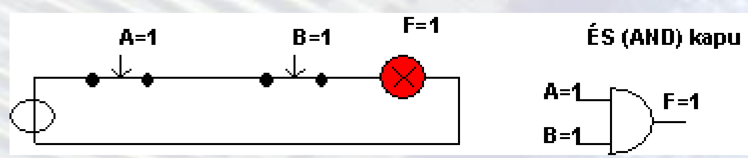
0 * 0 = 0



0 * 1 = 0



1 * 0 = 0

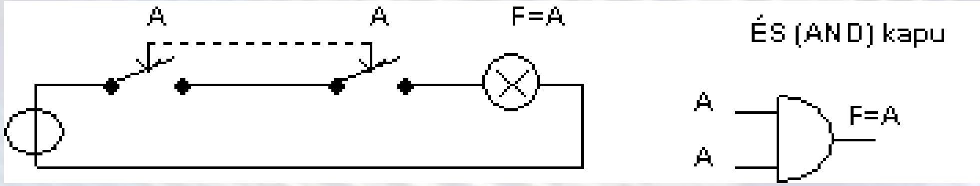


1 * 1 = 1

Boole algebra

A logikai szorzás (AND) további tulajdonságai és magyarázatuk kapcsolás algebrával. (Ezek egyben tételek is):

- **A*0 = 0** (Ha az A-val sorba kötött kapcsoló mindig nyitva, akkor nem világít a lámpa.)
- **A*1=A** (Ha az A-val sorba kötött kapcsoló mindig zárva, akkor az olyan, mintha ott sem lenne.)
- **A*A=A** (Egyszerre működtetett sorba kötött kapcsolók úgy működnek mint egyetlen egy.)



- **A*B = B*A** kommutativitás, felcserélhetőség (A sorba kötés sorrendje lényegtelen.)

Boole algebra

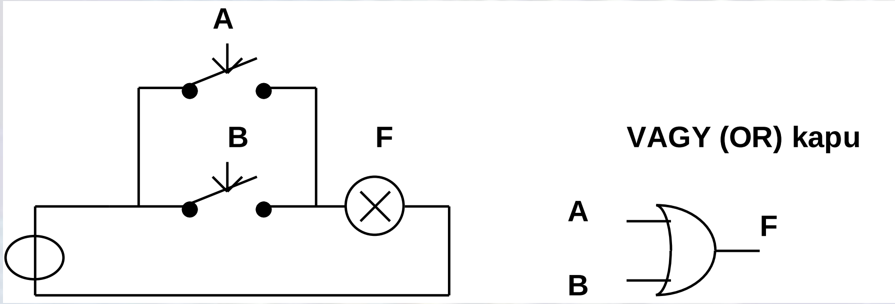
VAGY (OR)

Hagyományos leírás: **F = A + B** (Verilog: assign F=A | B;)

Kapcsolás algebrai VAGY:
Párhuzamosan kapcsolt kapcsolók. Áram akkor folyik, ha **bármely** kapcsoló le van nyomva.

VAGY (OR) kapu:
A kapu kimenete logikai 1-et ad, ha **bármely bemenetén logikai 1** van. (Valós áramkör esetén a fenti logikai jeleknek megfelelő feszültségszint.)

Tetszőleges számú változó OR kapcsolata **1-ed ad, ha bármelyik változó 1 értékű**. Tetszőleges számú változó OR kapcsolata **csak akkor 0, ha minden változó 0**. F = A+B+C+…= 0, ha A,B,C... = 0

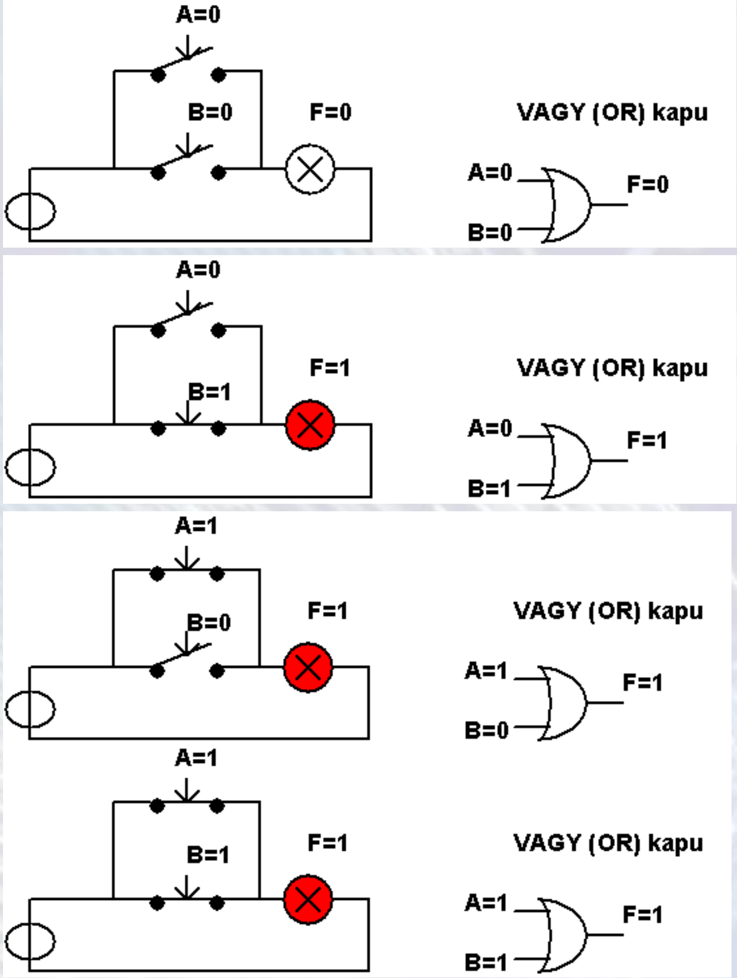


Boole algebra

VAGY (OR) művelet eredménye 2 változóval táblázatosan

(igazság tábla):

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



0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 1

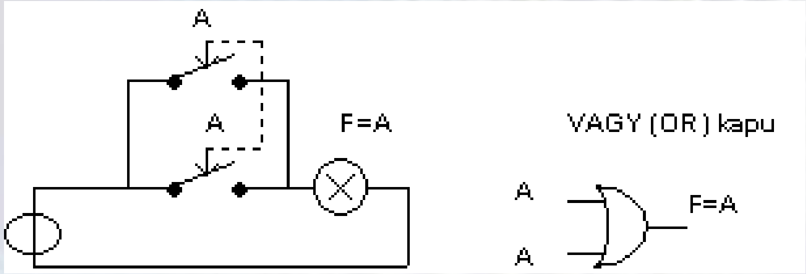
Boole algebra

A logikai összeadás (VAGY) további tulajdonságai és magyarázatuk kapcsolás algebrával. (Ezek egyben tételek is):

A + 0 = A (Ha az A kapcsolóval egy szakadást kötünk párhuzamosan, az nem befolyásolja a kapcsoló működését.)

A + 1 = 1 (Ha az A kapcsolóval rövidzárat kötünk párhuzamosan, a lámpa mindig világít, A értékétől függetlenül.)

A+A = A (tetszőleges számú egyszerre vezérlet párhuzamosan kötött kapcsoló ugyanúgy viselkedik, mint egyetlen egy.)



A+B = B+A kommutativitás, felcserélhetőség (A párhuzamosan kötés sorrendje lényegtelen.)

Boole algebra

Invertálás:

Hagyományos leírás: $F = \overline{A}$ (Verilog: assign F =~A)

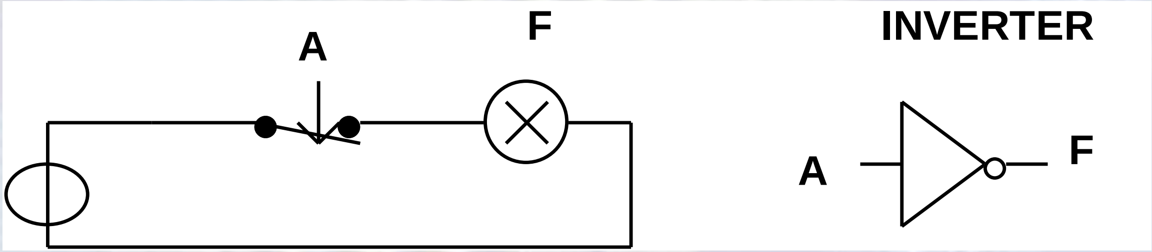
Igazságtábla: F = /A

A	F
0	1
1	0

Kapcsolás algebra:

A = 1, ha ***A*** kapcsoló le van nyomva. ***A kapcsoló itt lenyomott esetben (A=1) kikapcsol***. Áram folyik, ha a kapcsoló **nincs** lenyomva.

Logikai hálózat:



Boole algebra

A logikai invertálás (INVERTER) további tulajdonságai és magyarázatuk kapcsolás algebrával vagy egyéb módon. (Ezek egyben tételek is):

A * /A = 0 (Ha ugyanahhoz a változóhoz rendelt 2 sorba kötött kapcsoló közül az egyik invertáló, akkor az egyikük biztosan megszakítja az áram útját, mivel egyszerre működnek.)

A + /A = 1 (Ha ugyanahhoz a változóhoz rendelt 2 párhuzamosan kötött kapcsoló közül az egyik invertáló, akkor az egyikük biztosan zárja az áram útját, mivel egyszerre működnek.)

A+A = A (tetszőleges számú egyszerre vezérlet párhuzamosan kötött kapcsoló ugyanúgy viselkedik, mint egyetlen egy.)

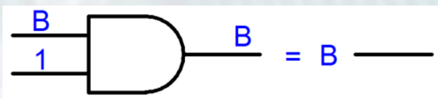
//A = A, ////A=A, ... Páros számúszor negálva egy változót, magát a változót kapjuk eredményül.

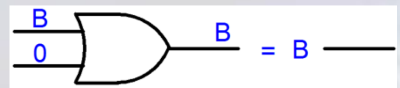
/A=///A=/////A ... Páratlan számúszor negálva egy változót, a negált változót kapjuk eredményül.

Boole algebra

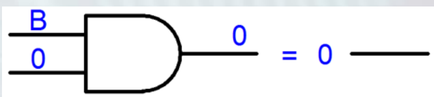
Egyetlen változóra vonatkozó **tételek** és duálisuk kapukkal lerajzolva (a tételek ismeretét elvárjuk):

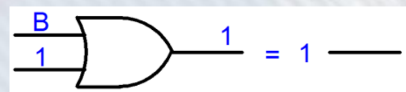
- T1: **B** * 1 = B T1d: B + 0 = **B**



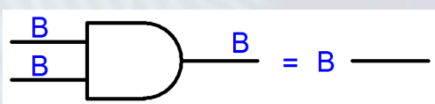


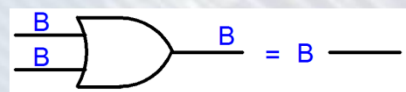
- T2: B * **0** = **0** T2d: B + **1** = **1**



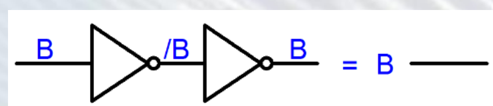


- T3: **B** * **B** = **B** **szokatlan!** T3d: **B** + **B** = **B**

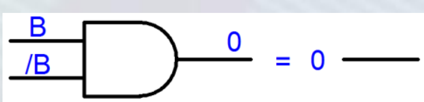


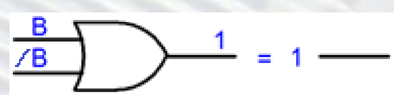


- T4: $\overline{\overline{B}}$ = B



- T5: B * $\overline{B}$ = 0 T5d: B + $\overline{B}$ = 1





Boole algebra

- **Két (vagy több) változóra vonatkozó tételek**
- **Az alábbi tételeket nem bizonyítjuk, de elvárjuk az ismeretüket.**

A bal oldali tételeket megjegyezve, a duálisuk (jobb oldali tételek) a dualitás tétele alapján könnyen adódik.

- **Kommutativitás** (felcserélhetőség) már bemutattuk
  T6: B*C=C*B T6d: B+C=C+B
- **Asszociativitás** (csoportosíthatóság). A hagyományos algebrában is hasonló.
  T7: (B*C)*D=B*(C*D) T7d: (B+C)+D=B+(C+D)
- **Disztributivitás** (szétválaszthatóság)
  T8: B*(C+D) =(B*C)+(B*D) T8d: B+(C*D) =(B+C)*(B+D)
  (T8 a hagyományos algebrában is hasonló.)

Boole algebra

- **Elnyelés**
  T9: B*(B+C)=B
  T9d: B+(B*D)=B
- **Összevonás** (egyszerűsítés)
  T10: (B*C)+(B* $\overline{C}$)=B
  Algebrai átalakítások a baloldalon:
  B kiemelése: B*(C+/C)
  C+/C =1, tehát B marad.

  T10d: (B+C)*(B+ $\overline{C}$)=B
  Algebrai átalakítások a baloldalon:
  beszorzás: B+B*/C+C*B+C*/C
  C*/C = 0, tehát B+B*/C+C*B marad,
  elnyelés után B+B*/C+C*B=B

Az ***összevonás*** tetszőleges kifejezés (KIF) esetén is használható:
KIF*A + KIF*/A = KIF* (A + /A) = KIF* 1 = KIF

- **De-Morgan tétel**

T11 2 változóra: $\overline{A * B} = \overline{A} + \overline{B}$ T11d: $\overline{A + B} = \overline{A} * \overline{B}$

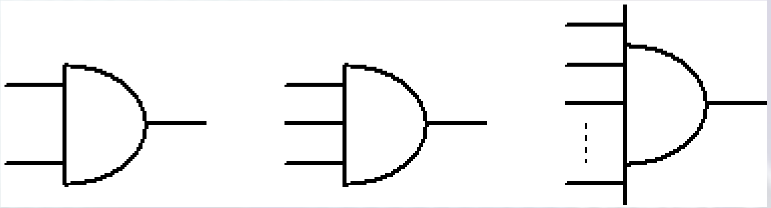
Tetszőleges változószám esetén: $\overline{B_0 * B_1 * B_2 \dots} = \overline{B_0} + \overline{B_1} + \overline{B_2} \dots$

Duálisa: $\overline{B_0 + B_1 + B_2 \dots} = \overline{B_0} * \overline{B_1} * \overline{B_2} \dots$

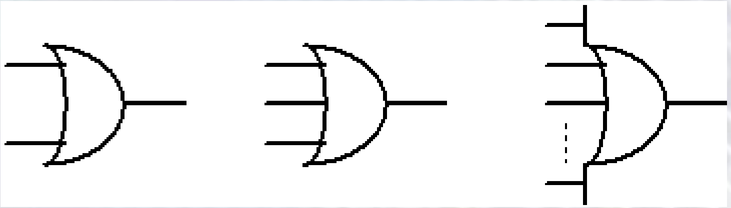
Boole algebra

Eddig csak 2 bementű kapukat rajzoltunk, azonban a kapuk bemenet száma tetszőleges lehet.

Pl: AND kapu

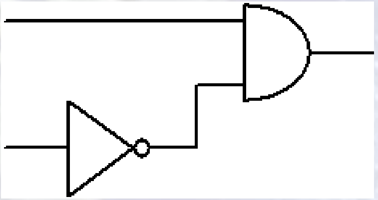


Pl: OR kapu

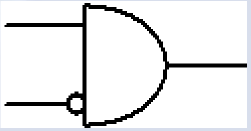


A rajzjelek hasonlóan módosulnak a többi kapu esetén is (NAND, NOR, XOR, XNOR)

Ha egy kapu bemenetére invertált jel kerül, azt az inverter helyett egy kis körrel jelölhetjük a kapu bemeneténél.



→

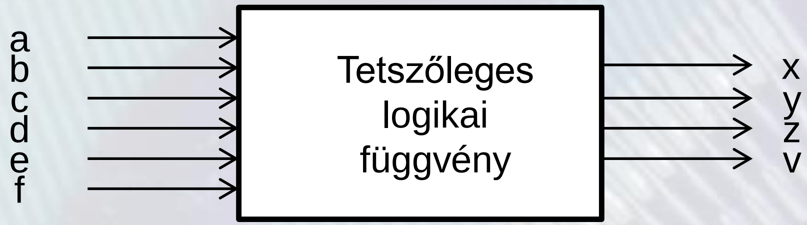


Logikai függvények

- **A logikai függvények: Boole változók között a Boole algebra szabályai szerinti értelmezett *leképezések*.**
  pl. Y = f(A,B,C) pl. Y = A + B*C + /A*/B*C + A*C
- Logikai függvények leírásakor előforduló **fontosabb kifejezések, elnevezések**
  - **Változók:** Az elsődleges logikai változók pl. A,B,C
  - **Literálok:** A fenti változók előfordulásai ***normál*** (ponált) ***vagy negált*** értelemben, azaz pl. A, /A, B, /B, C
  - **Szorzat (Product Term, PT):** Önálló literálok és **ÉS** kapcsolatú kifejezéseik, azaz A, /B, B*C, /A*/B*C, A*C
  - **Minterm:** Olyan ***szorzat***, amelyben ***minden változó szerepel, ponált vagy negált értelemben***
  - **Szorzatösszeg (Sum-of-Products, SOP):** A kifejezések azon formája, ami ***szorzatok VAGY kapcsolatából*** áll.

Logikai függvények

- A **logikai függvények** a bemeneti változók értékének minden lehetséges kombinációjához a kimeneti változók 0 vagy 1 értékét rendelik hozzá.



- A logikai függvényeket **kombinációs hálózatokkal** (pl. logikai kapuk kapcsolásaival) **realizálhatjuk**.

- **A logikai függvények értéke és így az azokat megvalósító kombinációs hálózatok minden kimenete csak a bemeneti változók aktuális értékétől függ.**

Logikai függvények

- **Logikai függvényeket megadhatjuk a következő formákban (**nem soroljuk fel mindet**):**
  - **Szöveges specifikációval**. (Később mutatjuk.)
  - Táblázatos formában, **igazságtáblával**. (Ahogy már az ÉS VAGY és INVERTER-nél mutattuk.)
  - **Általános algebrai alakban** (tetszőleges Boole algebrai kifejezéssel)
    (Pl. F = AB + /(AC + B/C)
    A szorzás jele 2 változó között elhagyható.
  - **Szorzat kifejezések összegeként** vagyis **SOP** (Sum Of Product) alakban. (Pl. F = AB + /BC + /ACD)
  - **Diszjunktív normál alakban** (**DNF**). (Később mutatjuk.)
  - **A specifikációk egymásba alakíthatók**.

Logikai függvények

A logikai függvények bonyolultsága erősen nő a bemeneti változók függvényében.

Az összes 2 bemenetű logikai függvény:

- 2 bemenet: $2^2 = 4$ bemeneti kombináció: 00,01,10,11, minden függvény értékeit (0,1) 2^2=4 helyre lehet beírni ezért $2^{2^2} =$ 16 féle 4 változós függvény létezik.
  Az *n* változós függvények száma: 2^{2^n}
- Az összes 2 változós logikai függvény igazságtáblája egyetlen táblázatban megadva:

BEM		KIM															
A	B	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	FA	FB	FC	FD	FE	FF
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
		0	A*B	A*/B	A	/A*B	B	A^B	A+B	/(A+B)	/(A^B)	/B	A+/B	/A	/A+B	/(A*B)	1

Logikai függvények

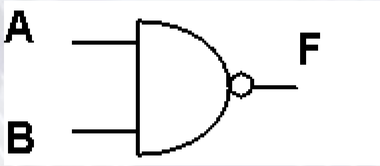
Az előbbi táblázatban a függvények között még van 4 eddig nem említett fontos elemi függvény, amelyhez kapcsolási rajz szimbólumot is rendeltek.

NAND (NEM ÉS)

Az ÉS kapu kimenetének meginvertálásával kapjuk meg.

Boole algebrai alakban: FE = /(A*B)

Kapcsolási rajz szimbóluma:

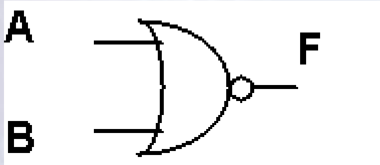


NOR (NEM VAGY)

Az OR kapu kimenetének meginvertálásával kapjuk meg.

Boole algebrai alakban: F8 = /(A+B)

Kapcsolási rajz szimbóluma:



A NOR kapu 1-el jelzi, ha minden bemente 0 értékű.

Logikai függvények

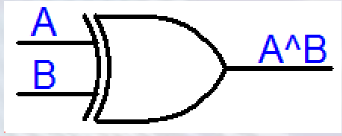
XOR (Kizáró VAGY, exclusive OR)

BEM		KIM
A	B	A^B
0	0	0
0	1	1
1	0	1
1	1	0

/AB

A/B

XOR kapu kapcsolási rajz szimbóluma:



- Az igazságtáblában 2 bemeneti kombinációnál van 1 értéke: AB=01 és AB=10 esetén.
- Az a függvény, ami csak AB=01 esetén AD 1-et, egyetlen ÉS kapcsolattal kifejezhető: /AB
  Hasonlóan, az a függvény, ami csak AB=10 esetén AD 1-et: A/B
- A két függvény VAGY kapcsolata mindkét bemeneti kombinációnál 1-et ad: F7=/AB + A/B

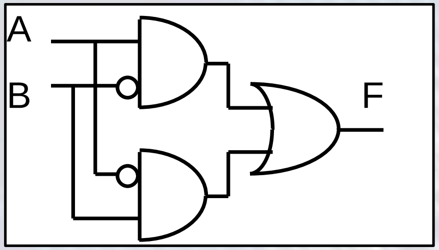
Logikai függvények

XOR Ezt a függvényt most több formában is megadjuk:

F6= /AB+A/B (Ez SOP és DNF alak is egyben).

F6= A $\oplus$ B (külön XOR műveleti jel) (**Verilog** assign F6 = A^B;)

Kapcsolási rajz (ÉS-ek VAGY kapcsolata, DNF, SOP**):**



Szöveges specifikáció:

Egy a változók EXOR kapcsolatával előállított függvény akkor 1 értékű, ha a bemeneti változók között páratlan számú 1-es van. (Az EXOR negáltja pedig akkor, ha a bemeneteken páros számú egyes van.)

Logikai függvények

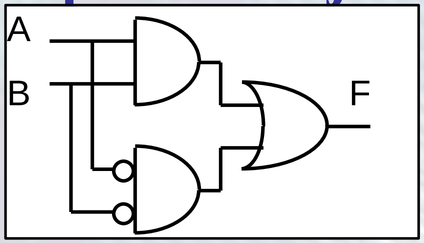
XNOR

Az XOR függvény invertálásával kapott függvény az XNOR.

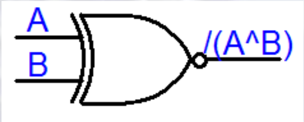
F9 = /(A $\oplus$ B)= AB+/A/B=A $\odot$ B (külön EXNOR műveleti jel)

BEM		KIM
A	B	/(A^B)
0	0	1
0	1	0
1	0	0
1	1	1

Kapcsolási rajz



XNOR kapu szimbóluma:



Logikai függvények megvalósítása kombinációs hálózattal

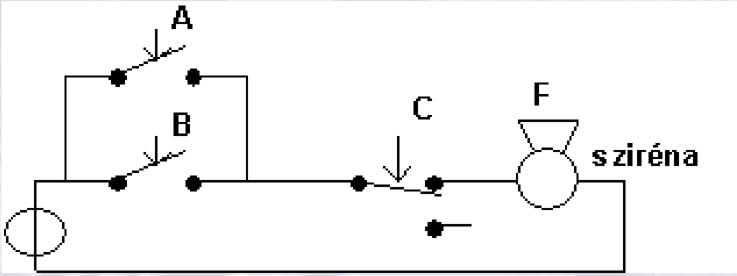
- A továbbiakban egy egyszerű, 3 változós logikai függvényen mutatjuk be a tervezést.

Szöveges specifikáció

- Tervezzünk riasztót. A riasztó akkor jelezzen, ha az ajtó érzékelő (*A*) vagy az ablak érzékelő (*B*) jelez és a riasztást tiltó kapcsoló (*C*) nincs tiltás állapotban. Az *A*, *B* egyszerű kontaktusok (kapcsolók), melyek az ajtó ill. ablak kinyitásakor záródnak. A C 2 állású nyomógomb.

Logikai függvények megvalósítása kombinációs hálózattal

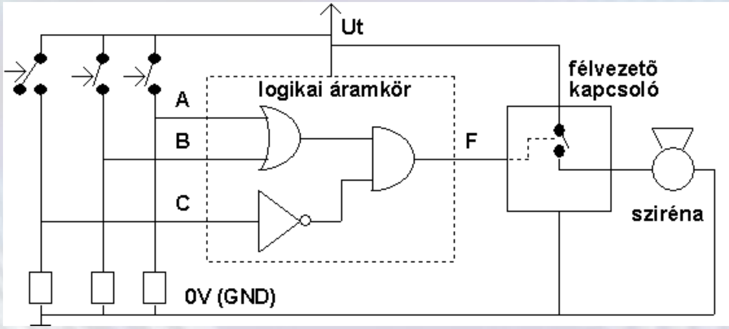
- A riasztó kapcsolási rajza először kapcsolás algebrai megoldással (ilyet többet nem rajzolunk):
- A *C* 2 állású kapcsolónak most olyan kapcsolót használunk, amely, lenyomott állapotban megszakít (invertáló kapcsoló).



- A szirénán folyik áram, ha a *C* risztás tiltó 2 állású nyomógomb le van nyomva (*C*=1).
- Ha a nincs riasztás tiltás (*C* = 0) és az *A* ajtó érzékelő vagy a *B* ablak érzékelő, vagy mindkettő lenyomódik (*A*=1 vagy *B*=1, vagy *A*=*B*=1), akkor megszólal a sziréna.
- A fenti szöveges specifikáció elég egyszerű ahhoz, hogy rögtön a legegyszerűbb Boole algebrai formába öntsük: F = (A + B)/C
- Riasztás van (a sziréna szól), ha az F függvény értéke 1 vagyis, ha C=0 és AB = 10 vagy 01 vagy 11.

Logikai függvények megvalósítása kombinációs hálózattal

- A riasztó kapcsolási rajza a logikai függvény alapján kombinációs hálózattal megvalósítva (F = (A + B)/C):



- A logikai áramkör a 0-nak és 1-nek megfelelő feszültségeket vár. Most ezek előállítását is bemutatjuk, de ilyet nem kérünk ZH-n.
- Ezeket most egy-egy ellenállással sorba kapcsolt kapcsolóval állítjuk elő.
- Ha a kapcsoló nyitott, az ellenálláson nem folyik áram és ezért az ellenállásnak a kapcsolóhoz csatlakozó végén 0V a feszültség, amit a logikai áramkör bemenete 0 logikai értéknek érez.
- Amikor egy kapcsoló zár, akkor az ellenállás és kapcsoló közös pontjára az Ut (tápfeszültség) kerül, ami logikai 1-nek felel meg.
- Ha a logika kimenete 1, akkor a félvezető kapcsoló a tápot rákapcsolja a szirénára. (A logikák kimenete csak kis áramot képes kiadni, ami nem tud meghajtani egy szirnát.)
- A továbbiakban a logikák be és kimeneteire kapcsolódó speciális áramkörökkel, feszültségszintekkel nem foglalkozunk, csak a logikákkal, azok megtervezésével, tulajdonságaival, felhasználásával.

Logikai függvények tervezése, egyszerűsítése

- A logikai függvények hagyományos tervezését a megadott kezdeti specifikációból kiindulva indítjuk. A legtöbb munkát a szöveges specifikáció adja.
- A szöveges specifikáció alapján most kitölthető a riasztó igazsátáblája, mert a riasztó egy egyszerű feladat: Ahol C=1 (tiltott), ott F=0. Ahol C=0 (engedélyezett) és AB=00, ott szintén F=0. Minden egyéb esetben F=1.
- Az igazságtábla **alapján állítsuk elő a függvény diszjunktív normál alakját** (DNF):

A	B	C	F	
0	0	0	0	
0	0	1	0	
0	**1**	**0**	**1**	**/AB/C**
0	1	1	0	
1	**0**	**0**	**1**	**A/B/C**
1	0	1	0	
1	**1**	**0**	**1**	**AB/C**
1	1	1	0	

Készítsünk az ABC változók felhasználásával olyan szorzatokat, amelyek értéke az igazságtábla F kimenetének 1-eseihez tartozó bemeneti kombinációk esetén 1.

ABC = 010 → /A*B*/C

ABC = 100 → A*/B*/C

ABC = 110 → A*B*/C

Szabály: Amely változók a bemeneti kombinációban 1-ek azok a szorzatban ponáltan szerepelnek, a többi pedig negáltan.

Logikai függvények tervezése, egyszerűsítése

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

- Elékezzünk, már szerepelt a fogalom:
  A *minterm* olyan *szorzat*, amelyben *minden változó szerepel, normál vagy negált értelemben.*
- Tehát az előbbi szorzatok az F függvény **mintermjei**.
- Egy függvény mintermjei önmagukban olyan függvények, amelyek igazságáblájában 1 db 1-es van (a hozzá tartozó bemeneti kombinációnál).
- Ezért a teljes függvény megkapjuk, ha a függvény összes mintermjét összeadjuk (VAGY kapcsolatba hozzuk). Az így kapott Boole algebrai alakot nevezik a függvény ***diszjunktív normál alakjának (DNF).*** Ez egyben SOP alak is (mivel szorzatok összege).

$$F_{DNF} = /AB/C + A/B/C + AB/C$$

Logikai függvények tervezése, egyszerűsítése

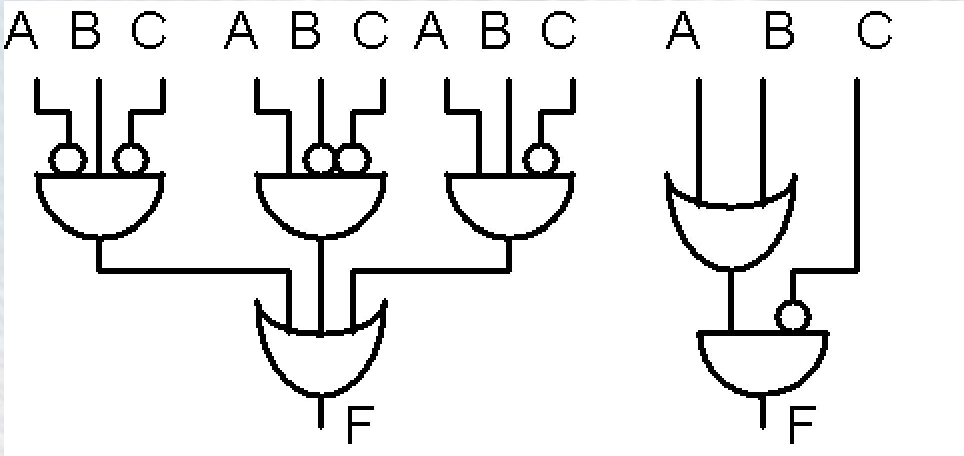
A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	**1**
0	1	1	0
1	0	0	**1**
1	0	1	0
1	1	0	**1**
1	1	1	0

Az F függvény DNF alakját és a legelején a szöveg alapján kitalált alakját összehasonlítva látható, hogy azok bonyolultságban nagyon különböznek:

F_{DNF} = /AB/C + A/B/C + AB/C F = (A + B)/C

A két függvény ugyanaz ($F = F_{DNF}$), hiszen az igazságtáblájuk megegyezik, csak más Boole algebrai alakban írtuk le őket.

Azonban a logikai kapus megvalósításuk alkatrészigénye jelentősen eltér, ahogy az alábbi kapcsolási rajzukból is látszik.



Logikai függvények tervezése, egyszerűsítése

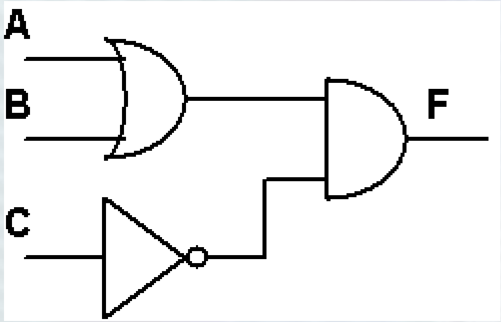
A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	**1**
0	1	1	0
1	0	0	**1**
1	0	1	0
1	1	0	**1**
1	1	1	0

Ezért az igazságtábla alapján felírt DNF alakot egyszerűsíteni (minimalizálni) szokták. Erre megfelelő számítógépes algoritmusok léteznek, melyek többek között a Boole algebra *egyszerűsítési tételét* alkalmazzák. Most kézi módszerrel mutatunk példát (Boole tételek alkalmazása):

/AB/C + A/B/C + AB/C

1. T8, kiemeljük /C-t: /C(/AB+A/B+AB)
2. T3d, bővítünk AB-vel:/C(/AB+AB +A/B+AB)
3. T10, egyszerűsítjük az első két és utolsó két összeget: /C(B + A) = **(A+B)/C**

Megkaptuk az F = (A+B)/C alakot



Az egyszerűsített Boole algebrai alak alapján felrajzoljuk a rendelkezésre álló kapu építőelemek felhasználásával a kapcsolási rajzot. Ezzel a logika megtervezése kész. (Azonban, ahogy az elején bemutattuk, a működő riasztóhoz még egyéb alkatrészek is kellenek...)

Logikai függvények realizációja

- **A specifikációk alapján a tervező több, egymással logikailag ekvivalens, de egyéb paramétereiben jelentősen eltérő megoldás közül választhat.**
- **Ez lehetőséget ad egyedi szempontok szerinti optimalizálásra. Tipikus optimalizálási szempontok:**
  - Legkevesebb alkatrész (jelentése technológia függő)
  - Leggyorsabb működés
  - Előre megadott alkatrészekből építkezés
- **Mindezek a célok a logikai függvények egyszerűsítésével, a redundanciák kihasználásával érhetők el. Ez a logikai függvények tervezésének tárgya.**
- A digitális technika tárgyban egyszerűsítő algoritmusokkal nem foglalkozunk, az egyszerűsítést elvégzik a számítógépes tervező rendszerek (Computer-aided Design, CAD).

1. EA vége