Digitális áramkörök

Analóg és digitális mennyiségek

Analóg mennyiség

Digitális mennyiség

Az analóg mennyiségek változása folyamatos (bármilyen értéket felvehet)

A digitális mennyiségek változása nem folyamatos, hanem ugrásszerű (csak diszkrét értékeket vehet fel)

Számrendszerek

Decimális (tízes) számrendszer

- **A mindennapi életben a tízes vagy decimális számrendszert használjuk.**
- **Már az ókori egyiptomiak is használták, de csak akkor vált teljessé, mikor i.sz. 400 körül a hinduk bevezették a 0 – át és számjegyként használták.**
- **Arab közvetítéssel jutott Európába és először 1202 – ben Fibonacci ismertette.**
- **Általánosságban egy N számot a következő zárt matematikai összefüggés határoz meg:**

$$N = a_n \cdot r^n + a_{n-1} \cdot r^{n-1} + \ldots + a_1 \cdot r^1 + a_0 \cdot r^0 + a_{-1} \cdot r^{-1} + a_{-2} \cdot r^{-2} + \ldots + a_{-m} \cdot r^{-m}$$

a_n – 0 és r-1 közé eső egész szám
r – a számrendszer alapszáma
m, n – egész számok

Bináris (kettes) számrendszer

- **A kettes számrendszer alapja r = 2, vagyis csak két elemet használ a számok ábrázolására (0 és 1).**
- **A számjegyeit biteknek is nevezik. (bit – Binary Digit)**
- **A bináris számrendszer tökéletesen összeegyeztethető a kétállapotú jeleket alkalmazó elektronikus áramkörökkel.**
- **Bináris szám formája:**
- **N = $101011{,}101_{(2)}$**
- **Szimbolikusan ábrázolva:**

$$N = 1\cdot 2^5 + 0\cdot 2^4 + 1\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 + 1\cdot 2^{-1} + 0\cdot 2^{-2} + 1\cdot 2^{-3}$$

- **Az eredmény : $43{,}625_{(10)}$**

Decimális ⇒ Bináris átalakítás

Alakítsuk át a $628_{(10)}$ számot kettes számrendszerbe!

628		: 2
314	0	: 2
157	0	: 2
78	1	: 2
39	0	: 2
19	1	: 2
9	1	: 2
4	1	: 2
2	0	: 2
1	0	: 2
0	1	

Kiolvasás innen felfelé

A kiolvasott bináris szám:

$1001110100_{(2)}$

MSB
Most Significant Bit

LSB
Least Significant Bit

Bináris → Decimális átalakítás

- **Alakítsuk vissza decimális számmá az $1001110100_{(2)}$ bináris számot!**

$$N = 1 \cdot 2^9 + 0 \cdot 2^8 + 0 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 =$$

$$N = 512 + 64 + 32 + 16 + 4 =$$

$$N = \underline{\underline{628_{(10)}}}$$

Jegyezzük meg a 2 hatványait legalább 10 – ig:

2^0,	2^1,	2^2,	2^3,	2^4,	2^5,	2^6,	2^7,	2^8,	2^9,	2^{10}
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
1,	2,	4,	8,	16,	32,	64,	128,	256,	512,	1024

Törtszámok átalakítása

Alakítsuk át a $0{,}625_{(10)}$ törtszámot bináris törtszámmá!

Kiolvasás innen lefelé

0,625		· 2
0,25	1	· 2
0,5	0	· 2
1	1	

A kiolvasott bináris szám:

$$0{,}101_{(2)}$$

$$N = 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3}$$

$$N = 0{,}5 + 0{,}125 = \underline{\underline{0{,}625_{(10)}}}$$

Gyakorló feladatok

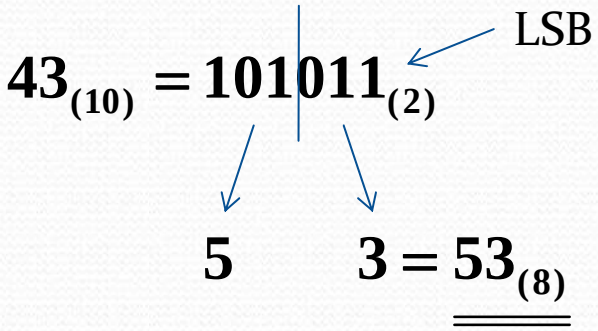
- **Alakítsd át a következő decimális számokat bináris számmá!**
- **255; 315; 455; 500; 1000 egész számok**
- **168,25; 192,6; 153,95 tört számok**

- **Alakítsd át a következő bináris számokat decimális számmá!**

- **1001001; 11100011; 101010101; 11110000; 10000011**
- **1001,101; 111,1111; 1010;001; 1100,010101**

Oktális (nyolcas) számrendszer

- **A számrendszer alapszáma r = 8.**
- **A felhasznált számjegyek 0 – 7 –ig terjednek.**
- **Átalakításakor a szám bináris megfelelőjéből indulunk ki.**
- **Az LSB felől 3 bitnyi csoportokra osztjuk a bináris számot, és egyenként decimális számmá alakítjuk.**
- **Ezek lesznek az oktális szám számjegyei.**



Hexadecimális (tizenhatos) számrendszer

- **A hexadecimális számrendszer elterjedt a digitális technikában főként a mikroszámítógépek világában.**
- **A számrendszer alapszáma r = 16.**
- **Ez annyit jelent, hogy 16 szimbólumot használ egy számjegy ábrázolásához.**
- **Ebből az első tíz a decimális számrendszer 0 – 9 –ig terjedő számjegyei.**
- **A következő hat számjegyet (10 – 15) A, B, C, D, E, F betűszimbólumok jelölik.**
- **Ezért a hexadecimális számrendszert alfanumerikusnak is nevezik.**

Hexadecimális (tizenhatos) számrendszer

- **Átalakítása során hasonlóan a nyolcas számrendszernél megismert módon az adott szám bináris megfelelőjéből indulunk ki.**
- **Az LSB felől 4 bitnyi csoportokra osztjuk a bináris számot, és egyenként decimális számmá alakítjuk.**
- **Ezek lesznek a hexadecimális szám számjegyei.**

$$43_{(10)} = 101011_{(2)}$$

LSB

Kiegészítés az MSB elé írt 0 -kal

$$00101011_{(2)}$$

$$2 \quad 11$$

$$2 \quad B \quad = 2B_{(16)}$$

Kódolás

Az információ kódolása

- **Az INFORMÁCIÓ valamely jelenségre vonatkozó értelmes közlést jelent, általános megfogalmazásban az INFORMÁCIÓ bizonyos fokú tájékozatlanságot szüntet meg.**
- **Az ADAT az információnak a konkrét megjelenési formája.**
- **Az információ szimbólumok sokaságából áll.**
- **Egy szimbólumhalmaz meghatározott rendszere alkotja a KÓDOT, amelyet KÓDSZAVAK alkotnak.**
- **Két szimbólumhalmaz egymáshoz rendelését KÓDOLÁSNAK nevezzük.**
- **A kódokat karakterkészletük szerint két csoportra osztjuk:**

1. **Numerikus kódok (csak számokat tartalmaznak)**
2. **Alfanumerikus kódok (betűket és számokat is tartalmaznak)**

Bináris kódolású számrendszerek

- **BCD – Binary Coded Decimal vagyis binárisan kódolt decimális számot jelent. (8 – 4 – 2 – 1 súlyozású)**
- **Egy decimális szám minden számjegyét külön átalakítjuk bináris számmá.**
- **$359_{(10)} = 0011 \mid 0101 \mid 1001_{(BCD)}$**

- **Háromtöbbletes kód (Excess – 3 , Stibitz – kód)**
- **A decimális számokhoz a náluk hárommal nagyobb szám bináris megfelelőjét rendeljük hozzá.**
- **Sajátossága, hogy minden kódszó tartalmaz legalább egy darab egyest és a kód önkomplementáló.**
- **$359_{(10)} = 0110 \mid 1000 \mid 1100_{(Stibitz)}$**

Bináris kódolású számrendszerek

- **Aiken kód: olyan négybites kódrendszer, ahol az egyes bitek súlyozása 2-4-2-1.**
- **A kódszavak a 0 – 4 tartományban megegyeznek a BCD kódszavakkal.**
- **A súlyozása miatt a legnagyobb szám a decimális 9.**

- **Gray – kód: alapváltozata 4 biten a decimális számjegyeket kódolja úgy, hogy az egymást követő kódszavak csak egy bitben térjenek el egymástól.**
- **Nagyon jól használható időzítési problémákból adódó kódhibák kivédésére.**

Bináris kódolású számrendszerek

- **Johnson – kód: szintén egylépéses kód, ahol a tiszta 0 értékű kód után az LSB felől minden lépésben feltöltődik egyesekkel, majd a tiszta 1 –est tartalmazó kód után nullákkal töltődik és visszatérünk a kiinduló kódhoz.**

Decimális szám	N-BCD	XS-3	Aiken	N-Gray-kód	Johnson-kód
0	0000	0011	0000	0000	0000
1	0001	0100	0001	0001	0001
2	0010	0101	0010	0011	0011
3	0011	0110	0011	0010	0111
4	0100	0111	0100	0110	1111
5	0101	1000	1011	0111	1110
6	0110	1001	1100	0101	1100
7	0111	1010	1101	0100	1000
8	1000	1011	1110	1100	
9	1001	1100	1111	1101	

Hibaellenőrző és hibajavító kódok

- **A digitális adatok átvitele során külső zavarok okozta kódhibák léphetnek fel. Egy sérült kódszó akár egy másik értelmes de eltérő jelentésű kóddá is változhat, ezért nagyon fontos, hogy az ilyen átviteli hibákat fel tudjuk fedezni és esetleg ki is javítani.**
- **Erre csak akkor van lehetőségünk, ha a kódolt hasznos információ mellé kiegészítő ellenőrző információt is beépítünk a kódba. (Redundáns információ)**
- **Az így kialakított kódrendszerek két csoportba sorolhatók:**

1. **Hibaellenőrző kódrendszerek (Error Detecting Code)**
2. **Hiba – javító kódrendszerek (Error Correcting Code)**

- **Valamely kódban felismerhető ill. javítható hibák számának mértéke az un. Hamming – féle „h” távolság.**
- **Két kódszó Hamming távolsága megegyezik a két kód eltérő bitjeinek számával.**

Paritás ellenőrző bit

- **A bináris kód minden egyes kódszavát egy redundanciát létrehozó paritásbittel egészítjük ki.**
- **A paritásbit tehát megnöveli a kódszavak hosszát, így kétszer annyi kódszó áll elő mint amennyi az információ kódolásához kellene. A kódrendszer Hamming távolsága 2.**
- **A paritásbittel való kiegészítés kétféleképpen lehetséges:**

1. **Páros paritás esetén a kódszóban található egyesek darabszámát a paritásbit párosra egészíti ki, vagyis ha a kódszó eredetileg páros számú 1 – est tartalmazott akkor a paritásbit 0, egyébként 1.**
2. **Páratlan paritás alkalmazásakor a kódszóban található egyesek darabszámát a paritásbit páratlanra egészíti ki.**

- **A hiba csak akkor fedezhető fel ha páratlan darabszámú 1 – es változott meg!**

Logikai hálózatok

Logikai algebra

- **George Boole (1815 – 1864) angol matematikus fektette le a logikai törvényszerűségek matematikai nyelven történő leírásának alapjait.**
- **Emiatt a logikai algebrát Boole – algebrának is nevezik.**
- **A Boole – algebrában a logikai igaz (true) állításnak az „1”, a logikai hamis (false) állításnak a „0” érték felel meg. Így a bináris számrendszerrel nagyon könnyen leírhatók a logikai folyamatok.**
- **A logikai algebra változói olyan mennyiségek amelyek csak „0” vagy „1” értéket vehetnek fel. A változókat általában az abc nagy betűivel jelöljük.**
- **Egy változó igaz értékét magával a betűvel, míg hamis értékét a betű fölé húzott vonással jelöljük.**

A – *igaz*

$\overline{A}$ – *hamis*

Logikai függvények

- **A logikai feltételek és hatásukra bekövetkező események közötti kapcsolat matematikailag logikai függvényekkel írhatók le.**

- **A logikai algebra alapfüggvényei a NEM, ÉS, VAGY kapcsolatok.**

  NEM – tagadás jelölése a felülvonás a változó betűjele fölött

  ÉS kapcsolat – jelölése a szorzásjel (·) pl.: A · B

  VAGY kapcsolat – jelölése az összeadás jele (+) pl.: A + B

E három függvényt tartalmazó logikai rendszereket NÉV rendszernek nevezzük!

Logikai függvények

- **Igazságtáblázat – olyan táblázat, amelyben egy logikai függvény a lehetséges összes bemeneti kombinációra adott kimeneti értékét tartalmazza.**
- **A logikai függvények adott bemeneti kombinációra adott válaszát az igazságtáblázatából olvashatjuk ki.**

NEM függvény

- **NOT vagy negáció**
- **Jelölése:** $F = \overline{A}$
- **Igazságtáblázata:**

A	F
0	1
1	0

Logikai függvények

ÉS függvény

- **AND vagy konjukció**
- **Jelölése: F = A · B**
- **Igazságtáblázata:**

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

VAGY függvény

- **OR vagy diszjunkció**
- **Jelölése: F = A + B**
- **Igazságtáblázata:**

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

Logikai függvények

NEM – ÉS függvény

- **NAND függvény**
- **Jelölése: $F = \overline{A \cdot B}$**
- **Igazságtáblázata:**

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

NEM – VAGY függvény

- **NOR függvény**
- **Jelölése: $F = \overline{A + B}$**
- **Igazságtáblázata:**

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

Logikai függvények

Kizáró – VAGY függvény

- **XOR (eXclusive OR) vagy ANTIVALENCIA**
- **Jelölése:** $F = \overline{A} \cdot B + A \cdot \overline{B} = A \oplus B$

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

Megengedő – ÉS függvény

- **EOR vagy EKVIVALENCIA**
- **Jelölése:** $F = \overline{A} \cdot \overline{B} + A \cdot B = \overline{A \oplus B}$

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

Logikai függvények jellemzői

- **Kommutativitás (felcserélhetőség): az ÉS ill VAGY kapcsolatban szereplő változók felcserélhetőségét jelenti.**

  $B + A = A + B$ vagy $A \cdot B = B \cdot A$

- **Asszociativitás (csoportosíthatóság): lehetővé teszi, hogy a függvényben szereplő azonos műveleteket tetszőleges sorrendben végezzük el.**

  $(C + B) + A = C + (B + A)$

- **Disztributivitás (szétválaszthatóság): e szerint mindegy, hogy először a VAGY és azután az ÉS kapcsolatot végezzük el, vagy fordítva.**

  $A \cdot (B + C) = (A \cdot B) + (A \cdot C)$

Logikai algebra alaptételei

1. $A \cdot \overline{A} = 0$
2. $A + \overline{A} = 1$
3. $\overline{\overline{A}} = A$
4. $A + A = A$
5. $A \cdot A = A$
6. $A \cdot 0 = 0$
7. $A + 0 = A$
8. $A \cdot 1 = A$
9. $A + 1 = 1$

De – Morgan azonosságok:

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

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

Logikai függvények szabályos alakja

- **A logikai függvények szabályos (kanonikus) alakjának alkalmazásával egy függvénykapcsolat csak egyféleképpen írható fel.**

Alapfogalmak

- **Term: független változók csoportja, amelyeket azonos logikai kapcsolattal kötünk össze.**
- **Minterm: a változók között ÉS kapcsolat van és minden változó ponált vagy negált alakban csakis egyszer szerepelhet.**
- **Maxterm: a változók között VAGY kapcsolat van és minden változó ponált vagy negált alakban csakis egyszer szerepelhet.**
- **Diszjunktív szabályos (normál) alak: olyan logikai függvény amely mintermek VAGY kapcsolatából áll.**
- **Konjuktív szabályos (normál) alak: olyan logikai függvény amely maxtermek ÉS kapcsolatából áll.**

Logikai függvények egyszerűsítése

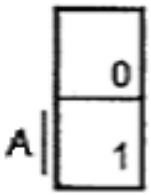
- **A műszaki gyakorlatban alapkövetelmény, hogy a logikai hálózatokat a lehető legkevesebb áramkörrel kell létrehozni.**
- **Ennek érdekében a logikai függvényeket a lehető legegyszerűbb alakra kell hozni.**
- **Az egyszerűsítéshez használhatjuk a már megismert Boole – algebra alaptételeit, ekkor algebrai egyszerűsítésről beszélünk.**
- **Pl.: egyszerűsítsük a következő függvényt:**

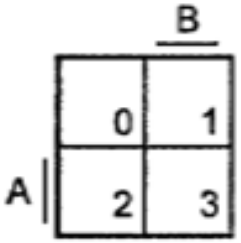
$$F^3 = A \cdot B + A \cdot \overline{C} + \overline{B} \cdot C + A \cdot C$$

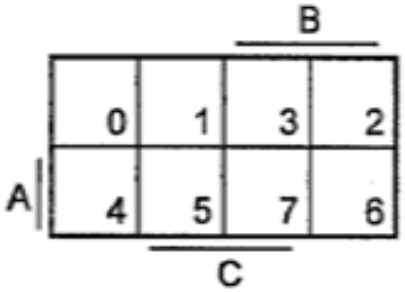
$$F^3 = A \cdot (B + \overline{C}) + C \cdot (\overline{B} + A)$$

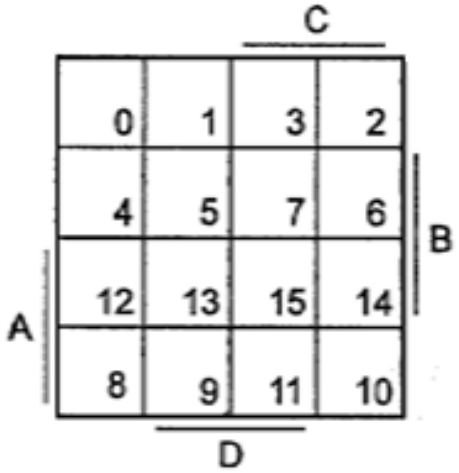
Logikai függvények grafikus egyszerűsítése

MINTERM táblák

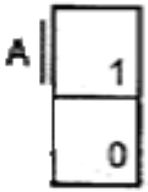


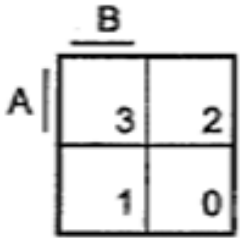


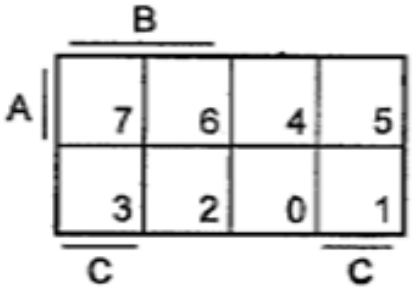


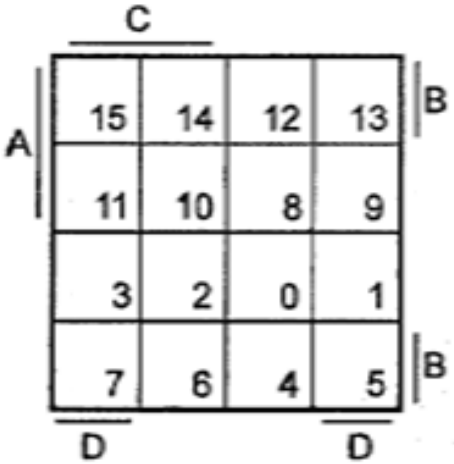


MAXTERM táblák

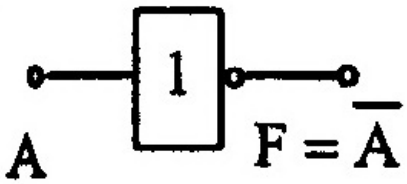




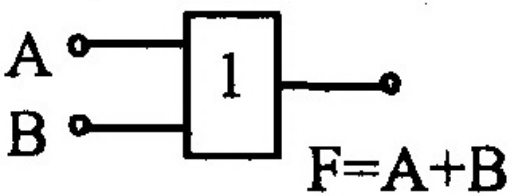




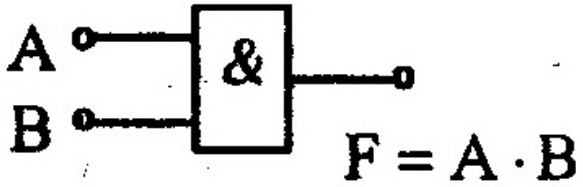
Kapuáramkörök



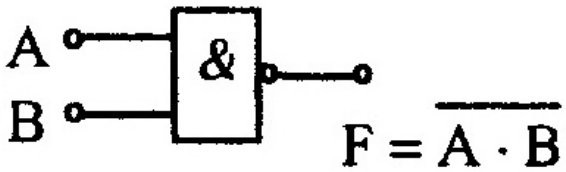
Inverter



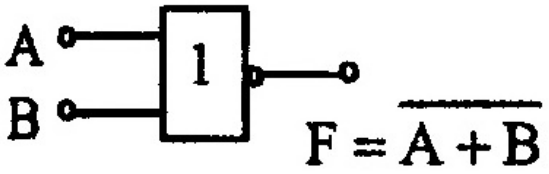
OR kapu



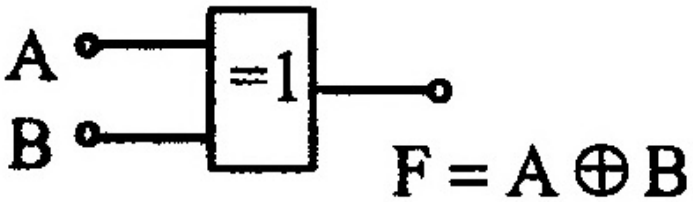
AND kapu



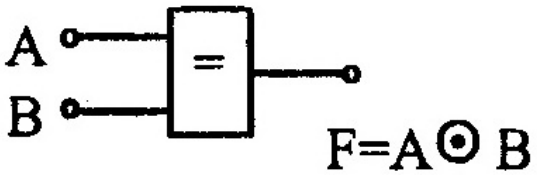
NAND kapu



NOR kapu

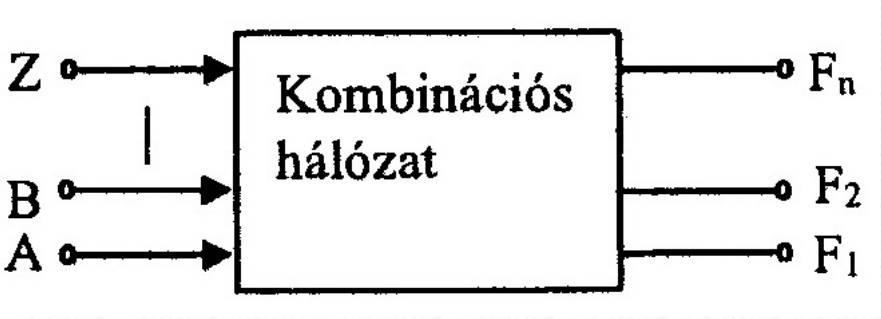


Antivalencia

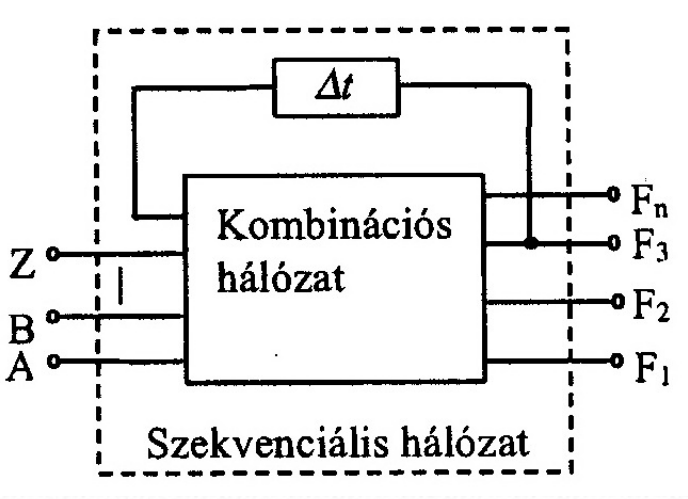


Ekvivalenciakapu

Kombinációs hálózatok tervezése



A kimenetek állapota csak a bemeneti változók értékétől függ!



A kimenetek állapota a bemeneti változók értékén felül a kimeneti változók előző értékeinek is függvénye!

Kombinációs hálózatok megadási módjai

1. **Szöveges megadási mód (működési leírás)**
2. **Táblázatos leírásmód (igazságtáblázat)**
3. **Logikai vázlat (kapuáramkörökkel, szimbólumokkal)**
4. **Algebrai alak (pl.: F =A + B)**
5. **Grafikus megadás (K – V táblában)**

Feladatmegoldás következik!

Funkcionálisan teljes rendszerek

- **Az előzőekben láthattuk, hogy INVERTER, ÉS, VAGY kapuk felhasználásával tetszőleges logikai hálózat megvalósítható.**
- **Ezért a NÉV rendszert funkcionálisan teljes rendszernek nevezzük.**
- **Hátrányos tulajdonsága viszont, hogy szinte minden hálózat háromféle kapuáramkört tartalmaz, ami a gyakorlati (gazdaságossági) szempontokból nem előnyös.**
- **A NÉV rendszeren kívül két funkcionálisan teljes rendszer létezik:**
  - **a) Tisztán NAND kapukból álló NAND RENDSZER**
  - **b) Tisztán NOR kapukból álló NOR RENDSZER**
- **Valójában ez azért lehetséges, mert NAND vagy NOR függvényekkel előállítható a NEM, ÉS, VAGY kapcsolatok mindegyike.**

INVERTER, VAGY, ÉS kapcsolatok

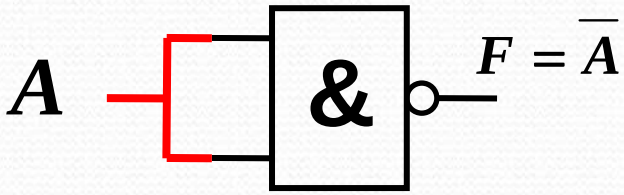
NAND függvény: $F = \overline{A \cdot B}$

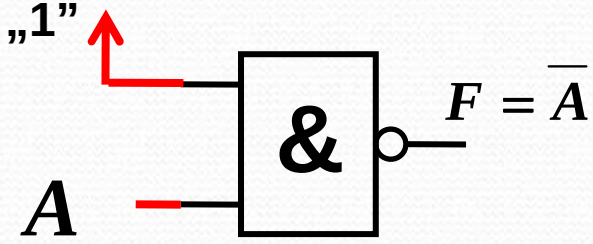
Igazságtáblázata

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

Inverter készítése

$$F = \overline{A}$$



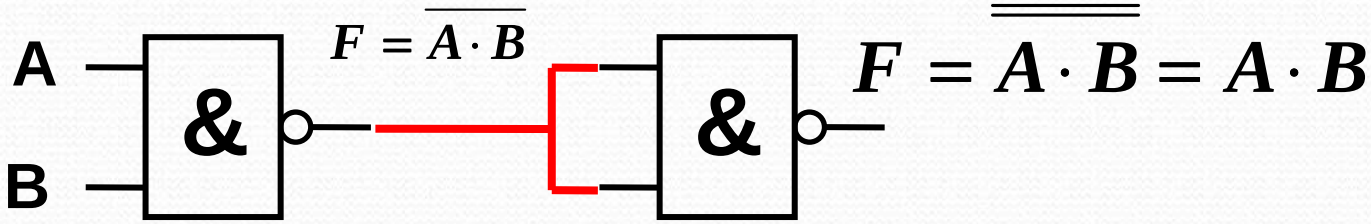


INVERTER, VAGY, ÉS kapcsolatok

ÉS kapcsolat : F = A · B

$NAND: \quad F = \overline{A \cdot B}$ **Negáljuk egyszer**

$F = \overline{\overline{A \cdot B}} = A \cdot B$ *tehát kész az ÉS kapcsolat*

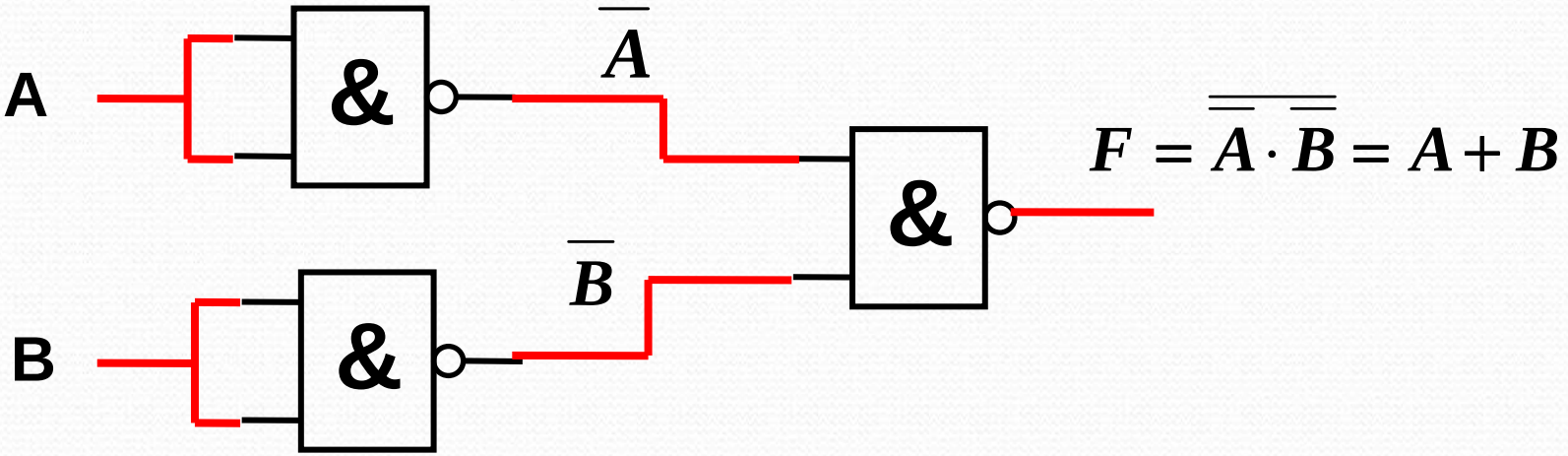


INVERTER, VAGY, ÉS kapcsolatok

VAGY kapcsolat: F = A + B

$F = A + B$ *negáljuk kétszer*

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



NAND hálózat kialakítása

- **A tisztán NAND alakú függvény legkönnyebben a MINTERM KV táblából kiolvasott diszjunktív alakú függvényből alakítható ki.**
- **Az átalakításhoz a függvényt kétszer negáljuk (értéke nem változik) és a De – Morgan azonosságokat alkalmazva a negációt felbontjuk, így kialakul a NAND függvény!**
- **Lássuk a gyakorlatban:**

$$F = A \cdot B + \overline{C} \cdot D \quad \textit{kétszer negáljuk} \quad F = \overline{\overline{A \cdot B + \overline{C} \cdot D}}$$

Felbontjuk az egyik negációt és értelmezzük a tagok közti VAGY kapcsolatra

$$F = \overline{\overline{A \cdot B} \cdot \overline{\overline{C} \cdot D}}$$

(Realizáljuk)

Feladatmegoldás

- **Készítsük el az** $F^3 = \sum^3(0, 1, 3, 6, 7)$ **függvényt NÉV rendszerben és NAND rendszerben is!**

- **A kiolvasott függvény:** $F = \overline{A} \cdot \overline{B} + B \cdot C + A \cdot B$

- **Negáljuk kétszer a NAND alak eléréséhez:**

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

$$F = \overline{\overline{\overline{A} \cdot \overline{B}} \cdot \overline{B \cdot C} \cdot \overline{A \cdot B}} \quad \textit{ez NAND alak!}$$

Feladatmegoldás

Maximális pontszám: 15

Kombinációs hálózat tervezése

Adott a logikai függvény diszjunktív sorszámos alakja:

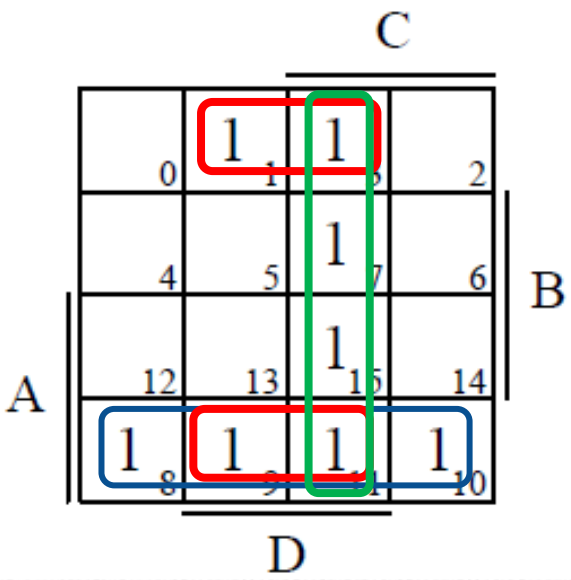
$$F^4 = \Sigma^4(1, 3, 7, 8, 9, 10, 11, 15)$$

Feladatok:

a) Egyszerűsítse a diszjunktív függvényt grafikus módszerrel!
A legnagyobb helyiértékű változót „A”-val jelölje!
b) Valósítsa meg az egyszerűsített függvényt NOT-AND-OR kapukkal! A változók csak ponált alakban állnak rendelkezésre.
c) Valósítsa meg az egyszerűsített függvényt NAND kapukkal! A változók csak ponált alakban állnak rendelkezésre.

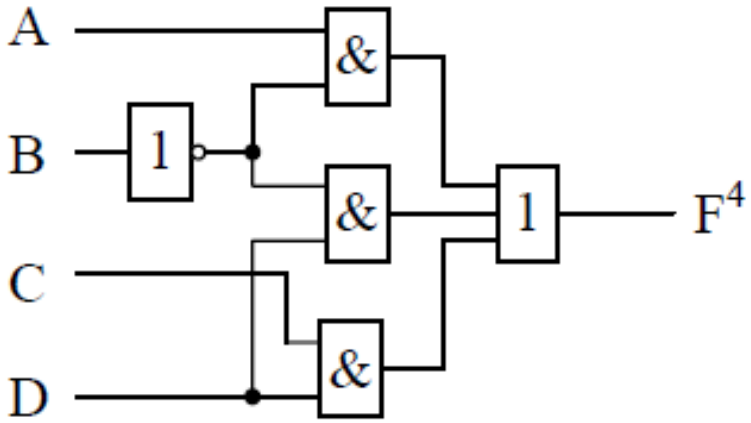
Feladatmegoldás

a)



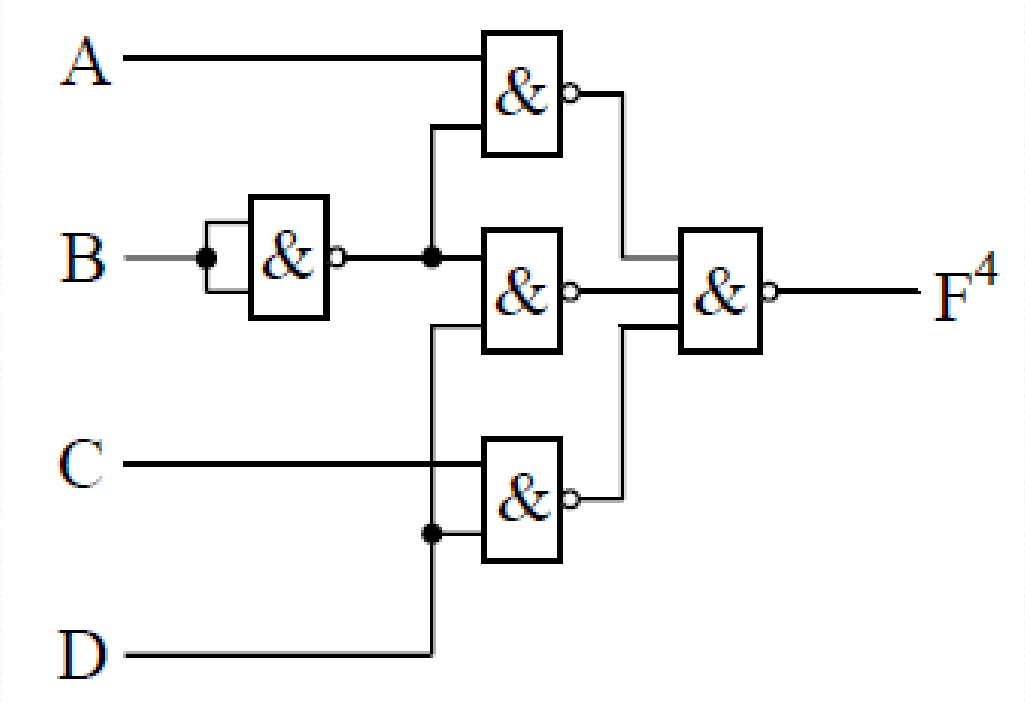
$$F^4 = A \cdot \overline{B} + \overline{B} \cdot D + C \cdot D$$

b)



Feladatmegoldás

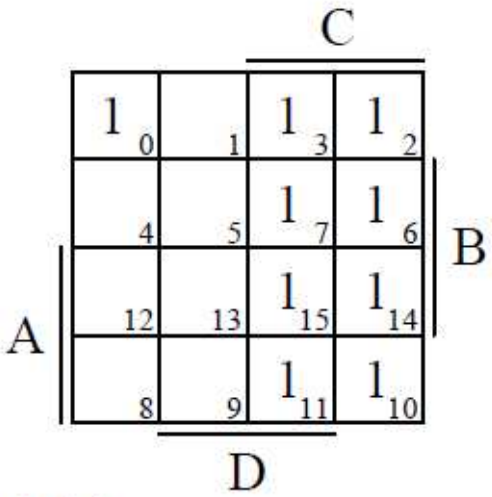
c) $$F^4 = A \cdot \overline{B} + \overline{B} \cdot D + C \cdot D = \overline{\overline{A \cdot \overline{B} + \overline{B} \cdot D + C \cdot D}} = \overline{\overline{A \cdot \overline{B}} \cdot \overline{\overline{B} \cdot D} \cdot \overline{C \cdot D}}$$



Feladatmegoldás

Kombinációs hálózat tervezése

Adott a logikai függvény Veitch-táblája:



Feladatok:

a) Írja fel a függvény sorszámos alakját!

b) Egyszerűsítse a függvényt grafikus módszerrel!

c) Valósítsa meg a függvényt NOT, AND és OR kapukkal!

(A változók csak ponált alakban állnak rendelkezésre.)

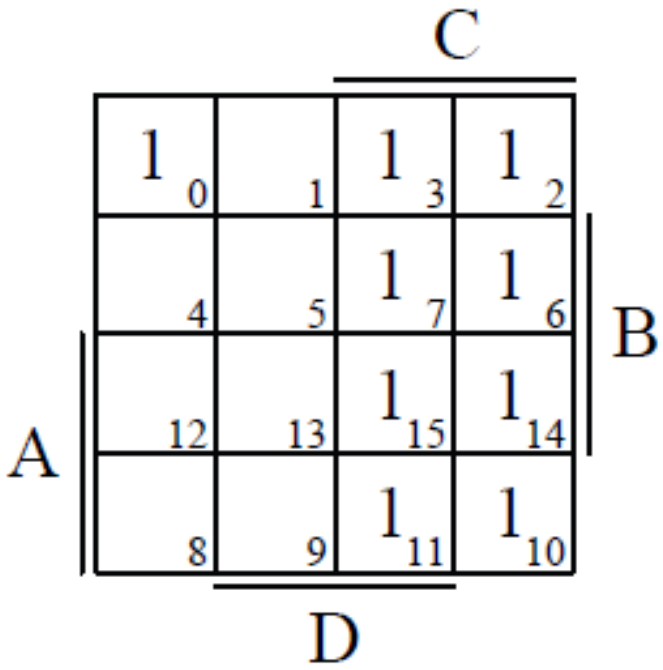
d) Valósítsa meg a függvényt NAND kapukkal!

(A változók csak ponált alakban állnak rendelkezésre.)

Feladatmegoldás

a) $F^4 = \Sigma^4 (0, 2, 3, 6, 7, 10, 11, 14, 15)$

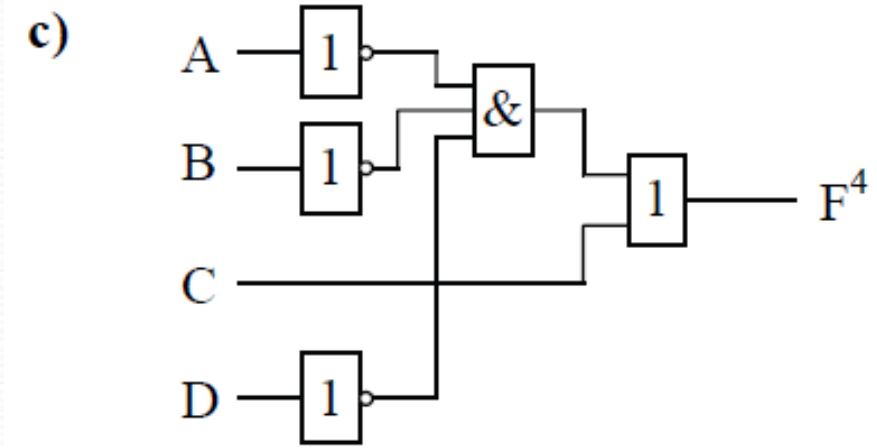
b)



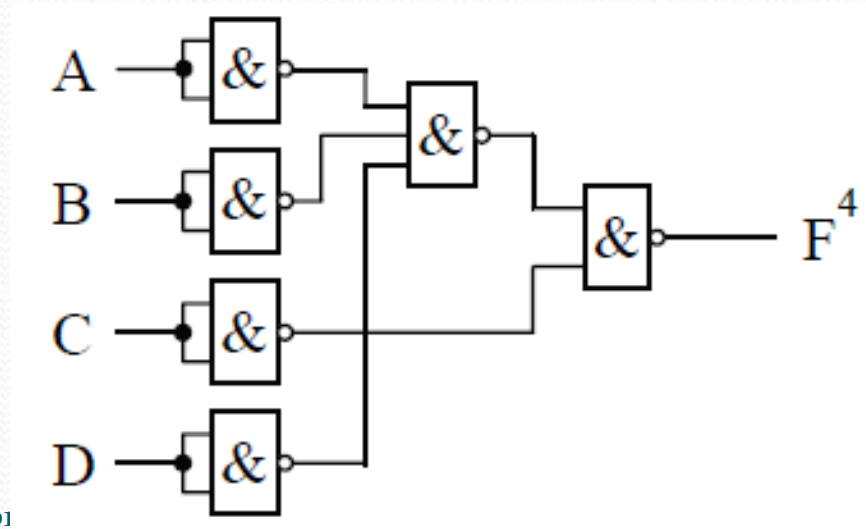
$F^4 = C + \overline{A} \cdot \overline{B} \cdot \overline{D}$

Feladatmegoldás

c)



d) $$F^4 = C + \overline{A} \cdot \overline{B} \cdot \overline{D} = \overline{\overline{C + \overline{A} \cdot \overline{B} \cdot \overline{D}}} = \overline{\overline{C} \cdot \overline{\overline{A}} \cdot \overline{\overline{B}} \cdot \overline{\overline{D}}}$$



Feladatmegoldás

Kombinációs hálózat tervezése

Adott a logikai függvény igazságtáblázata:

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

Feladatok:

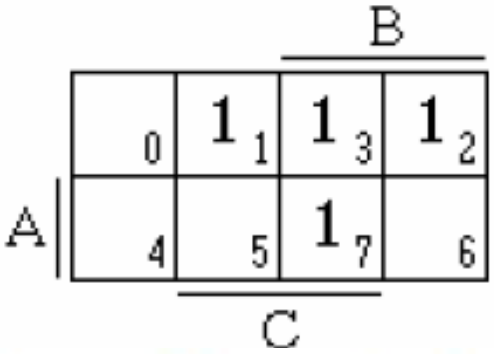
a) Adja meg a függvény algebrai alakját!
b) Egyszerűsítse a függvényt!
c) Valósítsa meg a függvényt NOT, AND és OR kapukkal! (A változók csak ponált alakban állnak rendelkezésre.)
d) Valósítsa meg a függvényt NAND kapukkal!

A legnagyobb helyiértékű változót „A”-val jelöltük.

Feladatmegoldás

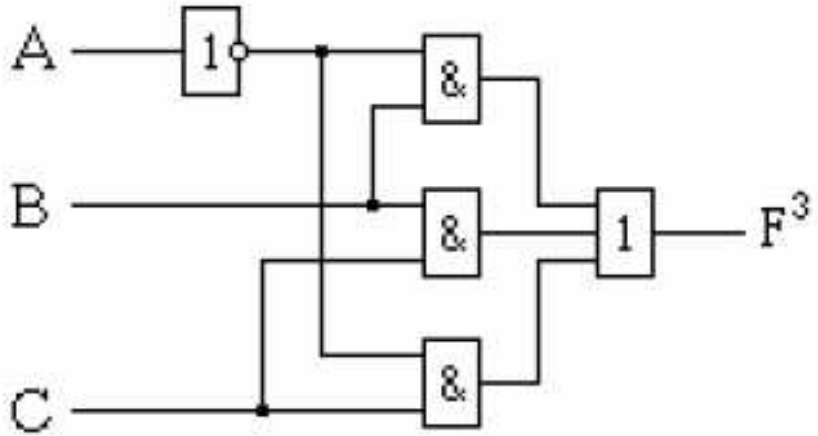
a) $$F^3 = \overline{A} \cdot \overline{B} \cdot C + \overline{A} \cdot B \cdot \overline{C} + \overline{A} \cdot B \cdot C + A \cdot B \cdot C$$

b)



$$F^3 = \overline{A} \cdot B + \overline{A} \cdot C + B \cdot C$$

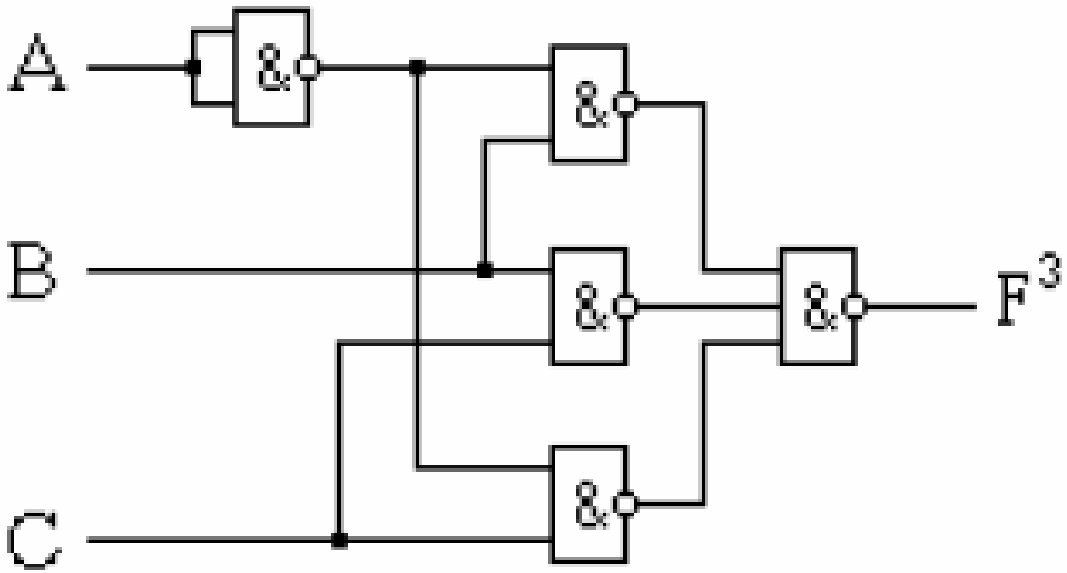
c) Megvalósítás NOT–AND–OR kapukkal:



Feladatmegoldás

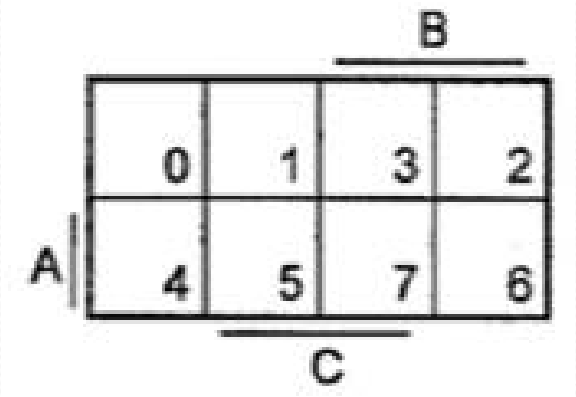
d) Megvalósítás NAND kapukkal:

$$F^3 = \overline{A} \cdot B + \overline{A} \cdot C + B \cdot C = \overline{\overline{A} \cdot B + \overline{A} \cdot C + B \cdot C} = \overline{\overline{A} \cdot B} \cdot \overline{\overline{A} \cdot C} \cdot \overline{B \cdot C}$$

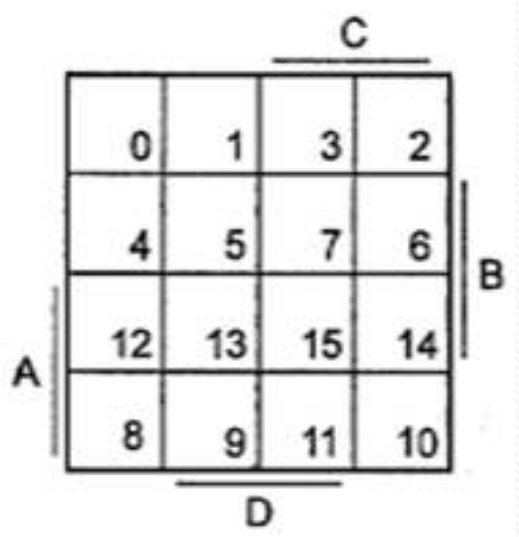


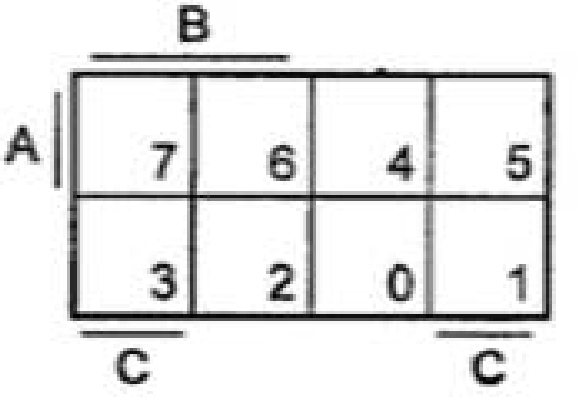
MAXTERM tábla és a konjuktív alak

A MAXTERM tábla a MINTERM tábla inverze (negáltja)!

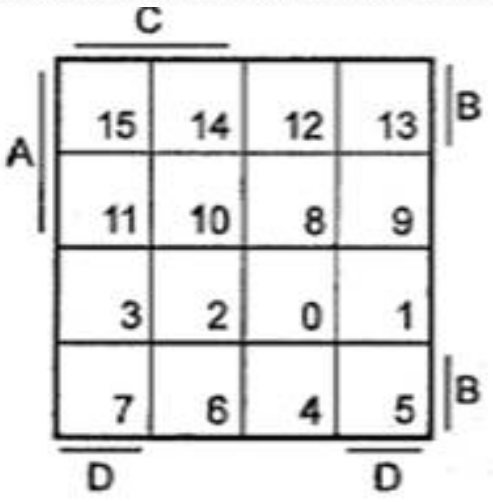


MINTERM



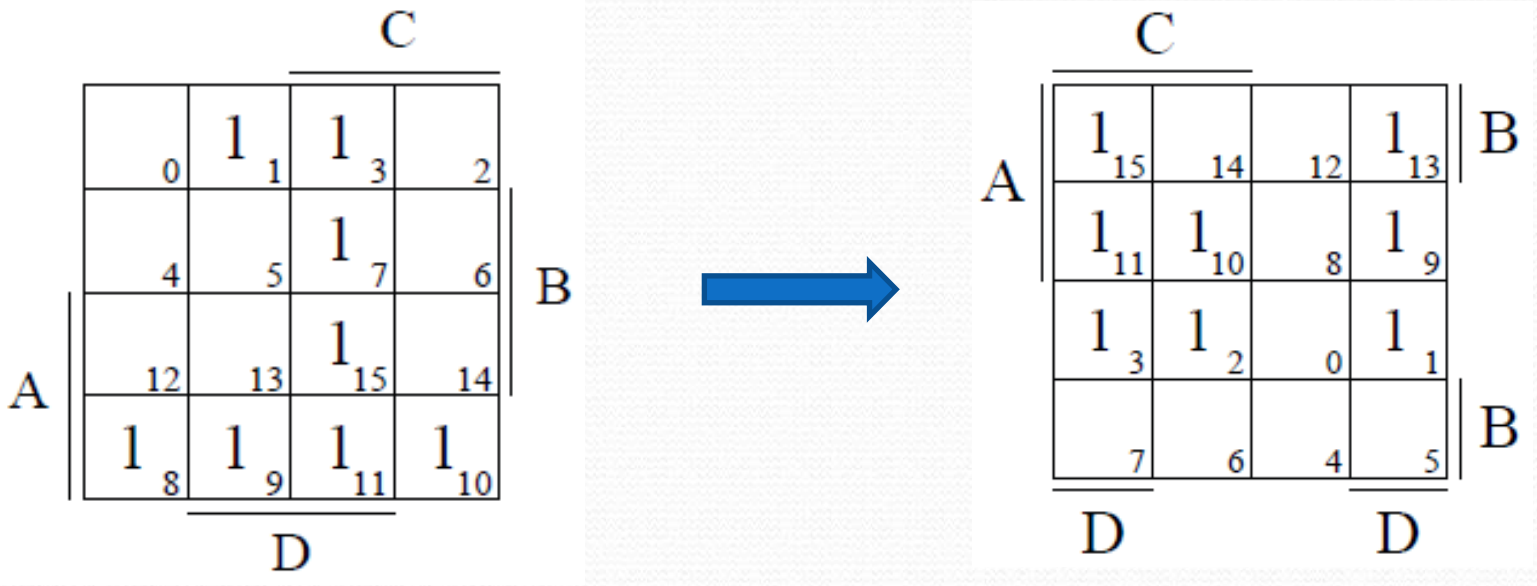


MAXTERM



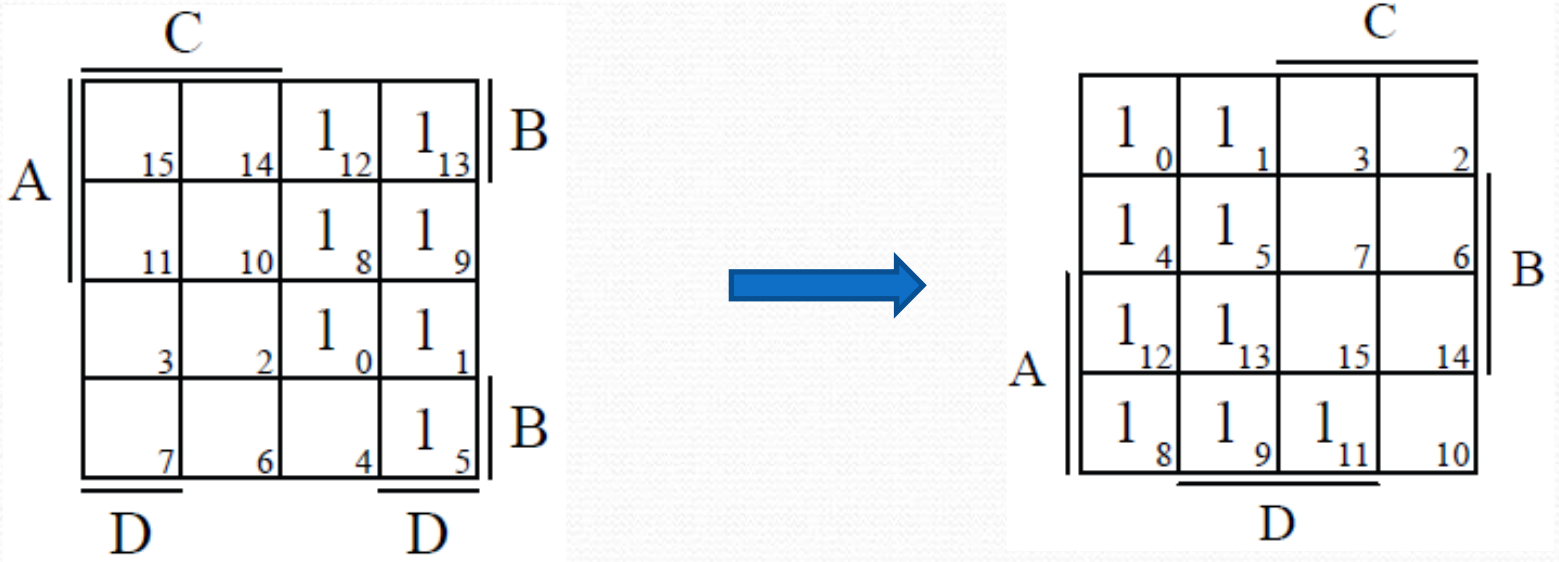
Minterm → Maxterm átalakítás

- **Az átalakítás fő szabálya, hogy a MINTERM táblában található egyesek („1”) pozíciójába a MAXTERM táblában nullákat („0”) írunk, és természetesen a nullák helyére egyeseket.**
- **Az így kapott MAXTERM K – V táblából a függvény konjuktív alakját olvashatjuk ki.**
- **Konjuktív alak: a változók között VAGY kapcsolat, TERMek között ÉS kapcsolat van.**



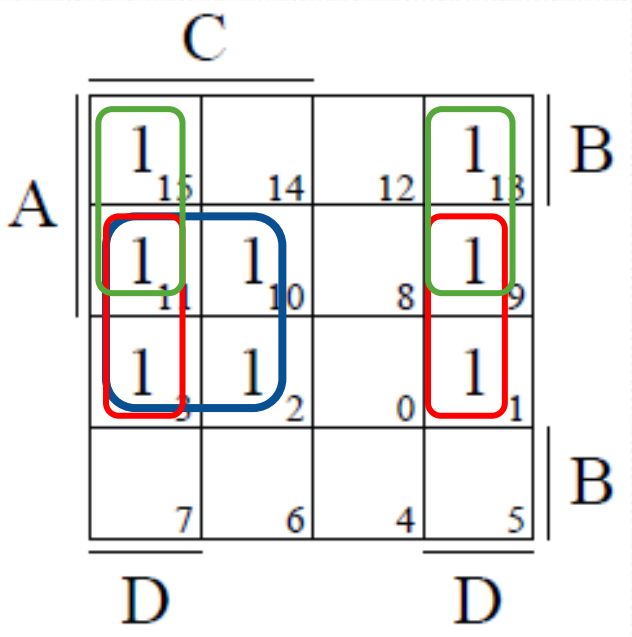
Maxterm → Minterm átalakítás

- **Az átalakítás fő szabálya, hogy a MAXTERM táblában található egyesek („1”) pozíciójába a MINTERM táblában nullákat („0”) írunk, és természetesen a nullák helyére egyeseket.**
- **Az így kapott MINTERM K – V táblából a függvény diszjunktív alakját olvashatjuk ki.**
- **Diszjunktív alak: a változók között ÉS kapcsolat, TERMek között VAGY kapcsolat van.**



Függvény kiolvasás MAXTERM táblából

Olvassuk ki az alábbi MAXTERM táblából az egyszerűsített konjuktív alakú függvényt!



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

MAXTERM sorszámos alak

- **A MAXTERM sorszámos alak a MINTERM sorszámos alakhoz hasonlóan az adott táblázatban található egyesek („1”) celláinak sorszámát sorolja fel!**
- **Mivel a táblák egymásba könnyen átalakíthatók, ezért egy függvény bármely alakjából eljuthatunk a diszjunktív és a konjuktív egyszerűsített alakhoz egyaránt.**
- **A MAXTERM sorszámos alak formája:**

$$F^4 = \Pi^4(0, 1, 5, 8, 12, 13, 15)$$

Pí

NOR hálózat kialakítása

- **A NOR kapu is univerzális építőelem, hasonlóan a NAND kapuhoz.**
- **Ez azért lehetséges mert a három logikai alapművelet mindegyike kialakítható tisztán NOR kapukból álló hálózatból.**
- **A NOR rendszer tehát funkcionálisan teljes rendszer!**

INVERTER, VAGY, ÉS kapcsolatok

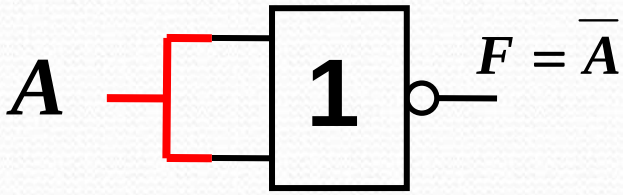
NOR függvény: $F = \overline{A + B}$

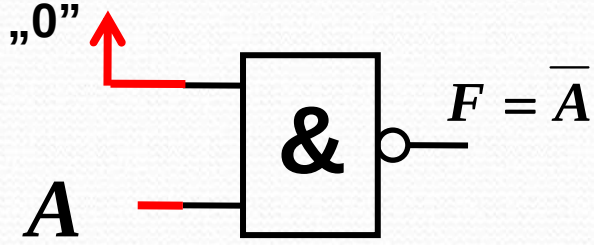
Igazságtáblázata

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

Inverter készítése

$$F = \overline{A}$$



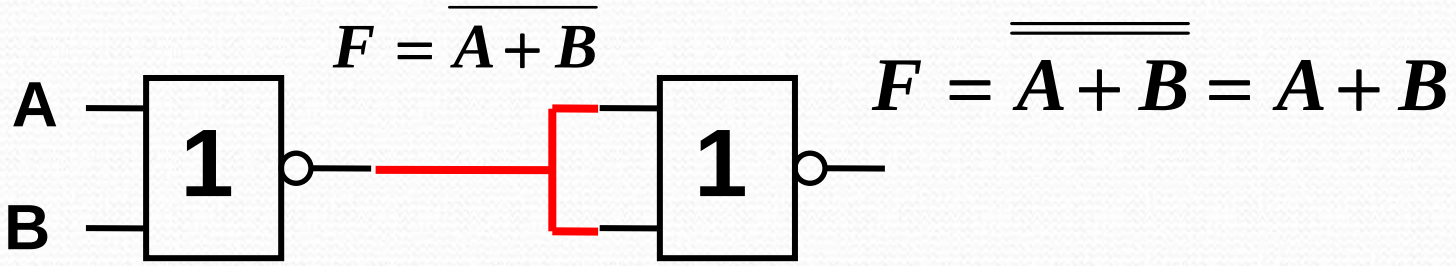


INVERTER, VAGY, ÉS kapcsolatok

VAGY kapcsolat : F = A + B

$NOR: \quad F = \overline{A + B}$ **Negáljuk egyszer**

$F = \overline{\overline{A + B}} = A + B$ *tehát kész a VAGY kapcsolat*



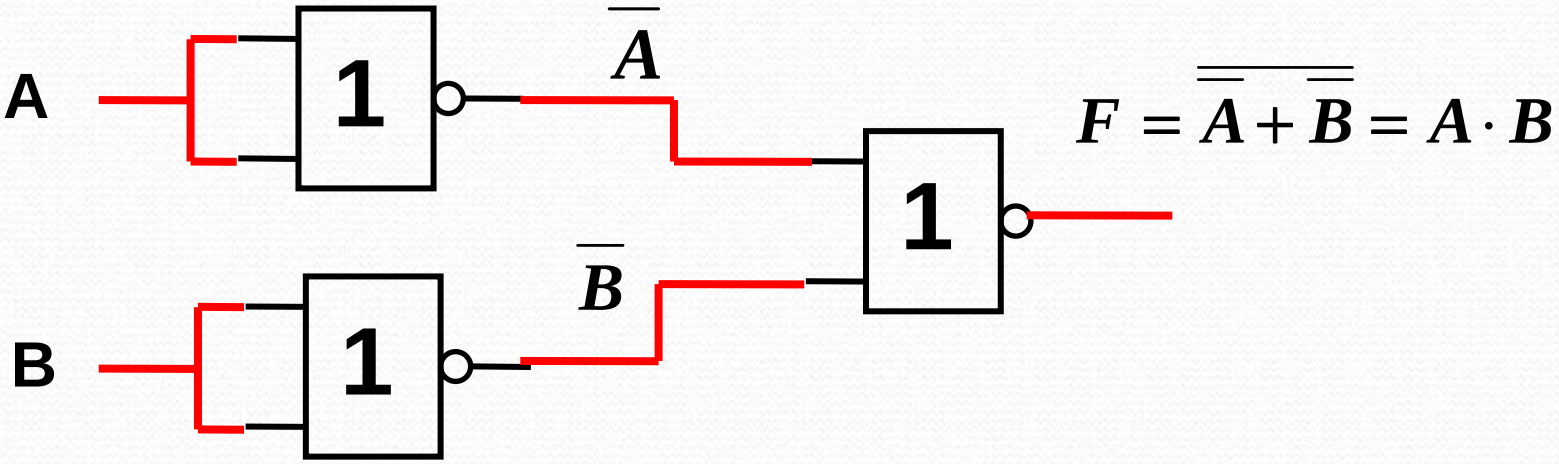
INVERTER, VAGY, ÉS kapcsolatok

ÉS kapcsolat: F = A · B

$$F = A \cdot B \quad \textit{negáljuk kétszer}$$

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

Ez már NOR alak



NOR hálózat kialakítása

- **A tisztán NOR alakú függvény legkönnyebben a MAXTERM KV táblából kiolvasott konjuktív alakú függvényből alakítható ki.**
- **Az átalakításhoz a függvényt kétszer negáljuk (értéke nem változik) és a De – Morgan azonosságokat alkalmazva a negációt felbontjuk, így kialakul a NOR függvény!**
- **Lássuk a gyakorlatban:**

$$F = (A + B) \cdot (\overline{C} + D) \quad \textit{kétszer negáljuk} \quad F = \overline{\overline{(A + B) \cdot (\overline{C} + D)}}$$

Felbontjuk az egyik negációt és értelmezzük a tagok közti ÉS kapcsolatra

$$F = \overline{\overline{A + B} + \overline{\overline{C} + D}}$$ (Realizáljuk)

Feladatmegoldás

- **Készítsük el az** $F^3 = \Pi^3(0, 1, 4, 6, 7)$ **függvényt NÉV rendszerben és NOR rendszerben is!**

- **A kiolvasott függvény:** $F = (A + B) \cdot (\overline{A} + \overline{B}) \cdot (\overline{B} + \overline{C})$

- **Negáljuk kétszer a NOR alak eléréséhez:**

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

$$F = \overline{\overline{(A + B)} + \overline{(\overline{A} + \overline{B})} + \overline{(\overline{B} + \overline{C})}} \quad \textit{ez NOR alak!}$$