Zalotay Péter

DIGITÁLIS TECHNIKA

Bevezetés

A digitális technika módszereivel az információ leképzés, műveletvégzés és az eredmények továbbítása kétértékű elemi információk (bitek) sorozatával, digitális szavakkal történik. A különböző műveletvégzések egyszerű logikai döntések sorozatára vezethetők vissza. Ugyancsak logikai műveleteket kell végezni, pl. két - különböző mennyiség értékét hordozó - információ közötti viszony (kisebb, nagyobb, egyenlő) megállapításához. Mielőtt a digitálistechnika alapjairól írnánk, röviden ismerkedjünk meg – a teljesség igénye nélkül – az e technikát megalapozó legjelentősebb személyek munkásságával.

George Boole (1815-1864) angol matematikus foglakozott legelőször a formális logika algebrai szintű leírásával és alkotta meg a róla elnevezett algebrát, melyet 1847-ben a " The Mathematical Analiysis of Logic " című könyvében tett közzé.

C. Shannon mérnök-matematikus 1938 -ban megjelent 'Switching Theory' című könyvében adaptálta először G. Boole algebráját, kétállapotú kapcsolóelemeket tartalmazó logikai rendszerek leírására. Az információelmélet megalapítása is nevéhez fűződik, az információ alapegységét is tiszteletére róla nevezték el

Azóta hihetetlen mértékű fejlődés következett be a technika és ezen belül is a logikai rendszerek fejlődésében és alkalmazásában. Ez a fejlődés mind az elmélet, a rendszertechnika mind pedig a technológia területén igen gyors volt és természetesen ma is még az. A technológia fejlődésén természetesen itt elsősorban az áramköri elemek és az ehhez kapcsolódó logikai illetve áramköri rendszerek szerelésének automatizálásra lehet gondolni. Érdekes megfigyelni - véleményem szerint a technika fejlődésében egyedülálló módon - hogy voltak időszakok, amikor a technológia fejlődése - konkrétan a nagy bonyolultságú integrált áramkörök, a mikroprocesszorok megjelenése - készületlenül érte az elméletet, szinte lehagyva azt.

A következő felsorolás teljesen önkényes, de mindenképpen olyan tudománytörténeti neveket tartalmaz akik igen nagy mértékben elősegítették a logikai rendszerek elméletének kidolgozását, fejlődését,

Evarist Galois (1812-1832)
Francai matematikus a modern algebra egyik ágának megalapítója. Az általa létrehozott és róla elnevezett csoportelmélet adja a kódolás elmélet, a kriptográfia elméleti hátterét. Rövid élete alatt hozta létre ezt a nem éppen könnyen elsajátítható elméletet, még egyetemista korában párbajban meghalt.

Wilkes angol matematikus aki 1954 es években kifejlesztette a mikro-programozás elméletét, amelyet a technológia akkori szintjén még igen költséges lett volna alkalmazni. Ez az elmélet többek között a számítógépek központi vezérlőegységének tervezéshez adott univerzális megoldást. Első alkalmazásai között az igen népszerű IBM 360 -as számítógép is szerepelt.

1964-65 években Mealey és Moore mérnökök a logikai rendszerek tervezésének egy olyan zárt jól alkalmazható elméletét adták meg, mely a kor eszközbázisának megfelelő alkalmazását tette lehetővé.

Az 1971-es évre tehető az integrált áramköri gyártástechnológia olyan mértékű fejlődése, hogy lehetőséggé vált a számítógépek központi egységének megvalósítása egy vagy több tokban, vagyis megjelent a mikroprocesszor. Azóta a fejlődés még inkább felgyorsult és szinte nincs az iparnak, a szórakoztató-iparnak, a kereskedelemnek, a mezőgazdaságnak, a szolgáltatásoknak olyan területe, ahol a nagy integráltságú és olcsó digitális rendszerek ne terjedtek volna el. Kis túlzással azt mondhatnánk, hogy az utolsó egy két évtized a digitális technika korszaka volt és talán még marad is. Az integrált áramkörök gyártástechnológiájának

fejlődését igen jól mutatja az , hogy az 1972-es évek közkedvelt I8080 típusú mikroprocesszora még csak megközelítően 4700 tranzisztort tartalmazott, míg ma a kereskedelemben lehet kapni olyan Pentium alapú mikroprocesszort és egyéb rendszertechnikai elemeket tartalmazó chipet mely 150 millió tranzisztorból épül fel

Természetesen nem csak mikroprocesszorokat fejlesztettek ki, de más univerzálisan, vagy nagy sorozatban használható áramköri készletek is kialakultak:

- Memóriák
- programozhat logikai elemek: FPGA, stb.
- berendezés orientált integrált áramkörök
- céláramkörök, pl. Quarz órák

Az integráltsági fok növekedésével egyre több funkció került egy tokba (chipbe), amely jelentősen megnövelte a kivezetések számát is. Ezeknek a nyomtatott áramköri lemezre való beültetésére a hagyományos technológia nem volt alkalmas, ezért kifejlesztették a felületszerelési technológiákat (angolul Surface Mount Technology = SMT) és alkatrészeket (angolul Surface Mountage Devices) SMD.

Az egy chip -ben leintegrált logikai funkciók olyan bonyolultakká váltak, hogy tesztelésükre már a hagyományos módon nem volt lehetőség, ezért ki kellett fejleszteni új megoldásokat erre a feladatra, és ezek a ma oly közkedvelt szimulációs programok illetve hardware leíró nyelvek (VHDL).

Nagyon kevés műszaki szakterületet lehet találni, amelynek csak megközelítően is akkora irodalma volna mint a digitális technikának illetve rendszereknek. Ugyanakkor és ez talán ellentmondásnak tűnik, hogy ritka az olyan szakterület is amelyben olyan rövid idő alatt lehet olyan tudásra szert tenni , mellyel már egész komoly logikai rendszerek építhetők fel. Az ellentmondást az oldja fel, hogy ma már nem elegendő ha egy rendszer működik, ez csak egy alapkövetelmény, de annak számos esetben igen nagy megbízhatósággal, könnyű szervizelhetőséggel, versenyképes áron kell megvalósulnia. És az ilyen "hiba tűrő" rendszerek tervezése és szervizelése nagy tudást igényel.

1. DIGITÁLISTECHNIKA ALAPJAI

1.1. Logikai alapismeretek

Halmazelméleti alapfogalmak

Halmazon valamilyen **közös tulajdonsággal** rendelkező dolgok összességét értjük. A halmazhoz tartozó "dolgok összességét" a **halmaz elemeinek** nevezik. Az adott tulajdonságokkal nem rendelkező dolgok összessége alkotja a **komplemens** vagy kiegészítő halmazt.

A halmazok lehetnek végesek vagy végtelenek a halmazt alkotó elemek számától függően. Két speciális halmazt is definiálnak: üres halmaz melynek egyetlen eleme sincs, és a teljes vagy univerzális halmazt, amelyet valamely halmaz és ennek **komplemens** - e alkot.

Egy halmaz általában további részekre úgy nevezett részhalmazokra is oszthatunk, mely úgy jön létre, hogy az adott halmazhoz még további szűkítő feltételt is rendelünk. Például vegyük egyszerűség kedvéért a természetes számok halmazát, ezeknek részhalmazai lehetnek a prímszámok, a 2-vel vagy 3-mal osztható számok stb.

Azon részhalmazt mely minden eleme része két vagy több halmaznak, azt a két halmaz közös részének (metszet) vagy latin kifejezéssel élve a két halmaz **konjunkció** - jának mondjuk. Előbbi példánkban az A halmaz a 2-vel a B halmaz a 3-mal osztható számokat jelentse, akkor azon számok melyek 2-vel és 3-mal is oszthatók a két halmaz közös részét más szóval metszetét képezik. Általánosan tehát az A halmaz elemei **2i** ahol **i** Ì **[1, ¥]**, a B halmazé **3j** ahol **j** Ì **[1, ¥],** és így a közös rész halmazát a **6k** ahol **k** Ì **[1, ¥]** számok képezik**.** A közös rész jelölésére halmazelméletben, L vagy Ç jelet, illetve a Boole algebrában a **logikai szorzást** és ennek jelét használják.

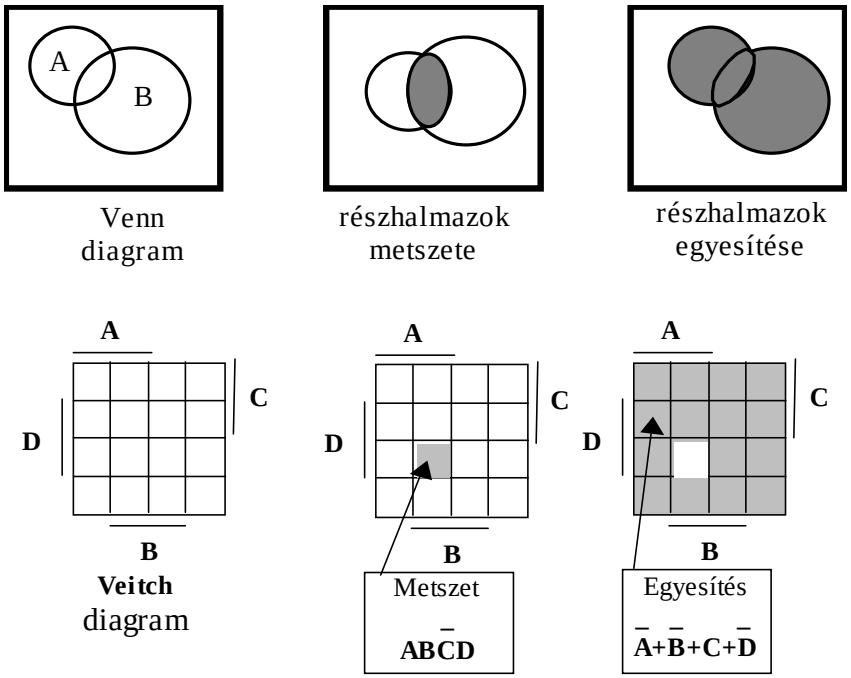
Tehát: **A Λ B = A ∩ B=A B**

Azon elemekből felépülő halmazt mely tartalmazza mind az A mind pedig a B (vagy esetleg több halmaz) elemeit a két halmaz egyesített halmazának vagy uniójának nevezik. Latin szóval ezt a műveletet **diszjunkció** - nak nevezik. Előbbi példánknál maradva az egyesített halmaz elmei 6i, 6i-2,6i-3,6i-4 (i=1,2,3...).

Az egyesített halmaz jelölésére az **U** vagy **V** szimbólumot illetve a Boole algebrában a **logikai összeadást** és ennek jelét használják:

A U B = A V B = A + B

A halmazok és a rajtuk értelmezett műveletek jól szemléltethetők (a J. Venn és Veitch matematikusról elnevezett) diagramokkal is. A teljes halmazt egy négyzettel, míg a részhalmazokat egy zárt alakzattal célszerűen egy körrel (Venn diagramban) vagy egy téglalappal (Veitch diagramban) jelölik.



A Veitch diagramban minden változóhoz a teljes halmaz (esemény-tér) fele, míg a másik térfél a változó tagadottjához tartozik. Az ábra négyváltozós halmazt ábrázol. A peremezésnél vonalak jelzik, hogy az egyes változók melyik térfelén IGAZ értékűek. Az ábrán a metszésnek (ÉS művelet) azt a változatát szemlélteti, amelyik mindegyik változó valamelyik értékének közös területe. Ez metszi ki a legkisebb elemi területet, ezért nevezik ezt **minterm** - nek. A másik ábrán az összes változó valamely értékeihez tartozó együttes terület. Az egyesített terület a legnagyobb részterület, amelyet **maxterm** - nek neveznek. Mind a két kitüntetett területből $\mathbf{2^n}$ – en darab van, ahol **n** a változók száma.

1.2. A logikai algebra

A logikai algebra a **Boole algebra** axióma rendszerét veszi alapul, de kiegészítésekkel a **digitális** rendszerek tervezésére, elemzésére alkalmas algebrává fejlődött.

A továbbiakban összefoglaljuk a logikai algebra alapjait. A logikai áramkörök később sorra kerülő ismertetésénél, valamint azok működésének megértéséhez az algebrai alapok biztos ismerete elengedhetetlen.

1.2.1. Logikai változók, és értékük

A logikai algebra csak **kétértékű logikai változók** halmazára értelmezett.

A logikai változók két csoportba oszthatók, úgymint

- független-, és
- függő változókra.

Mindkét csoport tagjait a latin ABC nagy betűivel (A, B, C . . . X, Y, Z) jelöljük. Általában az ABC első felébe eső betűkkel a független, az utolsó betűk valamelyikével pedig a függő változókat jelöljük.

A változók két logikai értéke az IGAZ, ill. a HAMIS érték. Ezeket 1-el, ill. 0-val jelöljük (IGAZ: 1; HAMIS: 0).

1.2.2. A Boole algebra axiómái

Az **axiómák** olyan előre rögzített kikötések, **alapállítások**, amelyek az algebrai rendszerben mindig érvényesek, viszont nem igazolhatók. Ezen állítások a halmaz elemeit, a műveleteket, azok tulajdonságait stb. határozzák meg. A **tételek**, viszont az axiómák segítségével bizonyíthatók.

A Boole algebra a következő axiómákra épül:

1. Az algebra **kétértékű** elemek halmazára értelmezett.
2. A halmaz minden elemének létezik a **komplemens** -e is, amely ugyancsak eleme a halmaznak.
3. Az elemek között végezhető műveletek a **konjunkció** (logikai ÉS), illetve a **diszjunkció** (logkai VAGY).
4. A logikai műveletek:

   Kommutatív –ak (a tényezők felcserélhetők),
   Asszociatív – ak (a tényezők csoportosíthatók),
   Disztributív – ak (a két művelet elvégzésének sorrendje felcserélhető).

5. A halmaz kitüntetett elemei az

   egység elem (értéke a halmazon belül mindig IGAZ), és a
   Nulla elem (értéke a halmazon belül mindig HAMIS).

A **logikai algebra** a felsorolt axiómákra épül. A logikai feladatok technikai megvalósításához a halmaz egy elemének komplemens - ét képező művelet is szükséges. Ezért a műveletek között a logikai **TAGADÁS** (NEM) is szerepel.

1.2.3. Logikai műveletek

A változókkal végezhető logikai műveletek az:

- **ÉS** (konjunkció) - logikai szorzás;
- **VAGY** (diszjunkció) - logikai összeadás;
- **NEM** (negáció, invertálás) - logikai tagadás.

A felsorolt műveletek közül az ÉS, ill. a VAGY logikai művelet két-, vagy többváltozós. Ez azt jelenti, hogy a változók legalább két eleme, vagy csoportja között értelmezett logikai kapcsolatot határoz meg. A tagadás egy változós művelet, amely a változók, vagy változócsoportok bármelyikére vonatkozhat.

1.2.3.1. Az ÉS (AND) művelet

A logikai változókkal végzett **ÉS** művelet **eredménye akkor** és **csak akkor IGAZ,** ha **mindegyik** változó értéke egyidejűleg **IGAZ**. A logikai algebrában az ÉS kapcsolatot szorzással jelöljük (logkai szorzás).

(Megj. a logikai szorzás jelet - akár csak az Euklideszi algebrában - nem szokás kitenni, így a továbbiakban mi is eltekintünk ettől).

Az

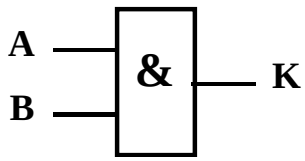
$$\mathbf{AB = K}$$

algebrai egyenlőségben **A** és **B** a független változók, és **K** a függő változó, vagy eredmény. Jelentése pedig az, hogy a K akkor IGAZ, ha egyidejűleg az A és a B is IGAZ.

Mint a későbbiekben látni fogjuk a logikai függvénykapcsolatok több féle képen is megadhatók, az egyik általánosan használt a táblázatos megadási mód. Mivel minden változó csak két értéket vehet fel ezért **n** változó esetén összesen $\mathbf{2^n}$ különböző kombináció különböztethető meg (2 elemből álló n-ed osztályú ismétléses variáció!). Így két változó esetén az összes lehetséges kombinációk száma négy. Az igazságtáblázat bal oldalán adjuk meg a bemeneti vagy független változók értékét, míg jobb oldalán a kimentei vagy függő változó értékei szerepelnek. Az ÉS műveletet leíró táblázat tehát az alábbi:

B	A	K=A B
0	0	0
1	0	0
0	1	0
1	1	1

Az algebrai alaknál sokkal áttekinthetőbb a művelet szimbolikus jellel történő megadása. Az ÉS művelet szimbolikus jele:



A művelet értelmezéséhez vegyünk egy nagyon hétköznapi példát. Ahhoz, hogy egy szobában a lámpa világítson, alapvetően két feltételnek kell teljesülni:

- legyen hálózati feszültség;
- a kapcsoló bekapcsolt állapotban legyen.

Szóban megfogalmazva: **ha** van hálózati feszültség **és** a kapcsoló bekapcsolt, **akkor** a lámpa világít. (Az egyéb követelmények teljesülését, hogy az áramkör elemei jók feltételezzük.) Ebben az egyszerű technikai példában a hálózati feszültség és a kapcsoló állapota a független-, a lámpa működése pedig a függő változó. Mindhárom tényező kétértékű.

1.2.3.2. A VAGY (OR) művelet

A logikai **változókkal** végzett **VAGY** művelet **eredménye** akkor **IGAZ,** ha a **független változók** közül **legalább** az **egyik IGAZ.**

Algebrai formában ezt a független változók összegeként írjuk le (logikai összeadás). Az

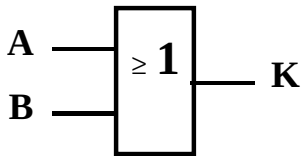
$$\mathbf{A + B = K}$$

alakú algebrai egyenlőségben a K eredmény akkor IGAZ, ha vagy az A, vagy a B, vagy mindkettő IGAZ.

A VAGY műveletet leíró táblázat tehát az alábbi:

B	A	K=A + B
0	0	0
1	0	1
0	1	1
1	1	1

A művelet szimbolikus jele:



Erre a logikai kapcsolatra ismert technikai példa egy gépkocsi irányjelzőjének működését ellenőrző lámpa. A vezető előtt a műszerfalon levő lámpa világít, ha a külső irányjelzők közül vagy a jobb oldali, vagy a bal oldali jelzőlámpacsoport világít. Azt az állítást, hogy jobb oldali jelzés van, jelölje **J** és azt, hogy bal oldali a jelzés pedig **B**. Az eredményt, hogy a belső ellenőrző lámpa világít, jelöljük **L**-lel. A működést leíró logikai egyenlőség:

$$\mathbf{B} + \mathbf{J} = \mathbf{L}$$

alakú lesz.

1.2.3.3. A tagadás (INVERS) művelete

A logikai tagadást egyetlen változón, vagy csoporton végrehajtott műveletként értelmezzük. Jelentése pedig az, hogy **ha** a **változó IGAZ, akkor** a **tagadottja HAMIS** és fordítva. Algebrai leírásban a tagadást a változó jele fölé húzott vonallal jelöljük. Ezek szerint a

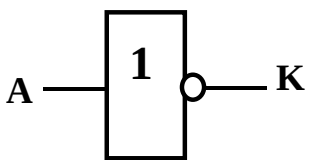
$$\mathbf{K} = \overline{\mathbf{A}}$$

egyenlőség azt jelenti, hogy a **K** akkor IGAZ, ha az **A** HAMIS. (Szóban A nem - nek, A felülvonásnak vagy A tagadottnak mondjuk.)

Táblázatban is megadható:

A	K
0	1
1	0

A művelet szimbolikus jele:



Az

$$\overline{\mathbf{A} * \mathbf{B}} = \mathbf{K}$$

összefüggés azt írja le, hogy az eredmény (K) csak akkor igaz, ha az A*B logikai ÉS művelet eredménye HAMIS értéket ad.

A tagadás műveletének előzőek szerinti értelmezése alapján abban a példában, amelyet az ÉS művelet magyarázatára hoztunk az A (A nem) azt jelenti, hogy nincs hálózati feszültség, ill. a B (nem) jelenti azt, hogy a kapcsoló nincs bekapcsolva. Az eredmény tagadása (K) azt fejezi ki, hogy a lámpa nem ég. Az előzőek alapján a gépkocsi irányjelzését ellenőrző lámpa működését leíró összefüggésben is értelmezhetjük a J-t (jobb oldali jelzés nincs),a B-t (bal oldali jelzés nincs) és az L - t (ellenőrző lámpa nem világít) jelölések technikai tartalmát.

1.2.3.4. A logikai műveletek tulajdonságai

A következőkben a logikai ÉS, valamint logikai VAGY műveletek tulajdonságait elemezzük.

Ø Kommutativitás (tényezők felcserélhetősége)

A leírt szemléltető példákat vegyük ismét elő. Azt állítottuk, hogy ha van hálózati feszültség, és a kapcsoló kapcsolt, akkor a lámpa világít. Az eredmény változatlan, ha az állítások sorrendjét **felcseréljük**, vagyis ha a kapcsoló be van kapcsolva és van hálózati feszültség, akkor világít a lámpa. Ez a látszólagos szójáték arra utal, - ami általánosan igaz - hogy az ÉS műveletekben a változók sorrendje felcserélhető, amely algebrai formában az

$$\mathbf{AB} = \mathbf{BA}$$

azonossággal írható le.

Az előzőekhez hasonlóan meggyőződhetünk arról is, hogy a VAGY műveletekben is felcserélhető -ek az egyes állítások. Érvényes a

$$\mathbf{J} + \mathbf{B} = \mathbf{B} + \mathbf{J}$$

azonosság.

Tehát mind az ÉS, mind pedig a VAGY művelet kommutatív.

Ø Asszociativitás (a tényezők csoportosíthatósága)

A két logikai művelet további tulajdonsága a műveleti tényezők **csoportosíthatósága** is, vagyis az **asszociativitás**. Algebrai alakban az

$$\mathbf{ABC} = \mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C} = \mathbf{B}(\mathbf{AC})$$

ill. az

$$\mathbf{A} + \mathbf{B} + \mathbf{C} = \mathbf{A} + (\mathbf{B} + \mathbf{C}) = (\mathbf{A} + \mathbf{B}) + \mathbf{C} = \mathbf{B} + (\mathbf{A} + \mathbf{C})$$

azonosságok írják le az asszociatív tulajdonságot. A zárójel - a matematikai algebrához hasonlóan - a műveletvégzés sorrendjét írja elő. Eszerint a háromváltozós ÉS, ill. VAGY műveletet úgy is elvégezhetjük, hogy előbb csak két változóval képezzük az ÉS, ill. a VAGY kapcsolatot, majd annak eredménye és a harmadik változó között hajtjuk végre az előírt műveletet.

Ø Disztributivitás (a műveletek azonos értékűek)

A harmadik jelentős tulajdonság, hogy a logikai ÉS, valamint a logikai VAGY azonos értékű művelet. Mindkettő disztributív a másikra nézve. Algebrai formában ez a következőképpen irható le:

$$\mathbf{A(B+C) = AB + AC}$$

$$\mathbf{A + BC = (A+B)(A+C)}$$

Az első azonosság alakilag megegyezik a matematikai algebra műveletvégzés szabályával. A második azonosság csak a logikai algebrában érvényes. Kifejezi azt, hogy egy logikai szorzat (ÉS kapcsolat) és egy állítás VAGY kapcsolata úgy is képezhető, hogy először képezzük a VAGY műveletet a szorzat tényezőivel és az így kapott eredményekkel hajtjuk végre az ÉS műveletet.

A logikai műveletek megismert tulajdonságai segítségével a logikai kifejezések algebrai átalakítása hajtható végre, és így lehetőség van a legegyszerűbb alakú kifejezés megkeresésére. Ezt a későbbiekben még részletesebben fogjuk tárgyalni.

1.2.3.5. A logikai algebra tételei

A továbbiakban felsoroljuk a fontosabb **tételeket**, azok részletes bizonyítása nélkül.

- A kitüntetett elemekkel végzett műveletek:

1*1 = 1	**0*0 = 0**
1*A = A	**0*A = 0**
1+1 = 1	**0+0 = 0**
1+A = 1	**0+A =A**

- Az azonos változókkal végzett műveletek:

$\mathbf{A*A = A}$	$\mathbf{A*\overline{A} = 0}$
$\mathbf{A+A = A}$	$\mathbf{A+\overline{A} = 1}$

Meg kell jegyezni, hogy az A-val jelzett logikai változó nem csak egy változó, hanem egy logikai **műveletsor** eredményét is jelentheti.

- A logikai tagadásra vonatkozó tételek:

$$\mathbf{\overline{\overline{A}} = A} \qquad \mathbf{\overline{\overline{\overline{A}}} = \overline{A}}$$

Általánosan: a páros számú tagadás nem változtatja meg az értéket, míg a páratlan számú tagadás azt az ellenkezőjére változtatja.

$$\mathbf{\overline{(A+B)} = \overline{A}*\overline{B}} \qquad \mathbf{\overline{A*B} = \overline{A}+\overline{B}}$$

Az előző két tétel az un. **De Morgan** - tételek, amelyek általánosan azt fogalmazzák meg, hogy egy logikai kifejezés tagadása úgy is elvégezhet, hogy az egyes változókat tagadjuk, és a logikai műveleteket felcseréljük (VAGY helyett ÉS, ill. ÉS helyett VAGY műveletet végzünk).

- Általános tételek:

$$\mathbf{A}(\mathbf{A}+\mathbf{B})=\mathbf{A} \qquad \mathbf{A}+\mathbf{AB}=\mathbf{A}$$

E két tétel a műveletek disztributív tulajdonsága és a már felsorolt tételek segítségével a következőképpen bizonyítható:

$$\mathbf{A}(\mathbf{A}+\mathbf{B})=\mathbf{AA}+\mathbf{AB}=\mathbf{A}(\mathbf{1}+\mathbf{B})=\mathbf{A}$$

$$\mathbf{A}+\mathbf{AB}=(\mathbf{A}+\mathbf{A})(\mathbf{A}+\mathbf{B})=\mathbf{A}(\mathbf{A}+\mathbf{B})=\mathbf{A}$$

További általános tételek

$$\mathbf{A}(\overline{\mathbf{A}}+\mathbf{B})=\mathbf{AB}$$

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

$$\mathbf{AB}+\overline{\mathbf{A}}\mathbf{B}=\mathbf{B}$$

$$(\mathbf{A}+\mathbf{B})(\overline{\mathbf{A}}+\mathbf{B})=\mathbf{B}$$

$$\mathbf{AB}+\mathbf{BC}+\overline{\mathbf{A}}\mathbf{C}=\mathbf{AB}+\overline{\mathbf{A}}\mathbf{C}$$

$$(\mathbf{A}+\mathbf{B})(\overline{\mathbf{A}}+\mathbf{C})=\mathbf{AC}+\overline{\mathbf{A}}\mathbf{B}$$

A legutóbb felsorolt tételek is bizonyíthatók az alaptulajdonságok segítségével.

1.2.4. Algebrai kifejezések

A továbbiakban ismertetünk néhány módszert, amelyeket az algebrai kifejezések átalakításánál gyakran használunk.

1. Az algebrai kifejezés **bővítése**.

Egy logikai szorzat nem változók, ha **1**-el megszorozzuk, vagyis

$$\mathbf{AB}=\mathbf{AB}*\mathbf{1}$$

Az 1-et pedig felírhatjuk, pl. $(\mathbf{C}+\overline{\mathbf{C}})$ alakban. Tehát:

$$\mathbf{AB}=\mathbf{AB}(\mathbf{C}+\overline{\mathbf{C}})=\mathbf{ABC}+\mathbf{AB}\overline{\mathbf{C}}$$

Egy logikai összeadás nem fog megváltozói, ha 0-t hozzáadunk:

$$\mathbf{D}+\mathbf{E}=\mathbf{D}+\mathbf{E}+\mathbf{0}$$

A 0-t kifejezhetjük $\mathbf{F}*\overline{\mathbf{F}}$ alakban. A bővítést végrehajtva az

$$\mathbf{D}+\mathbf{E}=(\mathbf{D}+\mathbf{E})+\mathbf{F}*\overline{\mathbf{F}}=(\mathbf{D}+\mathbf{E}+\mathbf{F})(\mathbf{D}+\mathbf{E}+\overline{\mathbf{F}})$$

azonosságot kapjuk.

Ennél a bővítésnél felhasználtuk a disztributivitást leíró egyik algebrai összefüggést, mely szerint

$$\mathbf{A}+\mathbf{BC}=(\mathbf{A}+\mathbf{B})(\mathbf{A}+\mathbf{C})$$

Az előzőben ismertetett bővítési szabály megfordítva egyszerűsítésre is felhasználható.

Példa:

Igazoljuk a tételek között felsorolt

$$\mathbf{AB + BC + \overline{A}C = AB + \overline{A}C}$$

azonosságot!

Első lépésként a baloldal mindhárom tagját kibővítjük úgy, hogy szerepeljen bennük mind független változó (A,B,C).

$$\mathbf{AB(C + \overline{C}) + BC(A + \overline{A}) + \overline{A}C(B + \overline{B}) =}$$

$$\mathbf{= \underline{ABC} + AB\overline{C} + \underline{ABC} + \underline{\overline{A}BC} + \underline{\overline{A}BC} + \overline{A}B\overline{C}}$$

Az így kapott hat szorzatot tartalmazó kifejezésben kettő - kettő azonos. Ezek közül egy - egy elhagyható az A + A = A tétel analógiájára (pl. ABC +…. + ABC = ABC). Ezeket jelöltük egyszeres, illetve kettős aláhúzással.

Második lépésként a bővítés fordítottját végezzük, vagyis ahol lehet az azonos tényezőket kiemeljük.

$$\mathbf{ABC + AB\overline{C} + \overline{A}BC + \overline{A}B\overline{C} = AB(C + \overline{C}) + \overline{A}C(B + \overline{B}) = AB + \overline{A}C}$$

A zárójelekben levő kifejezések 1 értékűek.
Ezzel igazoltuk az eredeti azonosságot.

2. Algebrai kifejezés tagadása (a **De Morgan** - tételek alkalmazása).

$$\mathbf{\overline{A\overline{B}C + \overline{A}BC + \overline{A}B\overline{C}} = (\overline{A\overline{B}C})(\overline{\overline{A}BC})(\overline{\overline{A}B\overline{C}}) =}$$

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

$$\mathbf{(\underline{\overline{A}A + AB + A\overline{C} + \overline{A}B + B\overline{B} + \overline{B}\overline{C} + \overline{A}\overline{C} + B\overline{C} + \overline{C}\overline{C}})(A + B + \overline{C}) =}$$

▼

Az átalakításnál először a De Morgan - tételt használtuk. A következő lépésként az első két zárójeles kifejezés logikai szorzatát (ÉS művelet) képeztük (az eredmény aláhúzva).

Az aláhúzott részt célszerű tovább egyszerűsíteni az $\mathbf{\overline{A}A = 0}$, és a $\mathbf{B\overline{B} = 0}$ tényezők elhagyásával, illetve a $\mathbf{\overline{C}\overline{C} = \overline{C}}$ helyettesítéssel. Majd tovább is egyszerűsíthető a $\mathbf{\overline{C}}$ kiemelésével.

$$\mathbf{(\underline{AB + A\overline{C} + \overline{A}B + \overline{B}\overline{C} + \overline{A}\overline{C} + B\overline{C} + \overline{C}}) = AB + \overline{A}B + \overline{C}(A + \overline{B} + \overline{A} + B + 1) =}$$

$$\mathbf{= AB + \overline{A}B + \overline{C}}$$

A zárójelben levő kifejezés azonosan 1, mert a logikai összeadás egyik tagja 1. Térjünk vissza az eredeti kifejezéshez, amelynél a zárójelbe tett kifejezések ”összeszorzása”, majd a lehetséges további átalakítás után (pl. az aláhúzott kifejezések értéke 0 stb.) kapjuk meg a végeredményt.

▼

$$= (AB + \overline{\overline{A}\overline{B}} + \overline{C})(A + B + \overline{C}) = ABA + \underline{\overline{\overline{A}\overline{B}}A} + A\overline{C} + ABB + \underline{\overline{\overline{A}\overline{B}}B} + \overline{C}B +$$
$$+ AB\overline{C} + \overline{\overline{A}\overline{B}}\overline{C} + \underline{C\overline{C}} = AB + 0 + A\overline{C} + AB + 0 + \overline{C}B + AB\overline{C} + \overline{\overline{A}\overline{B}}\overline{C} + 0 =$$
$$= AB(1 + 1 + \overline{C}) + \overline{C}(A + B + \overline{\overline{A}\overline{B}}) = AB + \overline{C}$$

Példa:

Igazoljuk a

$$\overline{\overline{D}\overline{F} + E\overline{F}} = F + D\overline{E}$$

azonosságot!

Első megoldás:

$$\overline{\overline{D}\overline{F} + E\overline{F}} = \overline{(\overline{D} + E)\overline{F}} = \overline{(\overline{D} + E)} + F = D\overline{E} + F$$

Második megoldás:

$$\overline{\overline{D}\overline{F} + E\overline{F}} = \overline{(\overline{D}F)}\,\overline{(E\overline{F})} = (D + F)(\overline{E} + F) = DF + FF + D\overline{E} + \overline{E}F =$$
$$DF + F + D\overline{E} + \overline{E}F = F(D + 1 + \overline{E}) + D\overline{E} = F + D\overline{E}$$

1.2.5. Logikai feladatok leírása igazságtáblázattal és logikai vázlattal

A logikai formában megfogalmazható, műszaki, számítási és irányítási feladatokban mindig véges számú elemi állítás szerepel. Ezek mindig csak két értéket vehetnek fel, vagy **IGAZ** - ak, vagy **HAMIS** - ak. Ebből következik, hogy a független változók lehetséges érték-variációinak a száma is véges. Minden egyes variációhoz a függő változó meg határozott értéke tartozik.

A logikai kapcsolat leírásának táblázatos formája az igazságtáblázat, amely felírását az alapműveletek tárgyalásánál már ismertettük. A táblázat tartalmazza a független változók összes érték-variációját és az egyes variációkhoz rendelt függvényértéket. Az igazságtáblázatban minden logikai változó IGAZ értékét 1-gyel, míg a HAMIS értéket 0-val jelöljük.

Összefoglalva: az igazságtáblázat oszlopainak száma az összes logikai változó számával (függő változók száma + független változók száma), sorainak száma pedig a független változók lehetséges értékvariációinak számával egyezik meg.

A lehetséges értékvariációk számát (V-t) általánosan a

$$V=2^n$$

összefüggéssel határozhatjuk meg, ahol **n** az összes független logikai változó száma.

A logikai szimbólumokkal ábrázolható egy összetett logikai kapcsolat is. Ezt nevezzük logikai vázlatnak. A logikai vázlat a hálózat tervezésénél hasznos segédeszköz. Ezzel a későbbi fejezetekben még részletesen foglalkozunk.

Megjegyezzük, hogy általában csak egy függő változót tartalmazó igazságtáblázatot írunk fel. Azokban az esetekben, ha egy logikai kapcsolat-rendszerben több függő változó van, célszerűbb mindegyikre külön-külön felírni az igazságtáblázatot. Ezzel áttekinthetőbb képet kapunk.

Példa:

Írjuk fel a $\mathbf{Z = A\bar{B} + \bar{A}B}$ logikai függvény igazságtáblázatát!

Első lépésként az igazságtáblázat oszlopainak és sorainak a számát határozzuk meg. Mivel két független-, (A, B) és egy függő változó (Z) van, az oszlopok száma 3. A sorok száma a független változók számából (n=2) a

$$\mathbf{V = 2^n = 2^2 = 4}$$

összefüggésből számolható.

B	A	Z

Második lépésként az értékvariációkat írjuk be. Célszerű ezt úgy végrehajtani, hogy az egyik oszlopban (A) soronként váltjuk a 0, és az 1 beírását. A következő oszlopban (B) párosával váltogatjuk az értékeket. (Nagyobb sorszámnál a következő oszlopoknál négyesével, majd nyolcasával variálunk s.i.t.) A beírásnak ez a rendszeressége biztosítja, hogy egyetlen variáció sem marad ki.

B	A	Z
0	**0**	
0	**1**	
1	**0**	
1	**1**	

Harmadik lépés az egyes sorokba írandó Z érték meghatározása. Ezt úgy végezhetjük el, hogy a független változóknak értékeket adunk, s az adott függvényt kiszámítjuk.

1. sorban**:** **A = 0, B = 0**
Z = 0*1 + 1*0 = 0

2. sorban: **A = 1, B = 0**
Z = 1*1 + 0*0 = 1

3. sorban: **A = 0, B = 1**
Z = 0*0 + 1*1 = 1

4.sorban: **A = 1, B = 1**
Z = 1*0 + 0*1 = 0

A teljesen kitöltött igazságtáblázat:

B	A	Z
0	**0**	**0**
0	**1**	**1**
1	**0**	**1**
1	**1**	**0**

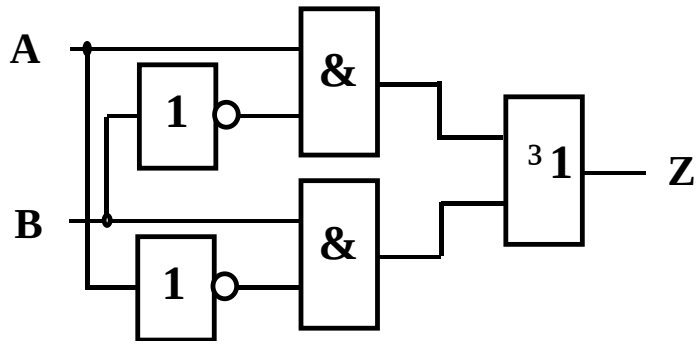
2. Rajzoljuk fel a példában szereplő logkai függvénykapcsolat logkai vázlatát!

A felrajzolást a Z-től érdemes kezdeni. A függvény

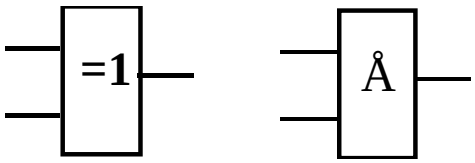
$$\mathbf{Z = (A\overline{B}) + (\overline{A}B)}$$

alakjából látjuk, hogy Z-t két kifejezés VAGY művelete határozza meg. Az egyes kifejezések (zárójelbe tett mennyiségek) két változó (egyenes és tagadott alakú) ÉS kapcsolata. Az A, ill. B állításokból a negált alakok egy-egy tagadással állíthatók elő.

A megismert szimbólumokkal megrajzolt logikai vázlat az ábrán látható.



Az előző példa egy sokszor használt függvény-kapcsolat, az un. **KIZÁRÓ-VAGY** (XOR) művelet. (Nevezik moduló összegnek is .) A művelet eredménye akkor 1, ha a két változó közül az egyik 1. Több változóval is végezhető **moduló - összegzés**, és eredménye akkor 1, ha páratlan számú változó értéke 1. A műveletnek a következő szimbólumok valamelyikét használják.



A KIZÁRÓ-VAGY tagadottja az EGYENLŐ (Equivalencia) művelet, amely akkor ad IGAZ értéket, ha az állítások egyformák.

1.2.6. Logikai függvény felírása az igazság-táblázatból

Az előző pontban megismerkedtünk az igazság-táblázattal, amely a logikai kapcsolatrendszer leírásának egyik módszere. Példa segítségével mutattuk be, hogy ismert logikai függvényből hogyan írható fel a táblázatos alak.

Ebben a részben azt fogjuk tárgyalni, hogy ha ismert az igazságtáblázat, hogyan lehet abból felírni a logikai függvényt.

Az igazságtáblázat egy sora a független változók adott kombinációját és az ehhez tartozó függvény értékét adja. Az **egy sorban** levő értékeket az **ÉS** művelettel lehet összekapcsolni. A különböző sorok pedig különböző esetnek megfelelő variációkat írnak le. Tehát egy adott időpillanatban vagy az egyik sor vagy egy másik sor variációja érvényes. A **sorok** logikai **kapcsolata VAGY** művelettel írható le.

Az igazságtáblázatból két módszerrel lehet felírni a feladat teljes alakú logikai függvényeit. Az alábbi példa kapcsán ismerkedjünk meg a módszerekkel, valamint a **teljes alakú** (diszjunkt, ill. konjunkt) **függvény-formákkal**.

· A **diszjunkt -** alakú függvény felírása.

C	B	A	K
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**	**1**
1	**1**	**1**	**0**

Az igazságtáblázat tartalmát a következőképpen olvassuk ki. A **K** jelű függő változó értéke **1** (IGAZ),

ha C=0 és B=0 és A=1 (2.sor),vagy

ha C=0 és B =l és A=0 (3.sor),vagy

ha C=0 és B=1 és A=1 (4.sor),vagy

ha C=l és B=l és A=0 (7.sor).

Az A,B,C és K változók közötti logikai kapcsolat az előbbiek szerint

$$\mathbf{K = A\overline{B}\overline{C} + \overline{A}B\overline{C} + AB\overline{C} + \overline{A}BC}$$

alakban is leírható

A felírás szabálya a következő:

1. *azokat a sorokat kell figyelembe venni, amelyeknél a függő változó értéke 1;*
2. *az egy sorban levő független változók között ÉS műveletet kell végezni, ahol a független változó igaz (egyenes, más kifejezéssel ponált) alakban írandó, ha értéke 1 és tagadott (negált) alakban, ha értéke 0;*
3. *az egyes sorokat leíró ÉS műveletű rész-függvények VAGY művelettel kapcsolódnak egymáshoz.*

Nézzük most meg az igazságtáblázatból felírt logikai függvény általános jellemzőit. A függvény rendezett **ÉS-VAGY** alakú. Az ÉS művelettel összekapcsolt részekben mindegyik változó szerepel egyenes vagy tagadott alakban, vagyis a Veitch diagramnál definiált **minterm.**

Az egyes **minterm** -ek között pedig VAGY műveleteket kell végezni. Az ilyen függvényalakot idegen szóval **diszjunktiv kanonikus** alaknak (teljes diszjunktiv normál formának) nevezzük.

· A **konjunkt** alakú függvény felírása.

Az igazságtáblázatból más módszerrel is felírhatunk logikai függvényt. Azt nézzük meg, hogy mikor nem IGAZ (HAMIS) a következtetés. A **K = 0**

ha C=0 és B=0 és A=0 (1.sor) vagy

ha C=1 és B=0 és A=0 (5.sor) vagy

ha C=1 és B=0 és A=1 (6.sor) vagy

ha C=1 és B=1 és A=1 (8.sor).

A leírt logkai kapcsolatot a

$$\overline{\mathbf{K}} = \overline{\mathbf{A}}\,\overline{\mathbf{B}}\,\overline{\mathbf{C}} + \overline{\mathbf{A}}\mathbf{B}\overline{\mathbf{C}} + \mathbf{A}\overline{\mathbf{B}}\overline{\mathbf{C}} + \mathbf{ABC}$$

függvénnyel írhatjuk le. Ebből a K értékét mindkét oldal tagadásával nyerhetjük.

$$\overline{\overline{\mathbf{K}}} = \overline{\overline{\mathbf{A}}\,\overline{\mathbf{B}}\,\overline{\mathbf{C}} + \overline{\mathbf{A}}\mathbf{B}\overline{\mathbf{C}} + \mathbf{A}\overline{\mathbf{B}}\overline{\mathbf{C}} + \mathbf{ABC}}$$

A bal oldalon K-t kapunk. A jobb oldal átalakítását a de Morgan - tételek alkalmazásával végezhetjük el.

$$\mathbf{K} = \mathbf{(}\overline{\overline{\mathbf{A}}\,\overline{\mathbf{B}}\,\overline{\mathbf{C}}}\mathbf{)(}\overline{\overline{\mathbf{A}}\mathbf{B}\overline{\mathbf{C}}}\mathbf{)(}\overline{\mathbf{A}\overline{\mathbf{B}}\overline{\mathbf{C}}}\mathbf{)(ABC)} =$$

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

A kapott függvényt elemezve, megállapíthatjuk, hogy annyi tényezőt (zárójeles mennyiséget) kaptunk, ahány helyen 0 értékű a K. A zárójeles mennyiségek mindhárom független változót (A,B,C) tartalmazzák egyenes vagy tagadott alakban, és VAGY művelet változói. Ezek **maxterm** -ek, melyeket a Veitch diagramnál definiáltunk. Az első maxterm az igazságtáblázat első sora szerinti állítás - vagyis, hogy az A=0 és B=0 és C=0 - tagadása. A további tagokat vizsgálva látjuk, hogy ezek is egy-egy olyan sornak a tagadásai, melyben K=0.

Az előzőek alapján most már megfogalmazhatjuk, hogy az igazságtáblázatból úgy is felírhatjuk a feladatot leíró logikai függvényt, hogy

1. *azokat a sorokat vesszük figyelembe, melyekben a függő változó értéke 0;*
2. *az egy sorban levő független változók között VAGY kapcsolatot írunk elő;*
3. *a független változót egyenes alakban írjuk, ha értéke 0 és tagadott alakban, ha értéke 1;*
4. *az egyes sorokat leíró VAGY függvényeket ÉS művelettel kell összekapcsolni.*

Azt a logikai függvényt, amely maxtermek logikai szorzata idegen szóval **konjunktív kanonikus** alakúnak, rendezett VAGY-ÉS függvénynek (teljes konjunktív normál alakúnak) nevezzük.

1.2.7. Teljes logikai függvények egyszerűsített felírású alakjai

Mivel a logikai változónak két értéke – 0, illetve 1 – lehet, ezért ezt tekinthetjük egy **bináris számjegy** -nek is.

1.2.7.1. A term –ek helyettesítése

A függvény egy maxtermjét, vagy mintermjét, bináris szám-ként is felírhatjuk. Ehhez először megadjuk a változókhoz rendelt **nagyságrend -** et**,** és a **ponált** változó helyére **1**-t, míg a **negált** helyére **0**-t írunk. Ez a szám lesz az adott maxterm, vagy minterm **sorszám**-a (súlya).

Például:

Legyen az $\mathbf{C \div 2^2, B \div 2^1, A \div 2^0}$ súlyozású. Ekkor a

$\mathbf{C\overline{B}A}$ minterm súlya: $\mathbf{1*2^2+0*2^1+1*2^0=101_B=5}$

$\mathbf{\overline{C}+B+\overline{A}}$ maxterm súlya: $\mathbf{0*2^2+1*2^1+0*2^0=010_B=2}$.

A mintermek az $\mathbf{m_i^v}$ kifejezéssel is felírhatók, ahol az **m** jelzi, hogy **minterm**, a felső index **v** a **változók számát**, az alsó index **i** pedig a **sorszámot** jelenti.

Hasonlóan a maxtermek is helyettesíthetők a $\mathbf{M_i^v}$ kifejezéssel. Az indexek (v,i) jelentése ugyan az, míg az **M** jelzi, hogy **maxterm**.

A leírtakat a példában szereplő kifejezésekre (ugyanazon változó súlyozásnál) a

$$\mathbf{C\overline{B}A \div m_5^3}$$

és a

$$\mathbf{\overline{C}+B+\overline{A} \div M_2^3}$$

helyettesítéseket alkalmazhatjuk.

1.2.7.2. Függvények helyettesítése

Az ismertetett helyettesítésekkel a diszjunktív, valamint konjunktív kanonikus alakú függvények is rövidebben írhatók. Vegyük példának az előzőekben felírt függvények alaki helyettesítését az $\mathbf{A \div 2^2, B \div 2^1, C \div 2^0}$ változó súlyozás alkalmazásával:

$$\mathbf{K = A\overline{B}\overline{C} + \overline{A}B\overline{C} + AB\overline{C} + \overline{A}BC}$$

$$\mathbf{K = m_4^3 + m_2^3 + m_6^3 + m_3^3}$$

$$\mathbf{K = (A+B+C)(A+B+\overline{C})(\overline{A}+B+\overline{C})(\overline{A}+\overline{B}+\overline{C})}$$

$$\mathbf{K = M_7^3 * M_6^3 * M_2^3 * M_0^3}$$

1.2.7.3. Indexelt felírás:

A függvények felírása tovább is egyszerűsíthető oly módon, hogy

- megadjuk a **függvény – alak** -ot
- a változók számát, és
- a függvényben szereplő **term** –ek sorszámait.

A diszjunktív alakot a $\overset{v}{\mathring{a}}$**(.....)** ,a

konjunktív alakot a $\overset{v}{\tilde{O}}$**(.....)** formában írjuk.

A két minta függvény egyszerűsített felírása (ugyanazon változó-súlyozást alkalmazva):

$$\mathbf{K} = \overset{3}{\mathring{a}}\mathbf{(2,3,4,6)},$$

$$\mathbf{K} = \overset{3}{\tilde{O}}\mathbf{(7,6,2,0)}$$

1.2.7.4. Kanonikus függvény-alakokok közötti átalakítás

Az előzőekben megismertük, hogyan lehet a logikai feladat igazságtáblázatából felírni a logikai függvény két kanonikus alakját.

Az **egyik** kanonikus alakú függvény egyszerűsített formája **alapján** nagyon egyszerűen felírható a **másik** rendezett alak egyszerűsített formája.

Az átalakítás menete a következő:

- az ismert függvény alapján felírjuk az **inverz függvényt** (amely az alap függvény tagadottja), ezt a hiányzó indexű term – ek alkotják, ha

  ismert a diszjunktív alak:

  $\mathbf{K} = \overset{3}{\mathring{a}}\mathbf{(2,3,4,6)}$ Þ $\overline{\mathbf{K}} = \overset{3}{\mathring{a}}\mathbf{(0,1,5,7)}$

  ismert a konjunktív alak:

  $\mathbf{K} = \overset{3}{\tilde{O}}\mathbf{(7,6,2,0)}$ Þ $\overline{\mathbf{K}} = \overset{3}{\tilde{O}}\mathbf{(5,4,3,1)}$

- az **inverz** függvény **tagadásával** nyerjük a másik alakú rendezett függvényt. A tagadáskor a függvény - típusjele az ellenkezője lesz, és mindegyik index (**i**) **kiegészítőjét** ($\bar{\mathbf{i}}$) kell vennünk a következő számítás alapján:

  $$\bar{\mathbf{i}} = \mathbf{(2^v - 1) - i}$$

  A tagadások elvégzése után

  $\overline{\overline{\mathbf{K}}} = \overset{3}{\mathring{a}}\mathbf{(0,1,5,7)}$ Þ $\mathbf{K} = \overset{3}{\tilde{O}}\mathbf{(7,6,2,0)}$

$$\overline{\overline{K}} = \overline{\overline{O}(5,4,3,1)} \quad \text{þ} \quad K = \overline{a}(2,3,4,6)$$

megkaptuk a keresett alakú függvényeket.

1.2.8. A logikai függvények egyszerűsítése

Az igazságtáblázat alapján felírt kanonikus alakú függvények a legtöbb esetben egyszerűsíthetőek. Az **egyszerűsítés** azt jelenti, hogy a logikai algebra megismert tételeinek felhasználásával olyan alakot nyerhetünk, amelyben **kevesebb művelet**, és vagy kevesebb **változó** szerepel. Az egyszerűsítésre azért van szükség, mert ez után a feladatot megvalósító logikai hálózat kevesebb áramkört, vagy programozott rendszer (mikrogép) programja kevesebb utasítást tartalmaz

Az **algebrai** módszer mellett kidolgoztak **grafikus**, illetve **matematikai** egyszerűsítési eljárásokat is.

A felsorolt egyszerűsítési (minimalizálási) eljárásokat az 1.2.6. fejezetben bemutatott igazságtáblázattal leírt logikai feladat segítségével ismertetjük. A feladat igazságtáblázata:

C	B	A	K
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	1
1	1	1	0

1.2.8.1. Algebrai egyszerűsítés

A logikai algebra tárgyalásakor már bemutattunk néhány átalakítási eljárást. Itt egy újabb példa segítségével végezzük el a feladat legegyszerűbb alakjának megkeresését.

a. Egyszerűsítés a diszjunktív alakú függvényből

$$K = A\overline{B}\overline{C} + \overline{A}B\overline{C} + AB\overline{C} + \overline{A}BC$$

Először keressük meg, hogy vannak-e közös részeket tartalmazó mintermek. És ezeket ”emeljük” ki !

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

A zárójelekben lévő mennyiségek értéke 1, ezért azok a logikai szorzatból elhagyhatók. A keresett, legegyszerűbb függvényalak a következő:

$$\mathbf{K = \overline{A}B + A\overline{C}}$$

b. Egyszerűsítés konjunktív alakú rendezett függvényből

$$\mathbf{K = (A + B + C)(A + B + \overline{C})(\overline{A} + B + \overline{C})(\overline{A} + \overline{B} + \overline{C})}$$

Hasonlóan az előző egyszerűsítéshez itt is végezhetünk – a disztributív tulajdonság alapján - ”kiemeléseket” a maxtermek -ből.

$$\mathbf{K = (A + B + C\overline{C})(\overline{A} + \overline{C} + B\overline{B})}$$

A $\mathbf{C\overline{C}}$ és $\mathbf{B\overline{B}}$ tényezők értéke 0 és ezért a logikai összegekből elhagyhatók. A keresett legegyszerűbb függvényalak tehát:

$$\mathbf{K = (A + B)(\overline{A} + \overline{C})}$$

c. Igazoljuk a két alakból kapott függvények azonosságát, vagyis hogy igaz az

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

egyenlőség.

Végezzük el a jobb oldalon a ”beszorzást”!

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

A kapott kifejezésben az első tényező 0. A negyedik tényezőt ”szorozzuk” 1-el.

$$\mathbf{0 + \overline{A}B + A\overline{C} + B\overline{C}(A + \overline{A}) = \overline{A}B + A\overline{C} + B\overline{C}A + B\overline{C}\overline{A}}$$

A közös részek ”kiemelése” után

$$\mathbf{\overline{A}B(1 + \overline{C}) + A\overline{C}(1 + B) = \overline{A}B + A\overline{C}}$$

a zárójeles kifejezések elhagyhatók, mivel értékük 1. A kapott eredménnyel igazoltuk az eredeti egyenlőség azonosságát.

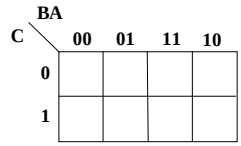
Ezzel bizonyítottuk, hogy az igazságtáblázatból a két - ismertetett - módszer bármelyikével ugyanazt a függvényt kapjuk.

Összefoglalva: megállapíthatjuk, hogy az igazság- táblázatból rendezett ÉS-VAGY (diszjunktiv kanonikus) alakú vagy rendezett VAGY-ÉS (konjunktív kanonikus) alakú logkai függvényt írhatunk fel. A két alak azonos függvényt ír le.

1.2.8.2. Grafikus egyszerűsítés Karnaugh –táblázattal

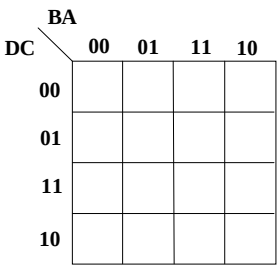
A logikai függvények grafikus ábrázolásainak közös jellemzője, hogy az elemi - egy műveletes - logikai függvényekhez (minterm, maxterm) sík-, vagy térbeli geometriai alakzatot rendelünk. A lehetséges egyszerűsítések közül csak a Karnaugh (ejtsd: karnó) – táblázatos eljárással foglalkozunk. A Karnaugh táblázat formailag a Veitch diagramból származtatott. A különbségek a változók megadásának (a peremezésnek) módjában, valamint abban van, hogy egy elemi négyszög mintermet, vagy maxtermet is jelképezhet. Egy **n** változós függvény $\mathbf{2^n}$ db elemi négyzetből álló táblázatban szemléltethető.

Az eljárás az ábra alapján követhető. A halmazt egy négyszögben ábrázoljuk. Minden változó 1 értékéhez a teljes terület egyik felét, míg a 0 értékéhez pedig a másik felét rendeljük. Az értékeket a négyszög szélére irt kódolással, adjuk meg. Több változó esetén a felezést úgy forgatjuk, hogy a változókhoz rendelt területeket jól meg lehessen különböztetni. Az a., és b ábrákon szemléltettük a 3 és 4 változós elrendezéseket. A változók kódolását úgy kell végezni, hogy az egymás melletti oszlopok, ill. sorok mindig csak egy változóban térjenek el egymástól.



a.

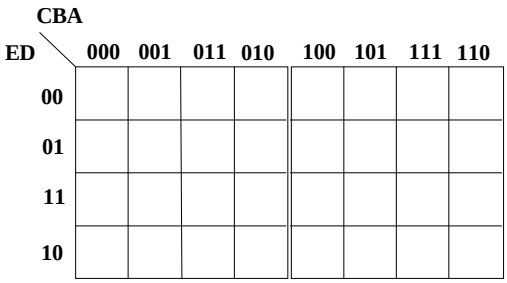
A **három**-változós Karnaugh - táblázat oszlopai a BA változó-pár lehetséges értékkombináció vannak rendelve. Az oszlop-peremezést úgy kell végezni, hogy a szomszédos oszlopok csak egyetlen változó-értékben különbözzenek. A harmadik változó C értéke szerint két sora van a táblázatnak. Az egyikben C=0,a másikban pedig C=1. Az egyes elemi négyszögekhez tehát a változók különböző értékvariáció tartoznak. A peremezés megváltoztatható, de csak úgy, hogy a szomszédos sorok, oszlopok egy változóban különbözhetnek. (A táblázat szélső oszlopai, illetve sorai mindig szomszédosak).



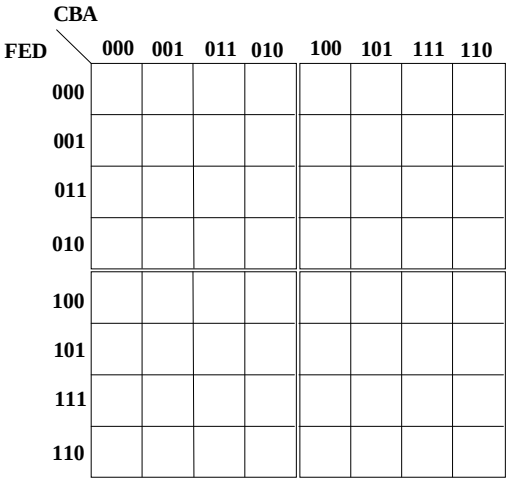
b.

A **négy**-változós Karnaugh - táblázatra is érvényesek az előző meghatározások.

Az ábrázolási mód legfeljebb 6 változóig alkalmazható szemléletesen. A következő ábrákon az 5, illetve a 6 változós táblázatot láthatjuk.



Az öt-változós táblázatot célszerű két négy-változós táblázatból úgy kialakítani, hogy a két rész peremezése csak az egyik változóban - itt pl a C – tér el egymástól.



A 6 változós táblázatnál függőlegesen duplázzuk meg a táblázat elemeit. Így négy 4 változós egységet kapunk. Az egyes rész-táblázatokban négy változó (ABED) azonosan variált. Az eltérés vízszintesen a C, míg függőlegesen az F változó.

Az eddigiekben csak az ábrázolás formai részével foglalkoztunk. Nézzük most meg a logikai tartalmat is. A két hozzárendelés szerint beszélünk **Kp** ill. **Ks** diagramról. A **Kp** jelölés az **ÉS-VAGY** (vagyis a mintermek logikai összege), míg a **Ks** a **VAGY-ÉS** (maxtermek logikai szorzata) műveletes összerendelést jelenti. A leírt kikötések betartásával - az előző fejezetben megismert - mindkét logikai függvényalak (diszjunktiv, ill. konjunktív) ábrázolható, és egyszerűsíthető Karnaugh – diagram segítségével. Először nézzük meg, hogyan tölthető ki a Karnaugh – diagram közvetlenül az igazságtáblázatból.

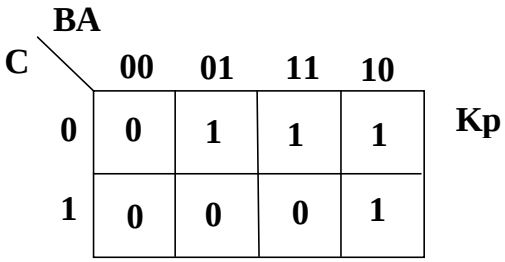
Példaként a **három-változós** – az előzőekben már megismert feladat igazság-táblázata:

C	B	A	K
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	1
1	1	1	0

a. Kp diagram használata.

A Karnaugh diagram egyes celláiba kell beírni a független változók (A,B,C) megfelelő kombinációihoz tartozó függő változó (K) értéket.

Az A=0,B=0,C=0 kombinációnál a K értéke 0, tehát a BA=00 oszlop és C=0 sor által meghatározott cellába 0-t kell írni és így tovább.



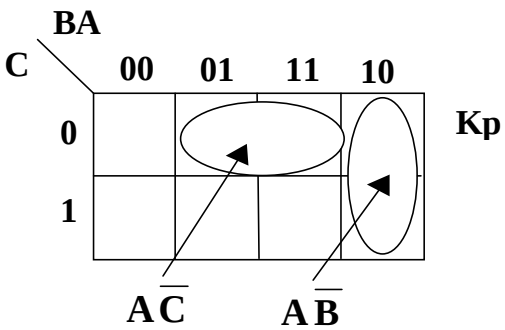
A 0 értékeket nem fontos beírni, ugyanis az egyszerűsítésnél csak az 1 értékű cellákat vesszük figyelembe.

Vizsgáljuk meg a diagram utolsó (BA=10) oszlopában lévő két cella tartalmát.

A felső cella tartalma az $\overline{A}B\overline{C}$, míg az alsó celláé $\overline{A}BC$ minterm. Mivel mindkét cella értéke 1, azt jelenti, hogy mindkét minterm a függvény tagja, és közöttük VAGY kapcsolat van. A két **minterm** -ből álló függvényrész egyszerűsíthető.

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

A példa alapján is bizonyítottnak tekinthetjük, hogy ha két – élben érintkező – cellában 1 van, akkor ezek összevonhatók, vagyis az a változó amelyikben különböznek a cellák kiesik. Az összevonhatóságot lefedő hurokkal szokás jelölni:



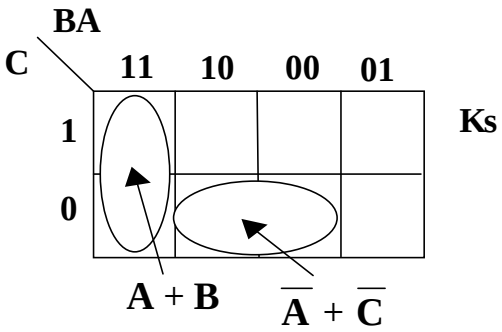
A lefedett (összevont) cellák VAGY kapcsolata adja az egyszerűsített függvényt:

$$K = \overline{A}B + A\overline{C}$$

b. Ks diagram használata.

Az egyszerűsített függvényalakoknál tárgyaltakhoz hasonlóan a Kp és a Ks diagramok is felrajzolhatók egymásból.

Az átrajzolásnál a peremezés, és a cella-értékek komplemens -ét kell írni, vagyis 0 helyett 1-e, és fordítva. A példa Ks diagramja:



A cellák most maxtermeket tartalmaznak, ezért az egyszerűsített függvény az összevonások (lefedések) közötti ÉS művelettel írható le:

$$K = (A + B)(\overline{A} + \overline{B})$$

c. Több **cella összevonása.**
A logikai függvények között vannak olyanok is, melyeknél többszörös algebrai összevonás is végezhető.

Keressük meg a következő négy (A,B,C,D) változós logikai függvény legegyszerűbb alakját!

A változókat súlyozzuk az **A ¸ 2^0,B ¸ 2^1,C ¸ 2^2,D ¸ 2^3,** szerint. A függvény egyszerűsített alakja:

$$F = \sum^{4} (8,10,12,13,14,15)$$

Rajzoljuk meg a függvény Karnaugh táblázatát.

CD \ AB	(0) 00	(1) 01	(3) 11	(1) 10
(0) 00				
(8) 01	1	1		
(12) 11	1	1	1	1
(4) 10				

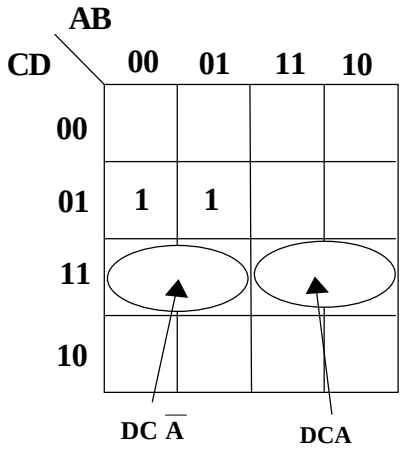
A Karnaugh diagram – egyszerűsített alakú függvény alapján történő – felrajzolását könnyíti, ha az egyes sorok és oszlopok súlyát decimálisan is jelöljük. Ezt tettük a zárójelbe írt számokkal.

Először írjuk fel a harmadik sor rész-függvényét algebrai alakban, mivel mindegyik cellában 1 értékű a függvény.

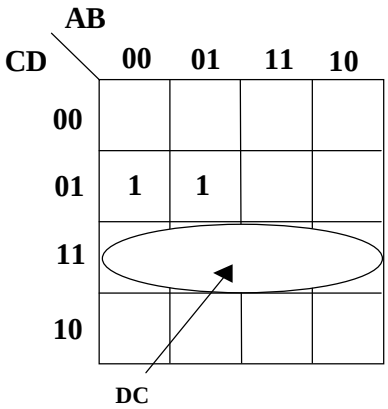
$$\overline{A}\overline{B}CD + \overline{A}BCD + ABCD + A\overline{B}CD = \overline{A}CD(\overline{B} + B) + ACD(B + \overline{B}) =$$

$$= \overline{A}CD + ACD = CD(\overline{A} + A) = CD$$

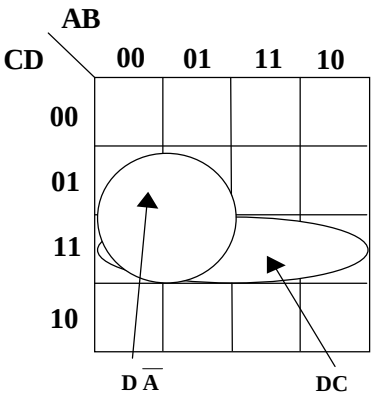
Az algebrai sorozatos kiemelések után két változó (A,B) kiesett.
Ugyanezt kövessük végig a Karnaugh diagramon is.



Mindkét lefedésnél kiesett a B változó. A két, három változós rész-függvényben közös a DC rész, tehát összevonható. A grafikus módszernél ez egy közös lefedéssel jelölhető.



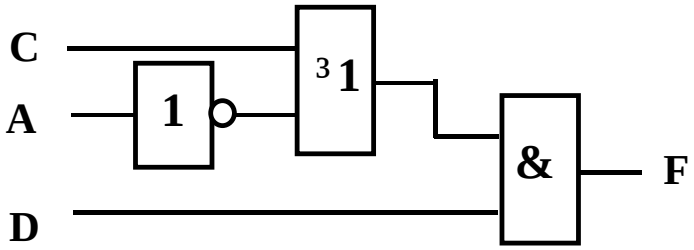
Hasonló négyes csoportot alkotnak a 8,10,12,14 sorszámú mintermek is, tehát összevonhatók.



Az egyszerűsített függvény a két részfüggvény logikai összege, amely még algebrailag tovább egyszerűsíthető:

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

Az utolsó egyszerűsítés eredményeként kaptuk a legkevesebb művelettel megvalósítható alakot. A logikai vázlata:



Összefoglalás:

A grafikus függvényegyszerűsítés szabályi:

- **2 páros** számú hatványa darab cella fedhető le (vonható össze , ha azok **kölcsönösen szomszédosak,**
- a kölcsönösen szomszédos meghatározást úgy kell érteni, hogy a **kiinduló** cellától indulva a szomszédos cellákon keresztül (2 páros számú lépés után) az **kiindulóhoz** jutunk vissza,
- a lefedett cellákból a **kitevőnek** megfelelő **számú** változó **esik ki**, amelyek a lefedés alatt változnak,
- **minden 1** -t tartalmazó cellát legalább **egyszer** le kell **fedni**.

2. EGYSZERŰ LOGIKAI ÁRAMKÖRÖK

Az előző fejezetben áttekintést adtunk azokról az alapvető matematikai és logikai ismeretekről, amelyek segítenek megérteni a digitális módon való műveletvégzés és jeltovábbítás módszereit, ill. az automatikus irányítóberendezések működésének elvét. Ebben a fejezetben a logikai műveleteket megvalósító alapvető **logikai áramkörökkel** ismerkedünk meg.

Részletesen fogúk ismertetni a **félvezetős kapu áramkörök** fizikai működését, logikai funkcióját és ezen elemi egységek egymáshoz csatlakoztatásának lehetőségeit, feltételeit. A tárgyalás során alapozunk a tanult elektromosságtani ismeretekre.

Logikai áramkörök

A megismert logikai műveletek (ÉS, VACY, NEM) technikai megvalósítása ma szinte kizárólag a **félvezető** alapú **digitális áramkörökkel** történik. Ezek részletesebb megismerése előtt célszerű a technikai fejlődést röviden összefoglalni.

Az elektronikus logikai áramköröket az alkalmazott áramköri elemek és az előállítási technológia alapján különböző generációkba soroljuk. Ez a besorolás egyúttal fejlődés-történeti csoportosítás is.

Az **első generációs** áramkörök **diszkrét passzív** áramköri elemekből (ellenállások, kondenzátorok, diódák stb.), valamint **elektroncsövekből** épültek fel. Felhasználásúkra elsősorban a negyvenes évek közepétől az ötvenes évek közepéig terjedő időszakban került sor.

A **második generációs** áramkörök ugyancsak **diszkrét passzív** áramköri elemeket tartalmaznak, de aktív elemeik már a **tranzisztorok**. Ezek az áramkörök a hatvanas évek közepéig voltak egyeduralkodók. Az áramkörök gyártástechnológiájára az alkatrészek nyomtatott áramköri lapokra szerelése a jellemző. Az egyszerű logikai funkciókat (ÉS, VAGY,NEM,TÁROLÁS) ellátó áramkörök egységes felépítésű - sorozatban gyártott - kártyákon (pl. EDS - kártyák) vagy térbeli elrendezésű, műgyantával kiöntött kockákban (pl. Terta kockák) kerültek forgalomba. Ezekből építették a különböző irányítóberendezéseket, mint pl. a forgalomirányító lámpák automatikus vezérléseit.

A **harmadik generációs** áramkörök csoportját alkotják kis és közepes bonyolultságú logikai (digitális) **integrált áramkörök** (IC- Integrated Circuit) (logikai kapuk, flip-flop -ok, regiszterek, számlálók stb.) alkalmazásával épített rendszerek. Az integrált áramkörök kb. 1 cm^3-es térfogatban (tokozással együtt) olyan nagyságrendű áramköri funkciót látnak el, amelyet a második generációs logikai áramkörökkel 1-2 dm^2-es nyomtatott áramköri lapon lehetett megvalósítani. A rendszerépítés az IC-kel is nyomtatott lapon történik. Ez a technika a hetvenes években vált egyeduralkodóvá, és napjainkban is ezt alkalmazzuk.

A negyedik generációs áramkörök közé a **nagy bonyolultságú** integrált áramkörök (a **mikroprocesszor**, kiegészítő **rendszerelemek**, **memóriák** stb.) tartoznak. A nagymértékű integrálás révén egyetlen tokban teljes rendszertechnikai egység (pl. központi egység) állítható elő. Néhány ilyen elem segítségével építhető egy teljes intelligens berendezés (mikroszámítógép, irányítástechnikai berendezés stb.).

A logikai áramkörök és egységek működésének megértéséhez elengedhetetlenül szükséges a diszkrét elemes félvezetős (második generációs), valamint a kis és közepes bonyolultságú integrált áramkörök (harmadik generációs) ismerete.

A digitális hálózatban az alapáramkörök végzik a logikai ÉS, VAGY, NEM (esetleg ezek kombinációjából álló) műveleteket, a tárolást, valamint a hálózat működését kisegítő, nem logikai funkciókat (időzítés, jelgenerálás, jelformálás stb.). Ezek alapján a következő logikai alapáramköröket különbözetjük meg:

- kapu áramkörök,
- tároló áramkörök (flip-f1opok),
- jelgenerátorok,
- késleltető áramkörök,
- jelformáló, illesztő áramkörök.

Az áramkörök elemzésénél használt gondolatmenet:

- az áramkör működésének,
- logikai funkciójának,
- csatlakoztatási feltételeinek

megismertetése lesz. A legfontosabb fogalmak közül, mint a

- villamos jelhordozók,
- terhelési viszony,
- jelterjedési idő

meghatározását előzetesen tárgyaljúk.

Külön kell még néhány mondatot szánni a **passzív**, ill. **aktív** áramköri elem fogalmának. A **passzív** elemek - mint pl. az ellenállás, kondenzátor, dióda - csak villamos **teljesítményt fogyasztanak**. Az **aktív** áramköri elemek - elektroncső, tranzisztor - villamos **teljesítmény átalakítására** is felhasználhatók. Önmaguk villamos energiát nem állítanak elő. A teljesítmény átalakításhoz (pl. erősítéshez) szükséges energiát a **tápforrásból** nyerik.

A logikai érték villamos jelhordozói

A különböző villamos áramkörökben a jelet - az információt - a villamos feszültség vagy áram hordozza. Amikor folytonosan változók az információ - pl. hangerő - a villamos jel minden egyes értéke (pl. nagysága) más és más információtartalmat képvisel. Az ilyen jelátvitelt nevezzük analógnak. A digitális technikában - mint ahogyan ezt már megismerték - az elemi információnak csak két értéke lehet (IGAZ, HAMIS). Amikor a logikai információt az áram hordozza, akkor az egyik értékhez rendeljük, hogy **folyik** áram, a másikhoz, pedig azt hogy **nem folyik** áram. Ez a jelhordozó-választás elsősorban az elektromechanikus relékkel megvalósított un. **relé-logikai** áramkörökben szokásos. Ez a téma terület nem anyaga a tantárgynak.

A félvezetős logikai áramkörökben (tananyagúnk témája) a logikai értéket hordozó villamos jellemző leggyakrabban a villamos feszültség. Mindkét logikai értékhez - egymástól jól elválasztva - egy-egy **feszültségtartományt** rendelünk. A logikai értékhez rendelt feszültségértékeket logikai feszültségszinteknek vagy rövidebben **logikai szinteknek** nevezzük.

Az egyes logikai értékekhez rendelt szintek egy-egy **feszültségsávot** jelentenek. E sávon belüli bár-mely feszültségérték ugyanazon elemi információt (logikai értéket) jelenti. Ez biztosítja azt, hogy az áramköri elemek tényleges értékének különbözősége (szórása) és a különböző környezeti feltételek (hőmérséklet, terhelés stb.) változósa az

információtartalmat nem módosítja. Ez is biztosítja a digitális jelfeldolgozás nagyobb zavarvédettségét az analóg módszerrel szemben.

A logikai **IGAZ** értékhez rendelt szintet **1** szintnek, vagy **IGEN** szintnek nevezik. A logikai **HAMIS** értékhez rendelt szint pedig a **0** vagy **NEM** szint.

Az angol eredetű áramköri leírásokban a pozitívabb logikai feszültségszintet **magas** vagy **H** (High) szintnek, a negatívabb feszültségszintet pedig **alacsony** vagy **L** (Low) szintnek is szokás nevezni.

A választott feszültségszintek egymáshoz viszonyított elhelyezkedése, valamint a megengedett feszültségsáv (szint tűrés) nagysága szerint többféle logikai szintrendszerről beszélünk.

A szintek egymáshoz való viszonya szerint megkülönböztetünk:

- **pozitív** és
- **negatív** logikai szintrendszert.

Pozitív logikai szintrendszerről akkor beszélünk, ha az **IGAZ** értékhez rendeljük a **pozitívabb** feszültségsávot. A **HAMIS** értéknek tehát a **negatívabb** feszültségsáv felel meg.

A **negatív logikai** szintrendszerben a **negatívabb** feszültségsávhoz (szinthez) tartozik az **IGAZ** érték és a **pozitívabb** szinthez, rendeljük a **HAMIS** értéket.

A technikai gyakorlatban az egyik szint mindig az áramköri rendszer közös 0 potenciálú értékét is magában foglaló feszültségsáv. A 1. ábra szemlélteti a pozitív, ill. negatív szintrendszer egy lehetséges elhelyezkedését a függőleges feszültségtengely mentén. Látható, hogy mindkét rendszerben a 0 logikai szint része a 0 feszültségérték.

A szintek tűrésének nagysága alapján megkülönböztetünk:

- **szabad** és
- **kötött** szintű

logikai áramköri rendszereket.

Szabad szintű a logikai áramköri rendszer, ha legalább az egyik feszültségszint széles határok között változhat. Általában ez a tűrés a tápfeszültség felével, egyharmadával egyezik meg.

Kötött szintű a logikai rendszer, ha mind az 1, mind pedig a 0 értékhez tartozó szint tűrése kicsi. Ennek értéke rendszerint a nyitott félvezető elemen (dióda, tranzisztor) eső feszültség két-háromszorosa.

A továbbiakban sorra kerülő áramköri elemzéseknél a logikai szintek és tűrések szélső értékeinek jelölésére – mint **statikus** jellemzőkre -a következő jelöléseket használjuk.

$\mathbf{U_1}$ - logikai **1** szint **névleges** értéke,

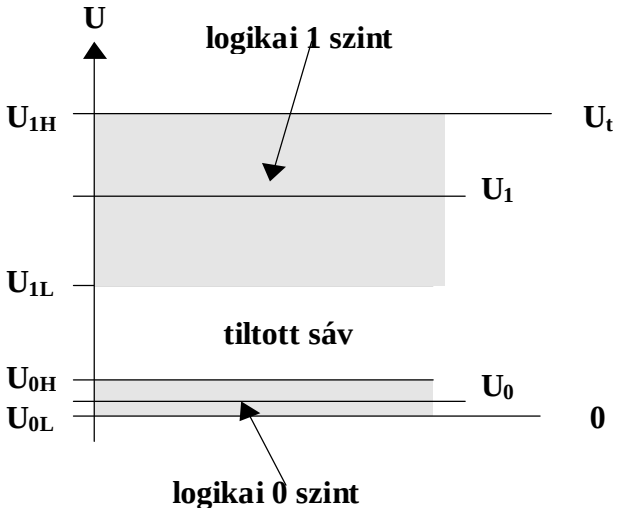
$\mathbf{U_{1H}}$ - logikai **1** szint **nagyobb** abszolút értékű szélső értéke,

$\mathbf{U_{1L}}$ - logikai **1** szint **kisebb** abszolút értékű szélső értéke,

$\mathbf{U_0}$ - logikai **0** szint **névleges** értéke,

$\mathbf{U_{0H}}$ - logikai **0** szint **nagyobb** abszolút értékű szélső értéke,

$\mathbf{U_{0L}}$ - logikai **0** szint **kisebb** abszolút értékű szélső értéke.



1. ábra

Az előző jelöléseket az 1.ábrán is feltüntettük.

2.1.1. Terhelési viszony

Összetett logikai hálózatokban egy áramkör - a logikai feladat függvényében - több áramkört is vezérelhet. Ezért ilyen esetekben azt is meg kell vizsgálni, hogy egy áramkör kimenetéhez hány további áramkör csatlakoztatható anélkül, hogy a megengedettnél nagyobb szinteltolódás vagy esetleg az áramköri elem tönkremenetele következne be.

Az egységesített áramkörrendszereknél a különböző funkciójú áramkörök legtöbb bemenete hasonló felépítésű, s így a bemeneti áram is azonos. Ezt szokták választani **egységterhelés**nek (terhelési egységnek). A **terhelési viszonyban** azt adják meg, hogy az egységterhelésnek hányszorosa az adott csatlakoztatásnál megengedett áram. Ez tehát egy **relatív érték**, egy nevezetlen szám.

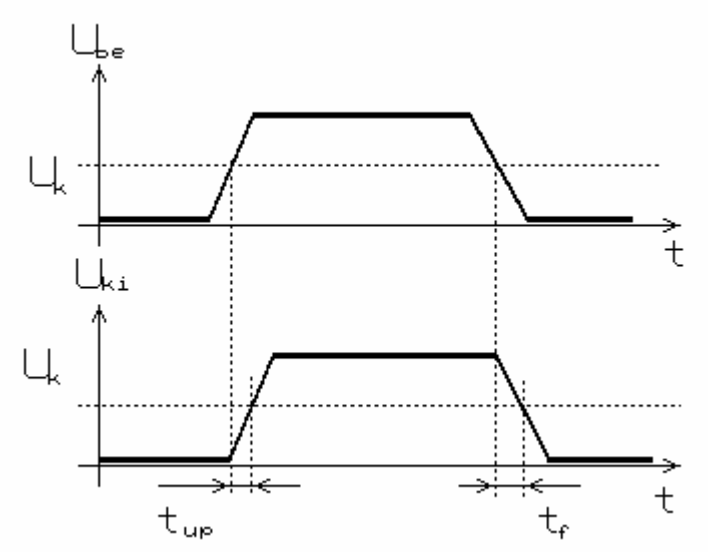
Bemeneti terhelési szám (fan-in) az áramkör bemeneti áramának és az egységterhelésnek a hányadosa.

Kimeneti terhelési szám (fan-out) az áramkör megengedett kimeneti áramának és az egységterhelésnek a hányadosa .A fan-out tehát megadja azt, hogy az áramkör hány áramkört tud vezérelni.

2.1.2. Jelterjedési idő

Bármely tetszőleges áramkör bemenetére jutó jelváltozóst a kimeneti jel változása mindig valamilyen késleltetéssel követi. A digitális áramkörökben a logikai információt hordozó villamos jel látszólag ugrásszerűen változik a 0-hoz és az 1-hez tartozó feszültségérték között. Valójában ez a szintváltás nem következhet be **nulla idő** alatt, mert ehhez **végtelen** nagy **energia** lenne szükséges. A tényleges változás az idő függvényében exponenciális vagy logaritmikus jellegű. Ezt szemléletesen láthatjuk egy oszcilloszkópon is, ha a vízszintes eltérítés frekvenciáját megnöveljük.

A 2. ábra szemlélteti egy négyszöghullámú bemeneti jellel vezérelt digitális áramkör be-, és kimeneti jeleinek időfüggvényeit.



2. ábra

Egy tényleges áramkör mindig késleltetve válaszol a bementi jelre. A késést a jelváltozások fél értékei (50%-os érték) között kell meghatározni. Rendszerint a két különböző irányú jelváltások ideje nem egyforma. A **0 -> 1** irányú változós késleltetését $\mathbf{t_u}$ -val (time-up ”emelkedési idő”), az **1 -> 0** váltás késleltetését pedig $\mathbf{t_f}$ **-**el (time-fall ”esési idő”) jelöljük.

Az áramkör **átlagos jelkésleltetési** idejét $\mathbf{t_{pd}}$ (propagation delay) a kétirányú változás késleltetésének számtani átlagaként számoljuk ki:

$$\mathbf{t_{pd}} = \frac{\mathbf{t_u} + \mathbf{t_f}}{\mathbf{2}}$$

Integrált áramköri kapuk

Az elektronikai ipar az elmúlt négy évtized alatt rendkívül gyors ütemben fejlődött. E fejlődés sor n az elektronikus berendezések és rendszerek bonyolultsága, és ezzel együtt mérete is rohamosan növekedni kezdett. A mind kisebb és mind megbízhatóbb elektronikus berendezések készítésére irányuló kutatásokat a hadiipar szükségletei indították el a második világháború idején. A háború befejezése utáni rövid visszaesést hamarosan megszüntette a tudományos és műszaki élet területén bekövetkezett fejlődés. A miniatürizálást nagymértékben indokolta a világűrkutatás rohamos fejlődése. A berendezések bonyolultsága olyan mértékben nőtt, hogy a megbízhatóságot már nem is annyira az alkatrészek megbízhatósága, mint az összeköttetéseké határozta meg. A kialakult hálózatban nyilvánvalóvá vált, hogy a problémák megoldása (miniatürizálás, megbízhatóság, stb.) új technológiai módszereket kíván. Az új technológiai módszerek, kidolgozása hozta létre az elektronika új ágát a mikroelektronikát és ezen belül az integrált áramkörök technikáját. Az integrált jelző arra utal, hogy az egy alaplemezen, azonos technológiai lépésekkel egyidejűleg létrehozott alkatrészekből álló áramkör nem bontható alkotóelemeire roncsolás nélkül.

A legkorszerűbb integrált áramkörök jelenleg az ún. monolit (félvezető alapú) integrált áramköri technikával készülnek. Ennek a lényege az, hogy a tranzisztorokat, diódákat, ellenállásokat, kondenzátorokat és az ezeket összekötő vezetékeket egyetlen szilícium kristályon alakítják ki, egymást követő technológiai lépések sorozatával.

A félvezető alapú integrált áramkörök bevezetésekor úgy tűnt, hogy a monolit technika főként digitális áramkörök realizálására alkalmas, elsősorban a nagy alkatrészszórás miatt. A technológia finomításával és újszerű áramkör konstrukcióval azonban olyan tulajdonságokkal rendelkező analóg áramkörök is készíthetők, amelyek a diszkrét elemekből felépülő áramkörökhöz képest is kedvezőbbek.

Bár az integrált áramkörök fejlődését kezdetben főleg a hadiipar és az űrkutatás serkentette, a polgári életben is élvezhetőek az eredményei. A ma technikája, a hétköznapi élet minden eszköze az integrált áramkörökre épül.

A napjainkban használt integrált áramkörök bonyolultságuk és alkatrészeik száma szerint a következő csoportokra oszthatók:

SSI (Small-Scale-Integration): **alacsony** fokú integrált áramkörök; egyszerűbb alapáramköröket tartalmaznak. Az egy tokban levő alkatrészek száma: **50...100**.

MSI (Medium-SI): **közepes** integráltságú áramkörök; bonyolultabb funkciókat elvégző egységeket tartalmaznak. Az egy tokban lévő alkatrészek száma: **500...1000**.

LSI (Large-SI): **magas** integráltságú áramkörök; tokonként egy-egy komplett rendszert alkotnak. Az alkatrészek száma: **1000...10 000**.

ELSI (Extra-LSI): az előbbinél több alkatrészt tartalmaznak, és bonyolultabb rendszereket valósítanak meg.

Az aktív logikai kapuk legkorszerűbb változatai a digitális integrált áramkörök választékaiban szerepelnek. Az egyetlen kristályban - integrálási technológiával - előállított áramkörök az **IC**-k (Integrated Circuit). A diszkrét elemes digitális áramkörökkel szemben sok előnnyel rendelkeznek. Jelentős a miniatűr méret, a sokkal nagyobb működési sebesség, kis disszipációs teljesítmény, valamint a nagy sorozatban való gazdaságos előállítás, tehát az alacsony ár.

A különböző integrált áramköri családok alapelemei a **NEM-ÉS** (**NAND**) vagy **NEM-VAGY** (**NOR**) kapuk. Ezek mellett megtalálhatók a háromműveletes alaplapúk (ÉS-VAGY-NEM), a tárolóelemek (**flip-flop** -ok),valamint a bonyolultabb logikai feladatokra használható **funkcionális áramkörök** (dekódolók, multiplexerek, számlálók, regiszterek stb.).

A különböző felépítésű integrált áramköri családok közül a tantárgyban a **TTL** (Tranzisztor –Tranzisztor - Logika) és a **CMOS** (Complement Metal-Oxid Semiconductor) rendszerű integrált áramkörökkel foglalkozunk. Ezek terjedtek el legjobban, a hazai felhasználásban. A **TTL** rendszert a **TEXAS INSTRUMENTS** cég fejlesztette ki az **SN74...** jelű sorozatával. Ma már több országban is gyártják az eredeti sorozattal csereszabatos (kompatibilis) TTL alapáramköröket. A **CMOS** családokat is számos világcég (pl. RCA) gyártja ma már. Létezik olyan sorozat is a CMOS áramkörök között, amely a TTL áramkörökkel funkció és láb-kompatibilis. Ezek típus-jele: **SN74C...**,amelyben csak a C betű utal a technológiai kivitelre. A többi szám azonos a megfelelő TTL áramkörével.

TTL rendszerű kapuk

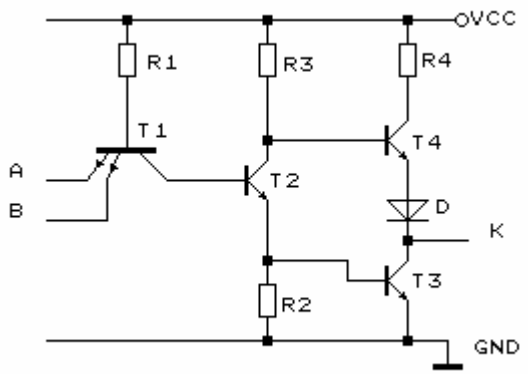
A TTL rendszerű integrált áramköri család pozitív logikai szinttel működik. A legfontosabb feszültség adatok a következők:

	Névleges	**Minimum**	**Maximum**
Tápfeszültség (Ucc)	+5 V	+4,5 V	+7 V
Bemeneti 1 szint (U_{iH})	+3,3 V	+2 V	+5,5 V
Bemeneti 0 szint (U_{iL})	+0,2 V	-1,5 V	+0,8 V
Kimeneti 1 szint (U_{OH})	+3,3 V	+2,4 V	+5,5 V
Kimeneti 0 szint (U_{OL})	+0,2 V	-0,8 V	+0,4 V

A **normál** TTL sorozat alap kapuja a NAND (NEM-ÉS) kapu. A családban kettő, három, négy és nyolc bemenetű NAND kapukat készítenek. A kapuk mind különböző kialakítású - tokozásban kerülnek a kereskedelembe. A leggyakoribb változat az un. **duál in line** tokozás, amely műanyag burkolatú, két oldalt elhelyezkedő kivezetései (lábak) van. Egy ilyen tokban – legtöbbször - több azonos kapu van.

A két-bemenetű NAND kapuból négy db, a három-bemenetűből három db, a négy-bemenetűből kettő db, és a nyolc-bemenetűből pedig egy db van a tokban. Mindezek a kapuk csak a bemenetszámban térnek el. Ezért a továbbiakban csak a két-bemenetű NAND kapu működését elemezzük.

A 3. ábrán látható a két-bemenetű **TTL NAND** kapu kapcsolási vázlata.



3. ábra

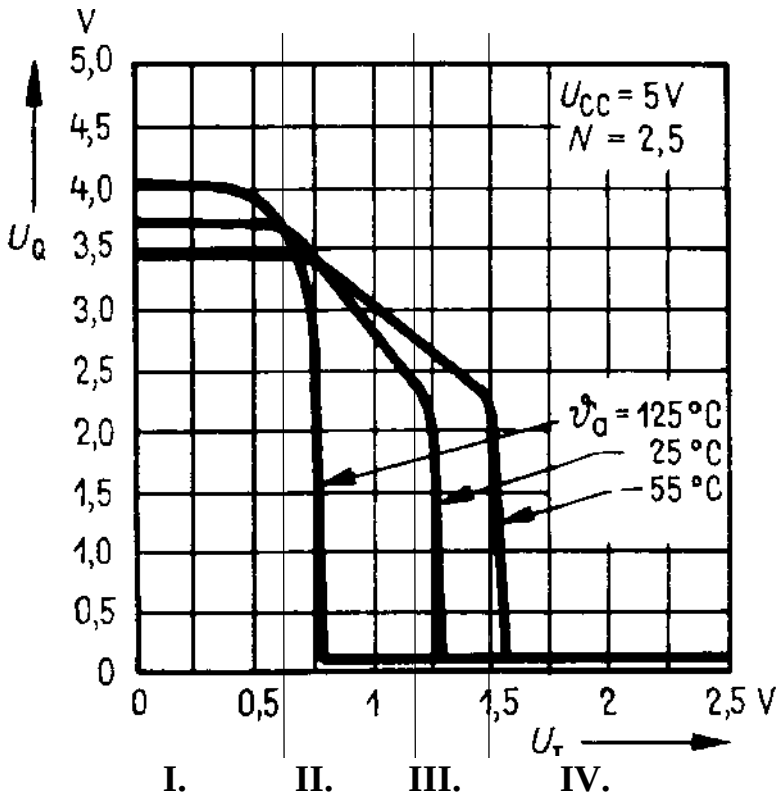
Az áramkör három fő egységre tagolható. Ezek:

- több emitter -es (múlti emitter) bemenet (T1 tranzisztor),
- vezérlő fokozat (T2 tranzisztor);
- teljesítményillesztő kimenet (T3,T4 tranzisztorok, totem-pole).

Az áramkör különválasztható az ÉS, valamint az invertáló funkciót ellátó részre. A T1 múlti emitter -es tranzisztor az **ÉS** kapu, míg a vezérlő és az ellenütemű (totem-pole) kimeneti fokozat feladata együtt az invertálás, valamint a szint-, és teljesítményillesztés.

Az áramkör elemzéséhez bemutatjuk a működést szemléltető, un. átviteli (transzfer karakterisztikát is. Ez a karakterisztika koordinátarendszerben ábrázolja a K kimenet feszültsége (U_{ki}) és a kimeneti szintet meghatározó U_{be} vezérlőfeszültség közötti kapcsolatot. A NAND kapunál mindig a legalacsonyabb szintű bemenő feszültség szabja meg a kimeneti szintet.

Az 4. ábrán látható a TTL rendszerű NAND kapu transzfer karakterisztikája. Az egyes görbék különböző hőmérséklethez tartoznak. A kb. szobahőmérséklethez tartozó karakterisztikát elemezzük.



4. ábra

A vízszintes tengely mentén négy jellemző tartományt különböztethetünk meg. Ezeket római számokkal jelöltük.

Az **I.** szakaszban az áramkör legalább egyik, vagy mindkét bemenetén az U_{be} feszültség a **0 <Ube <0,7 V** feszültségtartományba esik. Ekkor a **T1** tranzisztor **normál telitett** üzemmódban van, mivel bázisa az **R1** ellenálláson keresztül az **Ucc** tápfeszültségre kapcsolódik. A tranzisztor kollektor-feszültsége a maradék feszültséggel (0,1 . . 0,2 V) pozitívabb az emitter feszültségénél. Ez az alacsony szint még **zárva** tartja a **T2** tranzisztort. A **T3** tranzisztor is zárt, mivel nem kap nyitóirányú bázisáramot. A **T4** tranzisztor az **R3** ellenálláson folyó bázisáram hatására **vezet**. A **kimeneti** feszültség (U_{ki}) az **R4** ellenálláson, a nyitott **T4** tranzisztoron és a **D** szinttoló diódán keresztül **magas** pozitív feszültségű lesz, amely a logikai **1 szint**. Jellemző értéke terheletlenül **+3,6 V**.

II. szakasz, amikor az alacsonyabb szintű bemeneti feszültség, a **0,7 < Ube < 1,4 V** tartományba kerül, akkor már a **T2** tranzisztor nyitni kezd, és lineáris üzemmódba

kerül. A tranzisztor kollektor- feszültsége csökken, ezt a **T4** tranzisztor emitter -, és igy az áramkör kimeneti feszültsége is követi. A **T3** tranzisztor még **zárt**, mivel a bemeneti feszültség nem elégséges két **pn** átmenet (T2 és T3 bázis - emitter dióda) nyitó irányú előfeszítéséhez.

A **III. szakasz** az un. **billenési tartomány**. Amikor az alacsonyabb szintű bemenő feszültség eléri a **~ 1,4 V** -os értéket, a **T3** tranzisztor is **kinyit**. Ekkor az áramkör minden tranzisztora vezet és ellenütemű erősítő -ként üzemel. Miután a feszültség-erősítése nagy (A_u> 10) ezért kis bemenő-feszültség változás mellett nagy a kimenőfeszültség változása. A karakterisztika itt meredek.

A IV. szakasz, amikor $\mathbf{U_{be}}$**> 1,4 V**. Ebben a szakaszban a **T2** és **T3** tranzisztor is telítésbe kerül. A kimenő-feszültség logkai 0 szintű lesz, és értéke a telitett **T3** tranzisztor maradékfeszültsége (0,1... 0,2 V) lesz. Amikor a T2 telitetté válik, akkor kollektorán kb. 0,8 ... 0,9 V lesz a feszültség. Ez egyúttal a T4 tranzisztor bázisfeszültsége is. Ez az érték az U_{ki} -nél csak ~ 0,7 V -al pozitívabb, ami nem elég a T4 tranzisztor és a D dióda nyitva tartásához, tehát a **T4 lezár**. Az előzőekből lesz érthető a D szinttoló dióda szerepe. Megnövelte a T4 nyitásához szükséges bázisfeszültséget. Ez teszi biztonságossá annak lezárását is.

Ebben a működési szakaszban a T1 múlti - emitteres tranzisztor kollektor-feszültségét a két nyitott pn átmenet (T2, T3) 1,4 V értéknél megfogja. A tranzisztor bázisfeszültsége sem emelkedik 2,1 V fölé. Ezért a bemeneti feszültségek további növelésekor a bázis - emitter diódák lezárnak s a tranzisztor **inverz telitett** üzemmódba, kerül. Az inverz üzemmódban az emitter és kollektor szerepe felcserélődik. Ilyenkor a bemeneteken nagyon kis áram fog folyni.

Az áramkörök bemenő árama (I_{be}) különböző szintű vezérlésnél eltérő. A **0** szintnél a tipikus áramérték $\mathbf{I_{be0}}$ **= 1 mA**, de a legkedvezőtlenebb esetben is legfeljebb **1,6 mA**. Az **1** szintű vezérlésnél - az inverz üzemmódban működő tranzisztor emitter - árama - $\mathbf{I_{be1}}$ **= 5 µA** (határérték 40 µA).

Ezeket az áramértékeket tekintjük az áram-körkészlet terhelési egységének, amelyek alapján számolhatók a terhelési számok.

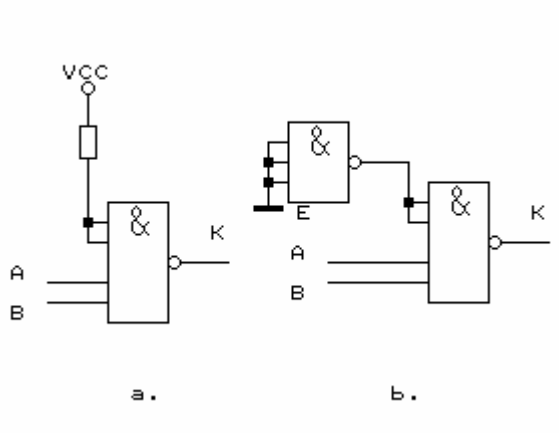
A kapuk terhelhetőségét a terhelési egységre vonatkoztatott terhelési szám, a **fan-out** adja meg. A tipikus fan-out érték 10. Ez abszolút terhelésben - 0 szintű kimenetnél - **16 mA**, 1 szintű kimenetnél **400 µA** határterhelést ad. Az áramkörcsalád újabb típusainál, szintnél **µA** a határérték, amely **20** egységterhelésnek felel meg.

A NAND kapuk vezérlésekor a kimeneti feszültség 0 -> 1, ill. 1 -> 0 irányú szintváltozósa különböző idejű késleltetéssel következik be. A lefutási késés $\mathbf{t_f}$ = 7-8 ns, a felfutási késés pedig $\mathbf{t_u}$ = 11 -13 ns. Az átlagos jelterjedési idő $\mathbf{t_{pd}}$ **= 10 ns**. Az átkapcsolási idők függenek a terhelés nagyságától, jellegétől, a tápfeszültségtől, valamint a hőmérséklettől. A tápfeszültség és a hőmérsékletfüggés általában elhanyagolható. A terhelésváltozós késleltető hatását - az áramkörök felhasználásakor - már figyelembe kell venni. A terhelés hatását a katalógusokban adják meg.

A késleltetéseket még növeli az is, ha a bemenetek közül egyet vagy többet nem kötünk sehova. (Ez a működést logikailag nem változtatja meg.) A bemeneti T1 jelű múlti emitteres tranzisztor árammentes bemeneteinek kapacitása 0,5 ... 1,5 pF értékű, ami üresen hagyott bemenetenként 1 ns - al növeli a késleltetési időt.

A járulékos késleltetés megszűnik, ha a fel nem használt bemeneteket egy vezérelt bemenettel kötjük össze. Ez a megoldás 1 szintű vezérlésnél növeli a bemenő áramot s

így csak a meghajtó áramkör terhelhetőségi határáig használható. Ezért előnyösek az 1 szintnél N = 20 terhelhetőségű kapuk. Ha a terhelési viszonyok nem engedik meg a bemenetek összekötését, akkor az 5. ábra szerint kell a fel nem használt bemeneteket R = 1 ... 5 kΩ közötti értékű ellenállással a tápfeszültségre (a. ábra) vagy egy szabad NAND kapu (inverter) 1 szintű kimenetéhez csatlakoztatni (b. ábra).

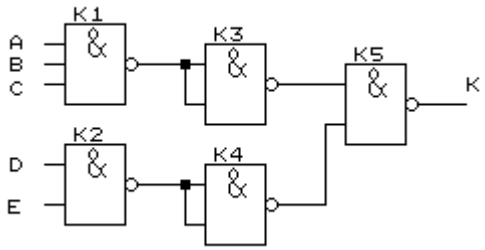


5. ábra

Késleltetés-növekedés e megoldásoknál is van, de értéke bemenetenként csak 0,5 ns.

A logkai kapuk tápáram felvétele (Icc) is változik a különböző vezérlési állapotokban. Kimeneti **0** szintnél a kapu áramfelvétele ~**3 mA**, az **1** szintnél pedig ~**1 mA**. (Ezek az értékek terhelet1enül érvényesek.) A 0 -> 1 átkapcsolások során az áramfelvétel átmenetileg megnövekszik, mert ilyenkor az ellenütemű kimenet mindkét tranzisztora (T3 és T4) rövid ideig együtt vezet.

A megismert NAND kapuk, felhasználásánál előfordulhat olyan eset is, hogy pl. nagyobb bemenetszámot kell megvalósítanunk, mint amilyen tokok rendelkezésünkre állnak. Erre példa a 6. ábra szerinti kapcsolás.



6. ábra

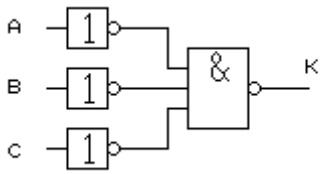
Itt öt bemenetű NAND kapcsolatot valósítottunk meg két és három bemenetű kapukkal. A logkai vázlat alapján fel írható a függvény-kapcsolat.

$$\mathbf{K} = \overline{\overline{(ABC)}\,\overline{(DE)}} = \overline{ABCDE}$$

A K1 jelű három-bemenetű kapu az első zárójeles mennyiség első tagadását, míg a második tagadást a K3 jelű kapu végzi. (A NAND kapu két bemenetét összekötve

invertert kapunk). A második zárójeles mennyiséget - az előzőekhez hasonlóan - a K2 és K4 jelű kapuk képezik. E két mennyiség közötti ÉS -NEM műveletet hozza létre a K5 jelű kapu. A megoldáshoz 1 tok kellett a két-bemenetű változatból (K2, K3, K4, K5) és egy a három-bemenetű kapukat tartalmazó tokból (K1).

Csak NAND kapuk segítségével ÉS-VAGY típusú logikai hálózat is megvalósítható. Ennek megértéséhez először nézzük meg, hogyan hozhatunk létre NAND kapuval VAGY műveletet. Az 7.ábra szerinti logikai vázlatnak megfelelően a NAND kapu bemeneteire az A,B,C változók tagadottjai jutnak.



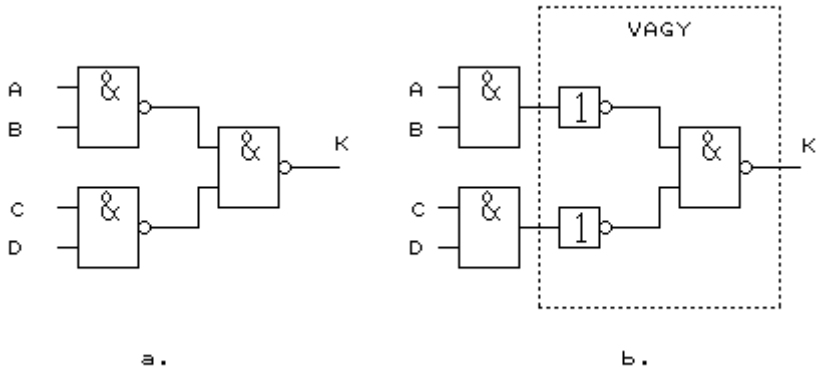
7. ábra

Felírva a logikai egyenletet a

$$\mathbf{K} = \overline{\overline{\mathbf{A}}\,\overline{\mathbf{B}}\,\overline{\mathbf{C}}} = \mathbf{A} + \mathbf{B} + \mathbf{C}$$

összefüggést kapjuk. Összefoglalva mondhatjuk, hogy a NAND kapu a bemeneteire jutó változók tagadottjainak VAGY kapcsolatát képezi.

Az 8.a.ábra szerinti logkai vázlatot felrajzolhatjuk a b. ábra szerint is, ha külön tekintjük a kapu invertereit. A szaggatott vonallal körülhatárolt részlet bemenetei között VAGY műveletet végez. Ezen két bemenet pedig **AB**, valamint **CD** értékű.



8. ábra

Ezek alapján a megvalósított függvénykapcsolatunk

$$\mathbf{K} = \mathbf{AB} + \mathbf{CD}$$

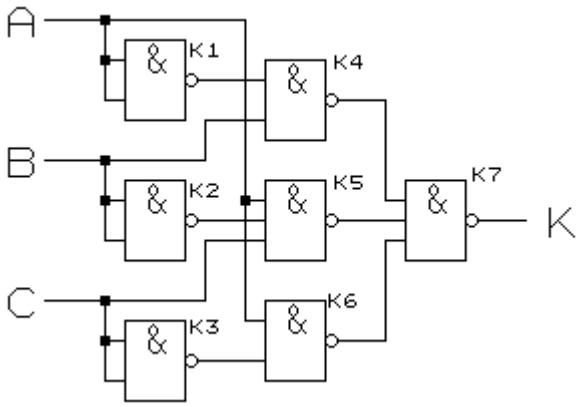
alakú függvénnyel adható meg.

A feladatot fordítva fogalmazva: az ÉS-VAGY alakú logikai függvény csak NAND kapukkal is megépíthető.

Példaként rajzoljuk meg a

$$\mathbf{K} = \overline{\mathbf{A}}\mathbf{B} + \mathbf{A}\overline{\mathbf{B}}\mathbf{C} + \mathbf{A}\overline{\mathbf{C}}$$

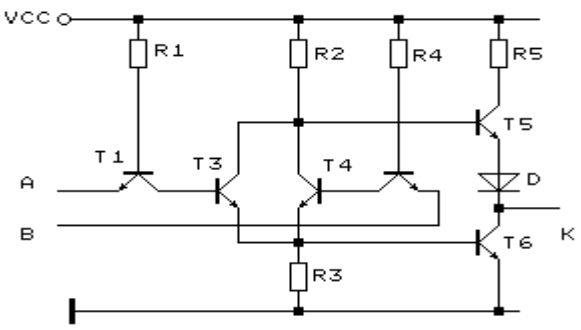
logikai függvénykapcsolatot létrehozó hálózat logikai vázlatát! A tagadásokat is NAND kapukkal állítsuk elő. A megoldást mutatja a 9. ábra.



9. ábra

A logikai hálózat két tokkal építhető meg, ui. négy két-bemenetű kaput (K1,K2, K4,K6) és 3 három-bemenetűt (K3,K5,K7) használtunk. Ezek pedig az SN 7400 (négy két-bemenetű NAND kapu) és az SN7410 (három darab három-bemenetű NAND kapu) típusú IC tokok.

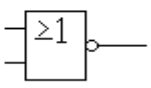
Az SN sorozatban - az eddigiekben tárgyalt NAND kapuk mellett - NEM-VAGY (NOR) kapu csak két-bemenetű változatban van. A kapu kapcsolási vázlatát mutatja az 10. ábra.



a.

A	B	K
0	0	1
0	1	0
1	0	0
1	1	0

b.

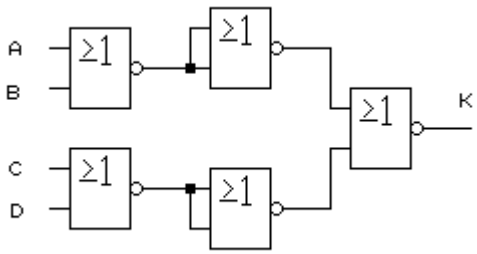


c.

10. ábra

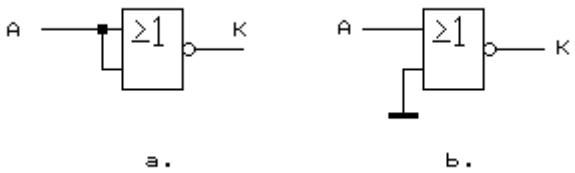
Az áramkör működése a következő. A kimenet logikai 1 szintű, ha a kimenő (totem-pole) fokozatot meghajtó **T3**, és**T4** tranzisztorok zártak. Ekkor a **T5** tranzisztor az **R2** ellenálláson keresztül telítésbe kerül, s ugyanakkor a **T6** tranzisztor lezár. A T3 és T4 tranzisztorok akkor zárnak, ha mind az **A**, mind pedig a **B** bemeneten logikai **0 szint** van. Ha a bemenetek valamelyike vagy mindkettő **1 szintű** vezérlést kap, akkor a bemeneti tranzisztor (ok) (T1 vagy T2, vagy mindkettő) inverz üzemmódban működik és a meghajtó tranzisztorok (T3,T4) közül az egyik vagy mindkettő nyit. A három kombináció mindegyikében a kimenet **T6** tranzisztora nyit s így kollektorán - a K kimeneten - logkai 0 szint lesz. A fenti működést írja le a b. ábra szerinti igazságtáblázat, amely a NOR függvénykapcsolatot adja. A kapu szimbolikus jele a c. ábra szerinti.

Több logikai változó NEM-VAGY kapcsolatát - több két-bemenetű kapuból - az 11. ábra szerinti kapcsolásban lehet megvalósítani.



11. ábra

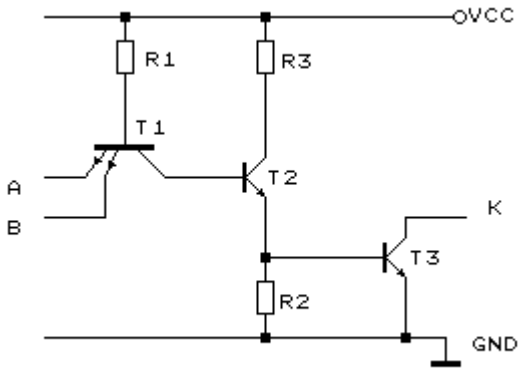
Az **inverter** áramkör - amely a logikai tagadás műveletét valósítja meg - tulajdonképpen egy-bemenetű kapu. Több bemenetű kapukból a bemenetek összekötésével, vagy egy bemenet használatával alakítható ki. Erre már a NAND kapu elemzésénél kitértünk. NOR kapuból az 12. ábra szerinti kapcsol sokkal alakítható ki inverter. A nem használt bemenetet - a logikai feltételekből adódóan - 0 szintre kell kötni.



12. ábra

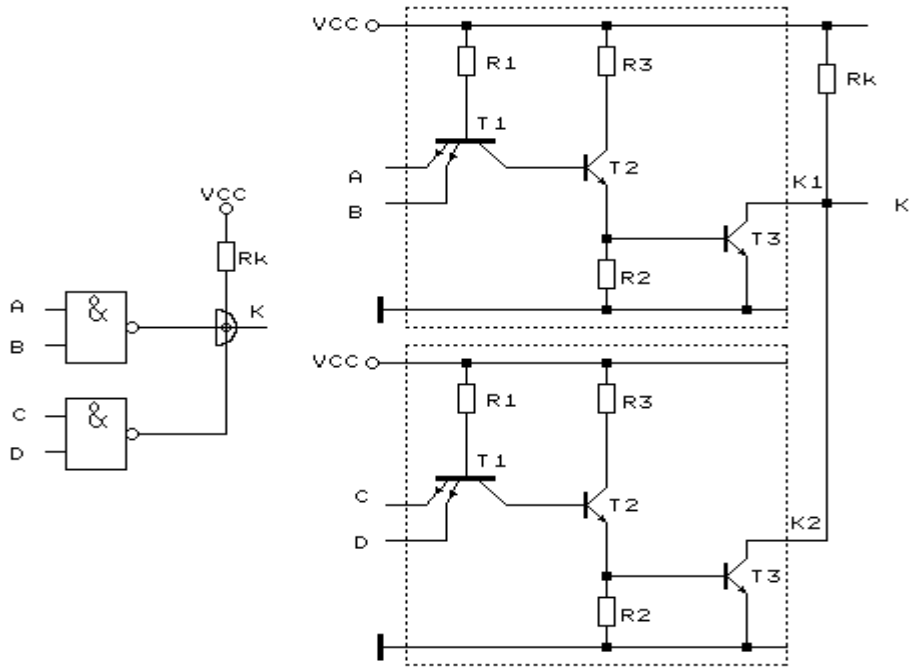
Az SN áramkör családban csak **inverter** -eket tartalmazó tokok is készülnek (6 db inverter 1 tokban). Ezek tulajdonképpen egy-bemenetű NAND kapunak tekinthetőek. Az inverterek működése a már leírtak alapján elemezhető.

Az áramkörcsalád speciális kapui a **nyitott kollektoros** (open-collector) változatok. Az ezekben levő kimenő fokozat egyetlen tranzisztor, amelynek szabadon hagyott kollektora van kivezetve. Ilyen kimenettel két-bemenetű NAND kapuk és inverterek készülnek. A két-bemenetű NAND áramköri kapcsolását az 13. ábra mutatja. A T3 tranzisztor munka-ellenállását kívülről kell bekötni.



13. ábra

A nyitott kollektoros NAND kapukkal többszintű logikai függvény is megvalósítható. Az un. **huzalozott ÉS** kapcsolatot kapjuk, ha két vagy több nyitott kollektoros NAND kapukimeneteit közös R_T munkaellenállásra kapcsoljuk. Négy bemeneti változóra az áramköri kapcsolást az 14.a.ábra mutatja. A kimeneten csak akkor lehet logkai 1 szint, ha mindkét NAND kapu kimenete 1 szintű, vagyis a kimeneti tranzisztorok zártak. Ez az egyes kapuk által megvalósított függvények ÉS kapcsolatát jelenti. A kapcsolás szimbolikus jelölését a b. ábrán láthatjuk.

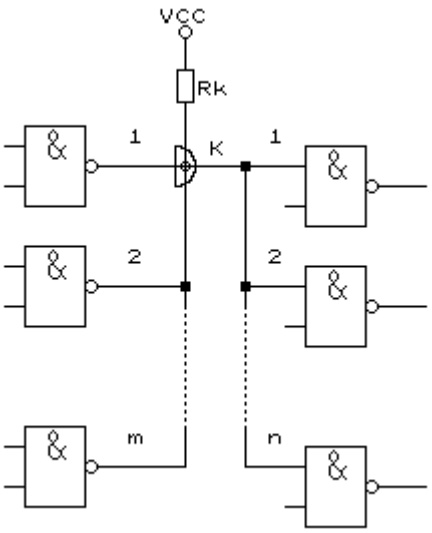


b. a.

14. ábra

A huzalozott kapcsolásokban alkalmazott külső munkaellenállás értékének megválasztásánál különböző feltételeknek kell teljesülnie.

Tételezzük fel, hogy m db nyitott kollektoros kapu kimenete van összekötve közös R_K munkaellenálláshoz. A K kimenet pedig n db további kapubemenetet vezérel az 15. ábra szerint.



15. ábra

Az R_K meghatározása a következők szerint végezhető:

1. A kimenet 0 szintű értékénél a legkritikusabb eset az, amikor egyetlen kimeneti tranzisztor vezet. Az áram nem haladhatja meg a tranzisztor határ-áramát $\mathbf{I}_{cmax}$ -ot. Ezen a tranzisztoron folyik keresztül munkaellenállás árama, valamint a kimenet által vezérelt n db kapu bemeneti árama (I_{be0}). Ezek alapján teljesülnie kell a következő egyenlőtlenségnek.

$$\frac{\mathbf{Ucc - U_{01}}}{\mathbf{R_{KMIN}}} + \mathbf{n(-I_{be0})} \pounds \mathbf{I_{c\,max}}$$

   Az I_{be0} értékénél a legkedvezőtlenebb érték - az 1,6 mA - veendő figyelembe.

2. Lezárt kimeneti tranzisztornál, vagyis 1 szintű kimenetnél az R_T ellenálláson folyik keresztül az m számú összekötött bemenet kollektor visszárama (I_{C0}) és az n számú vezérelt bemenet 1 szintjéhez tartozó árama (I_{be1}). Az összáram hatására sem csökkenhet a logikai 1 szint a megengedett alsó érték (U_{12}) alá. Ezt leíró egyenlőtlenség:

$$\mathbf{Ucc - R_{K\,max}(mI_{C0} + nI_{be0})} \;^{3}\; \mathbf{U_{12}}$$

Az I_{C0} és az I_{bel} értékeknél az alkalmazott áramkör páramétereinek legkedvezőtlenebb szélsőértékeit kell figyelembe venni. (A tápfeszültség Ucc értékét állandónak tekinthetjük.) Az előző egyenlőtlenségekből számolható ki az R_T ellenállás névleges értéke és megengedett tűrése.

A nyitott kollektoros áramkörök külön csoportját alkotják azok a változatok, amelyeknél a kimeneti tranzisztor 15 ... 30 V-os záró-feszültségű, ill. 40 mA áramterhelhetőség. Ezek az inverterek, ill. csak kapcsoló erősítők meghajtó áramkörökként, vagy magasabb logikai szintű és TTL rendszer illesztésére használhatók.

CMOS rendszerű kapuk

A digitális integrált áramkörök technológiai és áramköri fejlesztésében a 80-as évtizedben terjedt el a térvezérelt tranzisztorok (**FET**) szélesebb körű alkalmazása. A digitális áramkörcsaládok kialakításban szigetelt vezérlőelektródájú **MOS-FET** (Metal Oxide Semiconductor - Field Effect Transistor), vagy röviden **MOS** tranzisztorokat, használnak. Ezekben az áramkörökben nagy elemürüség érhető el, mert egy MOS tranzisztor helyigénye lényegesen kisebb, mint a bipoláris tranzisztoré.

A MOS integrált áramkör bemeneti ellenállása közel végtelen, ezért nagy egyenáramú (dc) fan-out érhető el. Gyakorlatilag a fan-out értékét csak a működési sebesség korlátozza .A működési sebesség általában alacsonyabb, mint a bipoláris tranzisztorokból kialakított IC-ké. (Mai áramkörök már elérik a TTL sebességét). Ez alapvetően abból adódik, hogy a MOS - elemek nagy impedanciája mellett a szórt és terhelő kapacitások hatása számottevőbb.

A MOS integrált áramkörök két nagy csoportba sorolhatók:

- MOS LSI és a
- CMOS áramkörökre.

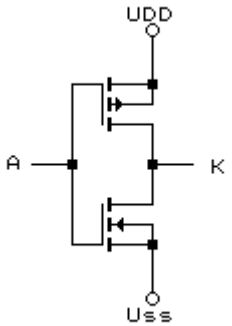
Az azonos típusú MOS tranzisztorokkal az alacsony integráltságú (SSI) digitális áramkörök (kapuk, flip-flopok stb.), illetve a közepes integráltságú (MSI) funkcionális egységek (számlálók, regiszterek stb.) gyártása gazdaságtalan. Ezért elsősorban a nagy

integráltságú (LSI) áramkörök (mikroprocesszorok, memóriák stb.) készülnek ilyen megoldásban.

A **komplementer** - **p** és **n csatornás** - MOS tranzisztorokat együttesen alkalmazva, készülnek a **CMOS** vagy más néven **COS-MOS** integrált áramkörök. A CMOS kialakításban kiváló tulajdonságú **SSI** és **MSI** digitális áramkörök kerültek forgalomba. (Kisebb volumenben mikroprocesszorok és memóriák is készülnek CMOS technológiával.) A fejezetben a CMOS kapuk alapvető felépítésével, jellemzőivel foglakozunk.

CMOS kapuk

A CMOS digitális áramkörök legegyszerűbb eleme a két komplementer tranzisztorból álló **inverter** (16. ábra). A két sorba kötött **T1** (n csatornás) és **T2** (p csatornás) **növekményes** típusú tranzisztor közösített vezérlőelektródája -**GATE**- az áramkör bemenete (A). A kimenet (K) az összekötött ”kollektorokhoz”- **DRAIN**- (nyelő) csatlakozik. A tranzisztorok ”emitterei” -**SOURCE**- (forrás) a tápfeszültség két pontjához csatlakozik.



16. ábra

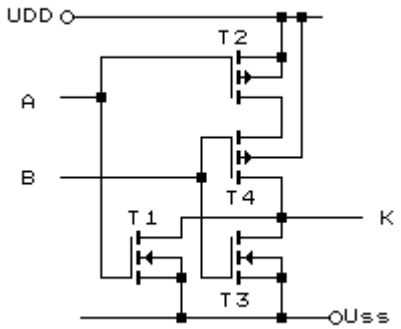
Az együttesen vezérelt komplementer tranzisztorok közül minden vezérlési állapotban (H agy L szintnél) csak az egyik vezet. Az $\mathbf{U_{SS}}$ szintű bemenő jelnél az n csatornás (T1) tranzisztor zár, mert a tranzisztorra jutó $\mathbf{U_{GS2}}$ vezérlőfeszültség meghaladja a küszöbfeszültséget. A K kimenet - a vezető T2 tranzisztor kis csatorna-ellenállásán keresztül - az $\mathbf{U_{DD}}$ tápfeszültség pontra kapcsolódik, s ezért a feszültsége (U_K) közel azonos lesz azzal. Az $\mathbf{U_{DD}}$ szintű vezérlésnél a tranzisztorok állapota felcserélődik, s ezért a kimeneti feszültségszint jó közelítéssel az $\mathbf{U_{SS}}$ értékével fog megegyezni. A logikai szintek névleges értéknek az $\mathbf{U_{SS}}$ -t ill. az $\mathbf{U_{DD}}$ -t választva, az áramkör a logikai tagadást valósítja meg. Jelentős előny, hogy mind pozitív, mind pedig negatív logikai rendszerben alkalmazható ugyanez az áramkör inverter -ként.

Az áramkör mindössze két aktív áramköri elemből áll. Mindkét logikai szintnél azonos a kimeneti ellenállás, és ezért a zavarvédettség is egyforma. A vezető tranzisztorok csatorna-ellenállása kisebb **1 kΩ** -nál. Jellemző - megengedett - kimeneti áram **0,5 mA**.

A bemenet feszültség-vezérelt, s csupán az átkapcsolásoknál - az elektróda kapacitások átpolarizálásához - kell **nA** nagyságú áramot szolgáltatnia a meghajtó áramkörnek. Ez az előnyös tulajdonság viszont néhány hátránnyal is jár. A vezérlőelektródák kapacitásai csökkentik a kapcsolási sebességet. A késleltetés miatt a két tranzisztor átkapcsolása között átfedés jöhet létre. Ennek következtében - amikor mindkét

tranzisztor vezet - átmenetileg megnő a tápáram felvétel. Ennek mértéke a táp-feszültség növelésével arányosan növekszik. (A táp-feszültség $\mathbf{U_{DD}}$ - $\mathbf{U_{SS}}$ 3 és 15 V, néhány típusnál 30 V közötti tetszőleges érték lehet.). Nagyon jelentős hátrány, hogy a szabadon hagyott bemenet kapacitása statikusan olyan mértékben feltöltődhet, hogy tönkremehet az áramkör. Ez viszont csak a korábbi típusoknál volt így. Ma már az áramkörökön belüli Zener diódás védőkapcsolásokkal gyártják az áramköröket.

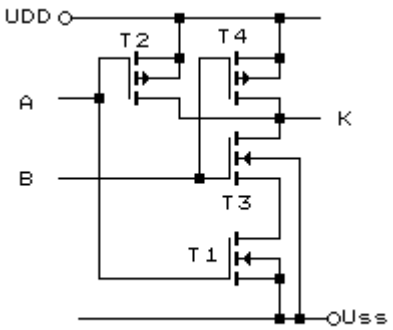
Komplementer MOS tranzisztorok vegyes kapcsolásával VAGY-NEM (**NOR**), ÉS-NEM (**NAND**), valamint összetett logikai műveleteket megvalósító kapukat is készítenek. A 17.ábra szerinti kapcsolású áramkör működése a következő. Amikor a bemenetek közül (A, B) legalább az egyik $\mathbf{U_{DD}}$ szintű vezérlést kap, akkor az ide kapcsolódó n - csatornás tranzisztorok (T1,T2) közül az egyik, vagy mindkettő vezet. A p - csatornás tranzisztorok (T3,T4) közül az egyik, vagy mindkettő zárt.



17.ábra

A K kimenet - a vezető tranzisztoron keresztül - az $\mathbf{U_{SS}}$ pontra kapcsolódik és feszültsége közel ezzel az értékkel lesz egyenlő. A kimeneti feszültség (U_K) csak akkor veszi fel az $\mathbf{U_{DD}}$ értéket, ha mindkét bemenet $\mathbf{U_{SS}}$ szintű vezérlést kap. Ha **pozitív logikai** szintet veszünk alapul, akkor az **1**-szint az $\mathbf{U_{DD}}$ és a **0**-szint pedig az $\mathbf{U_{SS}}$. Az áramkör ilyenkor VAGY-NEM (NOR) kapu. Negatív logikai rendszerben - az értelemszerű fordított szintválasztás eredményeként - az áramkör ÉS-NEM (NAND) kapu.

A 18. ábra szerinti áramkör is az előzőekhez hasonlóan elemezhető.



18. ábra

Az áramkör pozitív logikai rendszerben NAND, negatív logikai rendszerben pedig NOR kapu.

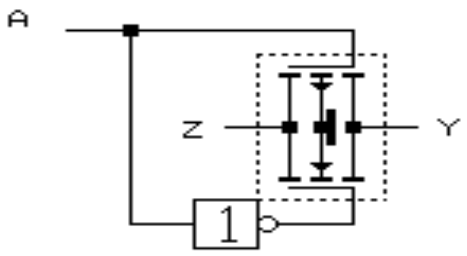
A CMOS áramkörökben kialakított növekményes MOS tranzisztorok küszöbfeszültsége Us = 2V. A vezérlő-elektródára megengedett feszültség (U_{GS}) maximuma 15-20 V. Az áramkör ezért használható széles tápfeszültség tartományban. Ez az áramkörcsaládok legtöbbjénél 3-15 V lehet.

Az áramkörök nyugalmi tápáram-felvétele nagyon kicsi, és a disszipáció is 10 nW nagyságrendű. A működési frekvencia növekedésével a disszipáció hatványozottan emelkedik.

A CMOS áramkörök korábbi változataiban az átlagos jelterjedési idő t_{pd}=50 ns. A legújabb fejlesztések eredményeként már léteznek a normál TTL sorozat késleltetési idejét megközelítő CMOS áramkörök is.

CMOS kapcsoló

Térvezérelt komplementer tranzisztor-párból, digitális jellel vezérelt kétirányú jelátvitelre alkalmas (ún. **bilaterális**), elektronikus kapcsoló alakítható ki. Áramkörileg két MOS-FET tranzisztorból áll (19. ábra), melyek közül a **T1 n** csatornás és **T2 p** csatornás. A két tranzisztor inverter - en keresztül ellenütemben kap vezérlést. Ha a **Z** pontot (közösített drain) tekintjük a bemenetnek, és az **Y** (közösített source) a kimenet, akkor a működés a következő. (Az U_{be} bemenő feszültség 0..+Up érték közötti lehet.) Az A = 0 szintű vezérlésnél mindkét tranzisztor zárt, mivel az n - csatornás T1 tranzisztor vezérlőfeszültsége a küszöbfeszültségnél negatívabb, ill. a T2 p - csatornás tranzisztornál pedig pozitívabb. Ezért a Z és Y pont között nagy impedancia mérhető. Az A = 1 szintű vezérlésnél - az Uz értékétől függően - legalább az egyik tranzisztor vezet, és így a Z és Y között kis impedanciájú a kapcsolat .A MOS tranzisztorok szimmetrikusak, ezért a source és drain felcserélhető. Ez az adott kapcsolásban a be-; és a kimenet (Z, Y) felcserélését is lehetővé teszi. Az integrált technológiával kialakított önálló bilaterális kapcsoló-elem az átvivő tranzisztorok mellett az invertert is tartalmazza.



19. ábra

3. ÖSSZETETT FELADATOKAT MEGVALOSITÓ LOGIKAI ÁRAMKÖRÖK

Az előző fejezetben megismerkedtünk azokkal a logikai kapukkal, melyek minden digitális berendezésben - mint alapáramkörök - megtalálhatók. A kapukka1 elvileg minden Ún. emlékezet nélküli digitális áramkör felépíthető. Ugyanakkor a legkülönbözőbb rendeltetésű hálózatokban megtalálhatók olyan nagyobb funkciókat ellátó egységek is, amelyek kapukból is megépíthetők, de gyakori használatúk miatt célszerű önálló áramkörként is gyártani. Ezeket általában **rendszertechnikai** (funkcionális) áramköröknek nevezzük.

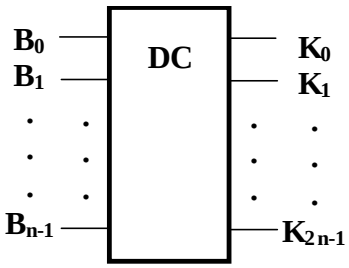
Ilyen funkcionális egységek a következők:

- kódolók, dekódolók;
- adatelosztók;
- adatkiválasztók;
- aritmetikai áramkörök.

E fejezetben csak e funkcionális egységek áramköri kialakításával és működésével foglalkozunk.

3.1. Dekódoló áramkörök

A dekódoló olyan több bemenetű kombinációs áramkör, amely a bináris vagy BCD kódból állit elő un. **1 az N - ből** kódot. Ez a kód azt jelenti hogy az N db bitből mindig csak egy - a bináris vagy a BCD kód által meghatározott - bit aktív logikai értékű. Az aktív logikai érték lehet 1, akkor a többi bit 0, vagy fordítva. A dekódoló blokkvázlatát a 1. ábra szemlélteti. A $\mathbf{B_0 \ldots B_{n-1}}$ jelű bemenetekhez csatlakozik az **n** bites átalakítandó kód (BCD kódnál n=4). A $\mathbf{K_0 \ldots K_{2n-1}}$ jelű ki-meneteken kapjuk az **1 az N-ből** kódot. (Bináris kódolású bemenetnél maximálisan $\mathbf{2^n}$ számú kimenet lehet.)



1. ábra

3.1.1. Bináris dekódoló

A 3 bites bináris kód (3-ról 8-ra) dekódolását végző kombinációs hálózat igazságtáblázata - logikai 1 szintű aktív kimenetet választva - a.2.a.ábrán a logikai vázlata pedig a 2.b ábrán látható.

A bináris kódban általánosan az A,B,C,D betűkkel jelöljük az egyes helyi értékeket, az $\mathbf{A \to 2^0, B \to 2^1, C \to 2^2, D \to 2^3, \ldots}$ stb. súlyozás választásával. Az igazságtáblázatból felírhatjuk a következő logikai függvényeket:

$$\mathbf{K_0 = \overline{A}\,\overline{B}\,\overline{C}} \qquad \mathbf{K_1 = A\,\overline{B}\,\overline{C}}$$

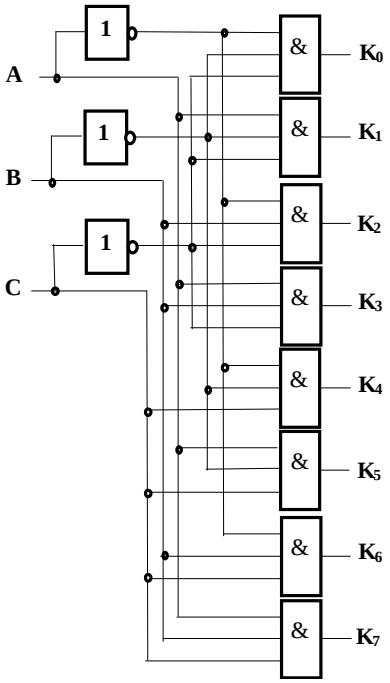
$$\mathbf{K_2 = \overline{A}\,B\,\overline{C}} \qquad \mathbf{K_3 = A\,B\,\overline{C}}$$

$$K_4 = \overline{A}\overline{B}C \qquad K_5 = A\overline{B}C$$

$$K_6 = \overline{A}BC \qquad K_7 = ABC$$

C	B	A	K_0	K_1	K_2	K_3	K_4	K_5	K_6	K_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

a.



b.

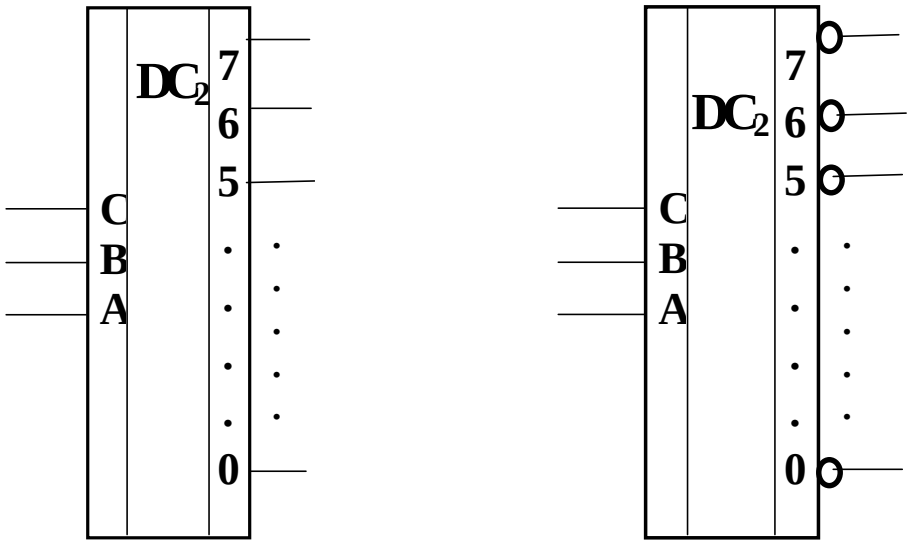
2. ábra

Az áramkör a minterm -eket megvalósító **ÉS** kapukból és a változók tagadott értékeit előállító **inverter** -ekből áll.

A 0 értékű aktív kimeneti logikai szintnél az előző összefüggések tagadásával kapjuk a logikai egyenleteket. Az áramkör pedig **NAND** kapukkal épül fel.

A dekódolandó bináris kód bitjeinek növelésével - az előbbiekben elemzett mindkét változatnál - a felhasznált kapuk és azok bemeneteinek száma növekszik.

A dekódolók szimbolikus jelölése látható a 3. ábrán. A 0- val aktív kimenetet a karika (tagadás) jelzi a b. ábrán.



a. b.

3. ábra

3.1.2. BCD dekódoló

A digitális áramköri készletek többségében van BCD decimális dekódoló áramkör. A négy bites **BCD** kód (8 4 2 1 súlyozású), és a tíz decimális számértéket a bináris kód első tíz (K0 . . . K9) kombinációjához rendeli.

A BCD dekódoló áramköri kialakításánál egyszerűsítésre felhasználhatók a kódban elő nem forduló (K10 . . . K15) kombinációk is. A legegyszerűbb felépítésű BCD dekóder logikai függvényei a következők:

$$K_0 = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} \qquad K_1 = A\overline{B}\,\overline{C}\,\overline{D}$$

$$K_2 = \overline{A}B\overline{C} \qquad K_3 = AB\overline{C}$$

$$K_4 = \overline{A}\,\overline{B}C \qquad K_5 = A\overline{B}C$$

$$K_6 = \overline{A}BC \qquad K_7 = ABC$$

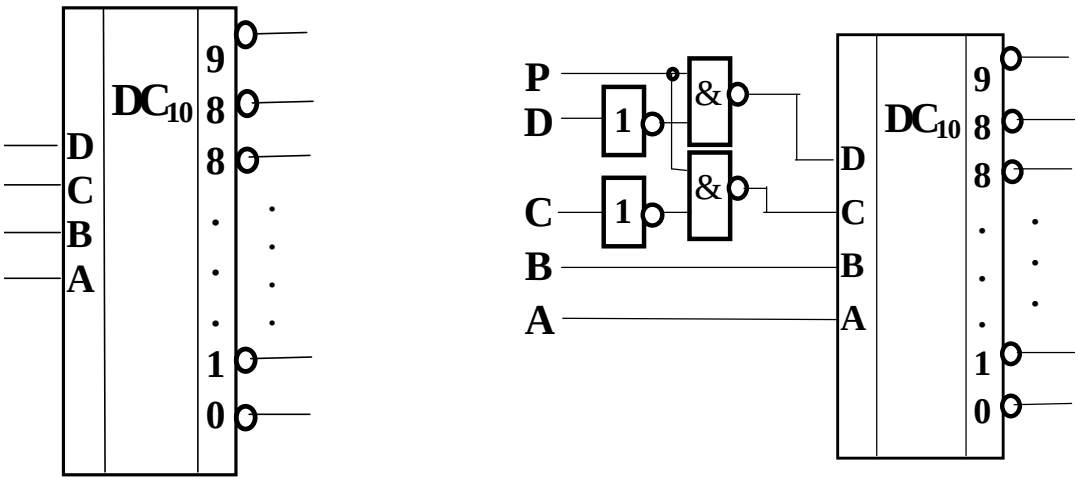
$$K_6 = \overline{A}D \qquad K_9 = AD$$

Az ilyen megoldás nem teljesen dekódoltnak nevezik, mivel a bemenetre adott tiltott kombinációk is aktiválhatnak kimenetet, (esetleg kimeneteket). Pl.: a DCBA kombináció hatására a K7 kimenet lesz aktív - 1 szintű.

3.1.3. Dekódolók alkalmazása

A dekódolók leggyakoribb felhasználása a digitális berendezéssel végzett mérés, műveletvégzés eredményének **megjelenítésénél** van. A jelfeldolgozás **bináris** vagy **BCD** kódban történik. A **dekódoló** alakítja át az eredményt **decimális** kóddá. Az áramkörök egy részét a dekódolás funkciója mellett kijelzők meghajtására alkalmas teljesítményfokozattal is ellátják. Ezeket nevezzük dekódoló-meghajtóknak (drivereknek).

A 4.a.ábrán az SN 7442 típusú dekódoló szimbolikus jele látható. Az áramkör **aktív kimenete** a logikai **0** szint. Ez a típus teljesen dekódolt áramkör, ami azt jelenti, hogy a tiltott bemeneti kód esetén egyik kimenet sem lesz aktív. Ezért áramköri kiegészítésekkel megoldható a külső jellel (P) való tiltás. Ennek logikai vázlata látható a 4.b.ábrán. **P=1**-nél a dekódolás **engedélyezett**, míg **P=0** vezérlésnél a dekódoló C,D bemeneteire logikai 1 szint kerül, s ez már - a bemeneti kódtól függetlenül - **tiltott** kombinációt ad, így egyetlen kimeneten sem lesz aktív jel.



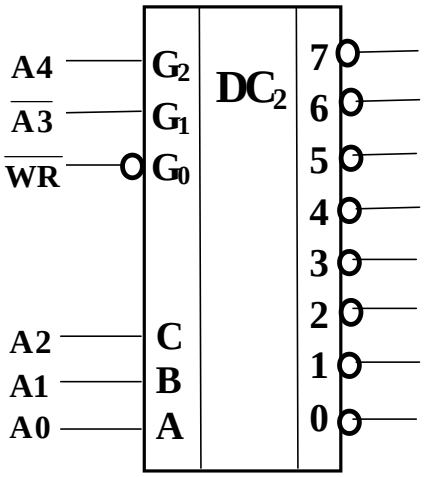
a. b.

4. ábra

Az integrált áramkörök alkalmazásának ”hőskorából” való az **SN 74141** típus, amely elsősorban a **Nixie - cső** (gáztöltésű számkijelző cső) vezérlésére közvetlenül alkalmas.

Az **SN 7445** típusú dekódoló-meghajtó nyitott kollektoros kialakítású, amelynek végtranzisztorai Ic=80 mA - el terhelhetők. (A zárófeszültség megengedett értéke 30 V.) Az áramkör **izzólámpák**, fénykibocsátó diódák (**LED**), **relék** meghajtására közvetlenül alkalmazható. TTL rendszeren belül közvetlen dekódolásra is felhasználhatjuk.

A mikroprocesszoros rendszerekben dekódolókat használnak az egyes memória-, illetve periféria IC-k kiválasztásához, az un. **címdekódolás** megvalósításával. Az 5. ábra szerinti kapcsolásban használt 74138 típusú (3-ról 8-ra) dekódoló a bemeneteire érkező $\mathbf{A_0}$ **…** $\mathbf{A_4}$ jelű cím bitek, valamint a $\overline{\mathbf{WR}}$ írást vezérlő jel hatására csak a megfelelő című – az aktív kimenethez csatlakozó - egységbe engedélyez adat írást. A kimenetek logikai egyenleteit is felírtuk.

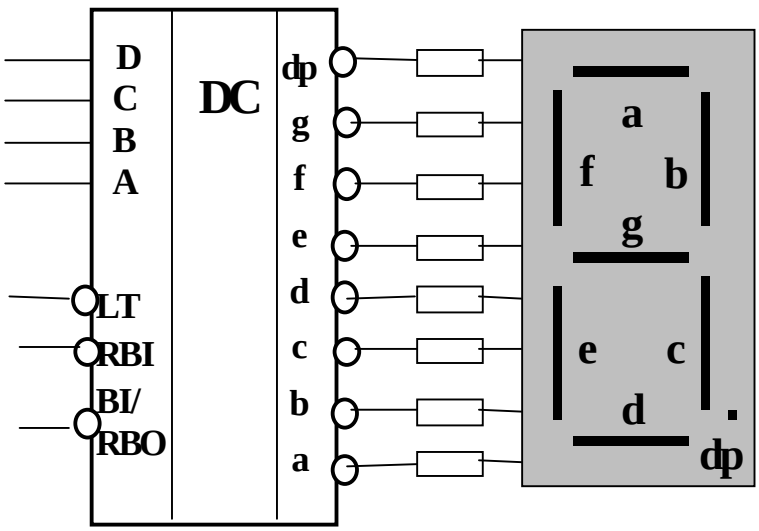


$$(A4 * \overline{A3})(A2 * A1 * A0)\overline{WR}$$
$$(A4 * \overline{A3})(A2 * A1 * \overline{A0})\overline{WR}$$
$$(A4 * \overline{A3})(A2 * \overline{A1} * A0)\overline{WR}$$
$$(A4 * \overline{A3})(A2 * \overline{A1} * \overline{A0})\overline{WR}$$
$$(A4 * \overline{A3})(\overline{A2} * A1 * A0)\overline{WR}$$
$$(A4 * \overline{A3})(\overline{A2} * A1 * \overline{A0})\overline{WR}$$
$$(A4 * \overline{A3})(\overline{A2} * \overline{A1} * A0)\overline{WR}$$
$$(A4 * \overline{A3})(\overline{A2} * \overline{A1} * \overline{A0})\overline{WR}$$

5. ábra.

Megjegyzés: Az ábra csak a periféria elemek engedélyező jeleit előállító egységet szemlélteti. A periféria egységek, és az adatvonalak itt nem szerepelnek.

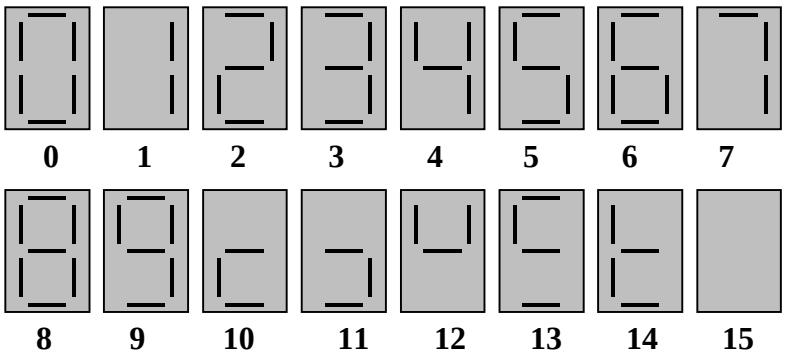
Az integrált áramköri **dekódoló-meghajtók** egy külön csoportja a **7 szegmenses** kijelzők vezérlésére használható. Miután napjainkban már ezek a kijelzők - mind LED -es, mind pedig folyadékkristályos (LCD) kialakításban - a legelterjedtebbek, ezért röviden tárgyaljuk ezek meghajtói közül az **SN 7446 N** típus alkalmazását.



6. ábra

A dekóder - meghajtó **BCD 8 4 2 1** súlyozású kódból állítja elő a **7 szegmensű** kijelző vezérlésére alkalmas jeleket az **a, b, c, d, e, f, g** jelű kimenetein. Kimeneti aktív szint a

0. Ezeken kívül különböző vezérlő bemenetei vannak az áramkörnek, amelyek szerepe a katalógusból olvasható ki. A dekóder és kijelző csatlakozását a 6. ábra mutatja. (A kijelző rajzán megjelöltük az egyes szegmensek betűjelzéseit.) Amikor a kimenetek közül valamelyik **0** szintű, akkor világit az azonos jelű szegmens. Az ABCD változók 16 kombinációjához tartozó kijelző-kép látható a 7.ábrán.

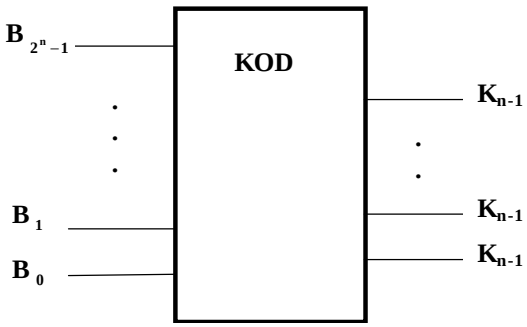


7. ábra

3.2. Kódoló áramkörök

A kódolás során **1 az N** - ből kódot (pl. 1 a 10-ből a decimális kód) kívánunk átalakítani **bináris, BCD** vagy **egyéb** kóddá. Ezt a feladatot megvalósító kombinációs hálózat a **kódoló**.

A kódolás tulajdonképpen a dekódolás duálja. A kódoló blokksémája a 8.ábrán látható. A maximálisan $\mathbf{2^n}$ számú bemenet ($\mathbf{B_0 \Lambda\ B_{2^n-1}}$) egyikére jut csak aktív logikai szint (1 vagy 0), s ennek alapján állítja elő az **n** db kimeneten ($\mathbf{K_0 \Lambda\ K_{n-1}}$) a megfelelő **n** bites bináris kódot.



8. ábra

Vizsgáljuk meg az egyik leggyakrabban használt kódoló áramkör, a decimális - BCD átalakító logikai függvényét és megvalósításának lehetőségeit. A 9.ábrán látható a kódolási feladat igazságtáblázata. A kimenetek jelölésére a szabványos A,B,C,D betűket használtuk.

A táblázat alapján felírhatók az egyes kimeneteket megvalósító logikai függvények.

$$\mathbf{A} = \mathbf{B}_1 + \mathbf{B}_3 + \mathbf{B}_5 + \mathbf{B}_7 + \mathbf{B}_9$$
$$\mathbf{B} = \mathbf{B}_2 + \mathbf{B}_3 + \mathbf{B}_6 + \mathbf{B}_7$$
$$\mathbf{C} = \mathbf{B}_4 + \mathbf{B}_5 + \mathbf{B}_6 + \mathbf{B}_7$$
$$\mathbf{D} = \mathbf{B}_8 + \mathbf{B}_9$$

B_9	B_8	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0	D	C	B	A
0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	0	1	0	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0	0	0	0	1	0	1
0	0	0	1	0	0	0	0	0	0	0	1	1	0
0	0	1	0	0	0	0	0	0	0	0	1	1	1
0	1	0	0	0	0	0	0	0	0	1	1	0	0
1	0	0	0	0	0	0	0	0	0	1	0	0	1

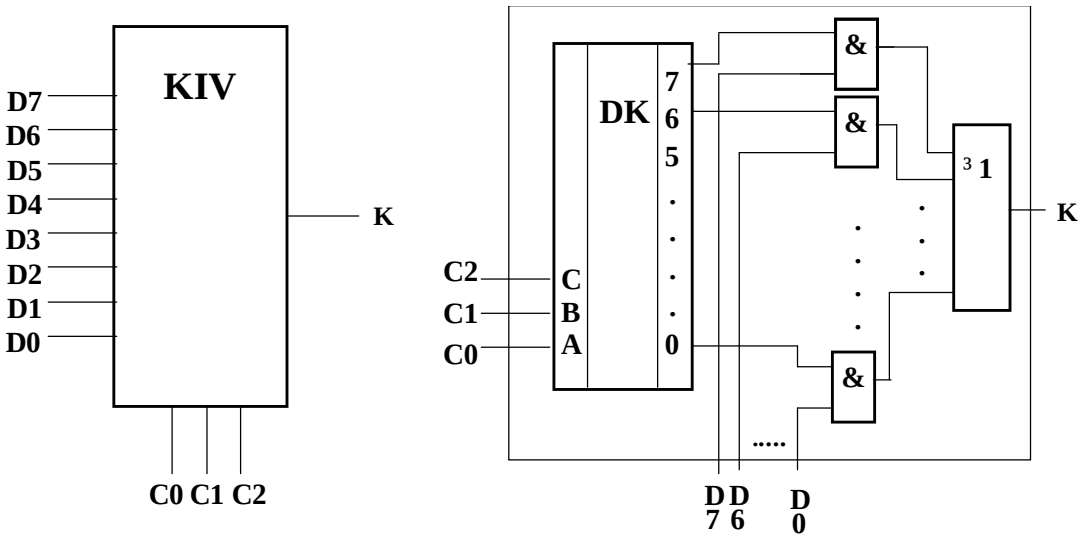
9. ábra

A logikai függvények ismeretében megtervezhető a kódolást megvalósító kombinációs hálózat.

A kódolók kitüntetett alkalmazási területe az adatbevitelre szolgáló billentyűzet (klaviatúra) és a digitális berendezés illesztése. Viszonylag egyedi felhasználásúk miatt integrált áramköri kialakításban nem készítenek ilyen kódolót. Diszkrét elemekből. IC kapukból könnyen megépíthetőek.

3.3. Kiválasztó áramkörök (multiplexerek)

A **kiválasztó** áramkör (adatszelektor) az **adat**-bemenetek ($\mathbf{D_1}$ . . $\mathbf{D_p}$) egyikének információját kapcsolja a **Q** kimenetre. A kiválasztást az **n** darab **címző** (kiválasztó) bemeneten ($\mathbf{C_0}$ ... $\mathbf{C_{n-1}}$) érvényes bináris kód határozza meg. (Az **n** bittel címezhető adatbemenet maximális száma $\mathbf{2^n}$.) Elvi blokkvázlata a 10.a. ábra szerinti.



a. b.

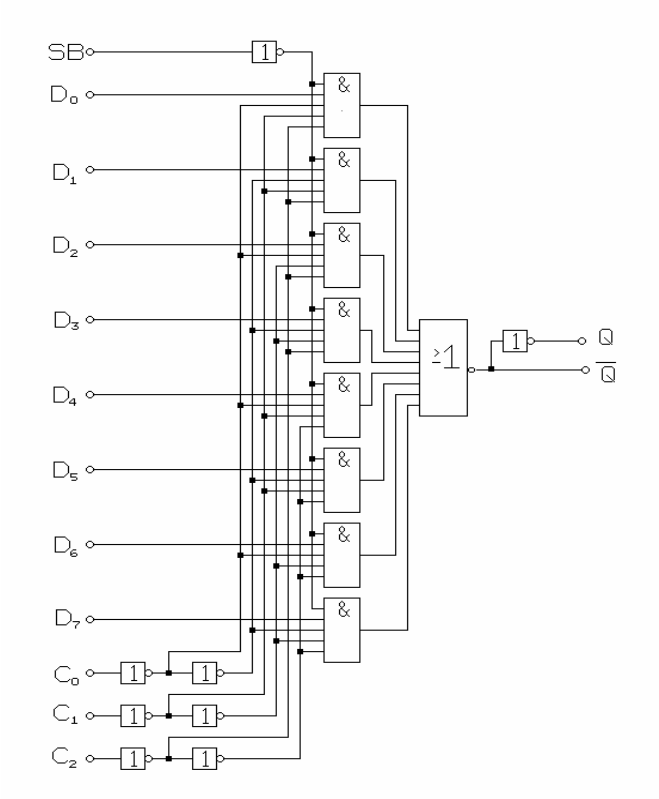
10. ábra

Egy n = 3 címző bemenetű multiplexer 8 adatból ($\mathbf{2^3}$) választ ki egyet. Ennek logikai függvénye a következő:

A függvény minden egyes logikai szorzatában szerepel valamelyik adat ($\mathbf{D_0}$ ... $\mathbf{D_7}$) és a címző bitek ($\mathbf{C_0,C_1,C_2}$) egyik kombinációja, mintermek (a zárójelbe tett mennyiségek), amelyek kimenetei közül egyidejűleg csak egyik lehet logikai 1 értékű. Ezért a **Q** kimenetre az az **adat-bit** ($\mathbf{D_i}$) jut, amelyhez tartozó címző variáció értéke1.

A függvénykapcsolat megvalósítható a címző $\mathbf{C_0,C_1,C_2}$ kódot **dekódoló** áramkörből és egy **ÉS-VAGY** hálózatból. Ennek logikai vázlat t mutatja a 10.b. ábra.

Az integrált áramköri elemkészletben több változatú multiplexer is van. A 11. ábra az SN 74151 típusú multiplexer (8-ról 1-re) logikai vázlatát mutatja.



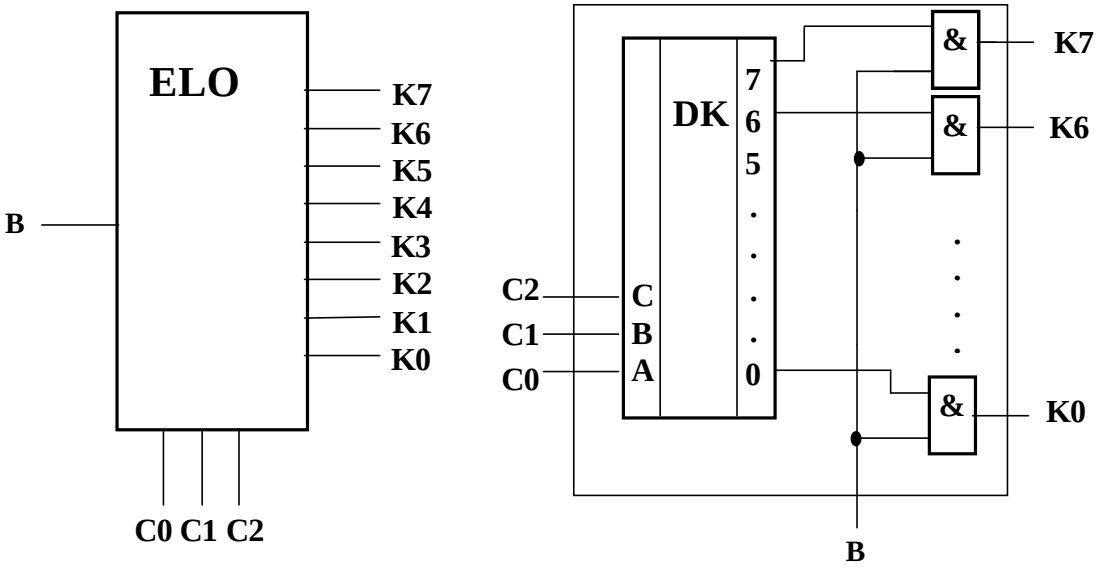
11. ábra

A címző (Data select) bemeneteken (D0 - D7) érvényes bináris kód választja ki a Q kimenetre jutó bemeneti információt. Az SB jelű kapuzó bemenetre (Strobe) adott 1 szinttel a választás letiltható.

3.4. Elosztó áramkör (demultiplexer)

Az adatelosztásra alkalmazható demultiplexer egyetlen adatbemenetről osztja szét az információt $\mathbf{2^n}$ számú kimenetre, ahol **n** a címző (elosztó) bemenetek száma.

Az áramkör elvi blokkvázlata a 12.a.ábrán látható.



a. b.

12. ábra

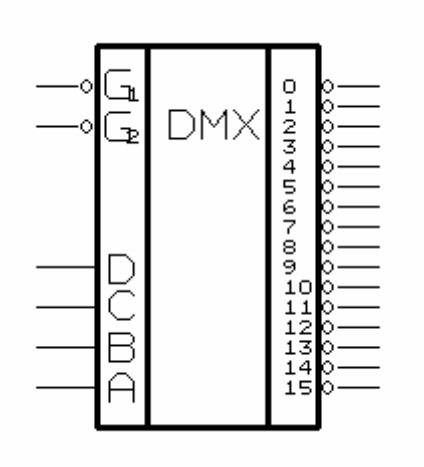
A függvényeket megvalósító hálózat felépíthető a címző bemeneteket dekódoló áramkörből, és ennek kimeneteit a D adattal kell kapuzni. A megvalósítás logikai vázlata látható a 12.b.ábrán.

Az elosztási feladat logikai függvényei n=3 esetén a következők:

$$K_0 = D(\overline{C}_2\overline{C}_1\overline{C}_0) \qquad K_4 = D(C_2\overline{C}_1\overline{C}_0)$$
$$K_1 = D(\overline{C}_2\overline{C}_1C_0) \qquad K_5 = D(C_2\overline{C}_1C_0)$$
$$K_2 = D(\overline{C}_2C_1\overline{C}_0) \qquad K_6 = D(C_2C_1\overline{C}_0)$$
$$K_3 = D(\overline{C}_2C_1C_0) \qquad K_7 = D(C_2C_1C_0)$$

(A zárójelekbe tett kifejezések a dekódoló kimeneteinek a függvényei). A demultiplexer kapuzott dekódolóként is alkalmazható, mivel a D bemenet 0 értékénél – a címző bemenetek vezérlésétől függetlenül – mindegyik kimenet 0 szintű lesz.

A TTL integrált áramköri elemcsaládban lévő **SN 74154** típusú dekódoló - demultiplexer szimbolikus jele látható a 13.ábrán. A címző bemenetek **A,B,C,D**. A **G1** és **G2** bemenetek közül az egyik adat a másik kapuzó bemenetként kezelhető (a kettő össze is köthető). A kimeneteken az aktuális adat negáltja jelenik meg. Ha G1 és G2 is 1 szintű, minden kimenet - a címtől függetlenül - 1 szintű lesz. Az áramkör bináris-decimális dekódolóként is használható, amennyiben G1 = G2 = 0. A kimeneti aktív szint a logikai 0.



13. ábra

3.5. Nagyság-komparátorok

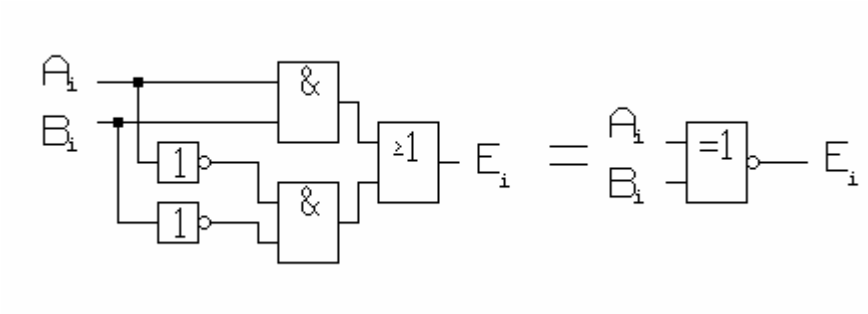
Nagyság-komparátornak nevezzük azt az áramkört, amely két bináris számot hasonlít össze, és kimenetein jelzi a számok közötti relációkat (egyenlő, kisebb, nagyobb).

Az összehasonlítás az összetartozó - azonos nagyságrend – bit-párok relációjának meg állapításán alapul.

Két bit (A és B) egyenlőségét az

$$\mathbf{E_i} = \mathbf{A_i B_i} + \overline{\mathbf{A}}_\mathbf{i}\,\overline{\mathbf{B}}_\mathbf{i}$$

logikai függvény (equivalencia) írja le. Az áramkör logikai vázlata a 14.ábrán látható, amely a kizáró-vagy tagadása



14. ábra

Több bites szám akkor egyenlő, ha az azonos helyértékű bitek egyenlők. Legyen a a két szám:

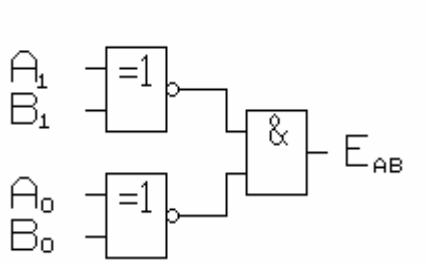
$$\mathbf{Z_A} = \mathbf{A_1 2^1} + \mathbf{A_0 2^0}$$

$$\mathbf{Z_B} = \mathbf{B_1 2^1} + \mathbf{B_0 2^0}$$

Az egyenlőséget leíró logikai függvény:

$$\mathbf{E_{AB} = (A_1B_1 + \overline{A}_1\overline{B}_1)(A_0B_0 + \overline{A}_0\overline{B}_0)}$$

A függvényt megvalósító áramkör logikai vázlata a 15. ábra szerinti.



15. ábra

További bővítés az előzőek ismétlésével történik.

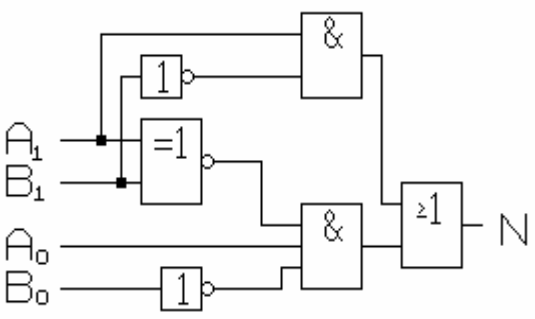
Két szám összehasonlításánál gyakran feladat - az egyenlőség jelzése mellett - a **kisebb**, ill. **nagyobb** viszony kijelzése is.

A következőekben vizsgáljuk meg - két-bites számok összehasonlításánál - az egyenlőtlenségi relációkat jelző áramkörök működési feltételeit és határozzuk meg a logikai függvényeket.

A $\mathbf{Z_A > Z_B}$ akkor igaz, ha $\mathbf{A_1 > B_1}$,ill. ha $\mathbf{A_1 = B_1}$ és $\mathbf{A_0 > B_0}$. A leírt feltétel teljesülését az **N** logikai változó jelölje. Logikai függvényben ez a következőképpen fogalmazható meg:

$$\mathbf{N = A_1\overline{B}_1 + (A_1B_1 + \overline{A}_1\overline{B}_1)A_0\overline{B}_0}$$

A függvény első logikai ÉS kapcsolata fejezi ki az $\mathbf{A_1>B_1}$ feltételt, ugyanis csak az $\mathbf{A_1=1}$ és $\mathbf{B_1=0}$ esetén ad 1 értéket. A zárójeles rész az $\mathbf{A_1=B_1}$ feltételt teljesíti, míg az $\mathbf{A_0\overline{B}_0}$ tag az $\mathbf{A_0 > B_0}$ relációt adja. A függvényt megvalósító áramkör logikai vázlata látható a 16.ábrán.



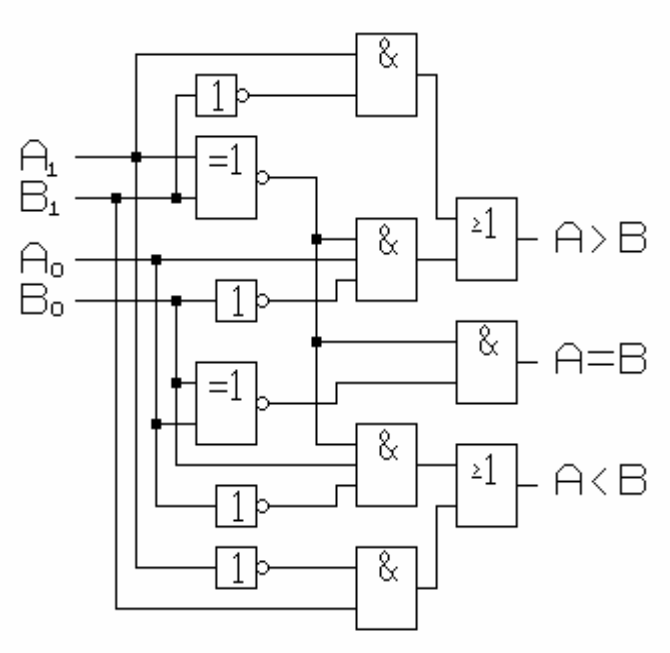
16. ábra

A $\mathbf{Z_A < Z_B}$ reláció logikai függvénye következik az előzőből, ha értelemszerűen felcseréljük a megfelelő biteknél a tagadást. Ezt a

$$\mathbf{K} = \overline{\mathbf{A}}_1\mathbf{B}_1 + (\mathbf{A}_1\mathbf{B}_1 + \overline{\mathbf{A}}_1\overline{\mathbf{B}}_1)\overline{\mathbf{A}}_0\mathbf{B}_0$$

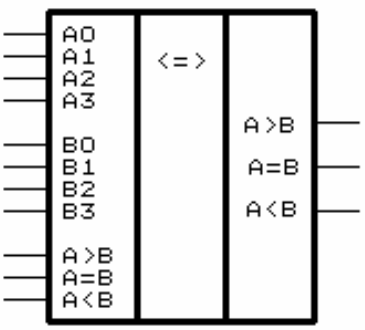
logikai függvény fejezi ki.

A kétbites számok teljes összehasonlítását végző komparátor logikai vázlata a 17.ábrán látható.



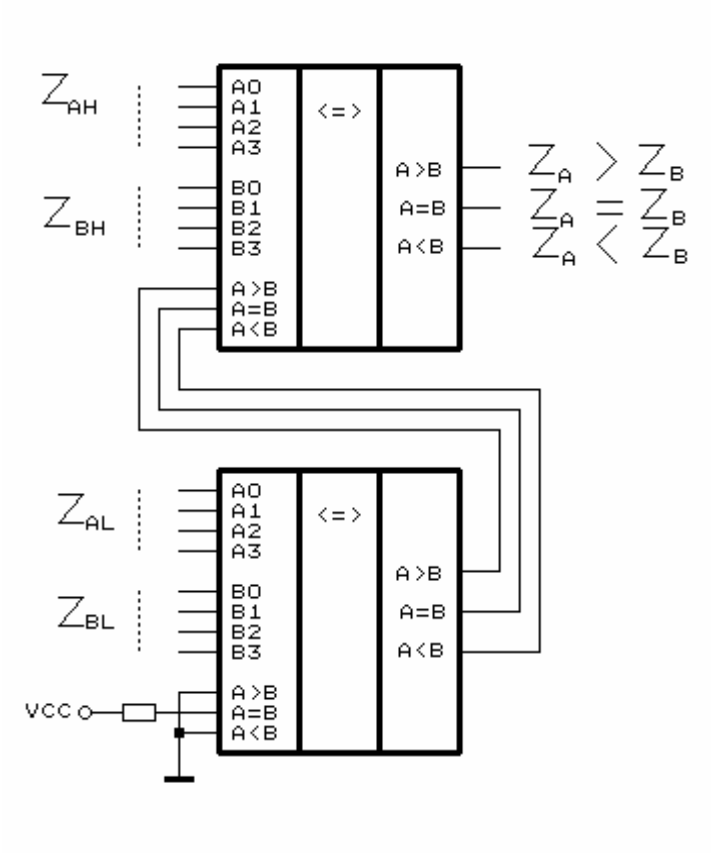
17. ábra

A TTL rendszerű integrált áramköri családban - egyetlen tokban - négy bites nagyság-komparátor az SN 7485 típusú áramkör. Szimbolikus jele a 18. ábra szerinti. Bemenetei a két összehasonlítandó szám bitjei (**A_0,A_1,A_2,A_3** és **B_0,B_1,B_2,B_3**) és az un. bővítő bemenetek **$A_i<B_i$, $A_i=B_i$, $A_i>B_i$**, amelyekre az alacsonyabb helyértékű négy bit összehasonlításának eredményét kell adni. Kimenetei a relációkat jelzik (**A<B, A=B, A>B**).



18. ábra

Hosszabb számok összehasonlításakor két vagy több komparátor köthető össze. Ilyenkor a kisebb nagyságrendeket összehasonlító áramkör kimeneteit kell összekötni a nagyobb nagyságrendeket összehasonlító áramkör bővítő bemeneteivel. Két nyolcbites számot összehasonlító nagyság-komparátor kapcsolási vázlata látható a 19. ábrán.



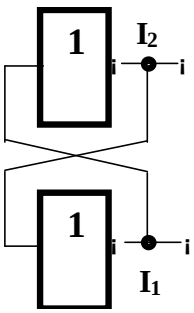
19. ábra

4. TÁROLÓK

A sorrendi, szekvenciális feladatok megvalósításához elemi tároló áramkörök szükségesek. A fejezetben ismertetjük a leggyakrabban alkalmazott tároló-elemek (flip-flop) felépítését, működését.

4.1. Tároló alapáramkörök

A tároló alapáramkörök - flip-flop -ok - **két stabil** állapotú áramköri kapcsolások. A két stabil állapot **1 bit** információ **tárolására** teszi alkalmassá a flip-flop -ot.
A kétállapotú elem két keresztbecsatolt inverter -ből alakítható ki. (1. ábra)



1. ábra

A két inverter keresztbe csatolása biztosítja a felvett állapot tarását. Az ábra szerint elrendezésben a tápfeszültség bekapcsolása után – a két inverter kapcsolási sebességének különbözősége miatt - véletlenszerűen alakul ki a stabil helyzet. Ha az $\mathbf{I_1}$ jelű inverter kimenetén **1** szint lesz, az az $\mathbf{I_2}$ bemenetére jutva biztosítja ennek a kimenetén a **0** szintet. A keresztbecsatolás révén ez a vezérlés fent tartja az $\mathbf{I_1}$ kimenetén az **1** szintet.

A fentiekben röviden elemzett áramkörnek nincs állapotváltozást vezérlő bemenete. A kívánt állapotváltozást a **vezérlő** bemenetek és **billentési módok** különböző változataival lehet megoldani. A megoldási módozatok alapján csoportosítjuk a flip-flop -kat.

4.1.1. Flip-flop típusok

A flip-flop -ok két nagy csoportba sorolhatók annak alapján, hogy az **információ** közlést és a **billentés**-t ugyanaz, vagy két különböző jel látja-e el. Ennek megfelelően:

- **közvetlen**, és
- **kapuzott** vezérlésű

tárolókat különböztünk meg.

A leggyakrabban alkalmazott típusok az

- **RS ,**
- **JK ,**
- **T** és
- **D** típusú flip-flop -ok.

Az egyes flip-flop -ok billentési módja szerint lehetnek:

- **statikus** és
- **dinamikus** billentésűek.

A kapuzott vezérlésű tároló elemek között - elsődlegesen az integrált áramköri kialakításban - további két nagy csoport létezik, a

- **együtemű,** és
- **kétütemű** vezérlésű

áramköri változat. A kétütemű vezérlést **közbenső tároló** alkalmazásával valósítják meg.

Az általános csoportosítás után a vezérlő bemenetek alapján megkülönböztetett típusokat elemezzük működés és áramköri kialakítás alapján.

Az **RS** flip-flop –nak két vezérlőbemenete van, amelyek közül az **S** jelű (set) a beíró és az **R** jelű (reset) a törlő bemenet. Ezek szerint a tárolt információ **1**, ha a **beíró** bemenetre (S) érkezik **aktív** logikai szintű vezérlő jel, és **0**, ha a **törlő** (R) bemenet kap ilyen vezérlést. A helyes működés feltétele, hogy a **két** vezérlőbemenet **együttesen nem** kaphat aktív vezérlést.

A **JK** flip-flop ugyancsak két vezérlőbemenettel rendelkezik. A **J** jelű bemenet **beírásra**, míg a **K** jelű a **törlésre** szolgál. Az RS flip-flop -tól az különbözteti meg, hogy **engedélyezett** a J és K **együttes** aktív vezérlése is. Ebben az esetben a flip-flop a tárolt állapot **ellenkezőjére** (komplemens -ére) vált át.

A **T** flip-flop egyetlen vezérlőbemenettel rendelkező tároló elem. A **T** bemenetre jutó aktív vezérlés a tároló állapotát **ellenkezőjére** változtatja.

A **D** flip-flop –nak ugyancsak egyetlen vezérlőbemenete van. A tároló mindenkor a D bemenet logikai értékét tárolja, vagy is **0** esetén **törlődik**, míg **1** értéknél **beíródik**. A leírt működés alapján a tárolót **adat** flip-flop -nak is nevezik.

4.1.2. Statikus billentésű flip-flop -ok

Statikusnak nevezzük azt a billentési módot, melynél a vezérlőjel logikai szintje a hatásos. A flip-flop mindaddig vezérelt állapotban van, míg a bemeneten az aktív logikai szint érvényes. Aktív lehet a logikai 1 és a logikai 0 szint is. A kívánt aktív vezérlés kiválasztása után a megvalósítandó flip-flop **működési táblázatából** (állapot-táblázat) felírhatók az **állapot-egyenletek**. Ezek alapján a szükséges vezérlési megoldás áramköri változata kialakítható.

S_n	R_n	Q_n	Q_{n+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. táblázat

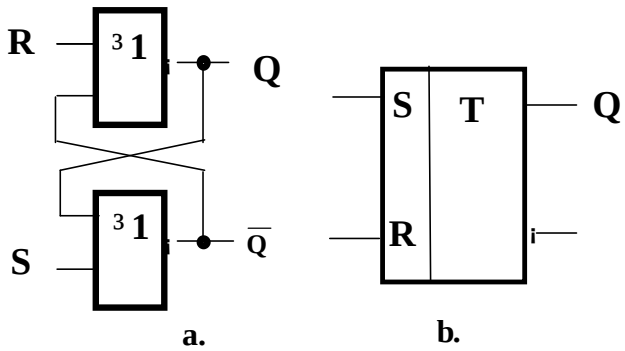
A statikus billentésű - logikai 1 szinttel vezérelt - RS flip-flop állapottáblázata az 1.táblázat szerinti.

A táblázat oszlopai között a $\mathbf{Q_n}$ mint bemenő változó szerepel (vezérlés előtti állapot). A vezérlés utáni **új állapotot** ($\mathbf{Q_{n+1}}$) a **vezérlés** (R_n, S_n) mellett az **előző állapot** (Q_n) is befolyásolja. A ***-al jelölt vezérlési kombinációk **tiltottak**. A táblázatból felírható **állapotfüggvények** az alábbiak:

$$\mathbf{Q_{n+1} = S_n + \overline{R}_n Q_n}$$
$$\mathbf{S_n R_n = 0}$$

(Az összefüggésben és a továbbiakban is az **n** index a vezérlés időpontjára utal.)

Ezek a logikai függvények az **RS** flip-flop **működését** írják le. Az összefüggés szerint az új állapot (Q_{n+1}) 1 szintű lesz - az előző állapottól függetlenül - ha a beíró (S) bemenet 1 szintű. Ugyancsak 1 szintű lesz a kimenet, ha már a vezérlés előtti állapotban is $Q_n = 1$ és az R bemeneten 0 szint van. A második összefüggés a **tiltott**



2. ábra

vezérlést írja le. Eszerint a két vezérlőbemeneten együttesen nem lehet 1 szint.

A fentiek szerint működő RS flip-flop két NOR kapuból alakítható ki a 2.a. ábra szerinti kapcsolásban. Az 1 szinttel vezérelhető RS flip-flop **szimbolikus** jele a b. ábra szerinti.

A kapcsolás működésének elemzése alapján könnyen belátható, hogy azért kell tiltani az együttes aktív vezérlést, mert ekkor mindkét kapu kimenete 0 szintű lesz. Az új állapot pedig a vezérlőjelek megszűnésének sorrendjétől függ, ezért előre meghatározhatatlan.

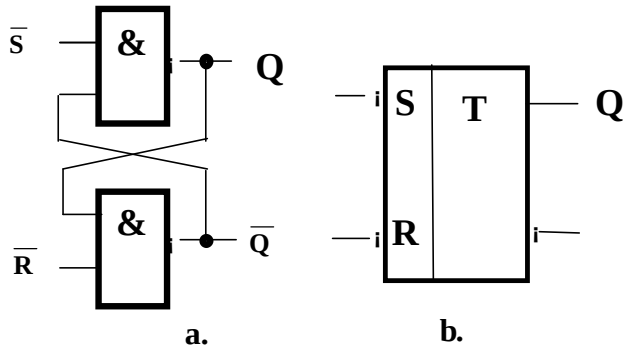
A **0** aktív vezérlési szintre billenő flip-flop működését az

$$\mathbf{Q_{n+1} = (\overline{S}_n + Q_n) R_n}$$
$$\mathbf{S_n + R_n = 1}$$

állapotegyenletek írják le .

Az összefüggés szerint az új állapot **1** lesz, ha a törlő bemenet (R) **1** és a beíró bemenet (S) **0** szintű, vagy vezérlés előtt is $Q_n = 1$ állapot volt.

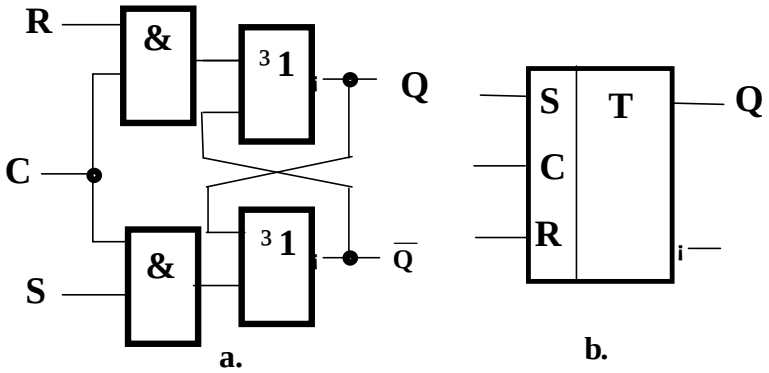
A második összefüggés írja le a tiltást, miszerint a két bemenet legalább egyikén 1 szintnek kell lenni. Áramkörileg NAND kapukkal valósítható meg statikus billentésű - 0 szinttel vezérelhető - RS flip-flop (3. ábra)



3. ábra

Az ismertetett két flip-flop **közvetlen** vezérlésű. A tárolandó információt hordozó **beíró** vagy **törlő** jel egyúttal a **billentést** is vezérli. A beírandó adatot hordozó jelet, és a billentő jelet kapuzással lehet fizikailag szétválasztani.

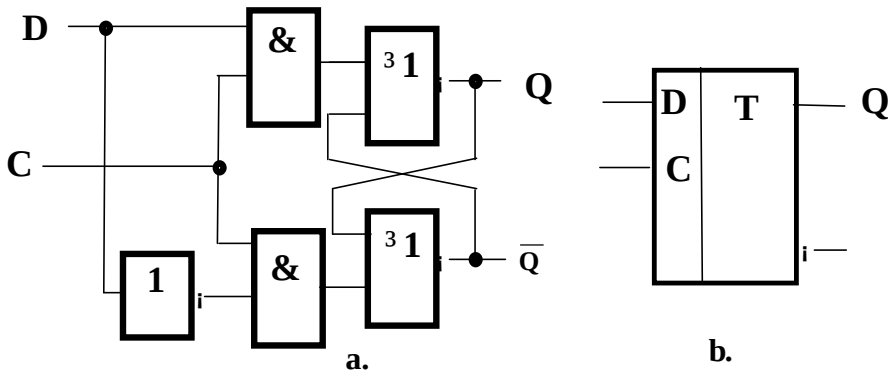
Statikus 1 szinttel vezérelt RS flip-flop vezérlőbemeneteit **C** billentő jellel kapuzva (4. ábra) kapjuk a kapuzott (szinkronozott) vezérlést. Ennél a flip-flop típusnál az R és S együttes aktív vezérlése csak a billentő (C) jel 1 szintjénél tiltott. Az **S** és **R** információs bemeneteken az **előkészítés** és a **C** jel hatására a tényleges beírás vagy törlés, vagyis az **adat** (információ) **bevitel** következik be.



4. ábra

A statikus billentésű 0 szinttel vezérelt RS flip-flop kapuzása VAGY kapukkal oldható meg, miután az aktív szint 0 mind a billentő, mind pedig az információs bemeneteknél.

A kapuzott RS flip-flop -ból alakítható ki a **D** flip-flop. Az inverter biztosítja a beíró és törlő bemenetek ellentétes szintű vezérlését (5. ábra). A **D** = 1 szintnél az S előkészítő bemeneten **1**, míg az R bemeneten **0** szint lesz. A C (szinkronozó) bemenetre érkező 1 szint a flip-flop –ot 1-be billenti, vagyis 1-et fog tárolni.



5. ábra

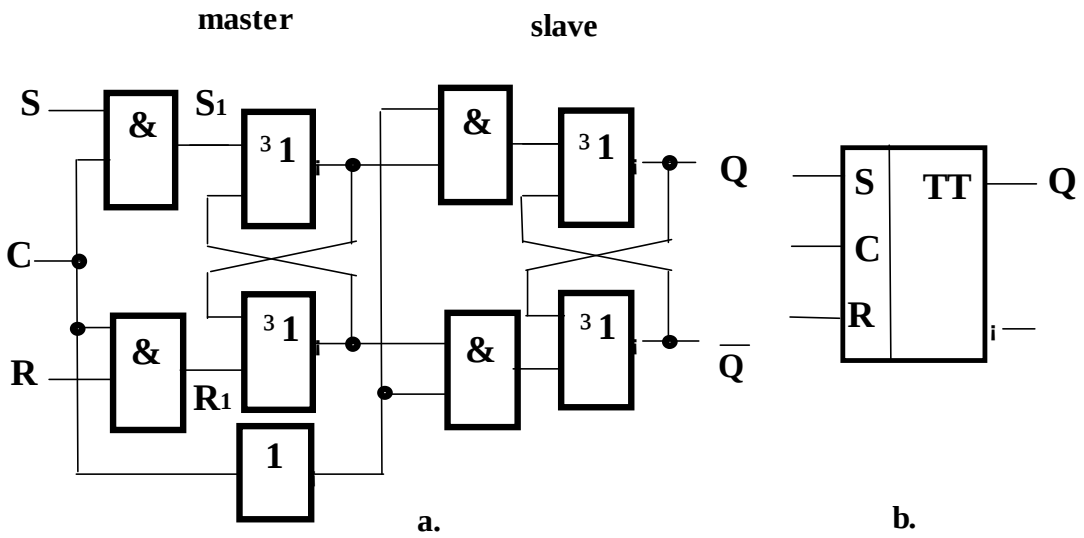
D = 0 esetében a törlés előkészítése, és a C jel hatására a **0** beírása következik. A D flip-flop két szinkronozó jel közötti időtartamra tárolja az információt. Ezt mutatja a 3.58.b. ábra szerinti idő-diaáram. A D tároló egyik legfontosabb fel-használási területe az információ szinkronoz s a C jel által meghatározott ütemezésben.

Az eddigiekben elemzett két flip-flop típus (RS és D) fő jellemzője, hogy kimenetén az új információ a vezérlőjel hatására - a billentési idő elteltével - azonnal megjelenik.

A digitális módon megvalósított jelfeldolgozásokban jelentős helyet foglalnak el azok a feladatok, melyekben az alkalmazott flip-flop -ok vezérlőbemenetére kimenetük értékét is vissza kell vezetni. Ilyen esetekben csak olyan flip-flop -ok alkalmazhatók, melyek kimenetén csak akkor jelenik meg az új állapot értéke, amikor a bemeneti vezérlés már hatástalan. Ez az igény **közbenső-tárolással** vagy **élvezérelt** billentéssel oldható meg.

4.1.3. Közbenső tárolós (ms) flip-fIop

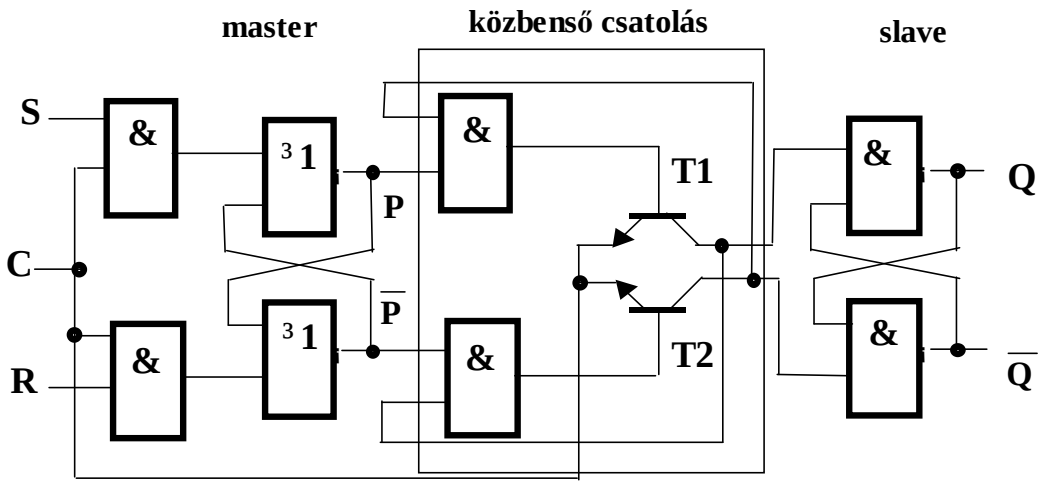
A közbenső tárolós **ms** (master-slave) flip-flop legegyszerűbb elvi változata a 6. ábra szerinti két kapuzott RS flip-flop -ból áll. Amíg a **C** billentő jel szintje **0**, addig a külső (RS) bemenetek szintjétől függetlenül az első flip-flop (master) $\mathbf{R_1}$ és $\mathbf{S_1}$ bemenetein is **0** szint van. A két flip-flop -ot elválasztó kapukra jutó C = 1 szintű jel hatására a master állapota átíródik a második (slave) flip-flop –ba.



6. ábra

Amikor a **C** jel logikai **1** szintű, akkor a bemeneti vezérlés határozza meg az első flip-flop állapotát, és **letiltódik** a két tároló közötti csatolás. A második flip-flop változatlanul tárolja az előző információt, és ezért a kimenet logikai értéke is változatlan. Az új információ a kimeneten csak C = 0 szintnél jelenik meg, amikor már a bemeneti vezérlés az első tárolóra hatástalan.

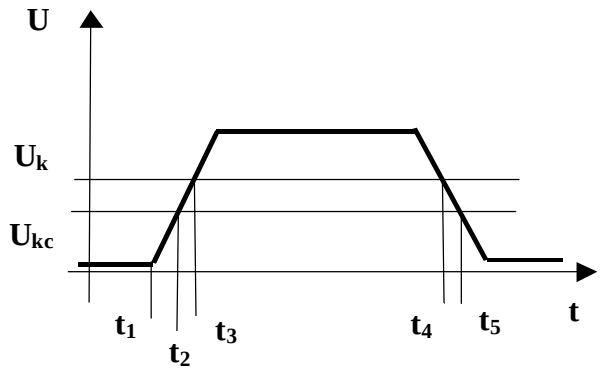
Az elemzett megoldás csak elvileg ad helyes működést. Ha az ellenütemű vezérlést biztosító inverter késleltetése nagyobb, mint a flip-flop billenési ideje, akkor a master még a slave vezérlésének tiltása előtt felveheti az új állapotot. Ez hibás működést eredményez. A tényleges áramköri megoldásoknál ezért a két flip-flop közötti csatolás letiltása hamarabb kell bekövetkezzen, mint a bemeneti kapuzás engedélyezése. Ezt a két komparálási szintű kapuzás biztosítja.



7. ábra

A 7. ábra a TTL rendszerű integrált áramköri készletben alkalmazott két komparálási szintű ms flip-flop logikai felépítését mutatja. A **master** NOR kapukból kialakított statikus - 1 szinttel billenthető (ponált) – RS flip-flop. A **slave** NAND kapukból kialakított - 0 szinttel billenthető (negált) - RS flip-flop. A két ÉS kapun és tranzisztoron (T1-T2) keresztül csatolódik a slave flip-flop a master flip-flop -hoz. A C billentő jel 0 szintjénél a csatoló tranzisztorok (T1, T2) emitterei 0 szinten vannak, így az a tranzisztor vezet, amelynek a bázisa 1 szintű. Ezt a master flip-flop kimenete vezérli a ÉS kapuk egyikén keresztül. A vezető tranzisztor 0 szinttel állítja be a slave flip-flop -ot a master által meghatározott állapotba.

Például, ha a P kimeneten van 1 szint, akkor T1 tranzisztor bázisára jut 1 szint s a vezetésbe kerülő tranzisztor 0 szintet kapcsol a slave felső kapujára. Ennek hatására a Q kimeneten 1 szint lesz. A P = 0 szintje miatt a T2 tranzisztor zárt, tehát a slave -nak csak egyik bemenetére jut 0. Ugyanakkor a C jel lezárja a bemeneti ÉS kapukat kapukat, amivel függetleníti a mastert a bemenetektől. A leírt állapotváltozást és a kettős komparálás fogalmát a billentő-jel időbeli változása alapján elemezzük. A 8. ábra a Cp bemenetre jutó billentő impulzus időbeli változását mutatja.



8. ábra

A t_1 időpontban kezdődik a billentő-jel felfutó éle, és ezzel együtt növekszik a tranzisztorok emitter feszültsége is. A **T1** kollektor feszültsége követi ezt a változást, de abszolút-értékben legalább U_m kollektor - emitter maradékfeszültséggel pozitívabb (0,6 V). Jelöljük a TTL kapuk komparálási szintjét U_k - val. A fentiekből adódik, hogy amikor a billentő jel feszültsége a t_2 idő-pillanatban eléri az

$$U_{kc} = U_k - U_m$$

értéket, ekkor a slave flip-flop bemeneteire 1 szint jut, és megszűnik a két flip-flop közötti csatolás.
Ugyanekkor a bementi kapuk még zártak, mivel a C feszültsége kisebb U_k - nál. A bemeneti mintavételezés csak az

$$U_c > U_k$$

feszültségtartományhoz tartozó $t_3 - t_4 = t_m$ idő alatt történik. A billentő-jel lefutó élénél is hamarabb következik be a bemeneti kapuk lezárása (t4), mint a csatoló kapuk nyitása (t5). Az újonnan beirt információ a kimeneten a t5 időpillanatot követően, jelenik meg. A csatoló-kapuk egy bemenetének a tranzisztor kollektorokkal való keresztbecsatolása a slave flip-flop együttes vezérlését akadályozza meg abban az esetben, ha a C bemeneten jelreflexióból adódóan negatív hullám jelenne meg. Az **ms** flip-flop fentiekben elemzett működése elvileg független a billentő-jel meredekségétől. Viszont lassú jelváltozóskor az U_k komparálási szint környezetében nagyon zavar-érzékeny a kapu és a legkisebb tápáram-zaj is nem kívánt átbillenést eredményez. Ezért a jelváltozás idejének 400 ns - nál kisebbnek kell lennie.

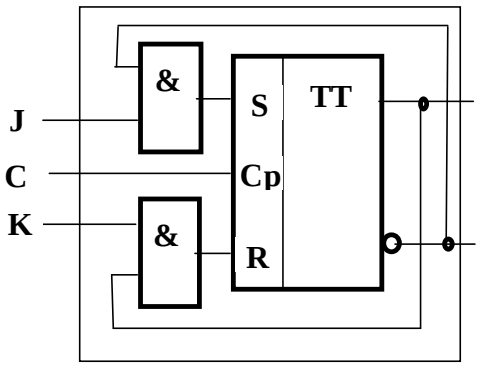
4.1.3.1. Közbenső tárolós JK flip-flop

A JK típusú flip-flop - amelynek működési feltételét a korábbiakban már elemeztük - csak szinkronizált vezérlőbemenetekkel alakítható ki. A flip-flop állapot egyenlete az alábbi:

$$Q_{n+1} = J_n\overline{Q}_n + \overline{K}_nQ_n$$

A JK flip-flop - közbenső tárolós RS flip-flop -ból a 9. ábra szerint épül fel. A bemeneti vezérlő jelek kapuzása a kimeneti jelekkel biztosítja a kívánt működést. Amikor a flip-flop 1-t tárol (Q = 1) akkor csak a K bemenetre jutó vezérlés eredményez állapotváltozást, ill. 0 tárolását követően (Q = 0) a J bemenetre jutó vezérlés hatásos .Ez

a kapuzás egyúttal engedélyezi a J és K bemenetek együttes vezérlését is. Ekkor ugyanis a flip-flop előző állapota határozza meg a billentő-jel hatására bekövetkező állapotváltozást.

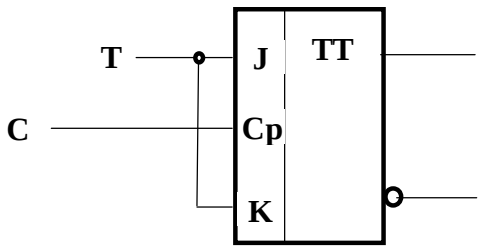


9. ábra

Az előzőekben elemzett JK flip-flop –ból T típusú tároló olyan módon alakítható ki, hogy a két vezérlő bemenetet (J és K) összekötjük s ez lesz a T vezérlő bemenet. Az állapotegyenlet - 1 szintű aktív vezérlésnél - a következő:

$$\mathbf{Q_{n+1} = T_n \overline{Q}_n + \overline{T}_n Q_n}$$

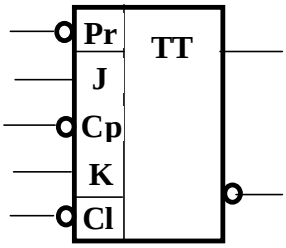
A tárolóba információt csak a T vezérlő bemenet 1 szintjénél lehet beírni. Az állapotváltozást a T = 0 vezérlés letiltja. A T flip-flop -ot elsősorban számláló áramkörök kialakítására használják. Integrált áramköri kialakításban ezt a flip-flop változatot önállóan nem gyártják, miután a JK típusból külső kötéssel kialakítható. A 10.a. ábra a T flip-flop logikai felépítését és szimbolikus jelét ábrázolja.



10.a. ábra

4.1.3.2. Közbenső tárolós flip-flopok aszinkron billentése

Az integrált áramköri közbenső tárolós - master-slave - flip-flopoknak **aszinkron törlő** és **beíró** bemenetei is vannak. Az aszinkron statikus vezérlés együtemű. Ez azt jelenti, hogy a vezérlőjel - a flip-flop mindkét tárolóját - egyidejűleg billenti a kívánt állapotba. A TTL rendszerű IC-s tárolóknál az aszinkron vezérlés aktív szintje - rendszerint - a 0 szint. A 10.b. ábra közbenső tárolós - TTL rendszerű integrált áramköri - **JK preset flip-flop** szimbolikus jelét mutatja. Az aszinkron vezérlő bemenetek, a **Cl** (Clear) **törlő** és a **Pr** (Preset) **beíró** bemenet.



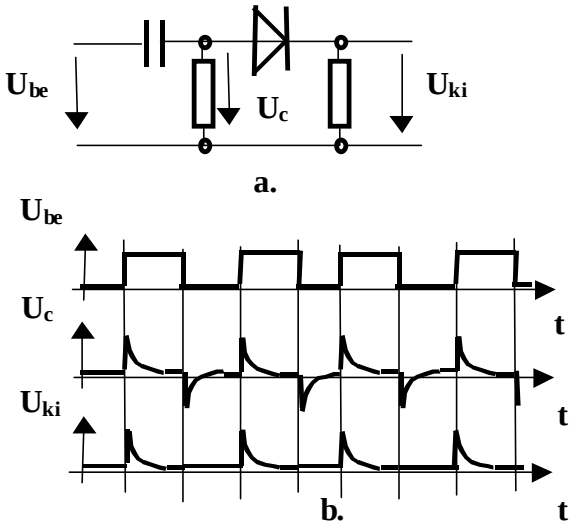
10.b. ábra

A jelölésben a bemeneti mező középső részéhez csatlakoznak a kétütemű vezérlésű ms flip-flop bemeneti jelei. A **Cp** billentő bemeneten lévő invertáló jel (karika) azt jelzi, hogy az új érték a kimeneteken a **billentő jel 0** szintjénél jelenik meg. A **Pr** aszinkron beíró, illetve Cl törlő bemenetek **aktív** szintje **0**. A két utóbbi bemenet szerint a tároló **RS típusú** együtemű flip-flop.

4.1.4. Dinamikus billentésű flip-flopok

Az eddigiekben elemzett flip-flopok közös jellemzője a statikus billentés. A tárolók másik nagy csoportját alkotják a **dinamikus** billentésű (**élvezérelt**) áramköri megoldások. A továbbiakban külön elemezzük a dinamikus billentés diszkrét ill. integrált áramköri megoldásait.

Az **élvezérlés** diszkrét elemekkel un **trigger - áramkörrel** alakítható ki. A trigger áramkör kimenetén csak akkor jelenik meg jel, ha bemenetén logikai szintváltás van. A trigger áramkör vagy más néven dinamikus csatolókapu egyik legegyszerűbb változata a 11. ábra szerinti.



11. ábra

Az a. ábrán a kapcsolási vázlat, míg a b. ábrán a jellegzetes feszültségalakokat szemlélteti. Ha a kapu bemenetére kapcsolt $\mathbf{U_{be}}$ feszültség négyszöghullám, akkor a belső ponton csak a bemeneti szintváltáskor mérhető feszültségugrás, mégpedig a szintváltás irányának megfelelő polaritású. A diódán csak a pozitív feszültség-változás hajt át áramot, ezért a kimeneti ponton a felfutó élekkor lesz jel. A diódát fordítva kötve, a negatív éleknél lesz a kimeneten jelváltás.

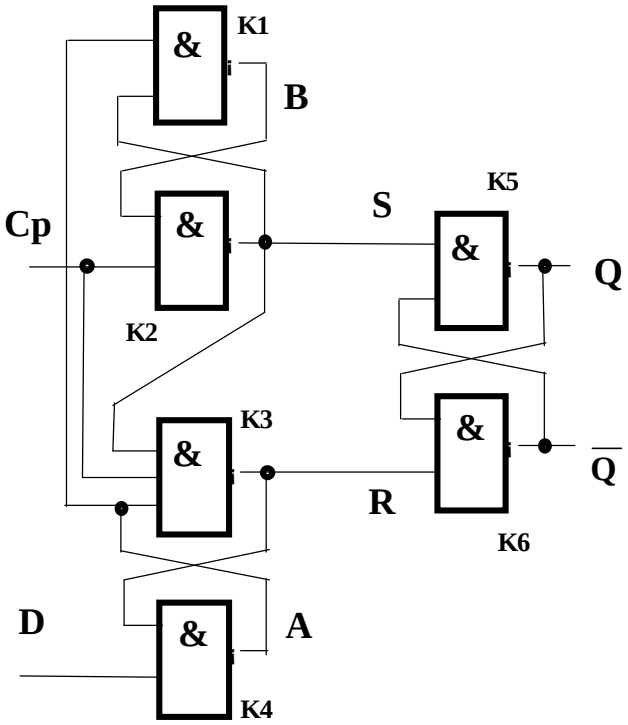
Az integrált áramköri tárolók dinamikus billentésű változatai az ún. élvezérelt flip-flopok. Ezekben az áramkörökben - technológiai és áramköri méretek miatt - kapacitív csatolókapu nem valósítható meg. Az élvezérlés lényege, hogy az áramkörben alkalmazott logikai kapuk **késleltetési idői** eredményezik a kis időtartamú billentő jelet a bemeneti vezérlés szintváltozásakor. Az állandósult jelek időtartama alatt a flip-flop leválasztódik a vezérlő bemenetekről.

A leírt elvi megoldásra példa a 12. ábra szerinti logikai felépítésű D flip-flop. A tároló a K5-K6 jelű NAND kapukból álló statikus - 0 szinttel billenthető - RS flip-flop. A C billentő jel 0 szintjekor a flip-flop nem kap vezérlést, mivel a K2 és K3 kapuk kimenete (R,S) - a D vezérlőbemenet értékétől függetlenül - 1 szintű. Ez az időszak az előkészítő fázis. Ekkor az áramkör A és B belső pontjainak logikai szintjeit a D bemenet logikai értéke határozza meg, az

$$\mathbf{A} = \overline{\mathbf{D}}$$

$$\mathbf{B} = \mathbf{D}$$

összefüggések szerint.



12. ábra

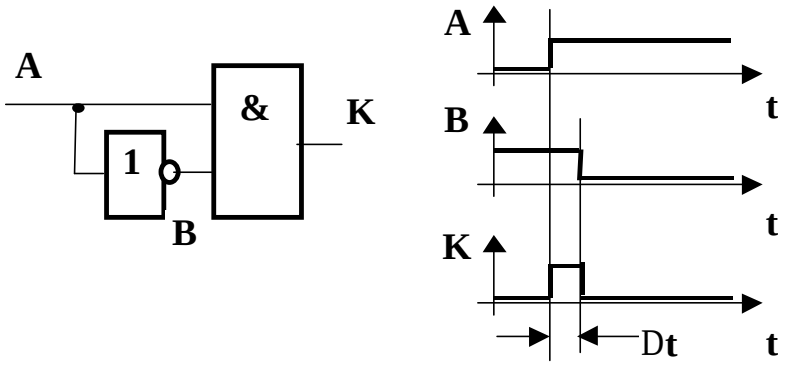
A belső pontok ezen értékeket a D változósa után, a kapuk késleltetését követően veszik fel. Mégpedig az A $\mathbf{t_{pd}}$, a B pedig a $\mathbf{2t_{pd}}$ idő elteltével. A billentő jel csak a D változását követő $\mathbf{2t_{pd}}$ idő múlva érkezhet. Ezt nevezik előkészítési $\mathbf{t_{su}}$ (set-up time) időnek. A billentő C jel pozitív jelváltozásakor a K1 és K2 kapuk kimeneti szintjét az A, ill. a B pontok logikai szintje határozza meg. Ennek megfelelően a flip-flop vezérlő bemenetei az

$$\mathbf{S} = \overline{\mathbf{B}} = \overline{\mathbf{D}}$$

$$\mathbf{R} = \overline{\mathbf{A}} = \mathbf{D}$$

logikai értékeket veszik fel. Az S és R bemenetek együttes 0 értékét a K2 és K3 kapuk közötti keresztcsatolás tiltja. A C billentő-jel 0 - 1 jelváltozósát követő $\mathbf{t_{pd}}$ idő elteltével a D újbóli változósa már nem változtatja meg az S és R logikai értékét, tehát még ennyi ideig kell a billentést követően a vezérlést a D bemeneten tartatni. Ez a katalógusokban adott tartási idő $\mathbf{t_h}$ (hold time). Újabb állapotváltozás, csak a C jel ismételt 0 - 1 élváltozásakor következhet be. A leírtak szerint működő TTL rendszerű élvezérelt flip-flop helyes működésének feltétele, hogy a C jel felfutási ideje kisebb legyen 250 ns - nál.

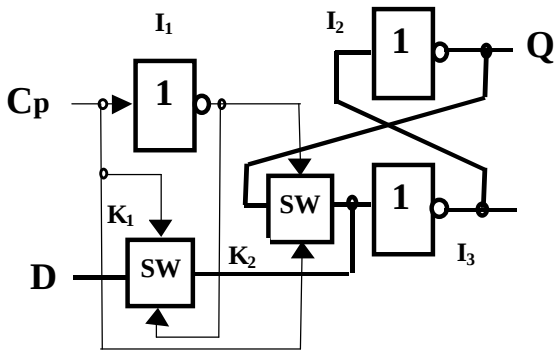
Az élvezérlés egy másik megoldásánál a billentő impulzus felfutó élénél mesterségesen létrehozott **hazárd** vezérli az állapotváltozást. Ennek elve, hogy ha a logikai ÉS kapu két bemenetén a jelek ellenkező értelemben változóak és az 1 - 0 átmenet késleltetett, akkor a kimeneten a késleltetéssel megegyező idejű – tű-impulzus (hazárd) jön létre. A kapcsolást és az időviszonyokat a 13. ábra szemlélteti. A D**t** jelű elem késleltetési ideje határozza meg a tű-impulzus szélességét.



13. ábra

4.1.5. CMOS flip-flop -ok

Az ipari vezérlésekben egyre inkább elterjedő **CMOS** integrált áramköri családokban a flip-flopok egy sajátos áramköri változata található. Ezt a megoldást szemlélteti a 14. ábra szerinti felépítésű D flip-flop. A $\mathbf{I_2}$ és $\mathbf{I_3}$ CMOS inverterek alkotják a flip-flopot, az eddigiektől annyiban térnek el, hogy az egyik csatolás az $\mathbf{K_2}$ jelű vezérelt kapcsolón (SW átvivő kapun) keresztül jön létre. Az $\mathbf{K_1}$ kapcsoló és $\mathbf{I_1}$ inverter a bemeneti vezérlő áramkörök. Az átvivő kapukat a C jel vezérli oly módon, hogy a C = 0 értéknél az $\mathbf{K_2}$ a kis impedanciájú és az $\mathbf{K_1}$ a nagy impedanciájú. Ezáltal a flip-flop mindkét csatolása biztosított és tárolja a beirt információt. A C billentő-jel 1 értékénél az $\mathbf{K_2}$ lesz nagy-, és az $\mathbf{K_1}$ kis impedanciájú. Ezáltal a flip-flop öntartó belső csatolása megszűnik és a D jel közvetlenül a $\mathbf{I_1}$ inverter bemenetére jut. A Q kimenet a kettős invertálás révén megegyezik D-vel. A C = 0 értéknél a kapcsolók állapota ismét ellenkezőjére vált és a flip-flop a beirt információt tárolja az újabb billentésig.



14. ábra

A CMOS technológiával kialakított JK ms flip-flop is hasonló felépítésű. Mindkét tároló-rész - az előzőekben elemzett - vezérelt elektronikus kapcsolós megoldásban épül fel.

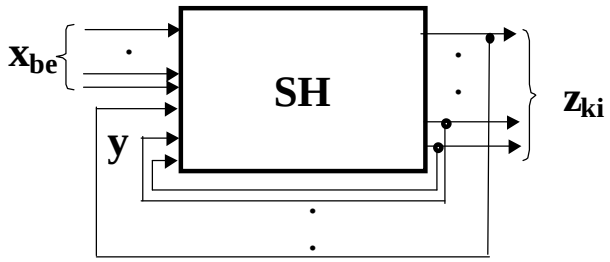
Az egyes típusok felépítését részletesen a katalógusokból lehet megismerni, ill. elemezni.

5. SORRENDI LOGIKAI HÁLÓZATOK

Az előző fejezetekben tárgyalt kombinációs hálózatok a logikai feladatok azon csoportjának megvalósítására használhatók, amelyeknél a következtetés(ek) egyedül az állítások időszerű kombinációjától függ(nek). Független viszont az állítás-kombinációk sorrendjétől.

A logikai feladatok jelentős hányadában - az éppen teljesülő állítások (feltételek) mellett - figyelembe kell venni az állítások megelőző kombinációját, ill. sorozatát is. Ezek a **sorrendi** vagy **szekvenciális** logikai feladatok.

Sorrendi feladat megvalósításához olyan hálózatra van szükségünk, amely a kimenetek aktuális kombinációja az éppen érvényes és a megelőző bemenőjelek adott sorozatának az eredménye. Akkor valósítható meg ilyen hálózat, ha a **bemeneteire** nem csak az éppen érvényes bemeneti jelek jutnak, hanem a kimenetek jelek is. Így biztosítható az előző állapotok hatása az új állapotokra. Általános blokkvázlata tehát a 1. ábra szerinti.



1. ábra

Az **SH** sorrendi hálózat két-két csoport bemenettel és kimenettel rendelkezik. Az **x** jelű **bemenetek**re jutnak a **külső** logikai változók. Az **y** jelű bemenetekre (állapot változók) **csatoljuk vissza** a hálózat előző állapotára jellemző **Z** kimeneti változókat. A visszacsatolás következtében a kimeneti változók függvényei **x**-nek és **y**-nak, vagyis

$$\mathbf{Z = f_z\,(x,\, y),}$$

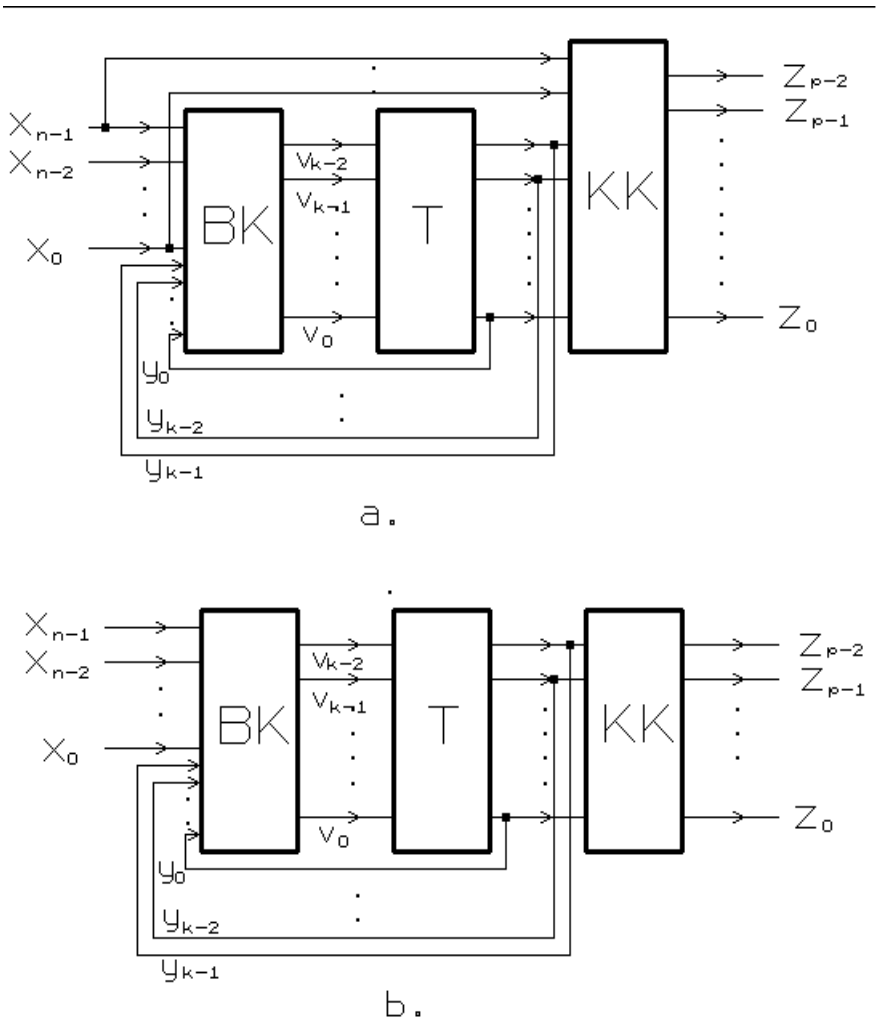
ahol $\mathbf{f_z}$ a kimenetek és a bemeneti változók, valamint az előző állapotok logikai kapcsolat-rendszerét adja meg.

Gondolati kísérlettel belátható, hogy az 1. ábra szerinti hálózat csak akkor stabil, ha **y = Y** és **x** bemeneti változók is **állandósultak**. Ez csak - egy bemeneti kombinációváltást követően - késleltetve, legkevesebb a hálózat ($\mathbf{t_H}$) késleltetési ideje múlva következhet be. Két **stabil** állapot között mind Z, mind Y változhat. Amennyiben két stabil állapot közötti tranziens időben (instabil állapotváltozások közben) változik a bemeneti kombináció, akkor újabb tranziens folyamat indul el.

Az ilyen felépítésű hálózatot **aszinkron sorrendi** hálózatnak nevezzük. Ezeknél az állapot-változást vagy azok sorozatát a bemeneti jelek változása indítja el. A következőkben az aszinkron magoldásnál csak számlálókat tárgyaljuk röviden, amelyek működése - az elemzőbb elméleti ismeretek nélkül is - elsajátítható.

Sorrendi hálózat kialakítható olyan működéssel is, amelynél az egymást követő **állapot-változásokat** nem közvetlenül az **x** bemeneti jelek megváltozása, hanem - egy ezektől független - ütemező jel, az un. órajel vagy **szinkron** jel okozza. Ehhez - a két órajel közötti időben - tárolni kell az állapotra jellemző információt. Ezek az **állapotjelek** (szekunder változók). Az órajelek **mintavételezik** az éppen aktuális **bemeneti**-, és **állapotjeleket**, és ezektől függően alakul majd ki - a soron következő órajel hatására - az új állapot.

Lényeges, hogy az órajel aktív ideje alatt a bemeneti és a szekunder változók értéke állandó legyen. Az előbbiek alapján működő hálózatot nevezzük **szinkron sorrendi hálózatnak**, amelynek általános felépítése a 2. ábrán látható. A bemeneti kombinációs hálózat (**BK**) állítja elő az **x** bemeneti-, és **y** állapot változókból a **T** állapottárolók új vezérlőjeleit (**vj**). Az állapot tárolókat az ütemező órajel (**Cp**) billenti a **vj** által meghatározott új állapotba. Az ábrán látható kétféle elvi felépítés. Az a. ábra szerint a kimenetek jeleit (Z) előállító kombinációs hálózat (KK) az állapotváltozókból és közvetlenül a bemeneti jelekből állítja elő. Ez a megoldás az un. **Mealy**- modell. A b. ábra szerinti változatban A kimenetekre csak az állapotváltozókon keresztül hatnak a bemeneti jelek. E változat az un. **Moore**- modell. Az utóbbi megoldásnál esetleg több tárolóra van szükség, de egyszerűbb a felépítés. A korszerű áramkörök alkalmazásával már elhanyagolandó szempont lett a tárolók száma (különösen a programozott rendszerekben), s ezért a Moore modell szerinti felépítés mind hardverben, mind pedig a szoftveres megoldásban nagyobb teret kap.



2. ábra

A kimeneti kombinációs hálózat (**KK**) - a bemeneti és az állapotváltózókból - állítja elő a hálózat kimeneti jeleit, $\mathbf{Z}_i$ -t.

Azonos funkciójú szinkron sorrendi hálózat kialakítható olyan változatban is, hogy nem használunk külön kimeneti kombinációs hálózatot, hanem az Y szekunder változókat állítjuk elő oly módon, hogy azok, vagy egy részük egyúttal a hálózat kívánt kimeneti változói is.

A hálózat állapota, s így a kimenő jelek is az órajel (Cp) ütemezésében váltanak értéket.

Tételezzük fel, hogy a vizsgált $\mathbf{t_i}$ időpillanatban - amely a két órajel közötti időpont - a hálózati tranziensek lejátszódtak, a hálózat állapotát és a kimeneti értékeket a $\mathbf{Z_i}$ kimeneti jelkombináció írja le. Ugyanebben az előkészítési fázisban az új bemeneti jelkombináció állandósult értéke $\mathbf{X_i}$. Ekkor a **KO** kombinációs hálózat bemenetén az $\mathbf{X_i}$ és $\mathbf{y = Z_i}$ bemeneti értékek érvényesek és előállítják a **T** tárolók $\mathbf{v_{ji}}$ vezérlőjeleit. A $\mathbf{t_{i+1}}$-edik időpontban érkező órajel fogja - a $\mathbf{v_{ji}}$ által meghatározott állapotba - billenteni a tárolókat. Ennek eredményeként alakul ki az új (Z_{i+1}) kimeneti jelkombináció, ami egyúttal a következő mintavételezéshez tartozó állapotjellemző is. Az előzőek alapján felírhatjuk a

$$\mathbf{Z_{i+1} = fz(v_{ji})}$$
$$\mathbf{v_{j_i} = fv(X_i, Z_i)}$$

függvénykapcsolatokat. Az **fz** kimeneti függvény az előállítani kívánt kimeneti (Z_{i+1}) és a tárolókat vezérlő $\mathbf{v_{ji}}$ jelek - billentés előtti - értékei közötti logikai kapcsolatot adja meg. Felírása az alkalmazott flip-flop típusok, un. állapotfüggvényei alapján és az előállítani kívánt új állapotok ismeretében történik. Az **fv** vezérlőfüggvény a tárolókat vezérlő jelek ($\mathbf{v_{ji}}$), az új állapotot előkészítő bemeneti jelek érvényes kombinációja (X_i) és a hálózat éppen aktuális állapota (Z_i) közötti logikai kapcsolatrendszert adja meg. Miután a függvény bemenő-, és kimenő változói egyazon időpillanatra (t_i) érvényesek, ezért kombinációs hálózattal valósítható meg.

A szükséges tárolók számát az előállítandó kimeneti kombinációk száma határozza meg. Abban az egyszerű esetben, amelynél nem alkalmazunk kimeneti kombinációs hálózatot (pl. számlálók) a tárolók és a kimenetek száma azonos. Kimeneti kombinációs hálózat alkalmazásakor kevesebb tároló is elégséges, miután ezek csupán a szükséges állapotok számát tárolják, és az állapotjelek száma kisebb, mint az általuk előállítható kombinációk száma. A szükséges állapotok optimális számának meghatározására különböző szisztematikus eljárások ismertek. Ezek a digitális rendszertechnika elméletének szerves részei, viszont nem témája tantárgyunknak. Ezért ebben a jegyzetben sem térünk ki az eljárások ismertetésére.

5.1. Sorrendi hálózatok logikai leírása

A sorrendi logikai feladatokat megvalósító szinkron sorrendi hálózatok tervezéséhez szükségünk van a kívánt működést egyértelműen megadó, az ismert hálózattervezési módszerek alkalmazását elősegítő leírásra. A továbbiakban röviden ismertetünk egy-egy

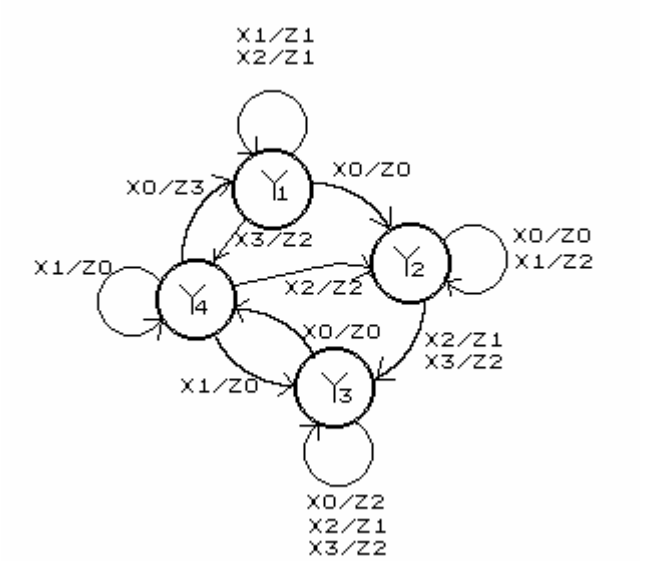
- grafikus és
- táblázatos

feladat leírási módszert.

5.1.1. Állapotgráf

A sorrendi logikai feladatokhoz gyakran használják az **állapot-gráfnak** nevezett szemléltető leírást. Ennek lényege, hogy a hálózat minden állapotát egy-egy **körrel** jelöljük. A körökbe az állapot-, és a kimeneti jellemzők kombinációját (Yi, Zi) írjuk. A körökből **nyilak** indulnak ki, amelyek vagy egy másik körben, vagy önmagában az induló körben (állapotban) végződnek.

Ezek a nyilak az egyes bemeneti kombinációkhoz tartozó állapotváltozások irányát jelzik. Minden körből annyi nyíl indul, amennyi a lehetséges bemeneti kombinációk száma. Az ábra egyszerűsíthető azzal, hogy ha több kombináció eredményez azonos állapotátmenetet, akkor azokat egyazon nyílra írjuk. A nyíl irányítása adja meg az állapotváltozás irányát, és erre írjuk rá az állapotváltozást kiváltó bemeneti-, (Xi) és a hozzá tartozó kimeneti kombinációt (Zi). Amennyiben a kimeneti kombináció egyúttal állapotjellemző is, akkor ezt a körbe írjuk. Azok a nyilak, amelyek a kiinduló körhöz térnek viasza olyan bemeneti kombinációhoz tartoznak, amelyek nem eredményeznek állapotváltozást.



4. ábra

Az elmondottakra példa a 4. ábra szerinti állapotgráf. A gráfból leolvasható, hogy a szemléltetett szekvenciális hálózatnak négy állapota van. Ezek állapotjellemzői **Y1**, **Y2**, **Y3**, **Y4**. A bemeneti jeleknek ugyancsak négy kombinációja van: $\mathbf{X_0, X_1, X_2, X_3}$. A kimeneti jelkombinációk száma három, $\mathbf{Z_0, Z_1, Z_2}$. Vizsgáljuk az Y2-vel jelzett állapotot. Ebből az állapotból csak az $\mathbf{X_2}$, vagy az $\mathbf{X_3}$ kombináció **hoz új állapotot** létre, míg az $\mathbf{X_0}$ **és** $\mathbf{X_1}$ **nem** viszi új állapotba a hálózatot, de változtatja a kimeneti kombináció értékét (X1 esetén Z1, X2 esetén Z3).

A jelzett állapot-átmenetek az ugyancsak jelzett kimeneti kombinációváltással járnak (X3-Z2, ill. X4-Z3). A hálózat az Y2 állapotba vagy az Y1-ből X1, vagy Y4-ből az X3 hatására kerül. Az állapot-változásokhoz Z1, ill. Z4 tartoznak. Két állapot között kétirányú átmenet is lehet. Pl. Y4 és Y3 között.

Az ismertetett állapotgráfon követhető a hálózat működése, vagy tervezésnél ezzel írhatjuk le a kívánt működést. Az áramköri tervezéshez viszont csak közvetve tudjuk felhasználni.

5.1.2. Állapottáblázat

Egy sorrendi hálózat állapotai, valamint a bemeneti-, és kimeneti változói közötti kapcsolatrendszert táblázattal is megadhatjuk. Az ilyen leírást nevezzük állapottáblázatnak (5. ábra).

Áll. vált.	**Bemeneti kombinációk**							
	X_0		**X_1**		**X_2**		**X_3**	
y_0	**Y_2**	**Z_1**	**Y_1**	**Z_2**	**Y_1**	**Z_2**	**Y_4**	**Z_3**
y_1	**Y_2**	**Z_1**	**Y_2**	**Z_3**	**Y_3**	**Z_2**	**Y_3**	**Z_3**
y_2	**Y_3**	**Z_3**	**Y_4**	**Z_2**	**Y_3**	**Z_2**	**Y_3**	**Z_3**
y_3	**Y_3**	**Z_1**	**Y_4**	**Z_1**	**Y_2**	**Z_3**	**Y_1**	**Z_3**

5. ábra

A hálózat **m** db belső állapotot (**y_m**) vehet fel. Ezt jelöljük a táblázat soraival. Minden egyes állapotban a bemenetekre juthat az n db bemeneti változó (x_1. . . x_n) **2^n** számú kombinációja (X_1. . . $X_2{}^n$). Tehát ennyi oszlopot kell megrajzolnunk. Az így kapott táblázat egyes elemeibe kell beírnunk azt, hogy az adott állapotban, az oszlopnak megfelelő bemeneti kombináció hatására mi lesz a szekunder, ill. a kimeneti változók kombinációja (Y_i, Z_i). A Z_i -vel a **p** db kimeneti változó (z_0. . . . . z_p) lehetséges kombinációit (Z_1. . . $Z_2{}^p$) jelöltük. A szekunder változók azok a jelek, amelyek az állapot-tárolókat a hálózat következő állapotának megfelelő kombinációiba vezérlik. Ezen kombinációinak száma tehát meg kell egyezzen a kívánt állapotok számával, **m** - el.

A táblázatban, pl. az első oszlop mutatja azt, hogy a lehetséges állapototok (az y sorok) az **X_1** bemeneti kombinációk hatására milyen állapotba megy át. Példa: a táblázat első sorában - az y_1 állapotban - az X_1 kombinációnál (első oszlop) Z_1 kimeneti és Y_2 szekunder változói kombináció jön létre, s az X_1 vezérlés a hálózatot az y_2 állapotba viszi át (függőleges nyíl jelzi az állapotváltozást). Amennyiben az állapot és a szekunder változó indexe azonos, nincs állapotváltozás, csak esetleg kimeneti értékmódosulás (pl. első sor második oszlopa).

Az ismertetett állapottáblázatból már külön tudjuk választani az, un. **vezérlési táblázatot** (6. a. ábra) és a **kimeneti táblázatot** (6.b. ábra). Ezek a táblázatok - a kombinációk kódolt értékeinek beírása után - a bemeneti és a kimeneti kombinációs áramkör részek tervezésében már közvetlenül is használhatók. A szükséges állapottárolók számát (k-t) az

$$\mathbf{m = 2^k}$$

összefüggésből határozhatjuk meg, ahol **m** a szükséges állapotok, **k** pedig az alkalmazandó tárolók száma.

y_1 y_0	x_1 x_0 00	01	10	11
0 0	Y_2	Y_1	Y_1	Y_4
0 1	Y_2	Y_2	Y_3	Y_3
1 0	Y_3	Y_4	Y_3	Y_3
1 1	Y_3	Y_4	Y_2	Y_1

y_1 y_0	x_1 x_0 00	01	10	11
0 0	Z_1	Z_2	Z_2	Z_3
0 1	Z_1	Z_3	Z_2	Z_3
1 0	Z_3	Z_1	Z_2	Z_3
1 1	Z_1	Z_1	Z_3	Z_3

6. ábra

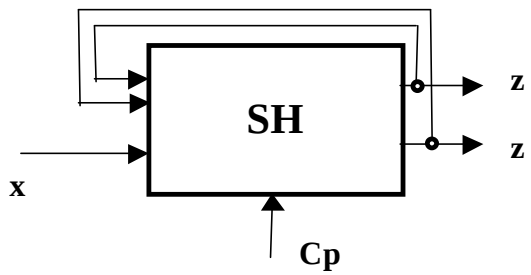
Áttekinthetőbb és jobban kezelhető az olyan sorrendi hálózatok állapottáblázata, amelyeknek az állapottárolók kimenetei egyúttal a hálózat kimenetei is. Ekkor a hálózat egy kombinációs hálózatból és a tárolókból áll. A tárolók számát az előállítani kívánt kimeneti kombinációkból határozhatjuk meg a fenti összefüggés alapján.

A kombinációs hálózatra a bemeneti és a visszacsatolt kimeneti változók csatlakoznak. Ez a hálózat kell előállítsa a tároló elemeket (flip - flopokat), vezérlő jeleket. E jelek készítik elő az új állapotot, ami a következő (billentő) órajel hatására áll elő.

Az állapottáblázat elemi négyszögeibe tehát csak a vezérlő jel kombinációkat kell írnunk. Tulajdonképpen ez megfelel az un. vezérlési táblázatnak.

5.1.3. Állapottáblázat használata tervezésnél

Példaként tervezzünk meg egy-két kimenetű és egy bemenetű szinkron sorrendi hálózatot (7. ábra).



a.

y	x X_0 x=0	X_1 x=1
y_0	Z_1	Z_1
y_1	Z_2	Z_0
y_2	Z_3	Z_1
y_3	Z_0	Z_0

b.

7. ábra

Az $\mathbf{X_0}$ bemeneti kombinációnál (x = 0) a $\mathbf{Z_0}$-$\mathbf{Z_1}$-$\mathbf{Z_2}$-$\mathbf{Z_3}$ kimeneti kombinációsorozatot, míg az $\mathbf{X_1}$-nél (x = 1) pedig a $\mathbf{Z_0}$-$\mathbf{Z_1}$-$\mathbf{Z_0}$-$\mathbf{Z_1}$ sorozatot kell előállítanunk. A működést a b. ábra szerinti állapottáblázat írja le. A feladatnál a kimenetek kombinációi ($\mathbf{Z_i}$) - a közvetlen visszacsatolás miatt – egyúttal az állapot jelek kombinációja is (Y_i=Z_i), tehát az állapottáblázatban nem kell külön felírni a két kombinációt.

A hálózat állapotainak száma **4**. Ehhez **k = 2** db tároló elem szükséges. Használjunk **T** típusú **ms** flip-flop -okat. A hálózat elvi blokkvázlata a 8. a. ábra szerinti. A kimeneti jel kombinációkat a

$$Z_0 = \bar{z}_0 \bar{z}_1$$

$$Z_1 = \bar{z}_0 z_1$$

$$Z_2 = z_0 \bar{z}_1$$

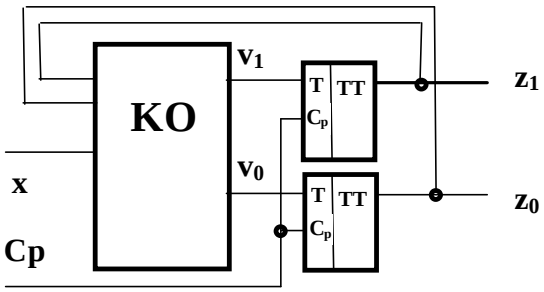
$$Z_3 = z_0 z_1$$

szerint választjuk. A v_0 és v_1 vezérlőjelek szükséges értékét az alkalmazott flip-flop típusnak megfelelően kell megállapítani.

Miután **T** flip - flopokat választottunk, ezért $v_i = 0$, ha nem kell állapotváltozás és $v_i = 1$, ha állapotváltozást kell előkészíteni. Az így megállapított kódolásokkal felirt állapottáblázat az 8.b.ábrán látható.

y	x	
	0	1
z_0 z_1	v_0 v_1	v_0 v_1
0 0	0 1	0 1
0 1	1 1	0 1
1 0	0 1	1 1
1 1	1 1	1 1

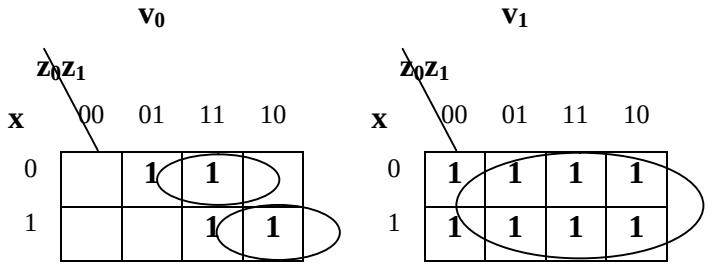
a.



b.

8. ábra

Az állapottáblázatból mind v_0 - ra, mind pedig v_1- re felírható egy-egy háromváltozós Karnaugh - diagram. Ezek segítségével pedig a kombinációs hálózat már megtervezhető (9. ábra).



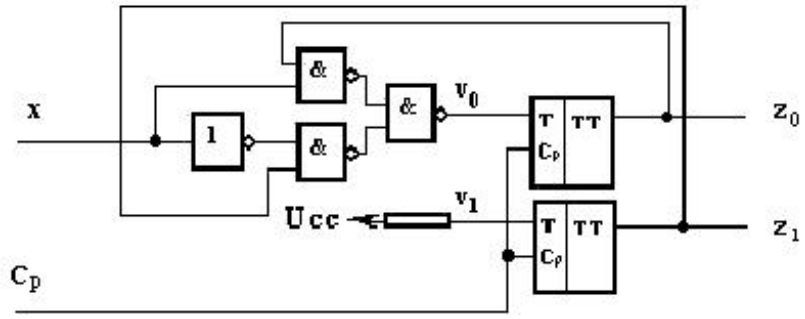
9. ábra

A két vezérlőjel logikai függvénye tehát:

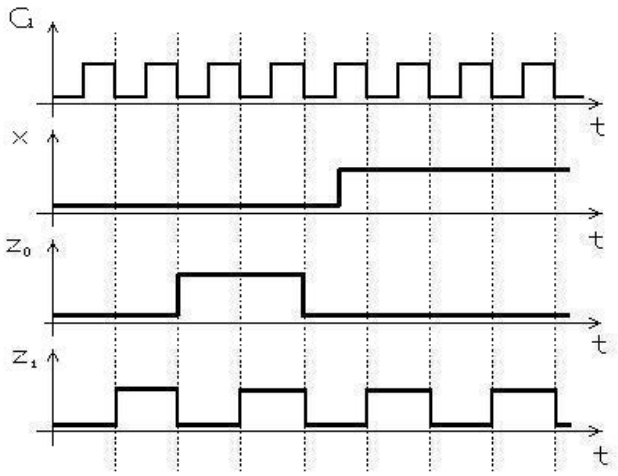
$$v_0 = \bar{x} \times z_1 + x \times z_0$$

$$v_1 = 1$$

A kitűzött sorrendi feladatot megvalósító áramkör logikai vázlata az 10.a. ábra szerinti. A 10.b.ábrán az órajel (**Cp**), valamint a be-, (**x**) és kimenetek (**z_0, z_1**) jeleinek időfüggvénye látható.



a.



b.

10. ábra

Összefoglalva: a tervezés lépései a következők:

1. Állapottáblázat felírása.
2. Tároló elemek számának és típusának meghatározása.
3. Vezérlési kombinációk meghatározása.
4. Be- és kimeneti kombinációk kódolása.
5. Kódolt állapottáblázat felírása.
6. Kombinációs hálózat tervezése.

Meg kell jegyeznünk, hogy általános esetben az optimális állapotkódolás meghatározására léteznek eljárások. Ezek nem tartoznak a tananyagunkba.

A továbbiakban az elektronikus vezérlésekben gyakran használt sorrendi hálózatokat, a

- számlálókat és a
- léptető regisztereket

tárgyaljuk részletesen.

5.2. Számlálók

A különböző irányítási, adatfeldolgozási és mérési feladatokban gyakran szereplő részfeladat a legkülönfélébb jelek, tágabb értelemben események számlálása. A funkciót ellátó áramköröket nevezzük **számláló**knak.

A számláláskor alapvetően két műveletet kell végezni, úgymint **tárolni** az eddig már bekövetkezett események **számát**, majd az újabb esemény hatására - a kiválasztott számlálási iránynak megfelelően - az eddigi értéket **növelni** (inkrementálás), ill. **csökkenteni** 1-gyel (dekrementálás).

A hozzáadás, ill. a levonás - mint ahogy ezt már megismertük - kombinációs logikai feladatként is kezelhető. Az előbbiek alapján tehát a számlálás sorrendi hálózattal megvalósítható.

Az egyirányú számlálók olyan speciális sorrendi hálózatok, amelyeknek állapotai csak egy meghatározott sorrendben követik egymást, s ez az állapotsorozat ciklikusan ismétlődik. A számlálandó jel fogadására a számlálónak egyetlen bemenete van. A kimenetek és a szükséges **állapottárolók** számát a **kapacitás** szabja meg. A **kapacitás** azt a legnagyobb számot jelenti, amellyel egy ciklus befejeződik. Ezt a továbbiakban **k**-val jelöljük. A számsorozat így **0, 1, 2. . . k** értékekből áll. A számlálónak tehát

$$\mathbf{m = k + 1}$$

különböző értéket kell megkülönböztetnie. Az **m** - et nevezzük a számláló **modulusának**.

5.2.1. A számlálók csoportosítása

A számlálókat többféle szempont alapján csoportosíthatjuk, úgymint:

- működési mód,
- számlálási irány,
- az információ-tárolás kódja,
- áramköri megvalósítás

szerint végezhetjük el a felosztást.

A **működési mód** szerint

- aszinkron és
- szinkron

számlálókat különböztetünk meg. Az **aszinkron** működés lényege, hogy a számlálandó jel csak elindítja a szükséges állapotváltozási sorozatot. A továbbiakban az egyes **flip - flop - ok billentik egymást**.

A **szinkron** működés alapja, hogy két számlálandó jel között történik a következő állapotba billentés előkészítése, s a flip-flop - okat **a számlálandó jel billenti**.

A számlálás **iránya** szerint

- előre (UP - fel),
- hátra (DOWN - le),
- előre-hátra (reverzibilis - UP/DOWN)

működések lehetnek. A **reverzibilis** számlálóknál a számlálási irányt külső vezérlőjel változtatja meg.

A számtartalmat (információt) **tárolhat**juk

- bináris

- BCD és
- egyéb

kódokban. Ezt a számláló megnevezésében jelöljük, pl. szinkron bináris előre számláló.

Az áramköri megvalósításnál

- diszkrét elemes és az
- integrált áramköri

számláló megkülönböztetés elsődlegesen formai és nem a működés lényegére utal.

A számlálókat - elsődlegesen az integrált áramköri kivitelben - ki szokták még egészíteni járulékos funkciókkal. Ilyen kiegészítés, hogy az egyes flip-flop - ok - a számlálási funkciótól függetlenül is - külső jellel beállíthatók 0 vagy 1 állapotba. Ezek az elő-beírású vagy PRESET számlálók. A másik gyakori megoldás, hogy a számlálás végszámát jelző áramkör is a számláló tartozéka (egyazon tokban van).

5.2.2. Bináris számlálók

Leggyakrabban használjuk a 2-es számrendszerben számláló bináris számlálók különböző változatait. Ebben a pontban részletesen foglalkozunk a számlálók e csoportjának logikai tervezésével, áramköri megvalósításával és néhány integrált áramköri változat működésével.

5.2.2.1. Bináris számlálók logikai tervezése

Egy **k** kapacitású **bináris számláló** logikai tervezésének lépései:

- a tárolók számának meghatározása,
- az állapottáblázat felvétele,
- az alkalmazott flip-flop típus kiválasztása,
- a kódolt állapottáblázat felírása,
- a vezérlő függvények meghatározása,
- a logikai vázlat megrajzolása.

A logikai tervezést egy **k = 7** kapacitású **szinkron bináris** számláló példáján ismertetjük.

A számlálónak **m = k + 1 = 8** állapotot kell megkülönböztetnie. A szükséges állapottárolók száma tehát **három** (2^3=8). A számláló kimenetein - az egyes ütemekben - a 0 - 7 értékek bináris kódját kell, kapjuk. Ezek az értékek három bináris helyértékkel kifejezhetők, tehát az állapottárolók és a hálózat kimenetei ugyanazok is lehetnek.

A számláló állapottáblázata (11. ábra) két oszlopot és nyolc sort tartalmaz. Miután egyetlen bemeneti jel van (C), két bemeneti kombináció lehetséges**,** $\mathbf{X_0}$**,** $\mathbf{X_1}$. Ezek közül egyik, amikor van számlálandó jel C = 1, a másik pedig, amikor C = 0. A nyolc kimeneti kombináció ($\mathbf{Z_0 - Z_7}$) pedig a bináris számértékeknek megfelelő állapotok. Az **állapotvezérlő** jelek kombinációi ($\mathbf{V_0 - V_7}$) billentik a flip-flop -okat a soron következő állapotkombinációba billentés csak a C=1 értéknél történik, ezért ekkor értékük már egyértelműen meghatározza a következő állapotot (első oszlop). Amikor nincs számlálandó jel - második oszlop - akkor a jelek pillanatnyi értéke 0 kell legyen, mivel nem szabad billenteni.

z	x	
	X_0	X_1
	C=1	C=0
Z_0	V_1	0
Z_1	V_2	0
Z_2	V_3	0
Z_3	V_4	0
Z_4	V_5	0
Z_5	V_6	0
Z_6	V_7	0
Z_7	V_0	0

11. ábra

Következő lépésként a kódolt állapottáblázatot kell felírni. A bemeneti kódolást már meghatároztuk, amely szerint

$$X_0 = 1, \qquad X_2 = 0.$$

A kimeneti kombináció-sorozat pedig **a bináris számsor** kell legyen, ahol a kimenetek a $z_0 = 2^0$, $z_1 = 2^1$, $z_2 = 2^2$ helyértékeket adják.

5.2.2.2. Szinkron bináris számlálók

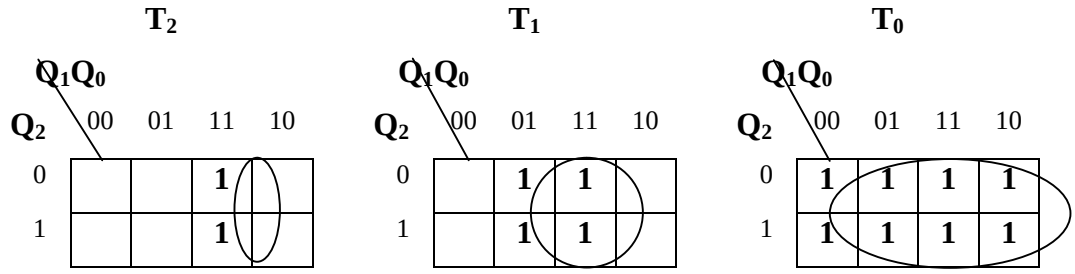
Számlálók **él-vezérelt** vagy **közbenső tárolós** (master-slave) flip-flop –ból építenek, mivel ezeknél lehet a billentés feltételébe a kimenetek jeleit visszacsatolni. Ugyanakkor a számlálandó jel mindegyik tároló billentő bemenetére vezethető, vagyis ketté választottuk az **előkészítést** végző jeleket, és a **billentő** jelet. Ez a számlálás **szinkron** üzemű megoldása.

A számlálót alakítsuk ki **T** típusú ms flip-flop –al. Ekkor az egyes tárolók T bemeneteire kell csatlakoztatni az állapotvezérlő jeleket. Ekkor az állapotváltozók kódolását abból a feltételből írjuk fel, hogy **1** szint **engedélyezi** a flip-flop billentését, **0** szint pedig **nem**. Az előbbi megállapodások szerinti kódolt állapottáblázat látható az .ábrán. (A táblázatban az állapotváltozókat a T_0,T_1,T_2 –al, a kimeneteket pedig Q_0,Q_1,Q_2 –al - a szakirodalomban legtöbbször használt betűkkel - jelöltük.)

Q_2	Q_1	Q_0	T_2	T_1	T_0
0	**0**	**0**	**0**	**0**	**1**
0	**0**	**1**	**0**	**1**	**1**
0	**1**	**0**	**0**	**0**	**1**
0	**1**	**1**	**1**	**1**	**1**
1	**0**	**0**	**0**	**0**	**1**
1	**0**	**1**	**0**	**1**	**1**
1	**1**	**0**	**0**	**0**	**1**
1	**1**	**1**	**1**	**1**	**1**

12. ábra

A kódolt állapottáblázatból az egyes engedélyező bemenetekre külön-külön felírhatunk egy-egy Karnaugh - diagramot. Ezek segítségével aztán a legegyszerűbb logikai függvények meghatározhatók. Ezzel tulajdonképpen a bemeneti kombinációs hálózat logikai felépítését is meghatározzuk.



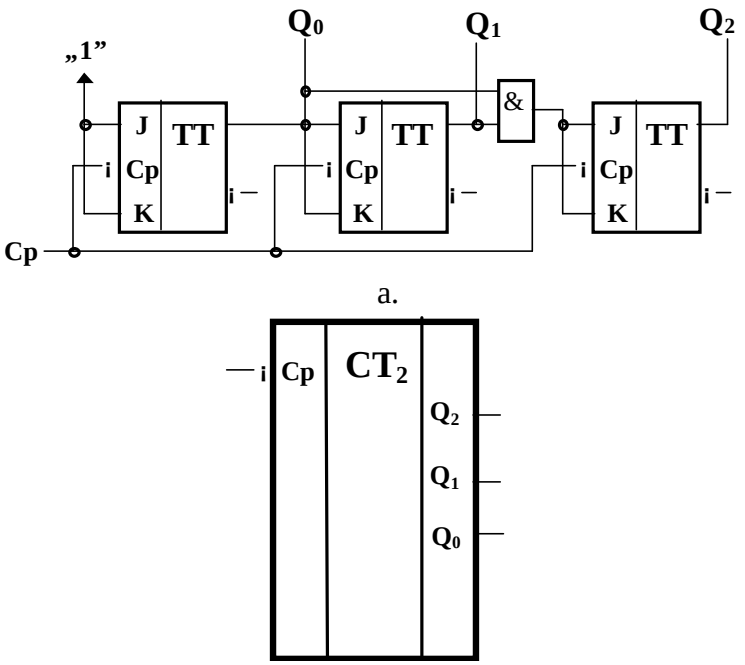
13. ábra

A 13. ábrán láthatók a T_0, a T_1 és a T_2 változókra felirt K diagramok, amelyek alapján az egyes vezérlőfüggvények:

$$T_0 = 1,$$

$$T_1 = Q_0,$$

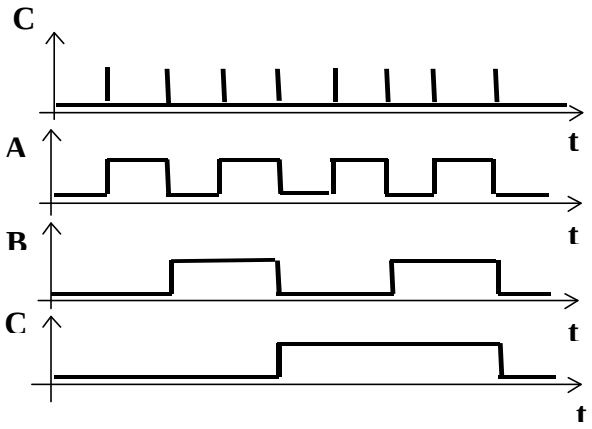
$$T_2 = Q_0Q_1$$



14. ábra

A számláló logikai vázlata az 14. a. ábrán, a számláló szimbolikus jele pedig a b. ábrán láthatók. A CT (COUNTER) jelölés melletti index a számrendszerre utal, pl. CT2 bináris számláló.

A kimenetek és a számlálandó jel időfüggvényeit mutatja a 15. ábra. Az időfüggvények felett feltüntettük az egyes ütemek kimeneti állapot - kombinációit. Ezek sorozata - a kitűzött célnak megfelelően - a növekvő bináris számsort adják.



15. ábra

A számláló kapacitását további flip-flop -okkal növelni lehet. Ezek vezérlőfüggvényeit az előzőekhez hasonlóan határozhatjuk meg. Ezt most mellőzve, az időfüggvényekből is következtethetünk a törvényszerűségre. A soron következő flip-flop mindig olyankor vált állapotot, amikor minden előző flip-flop nál 1 - 0 állapotátmenet van. Ennek alapján a az **i.** flip-flop vezérlőfüggvényének általános alakja:

$$T_i = Q_0 Q_1 Q_2 . . . Q_{i-1}$$

A függvény alapján megállapíthatjuk, hogy a kapacitásbővítéshez - az újabb flip-flop mellett - mindig 1-gyel több bemenetű ÉS kapu kell. Ezt a megoldást nevezzük

párhuzamos átvitelűnek. A megnevezés arra utal, hogy minden egyes előkészítő bemenetre egyidejűleg (párhuzamos csatornákon) jutnak a megfelelő kimenetek értékei. Ezáltal egy kapunyi jelkésleltetés múlva az újabb billentés előkészítése befejeződik.

Az összefüggések átalakíthatók a következők szerint:

$$\mathbf{T_0 = 1,}$$

$$T_1 = Q_0 = \mathbf{T_0\ Q_0}$$

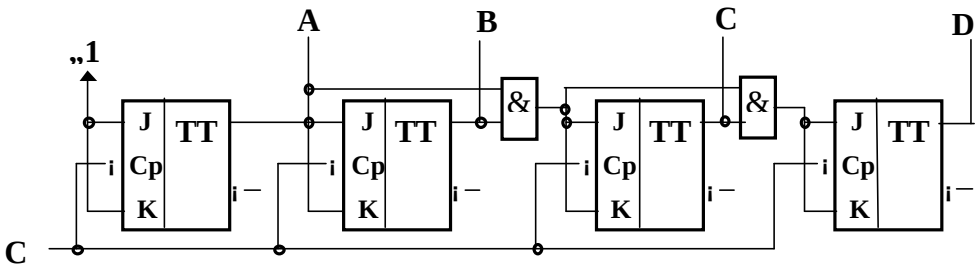
$$T_2 = Q_0Q_1 = \mathbf{T_1\ Q_1}$$

.

.

$$T_i = Q_0Q_1Q_2\ .\ .\ .\ \ Q_{i-1} = \mathbf{T_{i-1}\ Q_{i-1}}$$

Az átalakított vezérlőfüggvények szerint kialakított m = 16 modulusú bináris számláló logikai vázlatát mutatja a 16. ábra. A kimenetek elnevezésénél – a számlálóknál használt – **A,B,C,D** jelölést rajzoltuk.



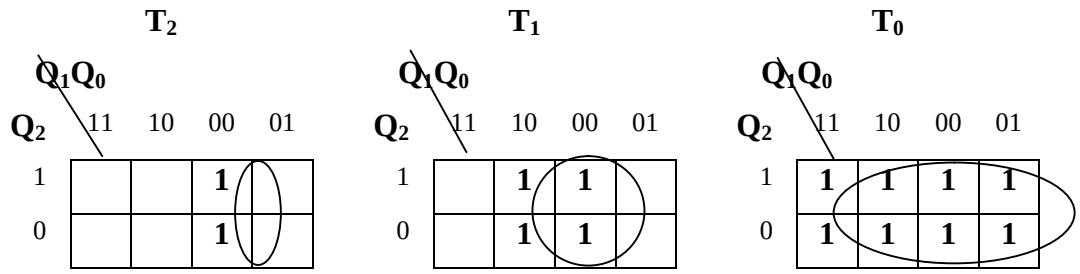
16. ábra

Ebben a megoldásban egységesen két bemenetű ÉS kapuk állítják elő a vezérlőjeleket. Az áramköri egyszerűsítés ára, hogy a számláló határfrekvenciája csökken, mert a legutolsó flip-flop előkészítő bemenetére a jel két sorba kötött kapun átjut. Ezt az áramköri megoldást nevezzük **soros átvitelűnek**. További kapacitásbővítés újabb kapukat, s így késleltetéseket iktat be.

Hárombites bináris **hátraszámláló** kódolt állapottáblázatát láthatjuk az 17. ábrán. Ezt is T típusú flip-flop -okból alakítjuk ki.

Q_2	Q_1	Q_0	T_2	T_1	T_0
1	**1**	**1**	**0**	**0**	**1**
1	**1**	**0**	**0**	**1**	**1**
1	**0**	**1**	**0**	**0**	**1**
1	**0**	**0**	**1**	**1**	**1**
0	**1**	**1**	**0**	**0**	**1**
0	**1**	**0**	**0**	**1**	**1**
0	**0**	**1**	**0**	**0**	**1**
0	**0**	**0**	**1**	**1**	**1**

17. ábra



18. ábra

Az előkészítő bemenetek vezérlőfüggvényei a Kp diagramok alapján (18. ábra) a következők:

$$T_0 = 1$$
$$T_1 = \overline{Q}_0$$
$$T_2 = \overline{Q}_0 \overline{Q}_1$$

A vezérlőfüggvény általános alakja:

$$T_i = \overline{Q}_0 \overline{Q}_1 \Lambda\, \overline{Q}_{i-1}$$

lesz. E függvények közvetlen megvalósításával párhuzamos átvitelű **bináris hátra-számlálót** kapunk. Logikai átalakítások után – az előreszámlálóhoz hasonlóan - a soros átvitelű bináris hátraszámláló is kialakítható.

Az előre-, ill. hátraszámláló vezérlőfüggvényeinek ismeretében felírhatjuk a **reverzibilis** bináris számláló függvényeit is. A számlálási irányt a **P** külső parancs vezérli (P = 0 előreszámlálás, P = 1 hátraszámlálás

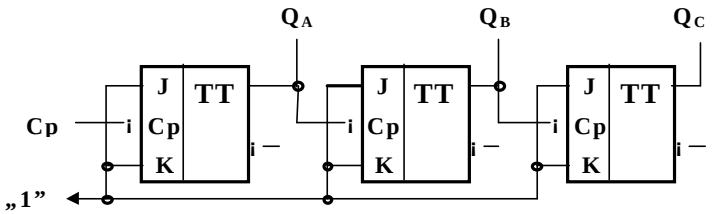
5.2.2.3. Aszinkron bináris számlálók

Az aszinkron működésű számlálóknál a számlálandó jel csak elindítja a soron következő állapotváltozást, de az egyes flip - flop -ok egymást billentik. A bináris előreszámláló kimeneteinek időfüggvényénél (15. ábra) láttuk, hogy mindegyik

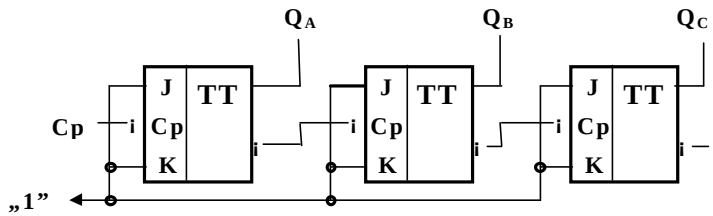
flip-flop a megelőző tároló 1 - 0 átmeneténél kell billenjen. Így **1 - 0** átmenetre billenő **T** flip-flop -ok 19.a. ábra szerinti kapcsolásával bináris előreszámlálót kapunk.

A bináris hátraszámlálónál az előző flip-flop -ok a megelőző tároló 0 - 1 állapotváltozás nál kell billenjenek. Ugyancsak 1 - 0 átmenetre billenő T flip-flop -okból kialakított hátraszámláló logikai vázlata a 19 b. ábrán látható.

Az aszinkron számlálók nagyon egyszerű felépítése mellett hátránya a kisebb határ-frekvencia. Ez abból adódik, hogy az új stabil állapot csak az egymást követő billenések befejezte után áll be. Ugyan-csak hátrány, hogy az átmeneti időszakban nem kívánt kombinációk is előfordulnak a kimeneteken. Ez a csatlakozó hálózatnál zavart okozhat.



a.



b.

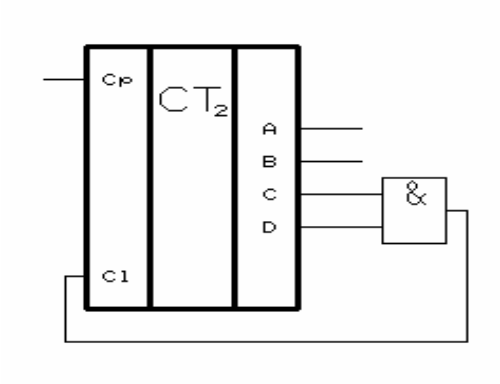
19. ábra

5.2.3. BCD kódolású számlálók

A különböző digitális mérő- és egyéb adatfel-dolgozó berendezésekben az adatok kijelzése, ill. bevitele rendszerint 10-es számrendszerben történik. Ezért a belső adatforgalomnál, így a számlálásnál is gyakran célszerű a dekád szervezése. Ezt teszik lehetővé a különböző **BCD** kódolású számlálók.

A tananyagban csupán a **BCD 8 4 2 1** súlyozású számlálókkal foglalkozunk. A megismert tervezési módszer alapján azonban a további BCD kódolású számlálók is megtervezhetők.

A **8 4 2 1** súlyozású BCD számláló működése a 4 bites bináris számlálótól abban tér el, hogy a tizedik impulzus hatására a kezdő **0 0 0 0** állapotba tér vissza a számláló.



20. ábra

Ennek a **modulus csökkentésnek** egy lehetséges megoldása, hogy egy 4 bites bináris számlálót - amelynek aszinkron törlő bemenete is van - olyan logikai hálózattal egészítünk ki, ami az **1 0 1 0** (decimális 10) állapot megjelenésekor minden flip-flop -ot töröl és ezzel **0 0 0 0** állapot áll be. Ezt a megoldást szemlélteti a 20. ábra.

E megoldás hátránya, hogy a 11. állapot egy rövid ideig - a kapu késleltetés és a billenési idő összegéig - bekövetkezik. Ez járulékos hibát okoz, különösen frekvenciaosztóként való alkalmazáskor.

A logikai tervezés során, a bináris számlálónál megismert induló fázisokat - az általános állapot táblázat és vezérlőfüggvényeinek felírását - elhagyjuk. A tervezést a kiválasztott flip-flop típusra érvényes kódolt állapottáblázat felírásával kezdjük.

5.2.3.1. Szinkron BCD számlálók

A szinkron BCD számlálók felépítését és működését **JK** típusú **ms** flip-flop -ok alkalmazásával ismertetjük.

Az állapottáblázat felírása előtt röviden összefoglaljuk a JK flip-flop vezérlésének feltételeit. Ezt mutatja a 21. ábrán levő táblázat. A $\mathbf{Q_i}$ a billentés előtti, $\mathbf{Q_{i+1}}$ pedig a billentő impulzus hatására bekövetkező új állapotot jelzi. Az **x** közömbös értéket jelent, vagyis 0 vagy 1 is lehet.

Q_i	Q_{i+1}	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

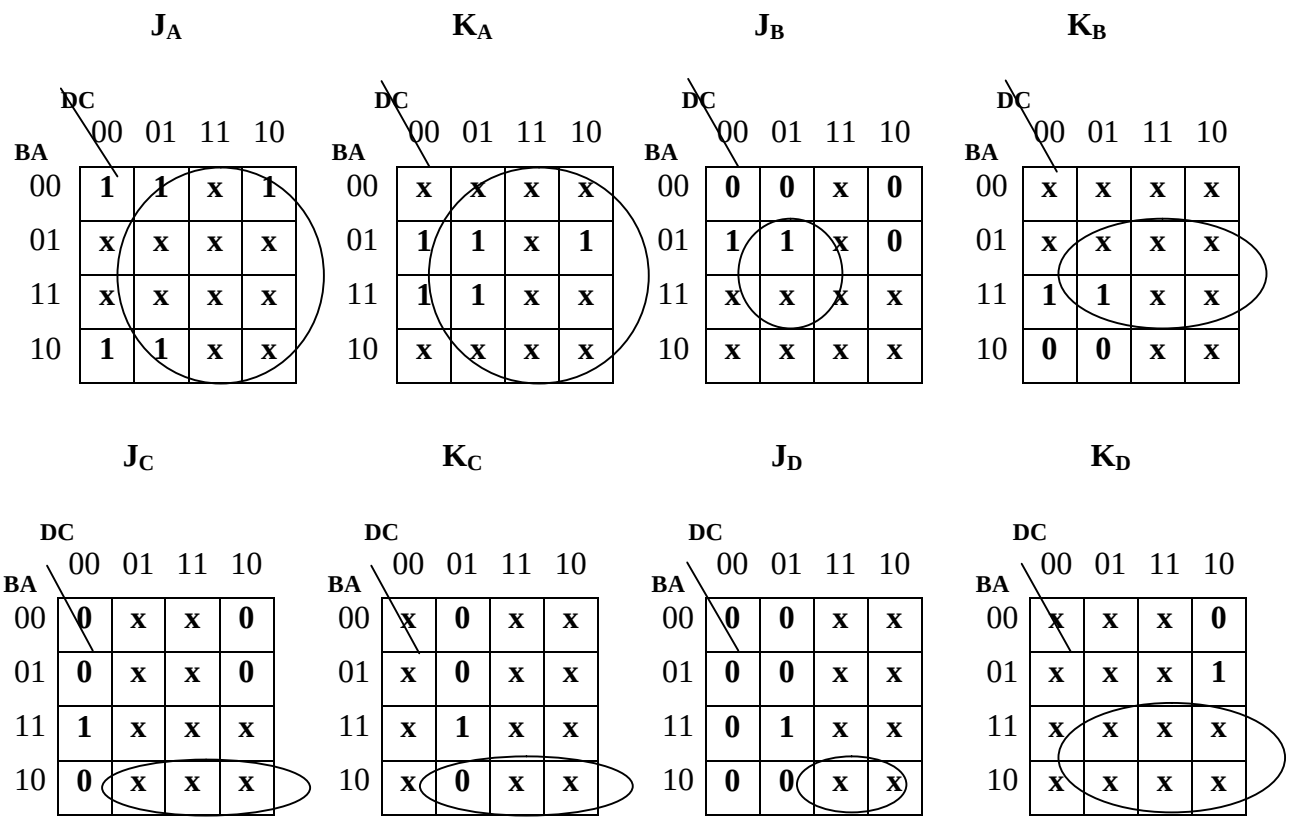
21. ábra

A BCD 8 4 2 1 súlyozású előreszámláló kódolt állapottáblázata látható a 22. ábrán. A kimeneteket - a nemzetközileg egységesen használt - A, B, C, D betűkkel jelöltük, ahol az $\mathbf{A=2^0}$ helyérték.

D	C	B	A	J_D	K_D	J_C	K_C	J_B	K_B	J_A	K_A
0	0	0	0	0	x	0	x	0	x	1	x
0	0	0	1	0	x	0	x	1	x	x	1
0	0	1	0	0	x	0	x	x	0	1	x
0	0	1	1	0	x	1	x	x	1	x	1
0	1	0	0	0	x	x	0	0	x	1	x
0	1	0	1	0	x	x	0	1	x	x	1
0	1	1	0	0	x	x	0	x	0	1	x
0	1	1	1	1	x	X	1	x	1	x	1
1	0	0	0	x	0	0	x	0	x	1	x
1	0	0	1	x	1	0	x	0	x	x	1

22. ábra

Az egyes flip-flop -ok J és K bemeneteinek vezérlési feltételeit 23. ábrán levő Kp diagramok segítségével határozzuk meg.

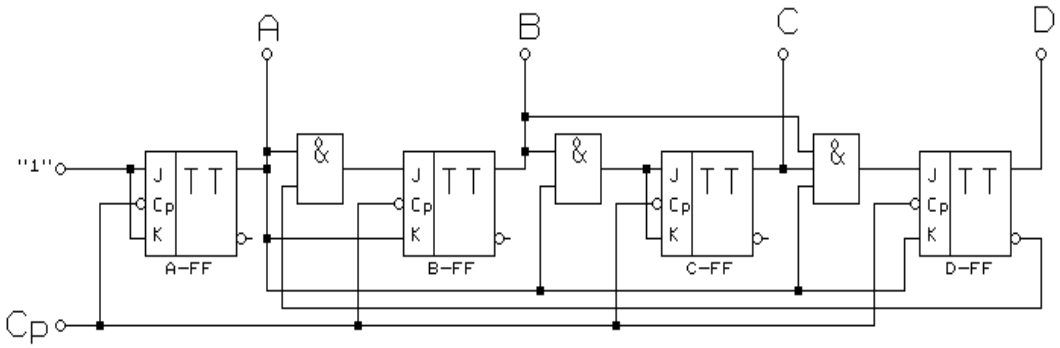


23. ábra

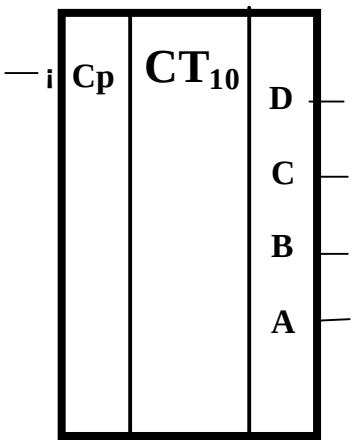
A BCD kódban nem szereplő kombinációkat is **x** -el jelölhetjük, s így a függvény egyszerűsítéseknél felhasználhatjuk. A logikai függvények:

$$J_A = 1 \qquad K_A = 1$$
$$J_B = A\overline{D} \qquad K_B = A$$
$$J_C = AB \qquad K_C = AB$$
$$J_D = ABC \qquad K_D = A$$

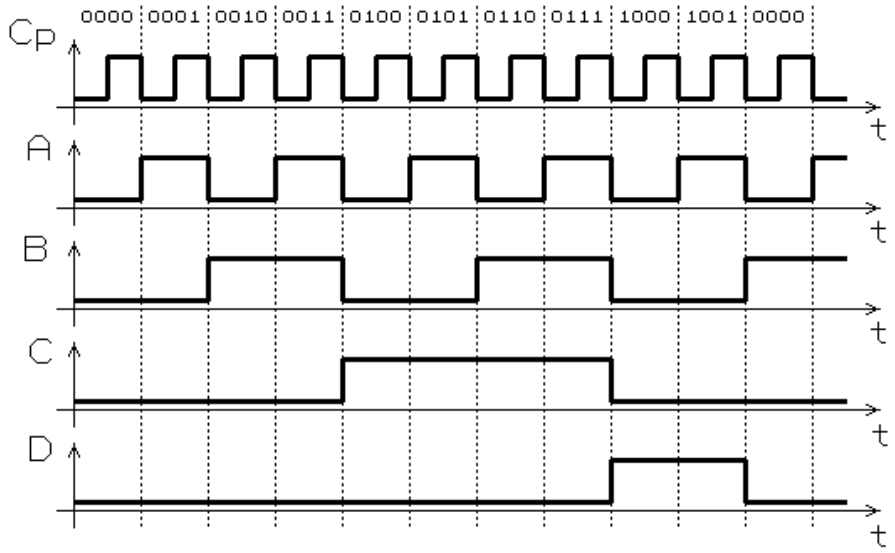
A szinkron BCD előreszámláló logikai vázlatát, szimbolikus jelölését, és a kimenetek időbeli változását a 24. a, b, c. ábrák mutatják.



a.



b.



c.

24. ábra

5.2.3.2. Aszinkron BCD számlálók

A legegyszerűbb **aszinkron** üzemű **BCD** előreszámlálót egyetlen billentő bemenettel rendelkező flip-flop -okból építhetünk. Az egyes tárolók billentési feltételeit a 24. c. ábra jelalakjai alapján is felírhatjuk. 1 - 0 átmenetre billenő flip-flop -nál az egyes billentési feltételek:

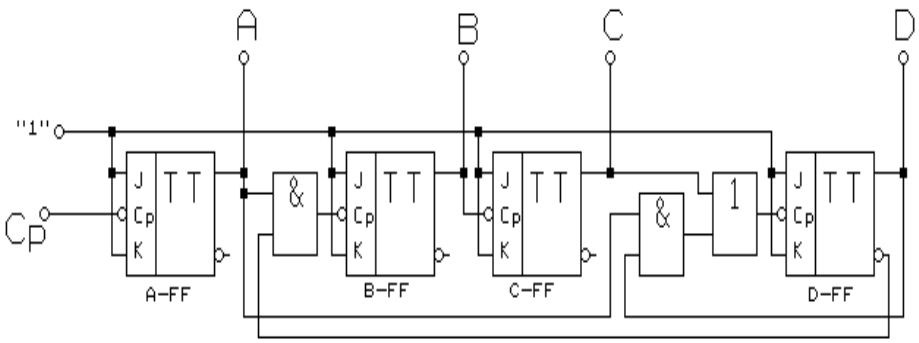
$$\mathbf{C_{pA} = C_s}$$

$$\mathbf{C_{pB} = A\overline{D}}$$

$$\mathbf{C_{pC} = B}$$

$$\mathbf{C_{pD} = C + AD}$$

Az élvezérelt flip-flop -okból kialakított aszinkron BCD előreszámláló logikai vázlata a 25.ábrán látható.



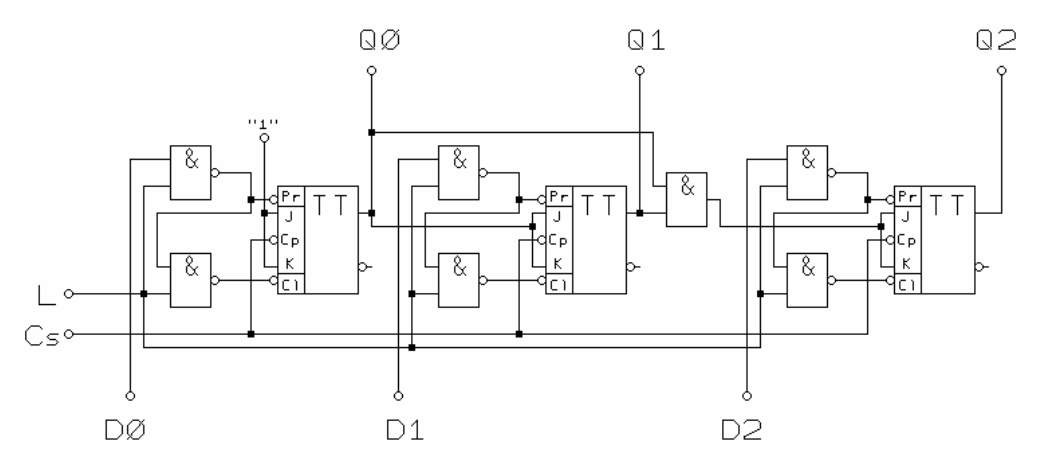
25. ábra

Az aszinkron hátra- és reverzibilis BCD számlálók kialakítása a megismert logikai tervezési eljárás segítségével lehetséges.

5.2.4. Preset számlálók

Számlálók alkalmazásakor szükség lehet arra, hogy a számlálást esetenként ne a 0-tól, vagy a kapacitás végértékétől kezdjük, hanem egy közbenső számtól. Ehhez szükséges, hogy külön külső parancs hatására, a számláló flip-flop - jait tetszőleges állapotkombinációba lehessen billenteni. Ezeket nevezzük preset (elő beírású) számlálóknak.

A számláló tartalmának - egyidejű párhuzamos - változtatása, programozása is történhet aszinkron, ill. szinkron módon.

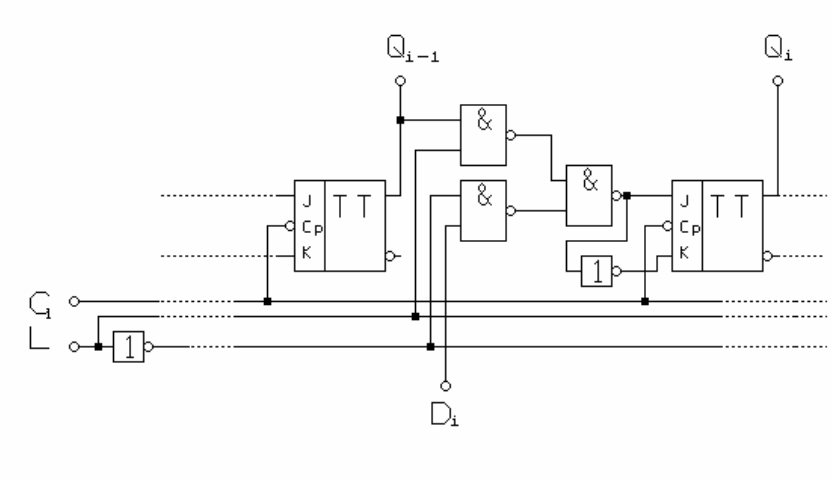


26. ábra

Az aszinkron programozás - az adatbeírás - a számlálandó jeltől függetlenül történik. A 26. ábrán látható 3 bites szinkron bináris előre számláló flip-flop - jai aszinkron üzemű beíró (**Pr**) és törlő (**Cl**) bemeneteit használjuk fel a párhuzamos adatbeírásra. Az adatbemenetek $\mathbf{D_0}$, $\mathbf{D_1}$, $\mathbf{D_2}$ és a beírást vezérlő jel az L (Load). Ha az L-re 0 szintet adunk, akkor a flip-flop - okba az adatbemeneteken érvényes információ íródik.

Amíg az L-en aktív jel van, addig az áramkör számlálóként nem működik. Az aszinkron bemenetek (Pr, Cl) hatása erősebb a Cp billentő jelnél.

A szinkronprogramozású megoldásnál a beírandó adatot mindig a soron következő Cp jel írja be a flip-flop - okba. Ennek egy áramköri megvalósítására mutat példát a 27. ábra.



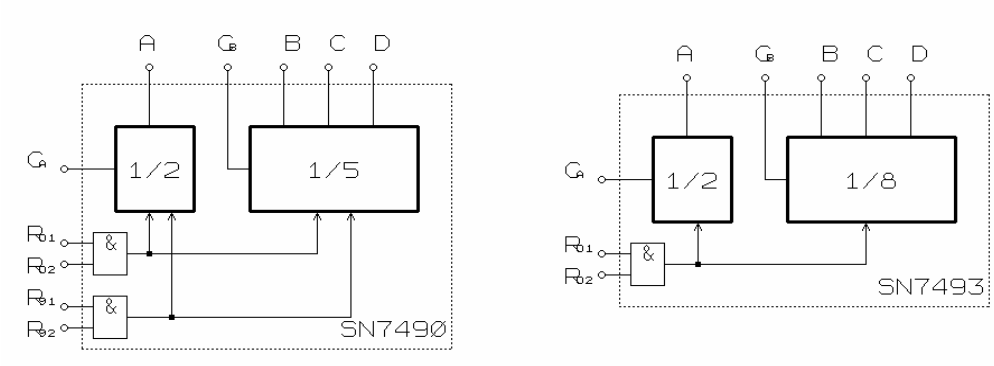
27. ábra

Mindegyik flip-flop előtt azonos felépítésű kiválasztó áramkör van. Az L beíró jel 1 szintjénél a $\mathbf{D_i}$ adatút tiltott és a számlálási feltételek kerülnek a flip-flop -ok előkészítő bemeneteire.

L = 0 vezérlésnél az adatok értéke jut a JK be-menetekre. A tényleges beírás ekkor is a Cp jel 1 - 0 átmenetekor következik be. Ameddig az L aktív, addig a párhuzamos beírás érvényesül.

5.2.5. Integrált áramköri számlálók

A különböző integrált áramköri családok (TTL, CMOS, ECL stb.) mindegyikében megtalálhatók a számlálók különböző változatai. Ebben a pontban a TTL és CMOS IC-k néhány - viszonylag gyakran használt - számláló típusának legfőbb jellemzőit tárgyaljuk.



a. b.

28. ábra

Az aszinkron működésűek közül az SN7490 és SN7493 típusú számlálók elvi blokkvázlata látható a 28. ábrán.

A decimális számláló (28.a. ábra) egy kettes és egy ötös osztóból áll. Aszinkron üzemű párhuzamos beírást tesznek lehetővé az ÉS kapukhoz csatlakozó bemenetek. Az $\mathbf{R_{0i}}$ jelű

bemenetre adott 1 szinttel mind a négy flip-flop 0-ba állítható (törlés). Az $\mathbf{R_{9i}}$ jelű bemenetek vezérlésével a BCD 1 0 0 1 kódot (9 beírása) tárolja a számláló.

Az áramkör **BCD 8 4 2 1** súlyozású számláló, ha $\mathbf{C_A}$ - ra adjuk a számlálandó jelet és az **A** kimenetet $\mathbf{C_B}$ - vel kötjük össze. Amennyiben $\mathbf{C_B}$-t vezéreljük impulzussorozattal és **D** -t a $\mathbf{C_A}$ - val kötjük össze, akkor olyan tízes osztót kapunk, amelynél az **A** kimeneten **szimmetrikus négyszögjelet** kapunk.

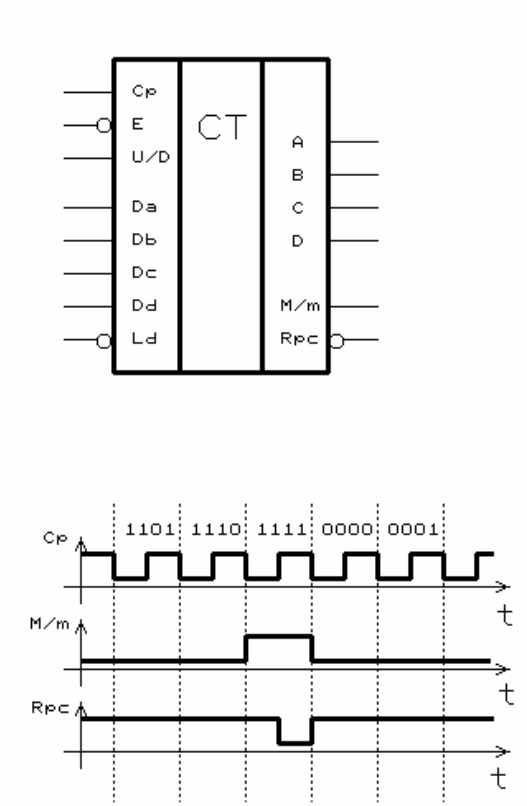
Az SN 7493 típus (28. b. ábra) 3, ill. 4 bites bináris **számlálóként** használható. A 3 bites számláló bemenete a $\mathbf{C_B}$. A kettes osztóval bővíthető a kapacitás. Az **R0** bemenetekre adott 1 szint hatására mind a négy tároló **törlődik**.

Röviden ismertetünk még a szinkronszámlálók közül két változatot. Ezek szimbolikus jelölései láthatók a 29. és 30. ábrákon.

Az SN 74190 és SN 74191 típusú TTL, ill. kompatibilis CMOS változatoknak felel meg a 29. ábra szerinti változat. Az áramkörök csak a kódolásban térnek el. A 74190-es BCD, a 74191-es pedig bináris. A két típusnak mind a bemenetei, mind pedig a kimenetei azonos funkciójúak, sőt a tokok láb - kompatíbilisak is.

E típusok 4 bites, kétirányú **preset szinkronszámlálók**, **végszámjelző áramkörrel** kiegészítve.

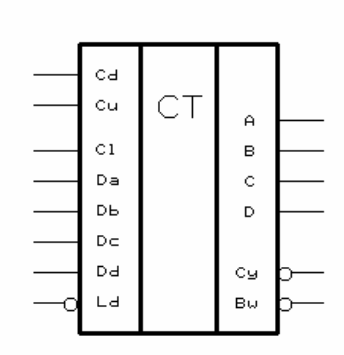
A számlálás irányát az **U/D** bemeneten érvényes logikai szint határozza meg. 0 szint előre-, 1 szint hátraszámlálást vezérel. Az **E** (Enable) jelű engedélyező bemenet 1 szinttel tiltja a számlálást (0 szint engedélyezi). A DA, DB, DC, DD adat-bemeneteken keresztül aszinkron üzemű párhuzamos beírás (programozás) történhet. Ezt az L (Load) bemenetre adott 0 szint vezérli.



29. ábra

A számláló A, B, C, D kimenetein kapjuk a szám-tartalom párhuzamos kódját. Az M/m (Max/min) kimeneten akkor kapunk 1 szintet, ha a számláló - a számlálási iránynak megfelelő - végszámának állapotában van. Az **Rpc** (Ripple Clock - órajel ismétlő) kimenet a végszám állapot ban ismétli az órajel 0 - 1 átmenetét. Az **M/m**, **Rpc** és **Cp** jelek időfüggvényeit láthatjuk a b. ábrán.

A 30. ábra az SN 74192, SN 74193 típusú TTL, és ezekkel kompatibilis CMOS rendszerű számlálók jelképi jele. Ugyancsak láb kompatíbilisak egymással a **BCD** (74192) és **bináris** (74193) áramkörök. Mindkét változat szinkron, kétirányú preset számláló. Az aszinkron párhuzamos **beírás** (programozás) az **L** bemenetre adott **0** szinttel vezérelhető. Az aszinkron üzemű **törlés** a **Cl** bemenetre adott **1** szinttel történik.



30. ábra

A két számláló bemenet közül a **Cu** -ra adott jelet előre, a **Cd**-re adottat pedig hátra számlálja. Párhuzamos kimenetein vehető le a számtartalom kódja. A **Cy** (Carry) kimenet az előreszámlálás végszámánál, míg a **Bw** (Borrow) kimenet a hátraszámlálás végszámánál ismétli az órajel 0 - 1 átmenetét.

5.2.6. A számlálók alkalmazása

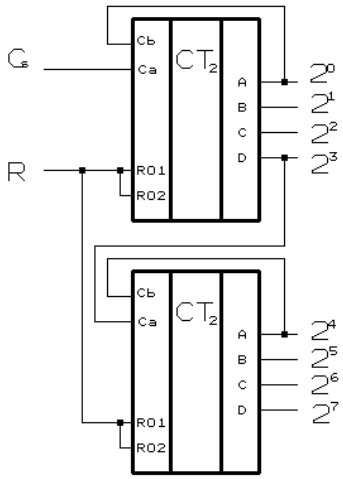
A számlálók nagyon sokrétű alkalmazása közül röviden foglalkozunk

- a kapacitásbővítés,
- a változtatható modulus és
- reverzibilis számlálók

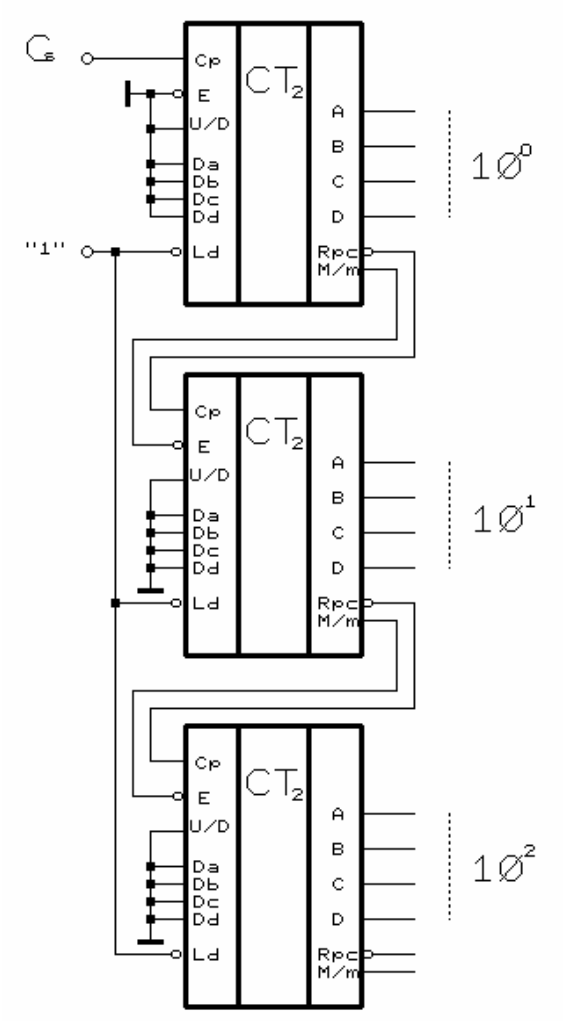
vezérlésének néhány áramköri megvalósításával.

A megismert integrált áramköri számlálók **kapacitásának bővítése** - aszinkron és szinkron változatoknál - különböző módon történik.

Az aszinkron számlálóknál a legnagyobb nagyságrendű kimenetet kell a következő tok számláló bemenetére kötni. Az 31. ábrán 8 bites bináris számláló kialakítása látható két SN7493 típusú tokkal. Mindkét tok egyidejűleg törölhető.



31. ábra



32. ábra

A szinkron számlálók bővítésénél az órajel ismétlő kimeneteket használjuk. A 32. ábrán - három SN 74190 típusú tokkal kialakított - 3 dekádos decimális számlánc kapcsolása látható. A nagyobb nagyságrendű dekád E bemenetére kötjük az előző dekád M/m kimenetét. Ez nagyobb zavarvédettséget biztosit azáltal, hogy csak akkor billenthető egy-egy dekád, ha az előző a végszámnál tart. A billentést az Rpc jele végzi.

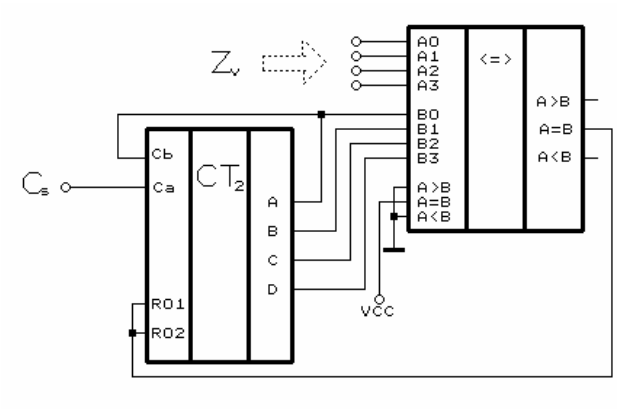
Programozható frekvenciaosztóként **változtatható modulusú** számlálót alkalmazunk. A modulus változtatás két alapvető elvi változata a

- végszám csökkentés és a
- kezdőszám változtatás.

A végszám csökkentésnél a számlálót törölni kell a kapacitásnál kisebb számérték elérésekor. A 33. ábrán látható ennek áramköri megvalósítása. A számláló kimeneteihez csatolt nagyság-komparátor aszinkron módon törli a számlálót, amikor annak tartalma megegyezik a kódkapcsolóval, vagy külső eszközzel beállított **Zv** értékkel. Az így kialakított programozható modulusú számláló eredő modulusa

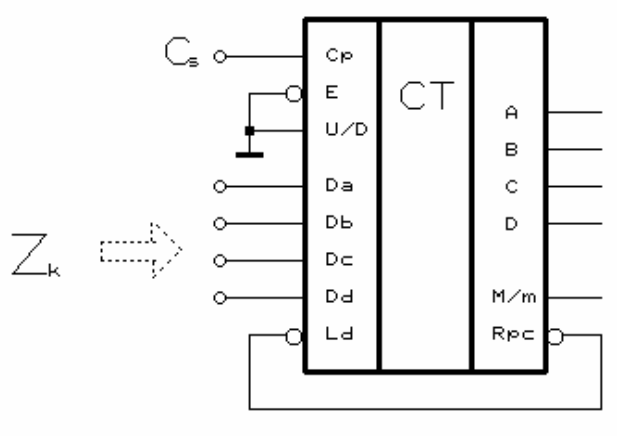
$$\mathbf{m' = Zv}$$

értékű lesz.



33. ábra

Preset számlálók alkalmazásával készíthető kezdőszám változtatással működő, változtatható modulusú számláló. Ebben a megoldásban a számláló túlcsordulása vezérli az adatbemeneteken érvényes – külső eszközzel történő - kezdőszám beírását. A számlálási ciklus innen folytatódik. A 34. ábrán a változtatható modulusú számláló kapcsolási vázlata látható.



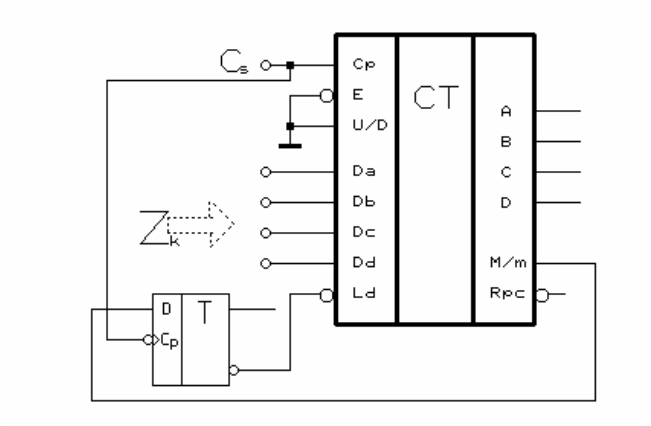
34. ábra

Az ábra szerinti kapcsolásban használt számláló (pl. SN 74190 vagy SN 74191 típus) a kapacitás végszámát elérve vezérli a párhuzamos beírást. A számlálás ezért a 0000 érték helyett a programozott **Zk** értékről folytatódik. A kapacitásnak megfelelő utolsó számérték csak az órajel 0 szintje alatt érvényes a kimeneten. A megváltozott modulus tehát

$$\mathbf{m' = m - Zk}$$

érték lesz, ahol **m** a számláló eredeti modulusa.

Szinkronozott kezdőszám beírási módnál (35. ábra) az utolsó számérték is teljes órajel ütemig áll fenn. Ebben az ütemben íródik a D flip-flop 1-be és készíti elő a program szerint érvényes Zk kezdőszám beírását, amelyet a következő órajel hajt végre.



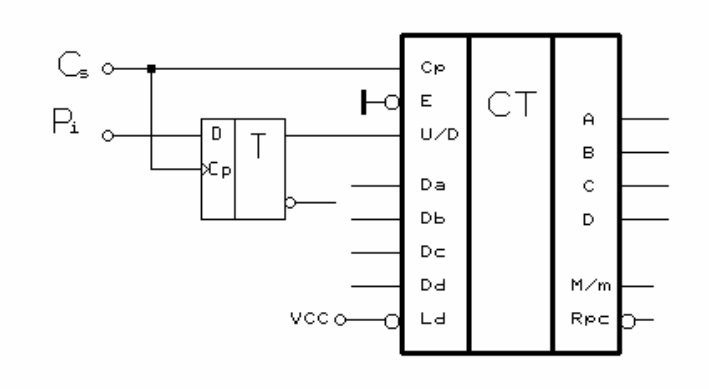
35. ábra

A nagyobb kapacitású - változtatható modulusú - számlálók kialakításánál az ábra szerinti megoldást célszerű alkalmazni. A kiegészítő áramkört csak annyiban kell módosítani, hogy a flip-flop beírási feltétele a számlálólánc minden **M/m** kimenetének egyidejű 1 értékénél kell, teljesüljön. Ez pedig - az ábrához képest - egyetlen ÉS kapubővítést jelent. Amennyiben a végszám módosítási eljárást alkalmaznánk, akkor

ugyanannyi nagyság komparátorra lenne szükségünk, mint amennyi a számláló tokok száma.

Az előző fejezetben megismert szinkron számlálók mindegyike előre-, és hátraszámlálóként is használható. Ezekkel megvalósított reverzibilis - paranccsal, programmal változtatható irányú - számlálók vezérlésénél biztosítani kell, hogy az irányváltás egyik számértéket se csorbítsa.

Azoknál a feladatoknál, amelyeknél az egy csatornán különböző időpontokban érkező jeleket kell **előre** vagy **hátra** számlálni, használjuk az SN 74190, ill. az SN 74191 típusú vagy ezekkel azonos funkciójú kétirányú számlálókat. Az U/D bemeneteken a parancsot - hibás számlálás elkerülése végett - csak impulzusszünetben szabad váltani. A tetsző-leges időpontban érkező külső parancs (Pi) szinkronozásának egy lehetséges áramköri megoldása látható a 36. ábrán. A számláló irányváltó jelét a D flip-flop kimenete szolgáltatja. Ez a flip-flop viszont - a külső Pi parancsnak megfelelő - állapotot csak az órajel 1 - 0 átmenetekor veszi fel.



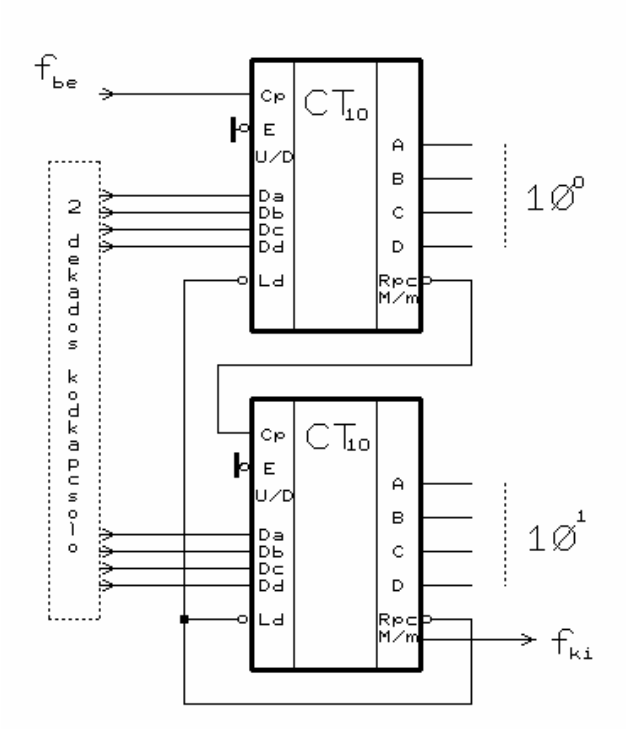
36. ábra

5.2.7. Változtatható modulusú számlálók alkalmazása

A változtatható modulusú számlálókkal alakítható ki

- programozható frekvenciaosztó, illetve
- programozható késleltető.

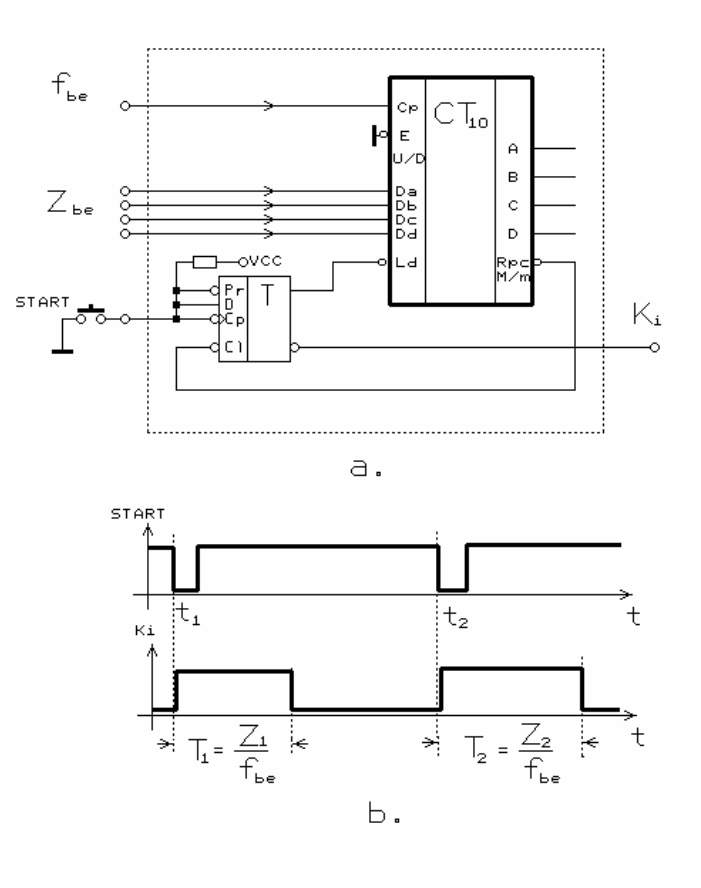
A **programozható frekvenciaosztó** egyik legegyszerűbb megoldását láthatjuk az alábbi ábrán. A két dekádos - szinkron BCD preset számlálókból álló – számlánc modulusa kezdőszám változtatás elvén változtatható. A bemeneti és a kimeneti jelek frekvenciájának aránya a mindenkori modulus. A számlánc utolsó dekádjának tulcsordulásakor következik be a kezdő szám bevitele, és innen folytatodik a számlálás a végértékig. A ciklus újból kezdődik. A $\mathbf{Z_{be}}$ szám megváltoztatása a kimeneten frekvenciaváltozást eredményez.



37.ábra

A programozható késleltetés (digitális időzítés) egyik megoldásának elvi logikai vázlata látható az alábbi a.ábrán. Alaphelyzetben a D flip-flop törölt állapotban van, s ekkor a számláló Load bemenetére 0 szint jut, és a számlálóba beíródik a Zbe szám. Mivel ez a bemenet statikus, ezért a számlálás nem indul. A START nyomógomb lenyomáskor létrejövő 1 – 0 szintváltás (negatív él) a D tárolót 1 –be billenti, s ekkor – a beírt számtól kezdődően - indul a számlálás. A számláló túlcsordulása ismét törli a D flip-flopot, és visszajutottunk a kiinduló helyzetbe.

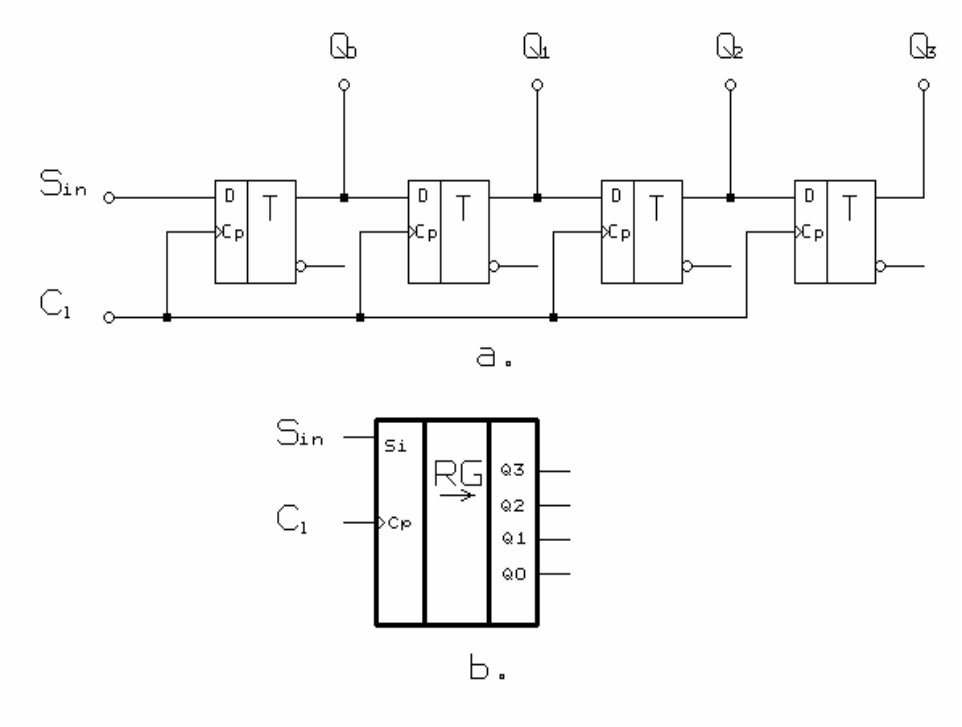
A b. ábra szemlélteti a START és a kimenet jelét. Elöször a t1 időpillanatban nyomták le a Start gombot. Ekkor a Z1 érték van az adat bemeneten. A második inditáskor Z2 az érvényes adat, ezért különböző időtartamú lesz a flip-flop beírt állapota. E jel lefutó éle következik be a programozott késleltetéssel.



38.ábra

5.3. Léptetőregiszterek

A léptető regiszterek (shift - regiszter) kettős feladatot ellátó funkcionális áramkörök. Egyrészt egy n bites **digitális szó tárolására**, másrészt egy-egy léptető jel hatására - a tárolt információ - jobbra vagy balra **léptetésére** használhatók. A 39. ábrán látható egy 4 bites - élvezérelt D típusú flip-flop -okból kialakított – léptető regiszter logikai vázlata. Léptető regiszternél - az információ átmeneti tárolása mellett - a szomszédos flip-flop -ok között olyan csatolást kell megvalósítani, amely biztosítja, hogy közöttük egy külső léptető jel hatására információ átadás történjen.



t	Q_0	Q_1	Q_2	Q_3
t_1	1	x	x	x
t_2	0	1	x	x
t_3	1	0	1	x
t_4	0	1	0	1

c.

39. ábra

A C_l léptető jel 0 - 1 átmeneténél mindegyik tároló elembe a D bemenetén érvényes logikai érték íródik. Azáltal, hogy az egyes D_i bemeneteket az előző flip-flop Q_{i-1} kimenetével kötöttük össze, a tárolt digitális szó léptetése történik. A legelső flip-flop - ba pedig az S_i jelű bemenet aktuális értéke íródik. A 39. b. ábrán a léptető regiszter szimbolikus jele látható. A 39. c. ábrán táblázatban szemléltetjük a működési ütemeket, ha a soros bemenetre (S_i) **1 0 1 0** jelsorozat érkezik a léptető-jel ütemezésében (az **x** a léptetés előtti ismeretlen tartalmat jelzi). Az egyes sorok az egymás után érkező léptető impulzusok hatására bekövetkező állapotokat tartalmazzák.

5.3.1. A léptető regiszterek fajtái

A léptető regisztereket csoportosíthatjuk

- az információ beírása és
- a kiolvasás

módja szerint, valamint a léptetés iránya alapján.

A beírás és a kiolvasás szerint megkülönböztetünk

- párhuzamos és
- soros

beírású, ill. kiolvasású léptető regisztereket.

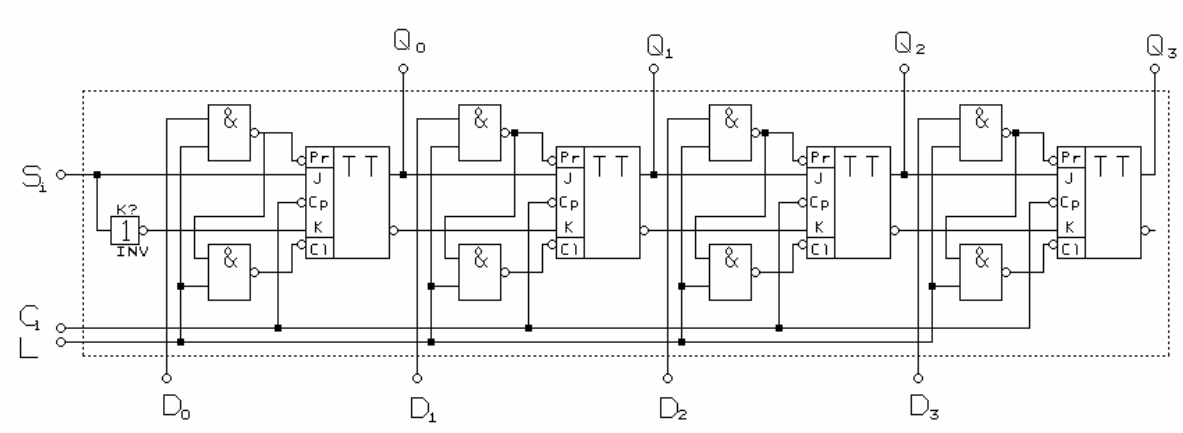
A léptetés iránya szerint

- jobbra,
- balra és
- kétirányú léptetésű

regiszterek vannak.

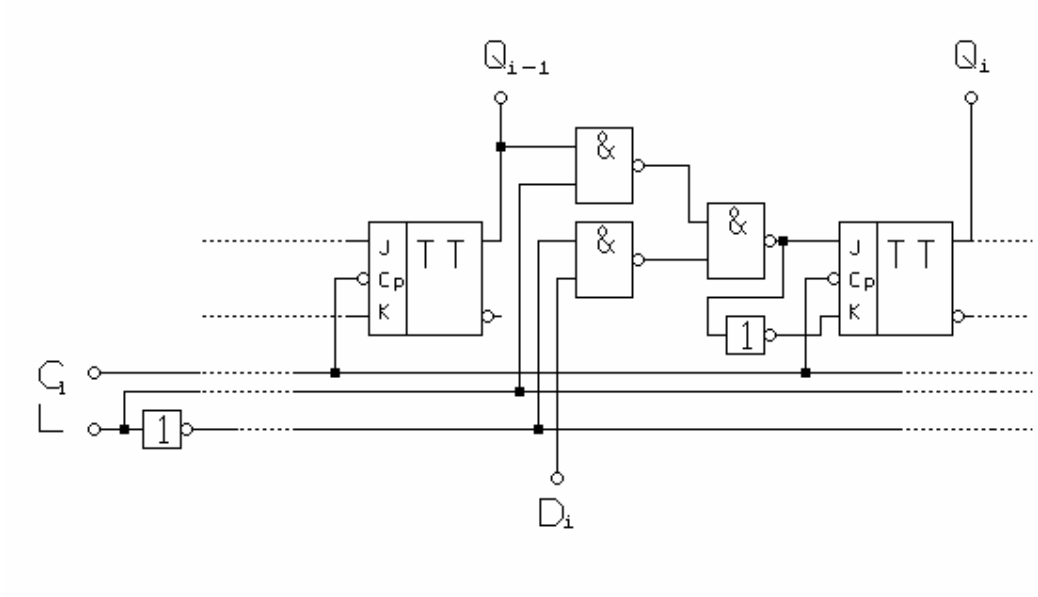
A 37. ábra szerinti léptető regiszter **soros beírású** jobbra léptető regiszter. Amennyiben a ki-menetek (**Q_0, Q_1, Q_2, Q_3**) mindegyike kivezetett, akkor a kiolvasás **párhuzamos**. Amennyiben csak a **Q_3** kimenethez csatlakozhatunk, akkor a **kiolvasás** módja **soros**. Ez az utóbbi megoldás elsődlegesen az integrált áramköri kialakításoknál használt a szükséges lábszám csökkentéséhez.

A párhuzamos információ-beírás - a számlálóknál már megismertekhez hasonlóan - **aszinkron** és **szinkron** módon történhet. Az 40. ábrán egy aszinkron párhuzamos beírású, jobbra léptető regiszter logikai vázlata látható. A párhuzamos szó beírása az **L**=1 értéknél történik az egyes flip-flop -ok **Pr** és **Cl** bemenetein keresztül, tehát aszinkron módon. Ekkor a **C_l** léptető jel hatása nem érvényesül. Az **L**=0 értéknél soros beírású (**S_i** = serial input), jobbra léptető regiszterként működtethető az áramkör.



40. ábra

A **szinkron** üzemű **párhuzamos beírás** egy áramköri megoldását mutatja a 41. ábra, amely egy léptető regiszter egy részletét mutatja. Az **L** beíró jel **1** értéke a léptetési üzemmódot választja. Ekkor az **i** - ik flip-flop -ba - a **C_l** 1 - 0 jelváltásakor - az **i-1** - ik flip-flop értéke íródik.



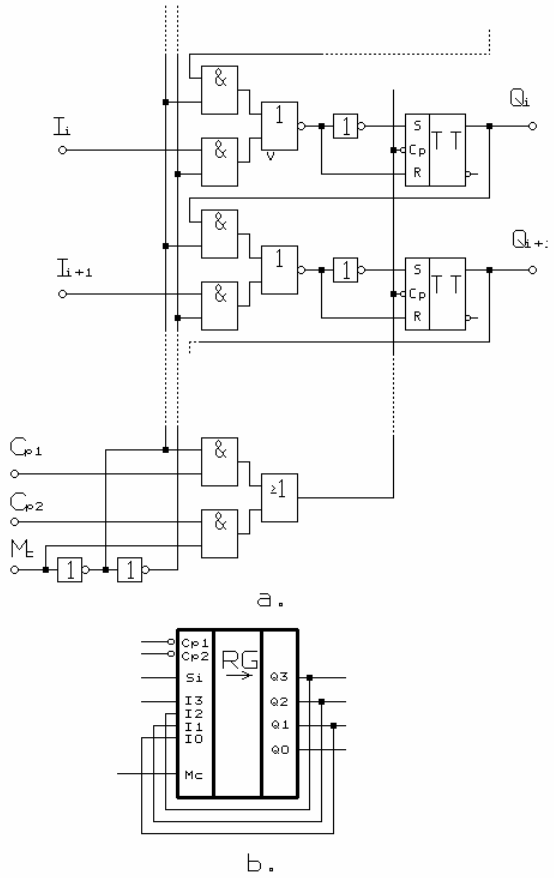
41. ábra

Az **L=0** vezérlésnél a $\mathbf{C_l}$ jel a $\mathbf{D_i}$ információ érvényes értékét írja az **i** - ik flip-flop -ba, vagyis párhuzamos beírás történik.

5.3.2. Integrált áramköri léptető regiszterek

A TTL rendszerű integrált áramköri család egyik – széleskörűen alkalmazható – léptetőregiszterét ismertetjük. Az áramkör működésének megismerésén túlmenően kitérünk az alkalmazás lehetőségeinek tárgyalására is.

A 42.a. ábrán az SN 7495 típusú 4 bites lépte-tőregiszter egy részletének logikai vázlata látható. A regiszterbe **adatbevitel** mind **sorosan**, mind pedig **párhuzamosan** lehetséges. Ez mindkét változatban szinkron üzemű. **Párhuzamos kiolvasást** tesz lehetővé az, hogy mind a négy flip-flop kimenete kivezetett. (Természetesen kiolvasás is lehetséges.)



42. ábra

Léptetés - s így **soros** beírás (**Si**) és kiolvasás (**Q_3**) - az üzemmód vezérlő bemenet **MC=0** (Mode Control) értékénél történik a **Cp1** léptető bemenetre adott impulzussorozattal (0 - 1 átmenetre érzékeny). Az **MC=1** vezérléskor a szinkron üzemű párhuzamos adatbeírás történhet a **Cp2** bemenetre adott impulzus 0 - 1 átmenetekor.

Az áramkörből – 42. b. ábra szerinti - külső kötéssel kétirányú léptető regisztert alakíthatunk ki. Ebben a kapcsolásban **MC=0** értéknél - a **Cp1**-re adott impulzussal - **jobbra**, míg **MC=1** értéknél - a **Cp2**-re adott impulzussal - **balra** léptetés történik. Jobbra léptetésnél **Si** a soros bemenet és **Q_3** a soros kimenet. Balra léptetésnél **I_3** a soros bemenet és **Q_0** a soros kimenet. A meghajtó áramkörök számára az MC kettő, míg a többi bemenet egy egység-terhelést jelent.

5.3.3. A léptetőregiszterek alkalmazása

A léptető regisztereket, mint átmeneti tárolókat (tartó áramkörök) szinte minden digitális berendezésben alkalmazzák. Ezzel itt részletesebben nem foglalkozunk. Viszont tárgyaljuk a léptetőregiszterek

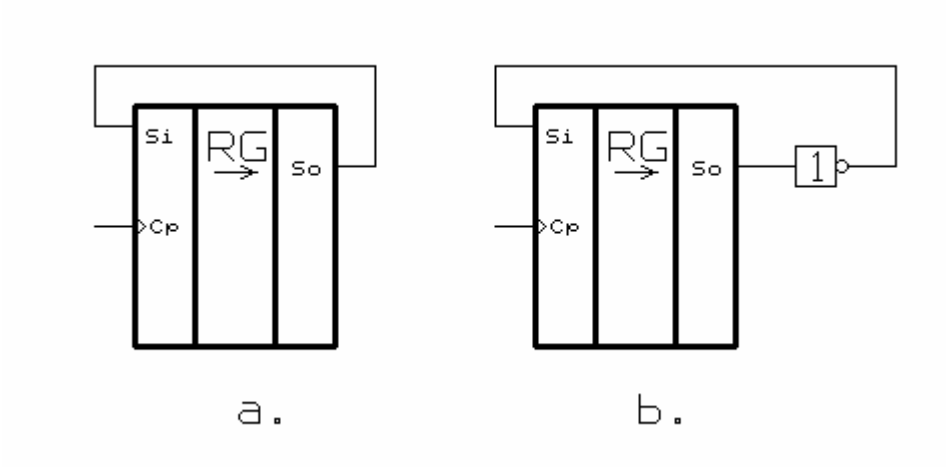
- gyűrűs számlálóként, ill.
- soros-párhuzamos és
- párhuzamos-soros

kódátalakítókban való felhasználását.

5.3.3.1. Gyűrűs számlálók

Amennyiben egy léptető regiszter soros kimenetét (**So**) a soros bemenettel (**Si**) összekötjük, akkor olyan áramkört kapunk, amelyben az információ kering (a kilépő bit beíródik az első tárolóba). Ezt a megoldást nevezzük **gyűrűs számlálónak**.

Az adat visszavezetése történhet egyenes és tagadott alakban is (43. ábra). Az a. ábra szerinti visszavezetési megoldással **n - modulusú**, míg a b. ábra szerint **2n - modulusú** gyűrűs számlálót kapunk, ahol **n** a regiszter tárolóinak száma.



43. ábra

Az **n - modulusú** gyűrűs számlálónál az eredeti információ az n. lépés után kerül vissza a regiszter megfelelő helyértékeire. Erre mutat példát a 44. ábra szerinti működési táblázat, amelyen egy 4 bites n modulusú gyűrűs számláló egyes ütemeinek állapota látható az **1 0 0 0** kezdő feltételből indulva.

t	**Q_0**	**Q_1**	**Q_2**	**Q_3**
t_1	1	0	0	0
t_2	0	1	0	0
t_3	0	0	1	0
t_4	0	0	0	1
t_5	1	0	0	0

44. ábra

Az áramkört felhasználhatjuk, pl. soros működésű aritmetikai egység átmeneti tárolójaként, ha az egyik tényezőt - műveletvégzés után - változatlanul kívánjuk megtartani.

Számlálóként is használhatjuk a gyűrűs számlálót. Ha a regiszterben egy darab 1-et léptetünk, akkor minden állapotban egyetlen kimenet értéke lehet 1 szintű. Ha az n kimenet mindegyikéhez egy N alapszámú számrendszer egy számjegyét rendeljük, akkor **1 az N-ből** kódolású számlálót kapunk.

A **2n modulusú** gyűrűs számlálóban **2n** számú léptetés után kapjuk vissza az eredeti állapotot. A 45.a. ábrán levő táblázat mutatja egy 4 bites 2n modulusú gyűrűs számláló állapotsorozatát, ha a **0000** állapotból indulunk ki. Ugyanezen gyűrűs számlálóban a 45. b. ábra táblázata szerinti állapotsorozat is kialakulhat. Mindkét sorozat 8-8 állapotból (2n) - két teljes ciklusból - áll. A kettő együtt tartalmazza a lehetséges 16 kombinációt.

Ütem	Q_3	Q_2	Q_1	Q_0
1	0	0	0	0
2	0	0	0	1
3	0	0	1	1
4	0	1	1	1
5	1	1	1	1
6	1	1	1	0
7	1	1	0	0
8	1	0	0	0
9	0	0	0	0

a.

Ütem	Q_3	Q_2	Q_1	Q_0
1	1	0	0	1
2	0	0	1	0
3	0	1	0	1
4	1	0	1	1
5	0	1	1	0
6	1	1	0	1
7	1	0	1	0
8	0	1	0	0
9	1	0	0	1

b.

45. ábra

Általánosan a következő törvényszerűség fogalmazható meg: egy **n** bites léptető regiszterből kialakított **2n** modulusú gyűrűs számláló **k** féle **teljes** ciklusban működtethető ahol

$$\mathbf{k} = \frac{\mathbf{2^n}}{\mathbf{2n}}$$

hányados egész része. Amennyiben az osztás eredménye nem egész szám, akkor csonka ciklus is van. Csonka ciklusnak nevezzük az olyan sorozatot, amely 2n lépésnél hamarabb veszi fel a kezdő kombinációt. A csonka ciklus állapotainak száma az osztásnál kapott maradékkal egyezik meg.

Példa:

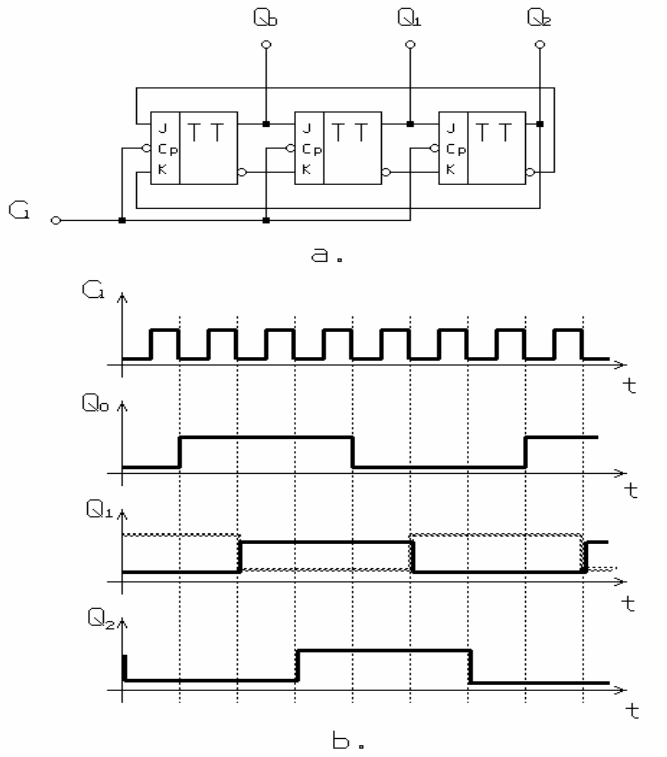
n=3 esetén **egy** 6 állapotú (2n = 6) **teljes** ciklus és **egy** kétállapotú **csonka** ciklus lehetséges. **n=5** bites gyűrűs számlálónál **három** 10 állapotú **teljes** ciklus és **egy** kétállapotú **csonka** ciklus létezik. A kezdőszám fogja meghatározni, hogy melyik ciklusban üzemel a számláló. Amennyiben több teljes ciklus is lehetséges, ezek közül azt tekintjük **alap-ciklusnak,** amely tartalmazza az összes bit 0 kombinációt.

Az **öt** bites **2n** modulusú gyűrűs számlálót decimális számlálóként is használjuk. A lehetséges három teljes ciklusból a 00000 állapotot is tartalmazó sorozatot (alap-ciklus) nevezzük **Johnson - kódnak**. Ahhoz, hogy a gyűrűs számláló mindig az alap-ciklusban üzemeljen, biztosítani kell, hogy az esetleges ciklustévesztés után (pl. külső zavar) automatikusan kerüljön vissza az alap-ciklusba. Egyik megoldás lehet, ha egy élvezérelt D flip-flop a soros kimenet 1 - 0 átmenetekor bebillen és törli a számláló flip-flop -jait. Ez a törlés a helyes működést nem zavarja, mivel az alap-ciklusban egyébként

is ez az állapot kell következzen. A következő órajel 1 szintje aszinkron módon törli a D flip-flop –ot. Ha valamilyen okból hibás állapot áll be, ezt - néhány ütem után - automatikusan törölni fogja a D tároló.

Példa: Vegyük azt, hogy valamilyen zavar eredményeként az **10010** hibás állapotot lép fel. A következő ütem az **11001**, majd **01100** lenne, de az utóbbi beálltakor a D flip-flop is bebillen s ez a számláló **00000** állapotát, állítja be. Ennek eredményeként csak egyetlen hibás ciklus lesz.
Röviden említést teszünk a **2n** modulusú gyűrűs számlálók egy speciális vezérléstechnikai felhasználásáról. Amennyiben **n=k*3**, vagyis a három egész számú többszöröse, akkor a számláló kimenő jeleiből mindig előállítható **3 fázisú szimmetrikus jelrendszer**.

A 46. a. ábrán **3** bites 2n modulusú gyűrűs számláló logikai vázlata látható. A b. ábra szemlélteti az órajel és a kimeneti jelek idő-függvényeit.



46. ábra

Mindhárom kimenet jele szimmetrikus négyszögjel, és frekvenciája

$$\mathbf{f_{ki}} = \frac{\mathbf{f_q}}{\mathbf{2n}}$$

ahol $\mathbf{f_q}$ a léptető jel frekvenciája, és n a regiszter bitjeinek száma.

A b. ábrán látható, hogy az egyes kimenetek jelei egy léptető-jel periódus idejével késnek egymáshoz képest. Mindegyik jel periódus-ideje 6 ütem, amit tekinthetek 360 villamos foknak. Ebből következik, hogy az egyes jelek közötti fázistolás:

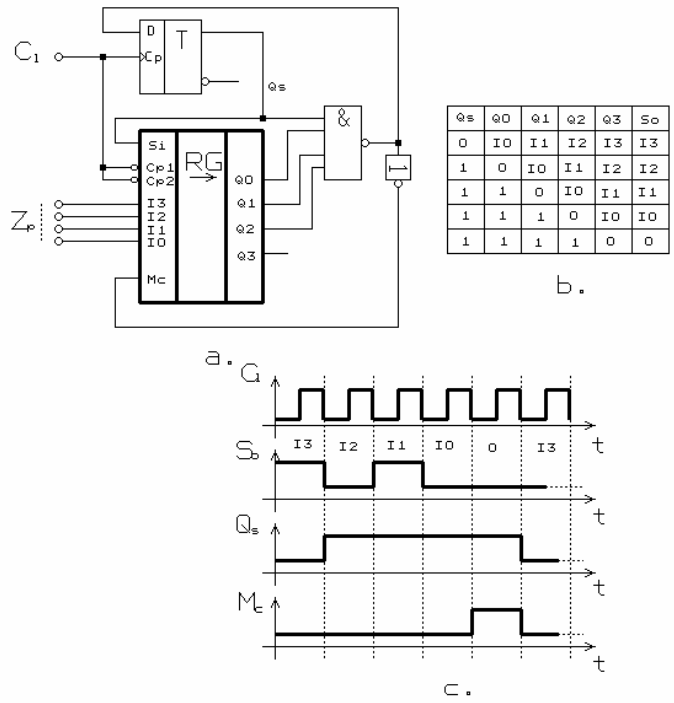
$$F = \frac{\mathbf{2}P}{\mathbf{n}} = \frac{\mathbf{360^o}}{\mathbf{6}} = \mathbf{60^o}$$

Amennyiben a $\mathbf{Q_0}, \overline{\mathbf{Q}}_1, \mathbf{Q_2}$ jelsorozatot tekintjük, ezek - bármilyen órajel frekvenciánál - pozitív sorrendű szimmetrikus háromfázisú rendszert alkotnak. Ezért háromfázisú rendszerek - pl. aszinkron motorok fordulatszám változtatásánál stb. - vezérlő jeleként felhasználhatók.

5.3.3.2. Párhuzamos-soros kódátalakítás

A fejezetben röviden ismertetjük a léptetőregiszterek alkalmazásával megvalósítható **párhuzamos-soros** kódátalakítást.

Az átalakítás elve, hogy az átalakítandó, párhuzamos kódolású információt a léptetőregiszterbe - a **párhuzamo**s adatbemeneteken keresztül - **írjuk be**. Ezt követően - az órajel ütemében - léptetve a regiszter tartalmát, annak **soros kimenetén** (So) időben egymás után - egyetlen csatornán - kapjuk az információ egyes bitjeit. Ezzel soros kódolásban áll rendelkezésünkre az eredeti információ. A kódátalakító áramkörnek biztosítania kell, hogy minden párhuzamos beírást - a szóhossznak megfelelő - **n** számú léptetés kövessen. Ezután ismét a párhuzamos beírás, vagyis az új információ fogadása következik.



Qs	Q0	Q1	Q2	Q3	So
0	I0	I1	I2	I3	I3
1	0	I0	I1	I2	I2
1	1	0	I0	I1	I1
1	1	1	0	I0	I0
1	1	1	1	0	0

47. ábra

A 47. a. ábrán egy 4 bites digitális szó párhuzamos-soros kódátalakítására alkalmas áramkör logikai vázlata látható. Az áramkörben felhasznált léptetőregiszter - az előzőekben már megismert - az SN 7495 AN típus. Párhuzamos beírás a Mode Control (MC) 1 szintjénél, míg léptetés MC=0 értéknél történik a Cp jel 1 - 0 átmenetekor. A kiegészítő áramkörök - D flip-flop és kapuk - automatikusan hajtják végre a párhuzamos beírást és a léptetés végének jelzését.

A működés elemzését az információ léptetésének kezdetétől végezzük el. Ebben a pillanatban az RG - ben van a 4 bites információ (Ip) és a D flip-flop törölt állapotú. Ennek hatására az MC vezérlőbemenet 0 szintet kap, s ezért a következő órajel a regiszter tartalmát eggyel jobbra lépteti. Ennek eredményeként a regiszterbe 0 lép be és

Q_0=0 lesz, ugyanakkor a **D** flip-flop -ba **1** íródik. A soros kimeneten viszont már a következő bit jelenik meg. A további órajelek a regiszterbe 1-et léptetnek be. A negyedik órajelre a K kapu mindegyik bemenetén 1 érték lesz, s ezért az MC is 1 szintre vált. Az ötödik órajel 1 - 0 átmenete írja be a regiszterbe a következő párhuzamos információt, s a D flip-flop -ba a 0-t. Ezzel kezdődik a következő átalakítási ciklus. A 45. b. ábrán látható táblázatban szemléltettük az egyes kimenetek értékeit ütemenként. Az átalakítandó információ (Zp) bitjei **I_3, I_2, I_1, I_0**.

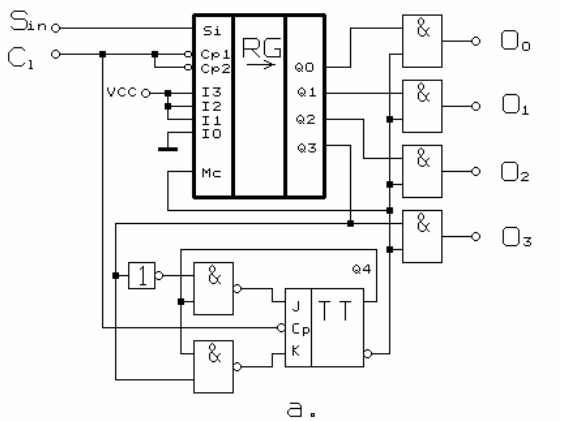
A 47. c. ábrán láthatók az **So** soros kimeneti csatornán kapott jelsorozat, a CL órajel, a D flip-flop, Qs kimenet és az MC jel időfüggvényei a **Zp = 1010** digitális szó átalakításakor. A szószünetet az MC=1 érték jelzi. E jel használható fel a soros jelet fogadó áramkörök szinkronozás hoz.

A párhuzamos-soros kódátalakítókat leggyakrabban a nagyobb távolságú adatátviteli rendszereknél használják. Soros kódban való információátvitelhez egyetlen adatcsatorna szükséges. Az ismertetett átalakító által előállított soros jel hibátlan vételéhez - dekódolás hoz - még a CL és a szószinkronozó (MC) jelet is továbbítani kell. Bármilyen bitszámú digitális szónál tehát összesen három jelvezetékkel valósítható meg ez az adatátvitel.

5.3.3.3. Soros-párhuzamos kódátalakítás

A soros-párhuzamos kódátalakítás elve, hogy az átalakítandó **n** bites **információt** CL órajel lépteti be a **regiszterbe**. Majd az **n+1**-edik ütemben (" szó szünet") kerül a párhuzamosan kódolt információ a kimeneti csatornákra.

A soros-párhuzamos kódátalakító kialakítható oly módon is hogy a szószünetet automatikusan állítja elő. Erre példa a 48. a. ábra szerinti áramkör, amely 4 bites digitális szó átalakítását végzi.



a.

Q0	Q1	Q2	Q3	Q4
0	1	1	1	1
I3	0	1	1	1
I2	I3	0	1	1
I1	I2	I3	0	1
I0	I1	I2	I3	0
0	1	1	1	1

b.

48. ábra

Az átalakítás egy párhuzamos beírással kezdődik. A regiszter első bitjét ($\mathbf{Q_0}$) **0**-ra, míg a továbbiakat és a kiegészítő JK flip-flop -ot is **1** értékre állítjuk be. Ennek hatására az **MC** jel **0** lesz és a következő órajelre, megkezdődik a **soros információ beléptetése**. Ugyanakkor tiltott a párhuzamos kimeneti csatorna. Az eredetileg $\mathbf{Q_0}$-ba irt **0** érték ütemenként tovább lép és a negyedik órajel hatására a JK flip-flop 0-ba billenése, engedélyezi a kimeneteket. Ugyanekkor előkészíti a következő ciklust indító párhuzamos beírást, mivel ekkor az MC és a flip-flop J és K bemenetére is 1 szint jut. Az órajel ötödik 1 - 0 átmenetekor kezdődik, a leirt ciklus elölről.

A regiszter feltöltésének ütemei a 48. b ábra táblázatában láthatók. Az ismertetett megoldású soros-párhuzamos kód átalakításnál az adótól csak az adatcsatornát és a léptető jelet kell csatlakoztatni. Ezzel a távátvitelhez szükséges csatornaszám kettőre csökkent.