Simon Alpár Tunyagi Arthúr Róbert

ELEKTRONIKA

LABORATÓRIUMI PRAKTIKUM

2

Digitális elektronika gyakorlatok

Presa Universitară Clujeană

Simon Alpár | Tunyagi Arthúr Róbert

ELEKTRONIKA

LABORATORIUMI PRAKTIKUM

2. Kötet

Digitális elektronika gyakorlatok és feladatok

Simon Alpár | Tunyagi Arthúr Róbert

ELEKTRONIKA

LABORATORIUMI PRAKTIKUM

2. Kötet

Digitális elektronika gyakorlatok és feladatok

Presa Universitară Clujeană / Kolozsvári Egyetemi Kiadó

2022

Referenți științifici:

Conf. dr. Járai-Szabó Ferenc

Conf. dr. Libál András

ISBN general: 978-606-37-1254-8
ISBN specific: 978-606-37-1688-1

Universitatea Babeş-Bolyai
Presa Universitară Clujeană
Director: Codruţa Săcelean
Str. Hasdeu nr. 51
400371 Cluj-Napoca, România
Tel./Fax: (+40)-264-597.401
E-mail: editura@ubbcluj.ro
http://www.editura.ubbcluj.ro/

TARTALOMJEGYZÉK

1. BEVEZETŐ

A „*Digitális elektronika gyakorlatok*” alcímet viselő kötet az „*Elektronika laboratóriumi praktikum*” munka második kötete, az első rész („*Elméleti és kísérleti alapok*”) szerves folytatása.

A jelen kötet egy laboratóriumi és szemináriumi útmutató, ahol már olyan konkrét digitális elektronika feladatokkal szembesül a hallgató, amelyeket akár analitikusan, akár gyakorlati-kísérleti úton oldhat meg.

A gyűjteményben olyan gyakorlatokkal találkozhatunk, amelyek lefedik a digitális elektronika fontosabb részeit és kihívásait, a kapcsolástechnikától és a logikai matematika alapoktól, a kapuzott egyszerű logikai áramkörökön át, a sokkal komplexebb kombinációs vagy sorrendi funkcionális hálózatokig.

A kötetet egy viszonylag bő, válogatott irodalomjegyzékkel és webográfiával zárjuk. Azok számára, akik a tárgyaltakat részletesebben, mélyebben és magasabb szinten kívánják megismerni több könyv-, tankönyv- és jegyzet címet gyűjtöttünk ide össze, a webográfia részben pedig olyan honlapcímeket adtunk meg amelyek akár elméleti szinten, akár hasznos gyakorlati információkat tartalmazva segítik a laboratóriumi munkát. Ugyanitt hívjuk fel a hallgatók figyelmét néhány igen jól használható ingyenes számítógépes szakszoftver és szimulációs programcsomag létezésére is.

Reménységünk szerint a kötetben található gyakorlatok és feladtok eredményes tanuláshoz és szakmai sikerekhez fognak vezetni.

dr. Simon Alpár
docens

dr. Tunyagi Arthúr
adjunktus

2. A BIPOLÁRIS TRANZISZTOR, MINT KAPCSOLÓ

1. A GYAKORLAT CÉLJA:

A bipoláris tranzisztor kapcsoló üzemmódjának megismerése és megértése, illetve a logikai kapuk fizikai működésének elektronikai megalapozása

2. SZÜKSÉGES ESZKÖZTÁR:

- dugaszolós próbapanel és csatlakozók
- *npn* típusú bipoláris tranzisztorok
- alkatelem teszter és multiméter
- feszültségforrás és vezetékek
- ellenállások
- oszcilloszkóp
- jelgenerátor
- mérőtűk

3. ELMÉLETI ALAPOK RÖVIDEN:

Az aktív áramköri elemeket tartalmazó logikai áramkörcsaládok működésének megértéséhez célszerű először részletesen megismerni és megérteni a bennük található aktív elem kapcsoló üzemmódját.

A mérnöki gyakorlatban többször előfordul, hogy egy adott rendszer (áramköri elem, két- vagy négypólus, egyszerűbb vagy bonyolultabb áramkör) kimenőjelének az értékét a bemenőjel folyamatos változtatása helyett ugrásszerűen, annak kapcsolgatásával tartjuk egy előírt értéktartományban. Ezt nevezzük *kapcsoló üzemmód*-nak.

A logikai függvények fizikai megvalósításához a gyártó cégek bizonyos alapáramkör (INVERTER, ÉS, VAGY, NAND, NOR, stb.) választékot alakítanak ki. Ezek kerülnek integráltáramkörös technológia segítségével egy hordozóra (chipre) és segítségükkel oldhatók meg, többszöri alkalmazással, a többé-kevésbé bonyolult feladatok.

Az alapáramkörök széleskörű felhasználhatóságát, az áramkörök jellemző adatainak katalógusokban való közlése és egységesítése teszi lehetővé.

A pozitív feszültségű rendszerben működő, pozitív logikájú logikai áramkörök legfontosabb jellemző adatai:

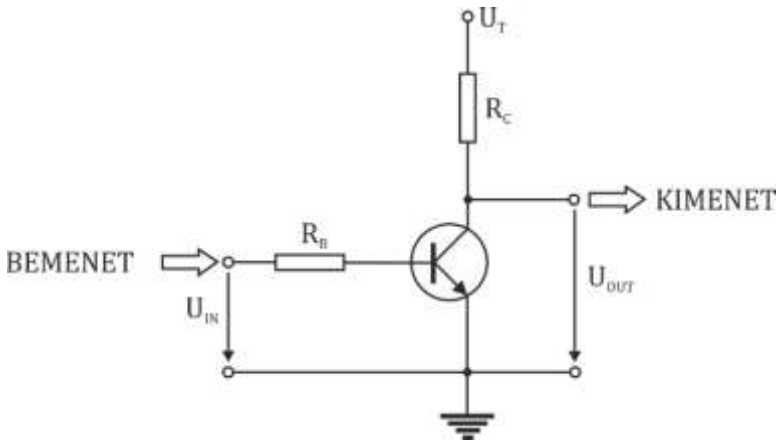
- *Tápfeszültség* (U_T) = az áramkör megfelelő működtetéséhez szükséges feszültség, a megengedhető tűrés feltüntetésével (emellett megadják még az áramkörre kapcsolható maximális feszültségértéket is U_{MAX})
- *Logikai szintek* (H, L) = a logikai szintek azok a feszültségtartományok, amelyek a logikai 1 (*magas*, *high* = H), illetve 0 (*alacsony*, *low* = L) értéket jellemző feszültségértékként még megengedhetők, mind a bemeneten, mind a kimeneten

- *Zajtartalék / Zajérzéketlenség* (*noise margin*, *noise immunity*) = az a feszültségtartomány, amelyen belül a feszültség változása nem változtatja meg a kapu logikai állapotát
- *Terhelhetőség* = az áramkör bemenetének és kimenetének áramviszonyai a *bemeneti terhelhetőség* (*FAN IN*) = az a maximális áramérték, amelyik az áramkör bemenetén átfolyhat mind a logikai 1, mind a logikai 0 állapotban, a *kimeneti terhelhetőség* (*FAN OUT*) = azon bemenetek száma, amelyeket egy áramkör kimenete – károsodás nélkül – képes árammal ellátni (meghajtani)
- *Teljesítményfelvétel* – nincs egységesen megadva, egyes katalógusok a *disszipáció*-t (vagyis az 50 %-os kitöltésű impulzus által vezérelt hővé alakuló teljesítményt) adják meg, mások pedig a különböző logikai szintekhez tartozó *tápáramigény*-t (ebből számítással határozható meg a tényleges teljesítményigény)
- *Jelterjedési idő* – a logikai áramkör bemenetére adott jel és a hatására a kimeneten létrejövő jel megjelenéséhez időre van szükség, ez az úgynevezett *jelterjedési idő* (t_{pLH}, t_{pHL}), ezt a kimenet L $\rightarrow$ H vagy H $\rightarrow$ L átmenete szerint adják meg, ugyanakkor megadható az *átlagos jelkésleltetési idő* (t_{pd}) amelyet a két jelterjedési idő számtani középarányosaként számítunk ki

Annak ellenére, hogy az *RT* (*Resistor – Transistor*, vagyis *ellenállás – tranzisztor*) logikai áramkörcsalád ma már nem használatos, igen megfelel a logikai kapuk elektronikai működésének didaktikai szemléltetésére. Az RT logika a digitális technika kezdeti szakaszát képviseli, átmenetnek tekinthető a telített logikájú tranzisztoros billenőkapcsolásoktól az IC technológiával készültek felé.

Két ellenállással és egy bipoláris tranzisztorral készített elektronikus kapcsoló tulajdonképpen egy inverter (negátor) és céltudatos, többszöri alkalmazásukkal a többi alapfüggvényt megvalósító logikai kapu is elkészíthető.

A negáció logikai műveletét a kacsoló üzemmódban működő emitterkapcsolású tranzisztorral valósítható meg (1. ábra). A digitális technikában ezt az áramkört *inverter*-nek nevezzük.

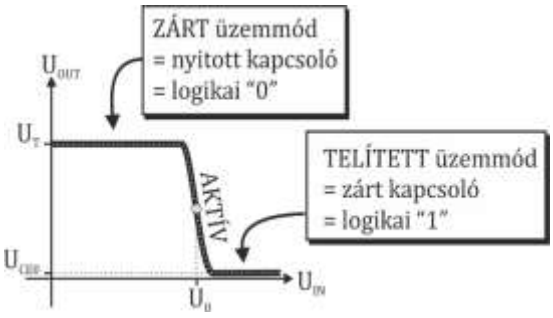


1.ábra A telítéses üzemű inverter

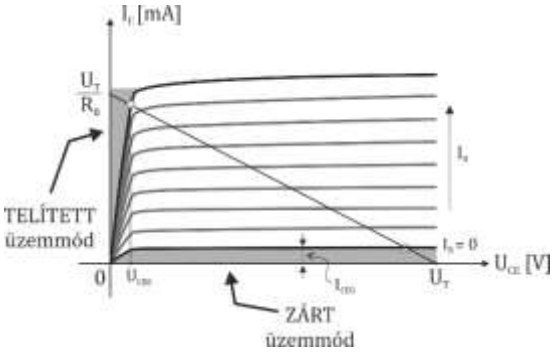
A telítéses üzemű inverter a bipoláris tranzisztort vezérelhető kapcsolóként használja. A tranzisztor bázis-emitter átmenetére adott vezérlő jel „dönti el", hogy a kollektor-emitter átmenet közel megszakítást (nyitott kapcsolót) vagy közel rövidzárlatot (zárt kapcsolót) képvisel:

- amennyiben a bázisellenállásra (R_B) kapcsolt bemeneti vezérlő feszültség (U_{in}) nem képes nyítni és előfeszíteni a bázis-emitter átmenetet, ezért a kimeneten (kollektor-emitter között) – az ott található esteleges további áramköröktől szinte teljesen függetlenül – csak egészen kicsi maradékáram (I_{CE0}I) folyik és a kimenet megszakítás-t képvisel
- nyitó irányú előfeszítéskor a vezérlő feszültség (U_{in}) és a bázisellenálás (R_B) által meghatározott bázisáram a tranzisztor lineáris, aktív üzemmódjára jellemző érték szélső, maximális megengedett értéke fele tart, a tranzisztor kinyit, az áram a kollektor és az emitter között szinte akadálytalanul folyhat, a tranzisztoron észlelhető feszültségesés pedig igen kicsi, a kimenet rövidzárlatot képvisel ($U_{out} = U_{CE0} < 0{,}2$)

Az inverter átviteli karakterisztikáját a 2.ábra, a bipoláris tranzisztor áram-feszültség karakterisztikáját pedig a 3.ábra szemlélteti.



2.ábra A telítéses üzemű inverter átviteli karakterisztikája



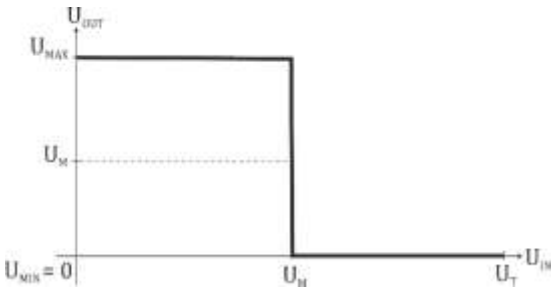
3.ábra A bipoláris tranzisztor áram-feszültség karakterisztikája

Logikai 0 feszültségű vezérlés esetén a tranzisztor zárt üzemmódba kerül, a kollektoráram megközelítőleg 0-ra csökken és a tranzisztor munkapontja a munkaegyenesen a $I_B = 0$ karakterisztikára vagy az alá kerül. Ennek a pontnak megfelelő igen kis áram a R_C kollektorellenálláson átfolyva elhanyagolhatóan kis feszültséget hoz létre, így a kimeneten mérhető feszültség gyakorlatilag megegyezik a tápfeszültséggel, tehát a bemeneti logikai 0-ra a kimeneten logikai 1 lesz a válasz.

Logikai 1 feszültségű vezérlés esetén a tranzisztor telített (szaturált) üzemmódba kerül. A nyitóirányú bemeneti feszültség hatására, a bázisellenállás által meghatározott nagyságú bázis nyitóáram indul meg. Ennek köszönhetően megjelenő kollektoráram az R_C munkaellenálláson feszültségesést hoz létre. Mivel a bázisáram többszöröse annak a szokványos bázisáramnak, ami a lineáris aktív üzemmódra jellemző, a kollektoráram is a szokványosnál sokkal nagyobb

lesz. Ezért a munkaellenálláson gyakorlatilag a teljes tápfeszültség esik, a kollektor-emitter átmenetnek nem marad csak egy maradékfeszültség (ez szilícium tranzisztorokra jellemzően, 0,2 – 0,3 V nagyságrendű). A kimenet gyakorlatilag majdnem 0 V-on, azaz logikai 0 szinten lesz.

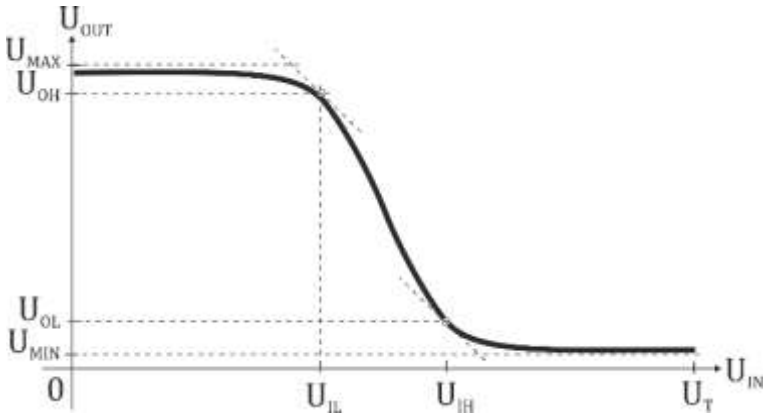
Az RT logika esetében a működtetés $U_T = +5\ V$ tápfeszültségről történik. A többi jellemző (logikai szint, zajtartalék stb.) minőségi és mennyiségi méretezéséhez célszerű megvizsgálni az ideális inverter átviteli karakterisztikáját (4.ábra).



4.ábra Az ideális inverter átviteli karakterisztikája

Az ideális inverter zajvédettsége maximális, hiszen akár a logikai 0, akár a logikai 1 vezérlőfeszültségekre maximálisan az U_M csúcsértékű zajfeszültség szuperponálódhat anélkül, hogy a kimeneti szint tévesen állna be.

A valóságos inverterek karakterisztikája (5.ábra) eltérő: U_{MAX} és U_{MIN} nem lesz tökéletesen U_T (+ 5 V), illetve 0 V, a komparációs szint is U_T -nél kisebb értéken valósul meg, a zajvédettség is csökken, mert keskenyebbek lesznek a megengedett logikai 0 és logikai 1 tartományok is.



5.ábra A valóságos inverter átviteli karakterisztikája és jellemző pontjai

A jellemző feszültségértékek definiálására meghúzzuk a „– 1” iránytangensű érintőket a karakterisztika inflexiós pontjaiba (ezek a zárt – aktív – telített üzemmódok közötti átmeneteknek felelnek meg).

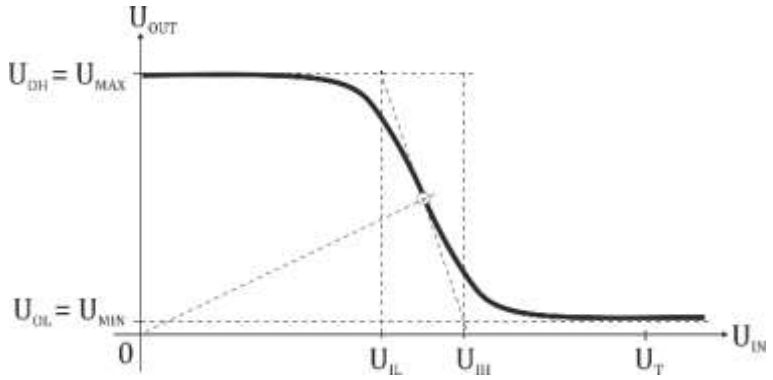
Jellemző feszültségértékek:

- $U_{MAX} = U_T =$ tápfeszültség (+ 5 V)
- $U_{MIN} = U_{CE0} < 0{,}2$ V
- U_{OH} és $U_{IL} =$ a zárt – aktív tartományátmenet inflexiós pontjának megfelelő értkek
- U_{OL} és $U_{IH} =$ az aktív –telített tartományátmenet inflexiós pontjának megfelelő értékek

Jellemző tartományok:

- $U_{IL} - 0 =$ megengedett bemeneti tartomány *L* (logikai 0) szinten
- $U_T - U_{IH} =$ megengedett bemeneti tartomány *H* (logikai 1) szinten
- $U_{OL} - U_{MIN} =$ garantált bemeneti tartomány *L* (logikai 0) szinten
- $U_{MAX} - U_{OH} =$ garantált kimeneti tartomány *H* (logikai 1) szinten
- $U_{IL} - U_{OL} =$ a logikai *L* szint zavartávolsága
- $U_{OH} - U_{IH} =$ a logikai *H* szint zavartávolsága
- $U_{IH} - U_{IL} =$ tiltott működési tartomány (az itt található bemeneti feszültségértékekre a gyártó nem tudja garantálni a kimenet logikai állapotát)

Mivel elég nehéz az inflexiós pontokat beazonosítani, létezik egy egyszerűsített módszer is, ezt a 6.ábra szemlélteti. Ebben az esetben az érintőt abban a pontban húzzuk meg, amely a karakterisztika és a koordinátarendszer szögfelezője találkozásánál van. A jellemző feszültségszintek az érintő és a zárt, illetve telített szakaszok meghosszabbításainak metszéspontjában lesznek.



6.ábra Az egyszerűsített módszer és az átviteli karakterisztika

4. A GYAKORLATI MUNKA MENETE:

(a) szemrevételezéssel azonosítjuk az ellenállásokat és teszterrel megmérjük pontos értéküket
(b) multiméteres méréssel azonosítjuk a tranzisztorok kivezetéseit, teszterrel megmérjük és lejegyezzük a közös emitterű sztatikus áramerősítési tényezőjüket (β)
(c) az 1. ábrán bemutatott kapcsolási rajzot elkészítjük a munkalapon
(d) beállítjuk az $U_T = +5\ V$ tápfeszültséget állandó értékre

(e) a bemeneti feszültséget megközelítőleg 0,1 V lépéssel folytonosan állítjuk és mérjük, a megfelelő kimeneti feszültségértékekkel együtt, a mérések eredményét táblázatba foglaljuk, majd ábrázoljuk az $U_{out} = f(U_{in})$ átviteli karakterisztikát
(f) a javasolt módszerekkel azonosítjuk a jellemző feszültségértékeket és tartományokat
(g) a jelgenerátor által szolgáltatott 50 %-os kitöltésű négyszögjelt kapcsolva az inverterre azonosítsuk és mérjük meg oszcilloszkóppal a jelterjedésre jellemző időket
(h) ismételjük meg méréseinket különböző ellenállásérték és áramerősítési tényező kombinációra, majd hasonlítsuk össze a kapott eredményeket, karakterisztikákat és vonjuk le a következtetéseket
(i) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

3. LOGIKAI ALAPFÜGGVÉNEK KAPCSOLÁSTECHNIKÁJA

1. A GYAKORLAT CÉLJA:

A logikai alapfüggvények kapcsolástechnikai megvalósítása mechanikai kapcsolók, nyomógombok segítségével

2. SZÜKSÉGES ESZKÖZTÁR:

- dugaszolós próbapanel és csatlakozók
- nyomógombok és DIP kapcsolók
- alkatelem teszter és multiméter
- feszültségforrás és vezetékek
- világítódióda (LED) és előtétellenállás
- mérőtűk

3. ELMÉLETI ALAPOK RÖVIDEN:

A digitális elektronika számára a matematikai alapot az úgynevezett Boole-féle algebra szolgáltatja, amelynek alapműveletei (operátorai):

a *TAGADÁS*, azaz *NEGÁCIÓ* vagy *INVERTÁLÁS*, jelei: „ $\neg$ ” vagy „ $-$ ”
az *ÉS*, azaz *KONJUNKCIÓ* vagy *LOGIKAI „SZORZÁS”*, jelei: „ $\wedge$ ” vagy „ $\cdot$ ”
a *VAGY*, azaz *DISZJUNKCIÓ* vagy *LOGIKAI „ÖSSZEADÁS”*, jelei: „ $\vee$ ” vagy „ $+$ ”

Minden további más logikai művelet összetettnek tekinthető és ezekből levezethető.

Mivel a digitális elektronikában leggyakrabban az úgynevezett *pozitív logika* használatos, ezért a Boole-algebra operátorait (műveleteit) és legfontosabb törvényszerűségeit (tételeit) ezzel a logikával adjuk meg, azaz:

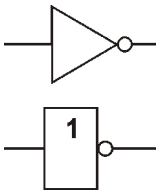
ha a *logikai érték* = 1	akkor a	*logikai állítás* = IGAZ *mechanikai kapcsoló* (*nyomógomb*) = ZÁRT *áramút* = ZÁRT *villamos potenciál* = MAGAS *elektronikus kapcsoló* = VEZET
ha a *logikai érték* = 0	akkor a	*logikai állítás* = HAMIS *mechanikai kapcsoló* (*nyomógomb*) = NYITVA *áramút* = MEGSZAKADT *villamos potenciál* = ALACSONY *elektronikus kapcsoló* = NEM VEZET

Az alapműveletek algebrai alakja és igazságtáblázata, illetve az őket megvalósító elektronikus kapcsolás áramköri (kapu) rajzjeleit a következők:

NEGÁCIÓ

$= \overline{A} =$ " *NON A* "

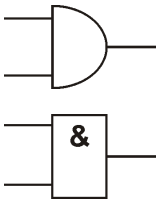
bemenet	*kimenet*
A	$\overline{A}$
0	1
1	0



KONJUNKCIÓ

$= A \cdot B =$ " A ÉS B "

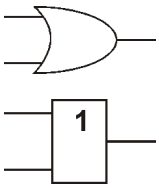
bemenetek		*kimenet*
A	B	$A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1



DISZJUNKCIÓ

$= A + B =$ " A VAGY B "

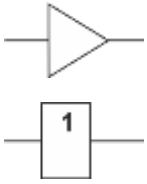
bemenetek		*kimenet*
A	B	$A + B$
0	0	0
0	1	1
1	0	1
1	1	1



Bár nem egy konkrét (alap)műveletet valósít meg, a digitális elektronikában mégis igen fontos szerep jut a negáció fordítottjának. Ezt a műveletet megvalósító kapuáramkört *PUFFER*-nek (*buffer*, *jelmásoló*, *jel erősítő*) nevezzük:

PUFFER

bemenet	*kimenet*
A	A
0	1
1	0



A *világító dióda* vagy LED (*LED = Light-Emitting Diode = fényt kibocsátó dióda*) félvezető anyagból készült fényforrás. Működése bizonyos mértékig hasonlít a félvezetőalapú diódák működéséhez: záróirányú előfeszítés esetén nem vezetnek, nyitóirányba előfeszítve a rajtuk

átfolyó nyitóirányú áram hatására inkoherens, viszonylag keskeny spektrumú fényt bocsátanak ki. A kibocsátott fény spektruma az infravöröstől, a láthatón át, az ultraibolyáig terjedhet. A konkrét szín a félvezető anyag összetételétől, részarányos ötvözőitől függ.

Nyitóirányú előfeszítésben az áram feszültség karakterisztika nagyon hasonlít az általános célú félvezető diódák karakterisztikájához, a legfontosabb különbség a jóval nagyobb nyitófeszültségekben rejlik. A viszonylag gyors exponenciális felfutás miatt LED-et soha nem használunk előtétellenállás nélkül, ennek értékét a

$$R_{előtét} = \frac{U_{táp} - U_{nyitó}}{I_{LED}}$$

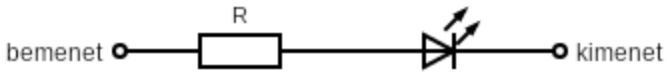
képlet alapján becsülhetjük meg. A világítódiódákon átfolyó áram áramerőssége rendszerint nem haladhatja meg a 20 mA értéket - az előtétellenállás méretezésénél ezt is figyelembe kell venni! Amikor az áramerősség értéke nem annyira fontos és meghatározó, egy 300 Ω körüli érték elegendő erre a célra. A szakkereskedelemben kapható 1% vagy5 % pontosságú 330 Ω-os ellenállások tökéletesen megfelelnek ennek a feladatnak az ellátására.

4. A GYAKORLATI MUNKA MENETE:

(a) szemrevételezéssel azonosítjuk a nyomógombokat, kapcsolókat, az előtétellenállást és a világítódiódát
(b) multiméteres és teszteres méréssel beazonosítjuk a világítódióda színét, anódját és katódját, majd megmérjük az előtétellenállás pontos értékét
(c) multiméteres méréssel beazonosítjuk a nyomógombok és a kapcsolók kivezetéseit, majd a mérések alapján letisztázzuk működési elvüket
(d) megtervezzük az alapfüggvények és a puffer által elvégzett logikai műveletek kapcsolását, majd kísérleti mérésekkel leellenőrizzük az elméleti bevezetőben és az igazságtáblázatokban foglaltakat – a logikai állapot kijelzésére LED-et használunk!
(e) javasoljunk megoldást az összetett a logikai függvények (NAND/NOR) megoldására is
(f) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

Egy kis segítség:

A *LED* működése mint logikai kijelző:



bemenet = magas potenciál-szint (+ 5 V)= logikai 1
kimenet = alacsony potenciál-szint (0 V)= logikai 0
$\Rightarrow$ a LED VILÁGÍT !!!

bemenet = alacsony potenciál-szint (0 V)= logikai 0
kimenet = magas potenciál-szint (+ 5 V) = logikai 1
$\Rightarrow$ a LED NEM VILÁGÍT !!!

A felhasználható nyomógombok és kapcsolók látképe:

nyomógomb

DIP1

DIP2

DIP4

mechanikai kapcsolók

Az egyváltozós logikai függvények megvalósításához javasolt kiellemző táblázat:

bemenet	*kimenet*
feszültségek	
0 V	
+ 5 V	
logikai szintek	
0	
1	

A kétváltozós logikai függvények megvalósításához javasolt kiellemző táblázat:

bemenet 1	*bemenet 2*	*kimenet*
feszültségek		
0 V	0 V	
0 V	+ 5 V	
+ 5 V	0 V	
+ 5 V	+ 5 V	
logikai szintek		
0	0	
0	1	
1	0	
1	1	

4. AZ ELLENÁLLÁSCSATOLÁSÚ TRANZISZTOROS LOGIKA

1. A GYAKORLAT CÉLJA:

A logikai alapfüggvények megvalósítása és tanulmányozása bipoláris tranzisztorok és ellenállások segítségével

2. SZÜKSÉGES ESZKÖZTÁR:

- dugaszolós próbapanel és csatlakozók
- bipoláris tranzisztorok és ellenállások
- alkatelem teszter és multiméter
- feszültségforrás és vezetékek
- mérőtűk

3. ELMÉLETI ALAPOK RÖVIDEN:

A digitális elektronika kezdeti szakaszában a műveletvégzésekhez csak mechanikai vagy mágneses üzemeltetésű kapcsolók álltak rendelkezésre – minden logikai művelet kapcsolók, vagy jelfogók segítségével valósult meg. Az analóg elektronika elektroncsöves szakaszában is léteztek próbálkozások a logikai műveletek megvalósítására. Az első programozható digitális számítógép, az ENIAC (Electronic Numerical Integrator And Computer, 1946) jellemzői: U-alak (30,5 méter hosszú, 1 méter széles, 3 méter magas), 140 kW teljesítményfelvétel, 5.000.000 kézi forrasztás, 17.468 elektroncső, 7.200 kristálydióda, 1.500 jelfogó, 70.000 ellenállás, 10.000 kondenzátor, 6.000 kapcsoló, műveletvégzési idők = 0,2 ms (összeadás, kivonás), 3 ms (szorzás) és 30 ms (osztás). A nagy méretek és energiafogyasztás, a nehézkes karbantartás és az igen magas előállítási és üzemeltetési költségek komoly akadályt jelentettek a széleskörű elterjedés szempontjából.

A tranzisztorok felfedezésével kezdetét veszi a félvezetőalapú elektronika és a digitális elektronika pedig forradalmasodik: tranzisztorok, diódák és ellenállások összekapcsolásával kis méretű, alacsony fogyasztású és nagyon gyors digitális áramköröket lehetett kialakítani. Ez a kezdeti szakasz, a későbbiekben az integrált áramkörök hódítanak teret. Ezekben a digitális kapcsolásokban a régi mechanikai- vagy mágneses kapcsoló szerepét az új elektronikus kapcsoló, a tranzisztor veszi át. A bipoláris tranzisztor elektronikus kapcsoló szerepköre a feszültségátviteli karakterisztikából következik.

Az ellenálláscsatolású tranzisztoros logika (*RTL*) diszkrét ellenállásokból és tranzisztorokból építi fel a logikai alapfüggvények megvalósításához szükséges kapcsolásokat. Ez egyfajta átmenetnek tekinthető a telített logikájú tranzisztoros billenőkapcsolások és az integrált technikával készült áramkörök között. Az RTL logika az első monolitikus integrált áramkörös logikai családnak az alapját képezték (1961 *Fairchild Semiconductor International*). Rendkívüli előnyük az egyszerű megvalósítás és a kisszámú felhasznált tranzisztor volt. A hátrányok közül megemlítendő a viszonylag magas hődisszipáció és az alacsony bemeneti terhelhetőség (maximum 3). A technológiai fejlődés rövid időn belül (1963, *Sylvania Electric Products*) két

másik, gyorsabb és performánsabb diszkrét logikával helyettesítette: DTL (diódacsatolású tranzisztoros logika) és TTL (tranzisztorcsatolású tranzisztoros logika).

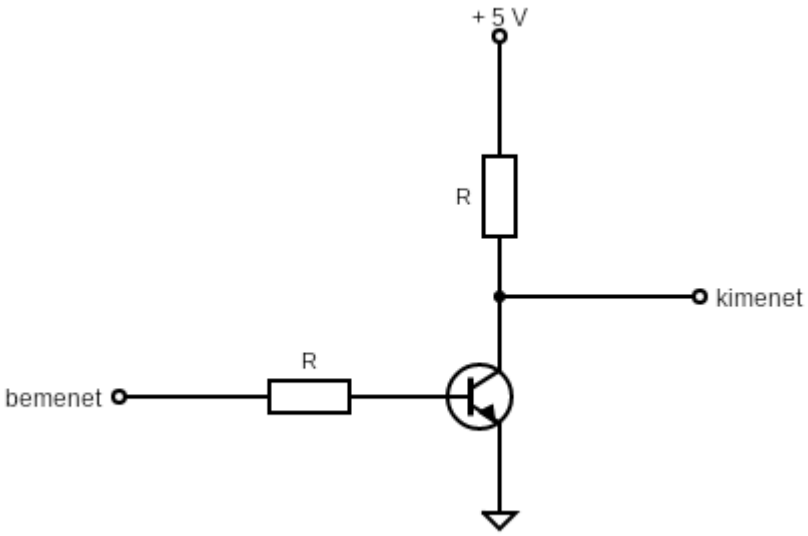
4. A GYAKORLATI MUNKA MENETE:

(a) szemrevételezéssel (színkód vagy felirat alapján) azonosítjuk az ellenállásokat, majd megmérjük pontos értéküket

(b) azonosítjuk az *npn* típusú tranzisztorok kivezetéseit (ez a helyes bekötéshez szükséges)

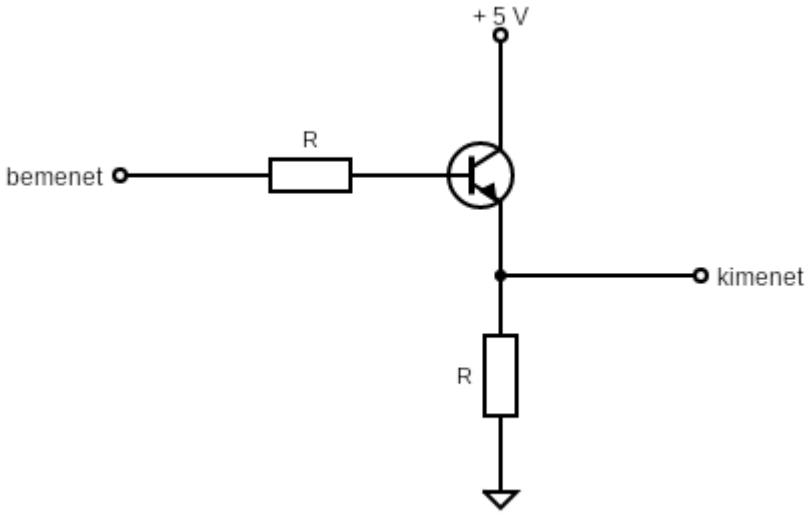
(c) rendre elkészítjük az alábbiakban bemutatott összes mérőkapcsolást, a bemenetekre felváltva 0 V-ot (logikai "0") és + 5 V-ot (logikai "1") csatlakoztatunk – mérjük a feszültségesést a kimenetnek jelölt ponton, a megmért szinteket táblázatba írjuk, majd kódoljuk (megadjuk az értéknek megfelelő logikai szintet/értéket), végül pedig azonosítjuk a megfelelő logikai függvényt

(d) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról



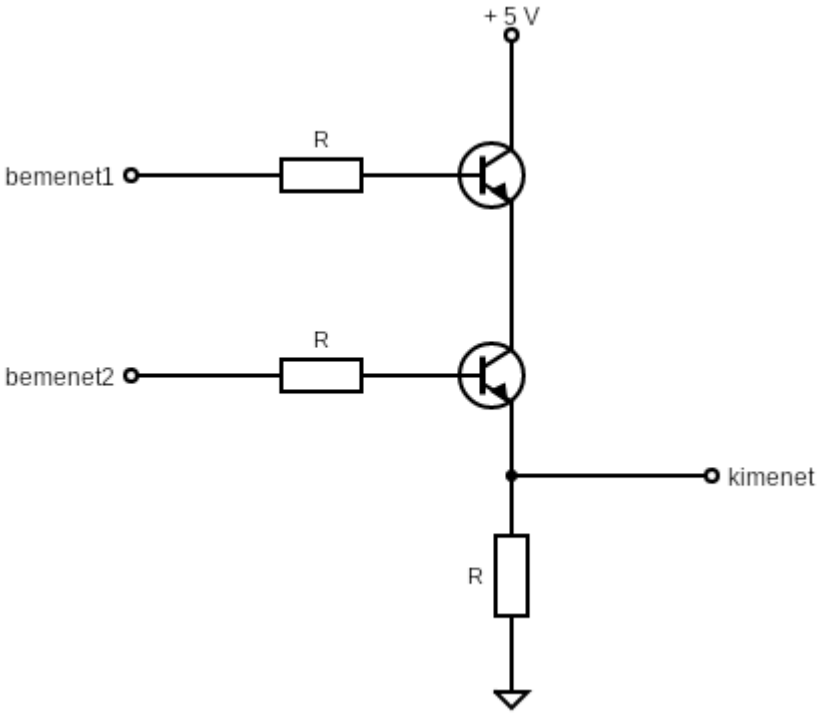
bemenet	*kimenet*
feszültségek	
0 V	
+ 5 V	
logikai szintek	
0	
1	

elnevezés: _____



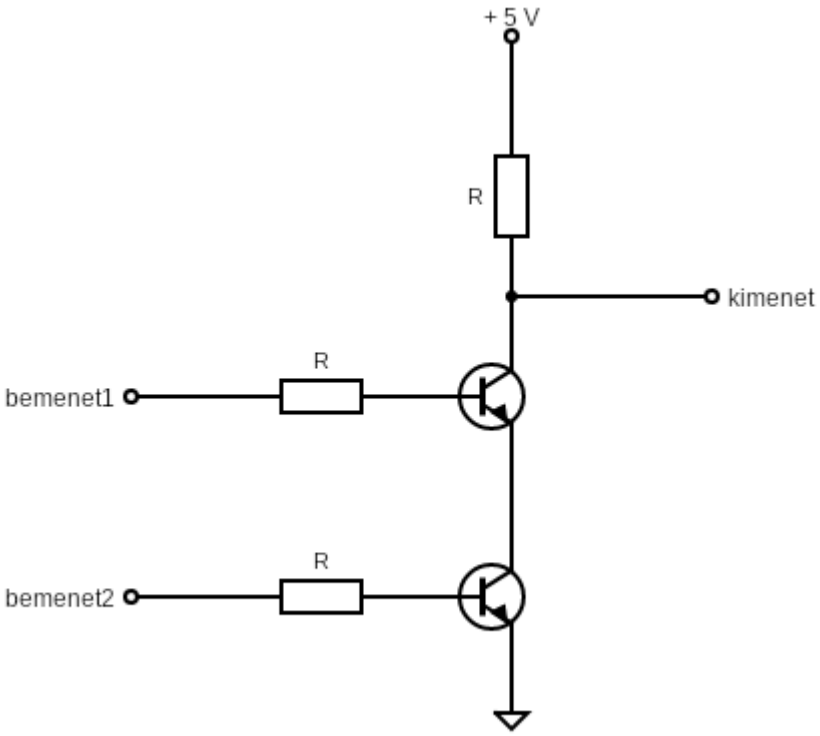
bemenet	*kimenet*
feszültségek	
0 V	
+ 5 V	
logikai szintek	
0	
1	

elnevezés: _____



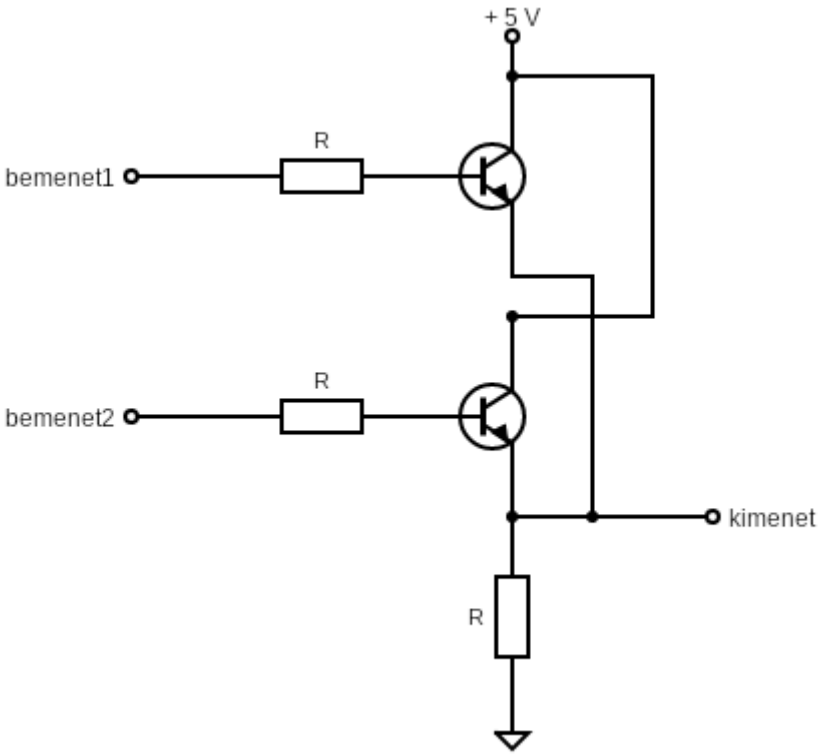
bemenet 1	*bemenet 2*	*kimenet*
feszültségek		
0 V	0 V	
0 V	+ 5 V	
+ 5 V	0 V	
+ 5 V	+ 5 V	
logikai szintek		
0	0	
0	1	
1	0	
1	1	

elnevezés: _____



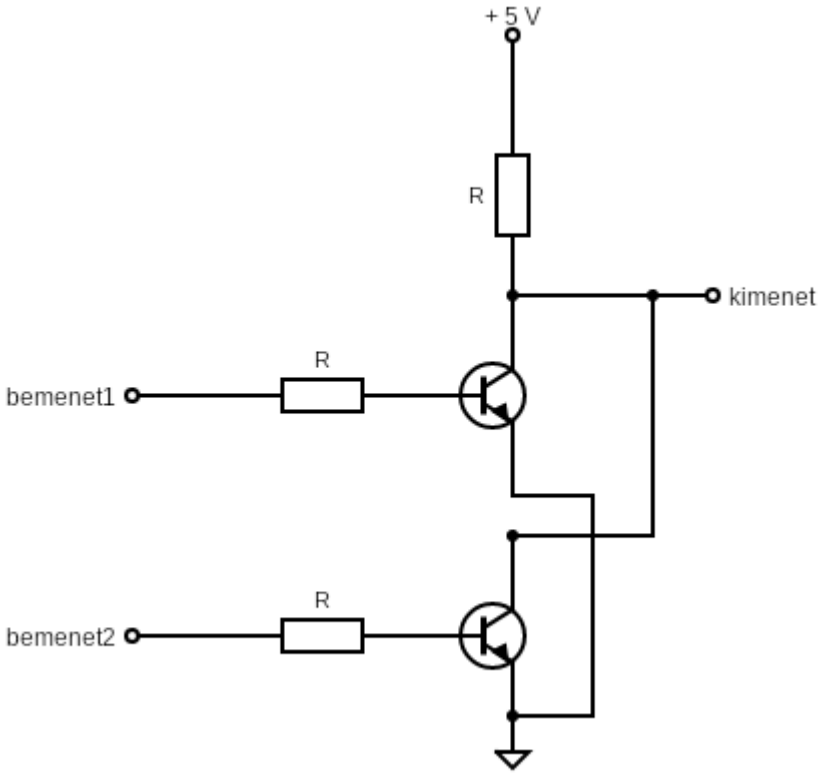
bemenet 1	*bemenet 2*	*kimenet*
feszültségek		
0 V	0 V	
0 V	+ 5 V	
+ 5 V	0 V	
+ 5 V	+ 5 V	
logikai szintek		
0	0	
0	1	
1	0	
1	1	

elnevezés: _____



bemenet 1	*bemenet 2*	*kimenet*
feszültségek		
0 V	0 V	
0 V	+ 5 V	
+ 5 V	0 V	
+ 5 V	+ 5 V	
logikai szintek		
0	0	
0	1	
1	0	
1	1	

elnevezés: _____



bemenet 1	*bemenet 2*	*kimenet*
feszültségek		
0 V	0 V	
0 V	+ 5 V	
+ 5 V	0 V	
+ 5 V	+ 5 V	
logikai szintek		
0	0	
0	1	
1	0	
1	1	

elnevezés: _____

Figyelem!

a logikai alapfüggvények (összetett)igazságtáblázata:

bemenet	*kimenet*	
A	A	$\bar{A}$
0	0	1
1	1	0
	PUFFER	*INVERTER*

bemenet 1	*bemenet 2*	*kimenet*			
A	B	$A \cdot B$	$\overline{A \cdot B}$	$A + B$	$\overline{A + B}$
0	0	0	1	0	1
0	1	0	1	1	0
1	0	0	1	1	0
1	1	1	0	1	0
		ÉS (AND)	NON-ÉS (NAND)	VAGY (OR)	NON-VAGY (NOR)

5. FUNKCIONÁLISAN TELJES RENDSZEREK

1. A GYAKORLAT CÉLJA:

Az alapfüggvények megtervezése és megvalósítása NÉV, NAND és NOR funkcionálisan teljes rendszerekben, illetve a működés tesztelése *Deeds (Digital Circuit Simulator)* software és/vagy a *Leybold Didactic* munkapanel és eszközpark segítségével.

2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator) programcsomag

3. ELMÉLETI ALAPOK RÖVIDEN:

*Funkcionálisan teljes rendszer*nek nevezzük azokat a logikai függvényekből és az őket megvalósító kapuáramkörökből álló rendszereket, amelyekből bármilyen tetszőleges logikai hálózat megvalósítható – ennek következtében egy adott logikai hálózat megvalósításához csak meghatározott típusú kapuáramkört kell használni – ezek lesznek majd a rendszer alkotóelemei.

A funkcionálisan teljes rendszerek előnyei a gyártási technológia (csak egy fajta kapu az IC -ben), a logisztikai folyamatok (rendelés, raktározás), a kivitelezés (alkalmazások tervezése, alkatelemek beültetése) és a hibaelhárítás, illetve javítást szintjein mutatkoznak meg.

Mivel bármelyik tetszőleges logikai függvény kifejezhető az ÉS, a VAGY és a TAGADÁS műveleteit megvalósító logikai alapkapuk megfelelően képzett kombinációjával … a negáció, a konjunkció és a diszjunkció függvényei, együtt az INVERTER, az ÉS, illetve a VAGY kapukkal egy olyan funkcionálisan teljes rendszert alkotnak, amelynek a szakmai gyakorlatban bevált neve *NÉV* (NEM – ÉS – VAGY) *funkcionálisan teljes rendszer* – a felhasznált kapuk magas száma és különbözősége miatt, ez a rendszer gyakorlati szempontból nem terjedt el – csak elméleti/didaktikai jelentősége van!

A szakmai gyakorlat két másik jól bevált funkcionálisan teljes rendszert is számontart, ezek a *NAND* (NOT AND = NEM-ÉS = NEGÁLT ÉS), illetve a *NOR* (NOT OR = NEM-VAGY = NEGÁLT VAGY) *funkcionálisan teljes rendszerek*. Óriási előnyük a viszonylag kis számú felhasznált, azonos típusú logikai kapu – ez energiafelhasználási, meghajtási és költségi előnyt is jelent!

4. A GYAKORLATI MUNKA MENETE:

(a) kizárólag NAND és/vagy NOR kapuk felhasználásával, illetve a két bemeneti változós de Morgan féle tételek segítségével, tervezzük meg a TAGADÁS, az ÉS, a VAGY, a NEM-ÉS és a NEM-VAGY logikai műveleteket elvégző logikai kapcsolásokat

(b) készítsük el a logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével

(c) a rendelkezésünkre álló eszközparkokkal készítsük el a logikai kapcsolások működőképes logikai áramköreit
(d) bizonyítsuk be kísérletileg, hogy a de Morgan tételek nemcsak két bemeneti logikai változó esetén érvényesek
(e) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

6. A BOOLE - ALGEBRA TÉTELEINEK ELLENŐRZÉSE

1. A GYAKORLAT CÉLJA:

A Boole algebra tételeinek gyakorlati ellenőrzése és kapcsolástechnikai megvalósítása NÉV, NAND és NOR funkcionálisan teljes rendszerekben, illetve a működés tesztelése *Deeds (Digital Circuit Simulator)* software és/vagy a *Leybold Didactic* munkapanel és eszközpark segítségével.

2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator) programcsomag

3. ELMÉLETI ALAPOK RÖVIDEN:

Mivel a digitális elektronikában leggyakrabban az úgynevezett *pozitív logika* használatos, ezért a *Boole-algebra* operátorait (műveleteit) és legfontosabb törvényszerűségeit (tételeit) ezzel a logikával adjuk meg.

A Boole-algebra alapműveletei (operátorai):

a *TAGADÁS*, azaz a *negáció* vagy *invertálás*
az *ÉS*, azaz a *konjunkció* vagy *logikai szorzás*
a *VAGY*, azaz a *diszjunkció* vagy *logikai összeadás*

Az alapműveletekkel való számolás szabályrendszerét a *Boole-féle algebra* határozza meg.

Ez a következő axiómarendszerre épül:

(1) A *semleges (nulla) elem* létezése: $A \cdot 0 = 0$
$A + 0 = A$

(2) Az *egység elem* létezése: $A \cdot 1 = A$
$A + 1 = 1$

(3) *Komplemens elem* létezése: $A \cdot \overline{A} = 0$
$A + \overline{A} = 1$

(4) *Kommutatívitás (felcserélhetőség)*: $A \cdot B = B \cdot A$
$A + B = B + A$

(5) *Asszociativitás (csoportosíthatóság)*: $(A \cdot B) \cdot C = A \cdot (B \cdot C)$
$(A + B) + C = A + (B + C)$

(6) *Disztributivitás (tagolhatóság)*: $A \cdot (B + C) = A \cdot B + A \cdot C$
$A + (B \cdot C) = (A + B) \cdot (A + C)$

(7) *Abszorbció* (*elnyelés*): $A \cdot (A + B) = A + A \cdot B = A$
$A + \overline{A} \cdot B = A + B$

(8) *A de Morgan-féle azonosságok*: $\overline{A+B} = \overline{A} \cdot \overline{B}$

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

(9) *Tautológia:* $A \cdot A = A$ *(idempotencia)*

$A + A = A$ *(idempotencia)*

$\overline{\overline{A}} = A$ *(involúció)*

4. A GYAKORLATI MUNKA MENETE:

(a) tervezzük meg az axiómákat megvalósító kapcsolásokat

(b) készítsük el a logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével

(c) a rendelkezésünkre álló eszközparkokkal készítsük el a logikai kapcsolások működőképes logikai áramköreit

(d) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

7. LOGIKAI FÜGGVÉNYEK ALGEBRAI ALAKBAN

1. A GYAKORLAT CÉLJA:

A logikai függvények megadása és egyszerűsítése algebrai alak alapján, majd kapcsolástechnikai megvalósításuk NÉV, NAND és NOR funkcionálisan teljes rendszerekben, illetve a működés tesztelése *Deeds (Digital Circuit Simulator)* software és/vagy a *Leybold Didactic* munkapanel és eszközpark segítségével.

2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator) programcsomag

3. ELMÉLETI ALAPOK RÖVIDEN:

Az *algebrai alak*ban megadott logikai függvények hátránya, hogy egy függvény több egymással ekvivalens módon is felírható, ennek kiküszöbölésére vezették be a szabályos kanonikus alakot. A szabályos alaknak igen nagy jelentősége van, mivel a logikai hálózatok tervezésénél használt szisztematikus (módszeres) függvényegyszerűsítő eljárásokat csak szabályos alak esetén lehet alkalmazni.

A független logikai változók azon csoportjait, amelyeket azonos logikai kapcsolatra jellemző szimbólumokkal kötünk össze *term*nek nevezzük. A logikai algebrában kétfajta termet különböztetünk meg (*n* a logikai változók száma, *i* és *j* a term sorszáma vagy indexszáma):

MINTERM m_i^n = a független változók logikai ÉS kapcsolata, amelyben minden változó (igaz vagy tagadott formában) egyszer és csakis egyszer szerepel

Pld.: $A \cdot \overline{B}$

MAXTERM M_j^n = a független változók logikai VAGY kapcsolata, amelyben minden változó (igaz vagy tagadott formában) egyszer és csakis egyszer szerepel

Pld.: $\overline{B} + C$

A minterm és a maxterm sorszámát a term változóiból bináris kód alapján képezzük, a változókat jobbról balra növekvő sorrendű bináris helyi értéknek tekintjük, majd az igaz változókat 1-nek, a tagadott változókat 0-nak tekintve a keletkezett bináris számot decimálissá alakítjuk:

$$A \cdot \overline{B} = 1 \cdot 2^1 + 0 \cdot 2^0 = 2 + 0 = 2 \rightarrow m_2^2$$
$$\overline{B} + C = 0 \cdot 2^1 + 1 \cdot 2^0 = 0 + 1 = 1 \rightarrow M_1^2$$

A kétfajta term közötti kapcsolat (a de Morgan féle azonosságok alapján):

$$m_i^n = \overline{M_{2^n-1-i}^n}$$
$$M_i^n = \overline{m_{2^n-1-i}^n}$$

A mintermek és a maxtermek segítségével kétfajta kanonikus alakot írhatunk fel:

DISZJUNKTÍV = olyan logikai függvény, amely mintermek VAGY (+) kapcsolatából áll

Pld.: $A \cdot B \cdot \overline{C} + \overline{A} \cdot B \cdot C$

KONJUNKTÍV = olyan logikai függvény, amely maxtermek ÉS (·) kapcsolatából áll – *a szakmai gyakorlat ritkábban használja*

Pld.: $(\overline{A} + C) \cdot (B + \overline{D})$

4. A GYAKORLATI MUNKA MENETE:

(a) felhasználva a Boole-féle algebra tételeit és azonosságait, amennyiben az lehetséges, egyszerűsítsük az alább felsorolt logikai kifejezéseket
(b) tervezzük meg NÉV, NAND és/vagy NOR funkcionálisan teljes rendszerben az egyszerűsített kifejezést megvalósító logikai kapcsolást
(c) készítsük el a logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével
(d) a rendelkezésünkre álló eszközparkokkal készítsük el a logikai kapcsolások működőképes logikai áramköreit
(e) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

Egyszerűsítendő logikai kifejezések:

(1.) $Y = A \cdot B \cdot C + A \cdot B \cdot \overline{C} + \overline{A} \cdot B \cdot C + A \cdot \bar{B} \cdot \bar{C} + \bar{A} \cdot \bar{B} \cdot \bar{C}$

(2.) $Y = \overline{A \cdot B \cdot (C + B)}$

(3.) $Y = \overline{\left(\overline{A} + B + \overline{C} \cdot D\right)} + \bar{A} \cdot \bar{B} \cdot \overline{D}$

(4.) $Y = \bar{A} \cdot \bar{B} \cdot \bar{C} + \overline{A} \cdot B \cdot C + A \cdot B \cdot C + A \cdot \bar{B} \cdot \bar{C} + A \cdot \bar{B} \cdot C$

(5.) $Y = A \cdot B \cdot C + \overline{A} \cdot B \cdot \overline{D} + \bar{A} \cdot \bar{C} \cdot \overline{D} + A \cdot \bar{B} \cdot C + A \cdot \bar{B} \cdot C \cdot \overline{D}$

(6.) $Y = A \cdot \bar{B} \cdot \bar{C} \cdot \overline{D} + B \cdot C \cdot \overline{D} + \bar{A} \cdot B \cdot \overline{D} + \bar{B} \cdot C + \bar{B} \cdot C \cdot \overline{D}$

(7.) $Y = A \cdot B \cdot C \cdot D + \bar{A} \cdot \bar{B} \cdot \overline{D} + A \cdot B \cdot \bar{C} + A \cdot \bar{B} \cdot D + B \cdot C \cdot \overline{D} + \bar{A} \cdot \bar{C} \cdot \overline{D}$

(8.) $Y = \overline{(\overline{A} + C) \cdot (B + \overline{D})}$

(9.) $Y = A \cdot B \cdot \overline{C} + A \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot \overline{C} + A \cdot B \cdot C$

(10.) $Y = A \cdot B \cdot C + A \cdot \overline{B} \cdot (\overline{\overline{A} \cdot \overline{C}})$

(11.) $Y = \overline{A \cdot \overline{B} + C}$

(12.) $Y = \bar{A} \cdot \bar{B} \cdot \bar{C} \cdot \bar{D} + \bar{A} \cdot B \cdot \bar{C} \cdot \bar{D} + A \cdot B \cdot \bar{C} \cdot \bar{D} + A \cdot \bar{B} \cdot \bar{C} \cdot \bar{D} + \bar{A} \cdot B \cdot \bar{C} \cdot D +$
$+ A \cdot B \cdot \bar{C} \cdot D + \bar{A} \cdot \bar{B} \cdot C \cdot \bar{D} + \bar{A} \cdot B \cdot C \cdot \bar{D} + A \cdot \bar{B} \cdot C \cdot \bar{D}$

(13.) $Y = A \cdot \bar{B} + \bar{C} \cdot D$

(14.) $Y = \bar{A} \cdot \bar{B} + C \cdot \bar{D}$

(15.) $Y = A \cdot B + \bar{B} \cdot C$

(16.) $Y = (A + \bar{C}) \cdot \bar{B} + B \cdot C$

(17.) $Y = A \cdot B + C \cdot D$

(18.) $Y = \bar{A} \cdot B \cdot C \cdot \bar{D} + \bar{A} \cdot B \cdot C \cdot D + A \cdot B \cdot \bar{C} \cdot D + A \cdot B \cdot C \cdot \bar{D} + A \cdot B \cdot C \cdot D$

(19.) $Y = (A \cdot B \cdot \bar{C} + \bar{B} \cdot C) \cdot \bar{A} \cdot B$

(20.) $Y = A \cdot (B + \bar{A}) + (A + C) \cdot C + B$

(21.) $Y = \bar{A} \cdot B \cdot C + \bar{A} \cdot B \cdot \overline{C} + A \cdot B \cdot C + A \cdot \bar{B} \cdot C$

(22.) $Y = (A + B \cdot \bar{C}) \cdot (\bar{A} + C) + \bar{A} \cdot B \cdot \bar{C}$

(23.) $Y = C \cdot (A + B) + D \cdot (B + C) + \bar{B} \cdot D + D \cdot C$

(24.) $Y = \bar{A} \cdot \bar{B} \cdot C + \bar{A} \cdot B \cdot C + A \cdot B \cdot C$

(25.) $Y = (\overline{A} + B) \cdot (A + C)$

(26.) $Y = \overline{A \cdot C} + \bar{A} \cdot \bar{C}$

(27.) $Y = A \cdot \bar{B} \cdot \bar{C} + \overline{A + \bar{C}}$

(28.) $Y = A \cdot \bar{B} \cdot C + \overline{B + \bar{C}}$

(29.) $Y = \overline{A \cdot \bar{B} \cdot \bar{C} + \bar{A} \cdot B \cdot C}$

(30.) $Y = \overline{A \cdot B \cdot \bar{C} + A \cdot \bar{B} \cdot C} + \bar{A} \cdot \bar{B} \cdot C$

(31.) $Y = \bar{A} \cdot B \cdot C + \bar{B} \cdot \bar{C} \cdot D + A \cdot \bar{B} \cdot C \cdot D$

(32.) $Y = \overline{A \cdot \bar{B} \cdot \bar{C} + \overline{B + \bar{A} \cdot B \cdot C}}$

(33.) $Y = A \cdot (B + C + D) + A \cdot \bar{B} \cdot C + A \cdot \bar{C}$

(34.) $Y = A \cdot B \cdot C + A \cdot B \cdot \overline{C} + \overline{A} \cdot B \cdot C + A \cdot \bar{B} \cdot C$

(35.) $Y = \overline{\bar{A} + B + \bar{C} + D} + \bar{A} \cdot \bar{B} \cdot \bar{D}$

(36.) $Y = \bar{A} \cdot \bar{B} \cdot \bar{C} \cdot \bar{D} + \bar{C} \cdot D + A \cdot \bar{B} \cdot C + \bar{D}$

(37.) $Y = (A + B) \cdot (\overline{A} + C) \cdot (\overline{B} + C)$

(38.) $Y = \overline{A} \cdot B \cdot \overline{C} + A \cdot B \cdot \overline{C} + B \cdot \overline{C} \cdot D$

(39.) $Y = \overline{\bar{A} \cdot B \cdot \bar{C}}$

(40.) $Y = \overline{\overline{A} + \overline{B} \cdot C}$

(41.) $Y = \overline{A \cdot B \cdot \overline{\overline{C}} \cdot \overline{\overline{D}}}$

(42.) $Y = \overline{A \cdot B \cdot \overline{C \cdot D}}$

(43.) $Y = \overline{A \cdot \left(\overline{B + \overline{\overline{C}}}\right) \cdot D}$

(44.) $Y = \overline{A \cdot B \cdot (C + D)}$

8. LOGIKAI FÜGGVÉNYEK SZÖVEGES FORMÁBAN

1. A GYAKORLAT CÉLJA:

A logikai függvények megadása és egyszerűsítése szöveges alak alapján, majd kapcsolástechnikai megvalósításuk NÉV, NAND és NOR funkcionálisan teljes rendszerekben, illetve a működés tesztelése *Deeds (Digital Circuit Simulator)* software és/vagy a *Leybold Didactic* munkapanel és eszközpark segítségével.

2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator)

3. ELMÉLETI ALAPOK RÖVIDEN:

A *szöveges forma* a logikai függvények egy olyan sajátos megadási lehetősége, amelynek során a logikai kapcsolatokat, műveleteket vagy törvényszerűséget szöveges leíró formában adjuk meg.

4. A GYAKORLATI MUNKA MENETE:

(a) "olvassuk ki" az alább felsorolt feladatok szövegéből a megoldást biztosító logikai kifejezés valamilyen formáját
(b) készítsük el a feladatok igazságtáblázatát és Karnaugh-Veitch diagramját
(c) egyszerűsítsük a kapott logikai kifejezéseket kizárólag az igazságtáblázat vagy a diagram alapján
(d) felhasználva a Boole-féle algebra tételeit és azonosságait, amennyiben az lehetséges, egyszerűsítsük a kapott logikai kifejezéseket
(e) tervezzük meg NÉV, NAND és NOR funkcionálisan teljes rendszerben az egyszerűsített kifejezést megvalósító logikai kapcsolást
(f) készítsük el a logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével
(g) a rendelkezésünkre álló eszközparkokkal készítsük el a logikai kapcsolások működőképes logikai áramköreit
(h) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

Megoldandó logikai szöveges feladatok:

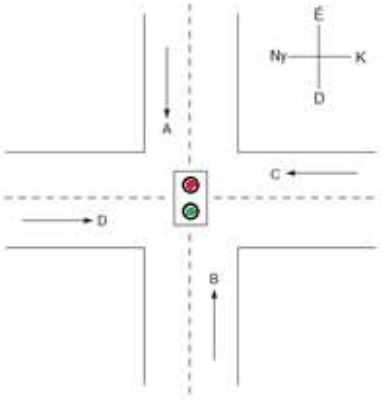
(1.) Tervezzünk egy olyan kombinációs logikai hálózatot, amely 3 vagy 5 szavazó esetén a többségi szavazatszámlálás elvén működik (tartózkodás nem megengedett)!

(2.) Tervezzünk egy olyan kombinációs logikai hálózatot, melynek kimenete 0 állapotban található ahányszor *C* jelen van, illetve *A* jelen van egyidejűleg *B*-vel vagy *D*-vel!

(3.) Tervezzünk egy olyan négy bemenetes logikai hálózatot, amelynek kimenete logikai 1 állapotban lesz valahányszor *A* és *B* egyidejűleg 1, ugyanakkor *C* és *D* egyidejűleg 0 vagy 1.

(4.) Egy értékdoboznak 3 kulcsa van (1 a tulajdonosnál, 1 a bankigazgatónál és 1 a teremőrnél). Egyszerre mindig 2 kulcs szükséges az értékdoboz kinyitásához, de a tulajdonos kulcsa nélkül a doboz nem nyitható! Tervezzünk egy olyan logikai hálózatot, amely az értékdoboz kinyitását valósítja meg.

(5.) Az ábra egy útkereszteződést szemléltet, ahol a *CD* irány a főutat jelöli, az *AB* irány pedig a mellékutat. Az úttesteken járműérzékelő szenzorok találhatók, ezek kimenetén a logikai 0 a járművek hiányát, a logikai 1 a járművek jelenlétét mutatja. Az útkereszteződést irányító jelzőlámpa a pozitív logikát követi (0 = piros, 1 = zöld). A kereszteződést vezérlő szabályok:



(a) K-Ny irány: zöld, ha *C*-n és *D*-n is autó van;
(b) K-Ny: zöld, ha *C* vagy *D* foglalt, de *A* és *B* nem egyszerre foglalt;
(c) É-D: zöld, ha *A* és *B* egyidejűleg foglalt, de *C* és *D* nem egyszerre foglalt;
(d) É-D: zöld, ha *A* vagy *B* foglalt, de *C* és *D* üres;
(e) Ha az útkereszteződés szomszédságában nincsenek autók, K-Ny irányban zöld a jelző.

Határozzuk meg az útkereszteződés jelzőlámpáját vezérlő logikai függvényt.

(6.) Határozzuk meg egy háromszintes épület felvonóját irányító logikai függvényt. A működést vezérlő logikai vázlatot a mellékelt ábra szemlélteti, ahol:

- M a felvonó mozgásállapotát kvantáló logikai változó ($M = 1$, ha a felvonó mozgásban van és $M = 0$, ha valamelyik emeleten áll)
- $E1$, $E2$ és $E3$ az épületszinteknek megfelelő logikai változók (normál állapotuk a *logikai 0*, csak akkor vannak *logikai 1* szinten, ha a felvonó az adott emeleten áll, Pld.: a felvonó a 2. szinten van $\Rightarrow E2 = 1$ és $E1 = E3 = 0$)

- a függő, kimeneti logikai változó a felvonó ajtója A, melynek 1-es logikai értéke azt jelenti, hogy az ajtó nyitva (természetesen, az $A = 0$ feltétel azt jelenti, hogy a felvonóajtó csukva)

(7.) Egy bank olyan riasztórendszert akar telepíteni, amely három mozgásérzékelőn alapul. A hamis riasztások kiszűrésére a riasztás csak akkor indul el, ha a három érzékelő közül legalább kettő egyidejűleg jelez. Határozzuk meg a riasztórendszert vezérlő logikai függvény egyszerűsített alakját és tervezzük meg az azt megvalósító logikai hálózatot.

9. EGYSZERŰSÍTÉS LOGIKAI VÁZLAT ALAPJÁN

1. A GYAKORLAT CÉLJA:

A logikai függvények megadása és egyszerűsítése logikai vázlat alapján, majd kapcsolástechnikai megvalósításuk NÉV, NAND és/vagy NOR funkcionálisan teljes rendszerekben, illetve a működés tesztelése *Deeds (Digital Circuit Simulator)* software és/vagy a *Leybold Didactic* munkapanel és eszközpark segítségével.

2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator) programcsomag

3. ELMÉLETI ALAPOK RÖVIDEN:

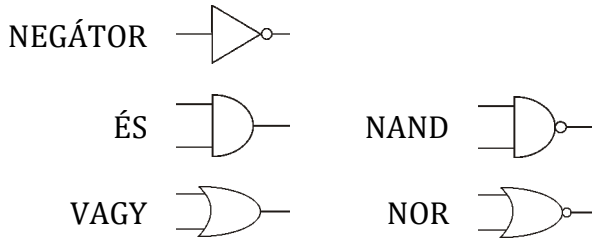
A *logikai vázlat* a logikai függvények egy olyan sajátos grafikus megadási lehetősége, amely az elvégzendő logikai műveleteket az őket megvalósító áramköri szimbólumokkal adja meg kapcsolási rajz formájában.

A kapcsolási rajzon kell végig követni a logikai változók alakulását, a bemenettől a kimenetig, figyelembe véve az összes elszenvedett logikai műveletet.

A logikai vázlat, összetettségétől és bonyolultságától függően, két vagy többszintes logikai hálózat.

A hálózat szintjeit a kimenettől a bemenet fele haladva számozzuk, azonosításuk úgy történik, hogy az utolsónak elvégzendő logikai művelettől kezdve visszafele haladva megszámoljuk az egymás után elvégzendő műveletek számát.

Rajzjelek:



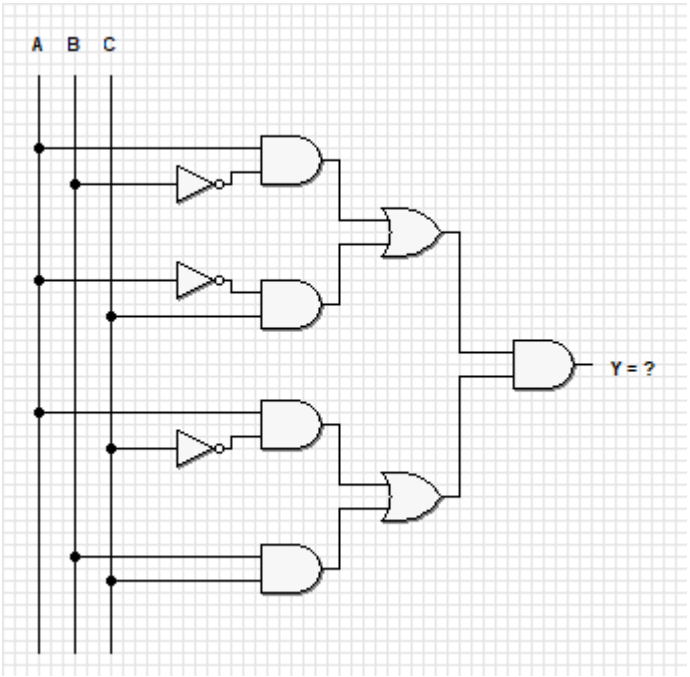
4. A GYAKORLATI MUNKA MENETE:

(a) olvassuk ki a logikai vázlatból a logikai függvény algebrai alakját
(b) határozzuk meg a logikai szintek számát
(c) készítsük el a logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével

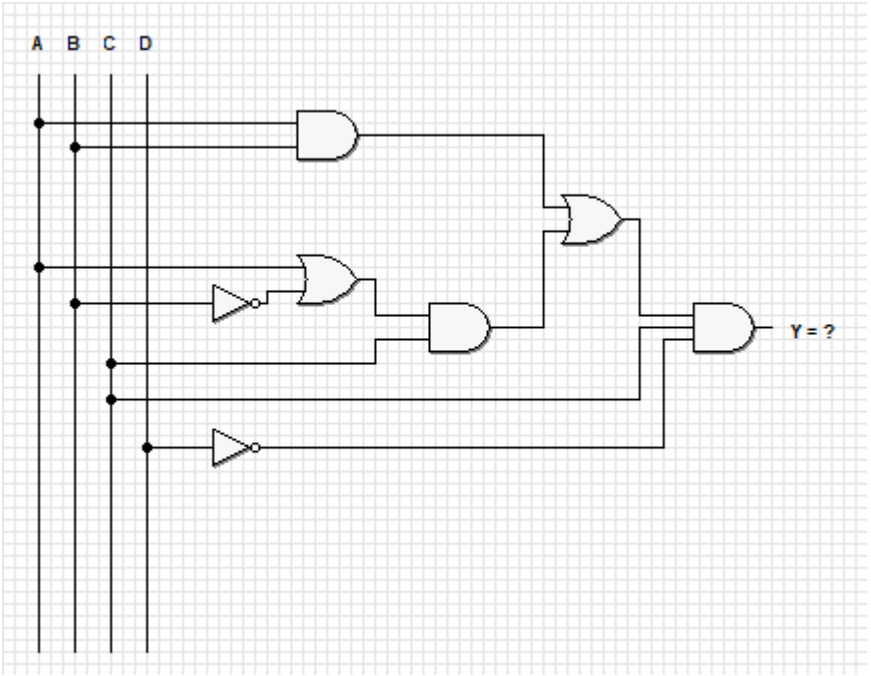
(d) tervezzük meg NÉV, NAND és/vagy NOR funkcionálisan teljes rendszerben az egyszerűsített kifejezést megvalósító logikai kapcsolást
(e) készítsük el az így kapott logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével
(f) a rendelkezésünkre álló eszközparkokkal építsük meg a logikai kapcsolások működőképes logikai áramköreit
(g) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

Egyszerűsítendő logikai vázlatok:

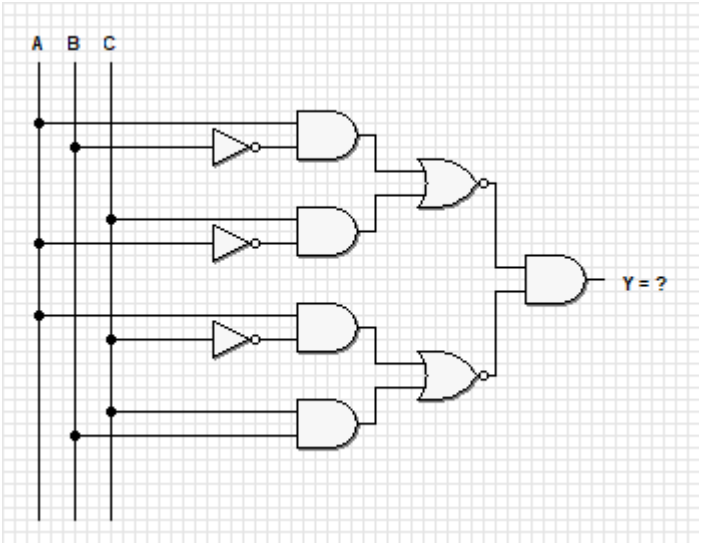
(1.)



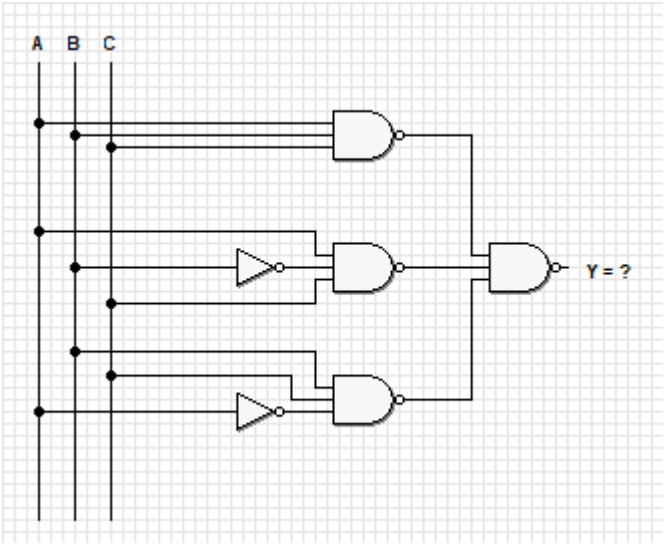
(2.)



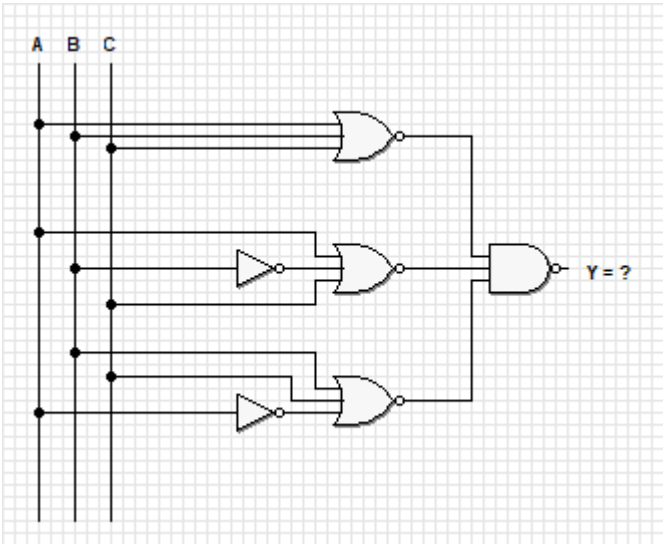
(3.)



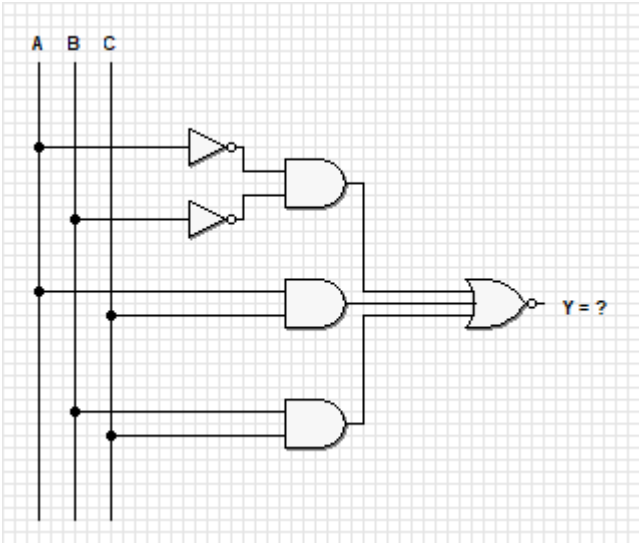
(4.)



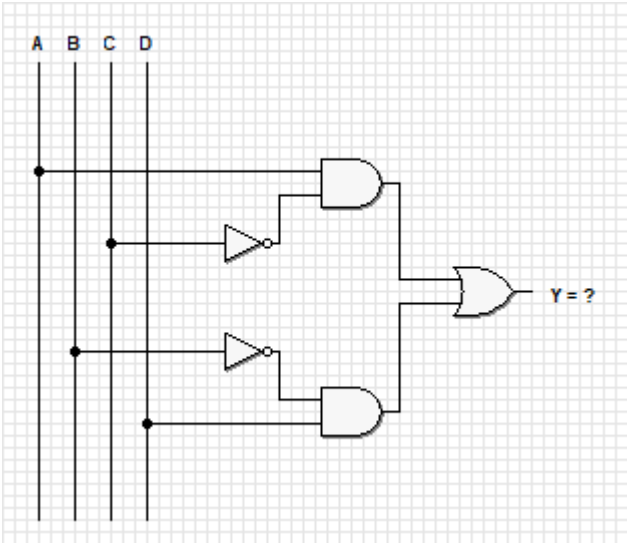
(5.)



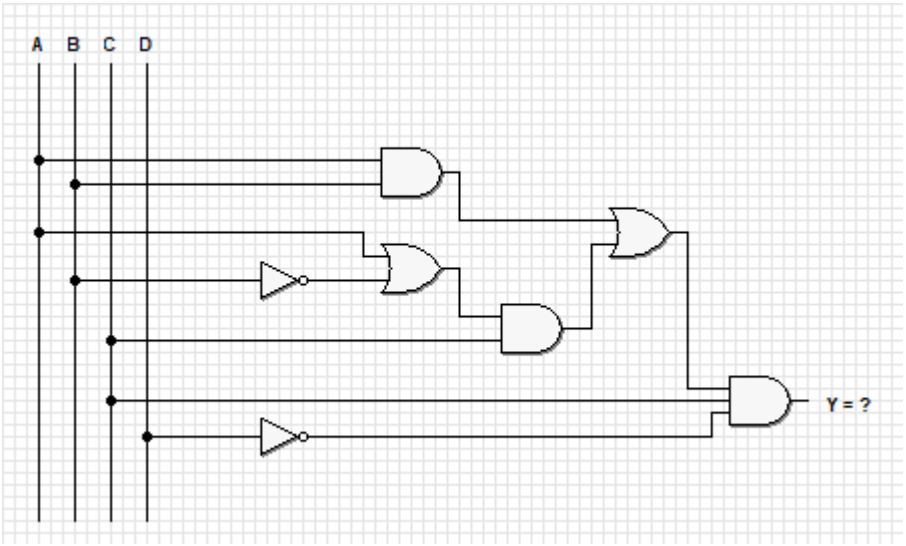
(6.)



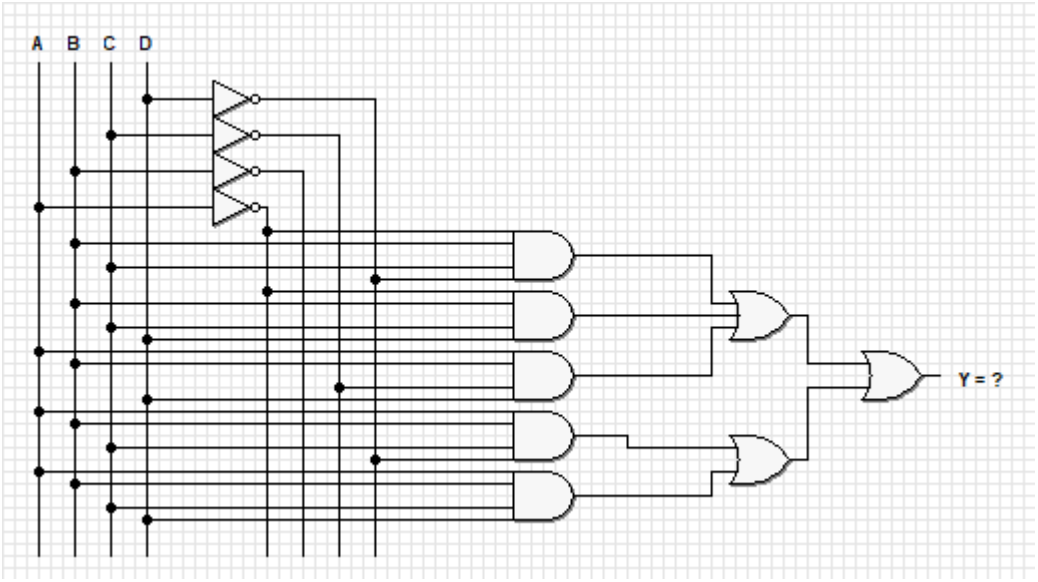
(7.)



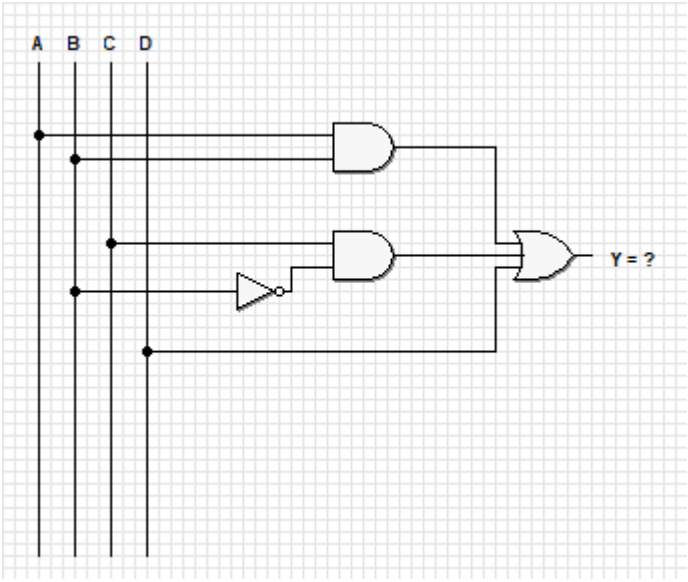
(8.)



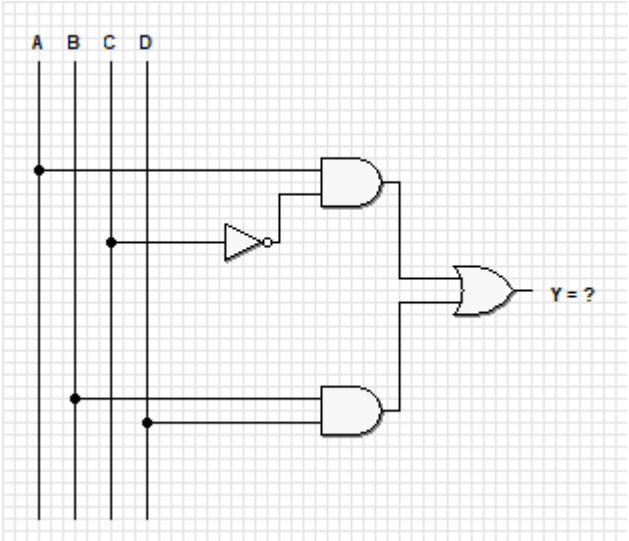
(9.)



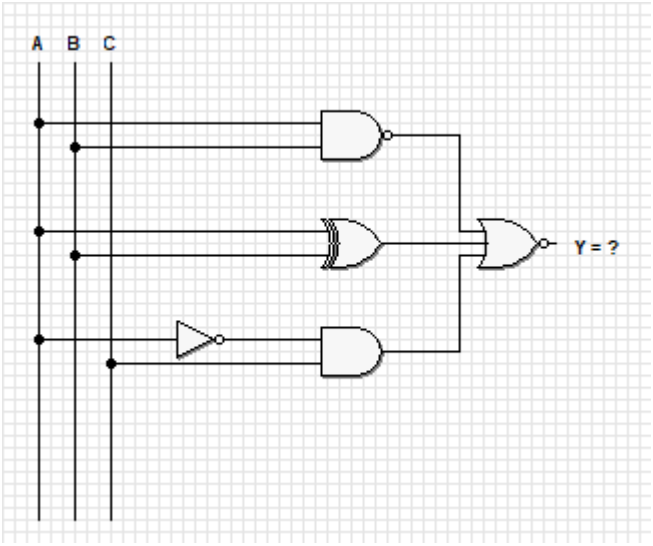
(10.)



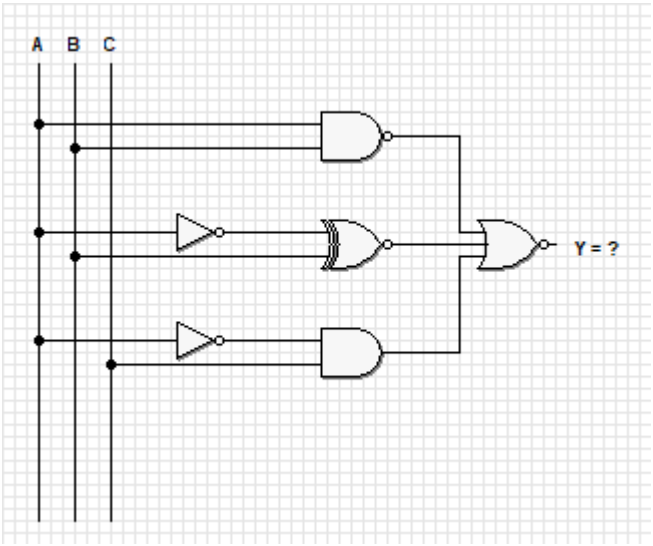
(11.)



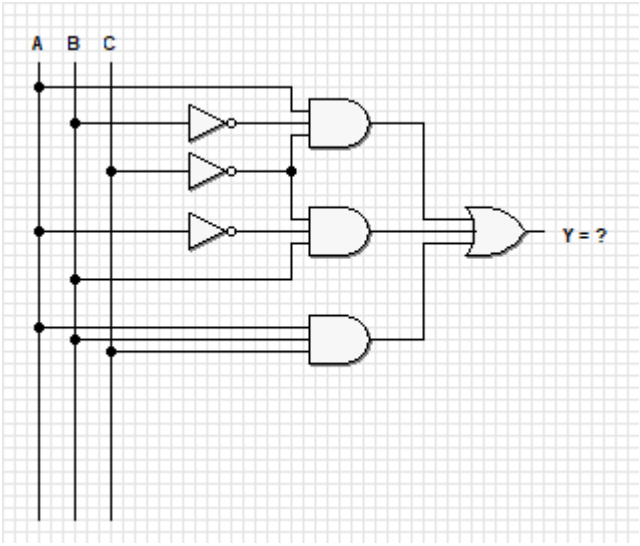
(12.)



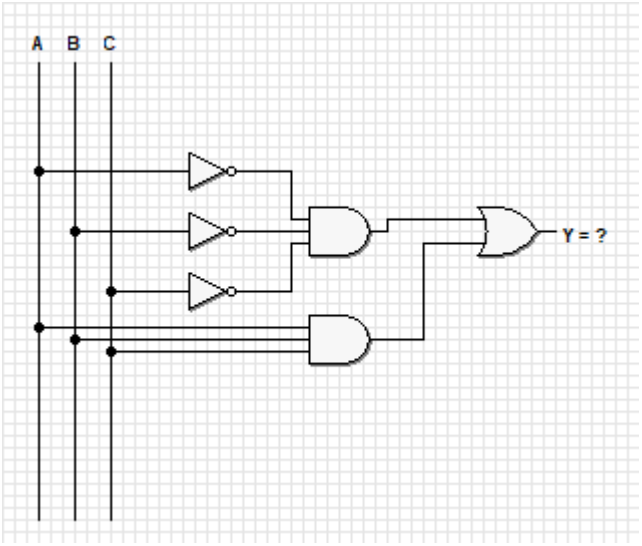
(13.)



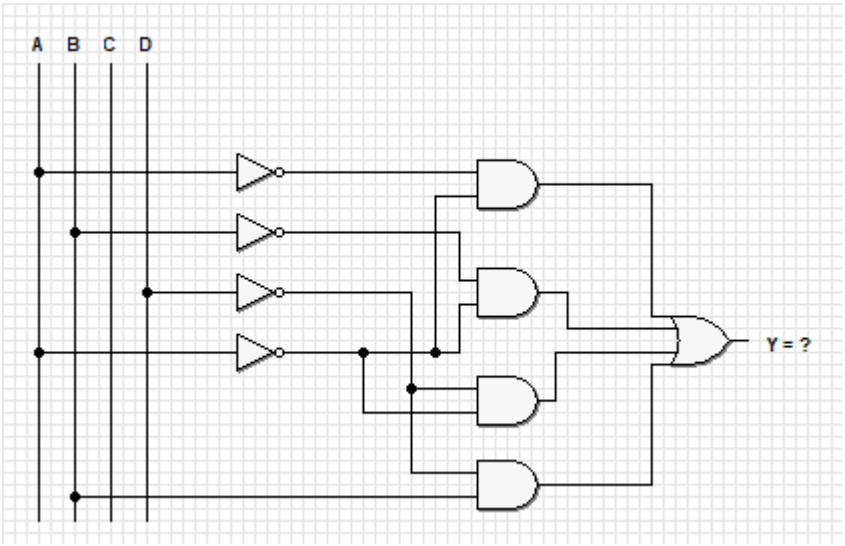
(14.)



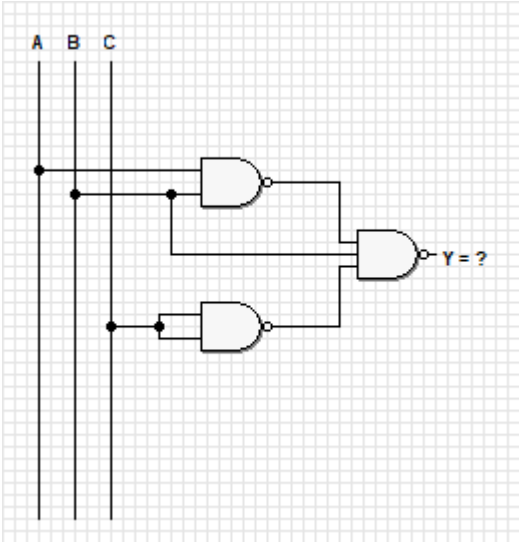
(15.)



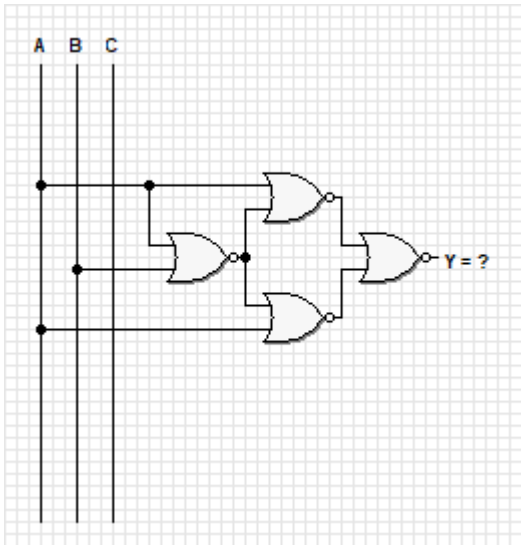
(16.)



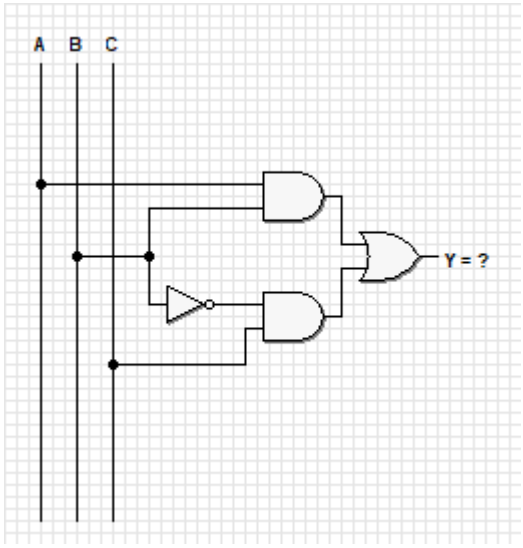
(17.)



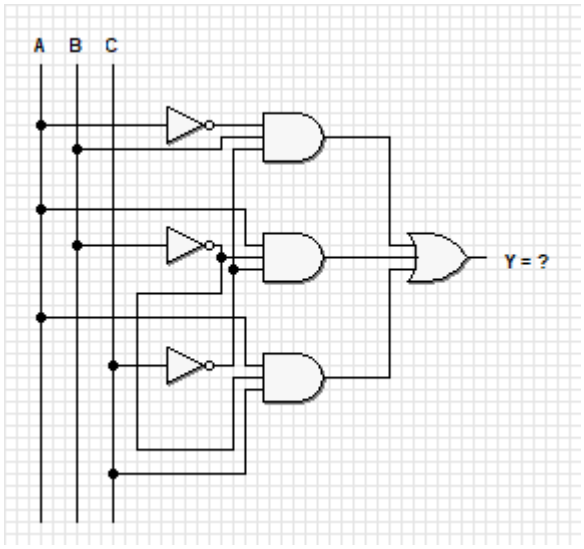
(18.)



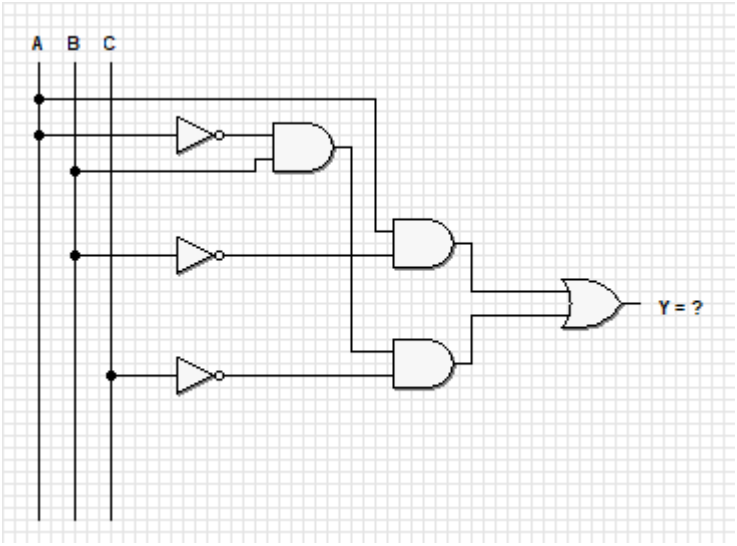
(19.)



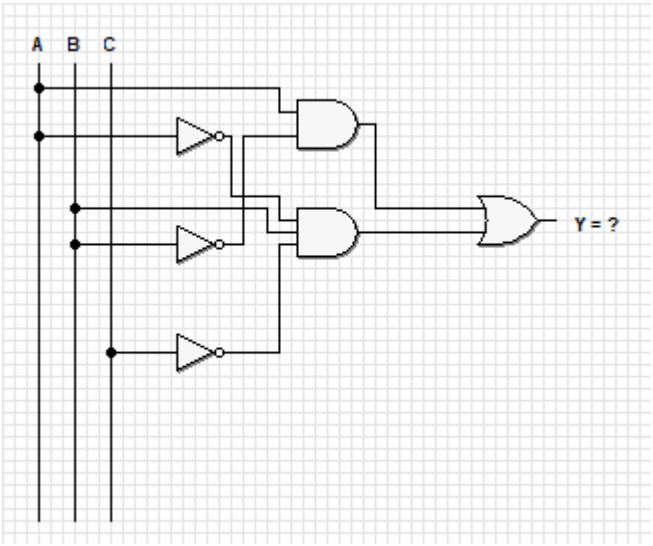
(20.)



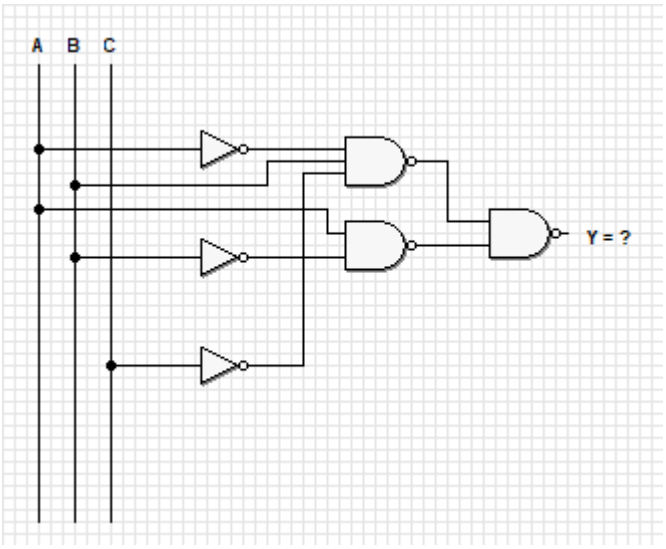
(21.)



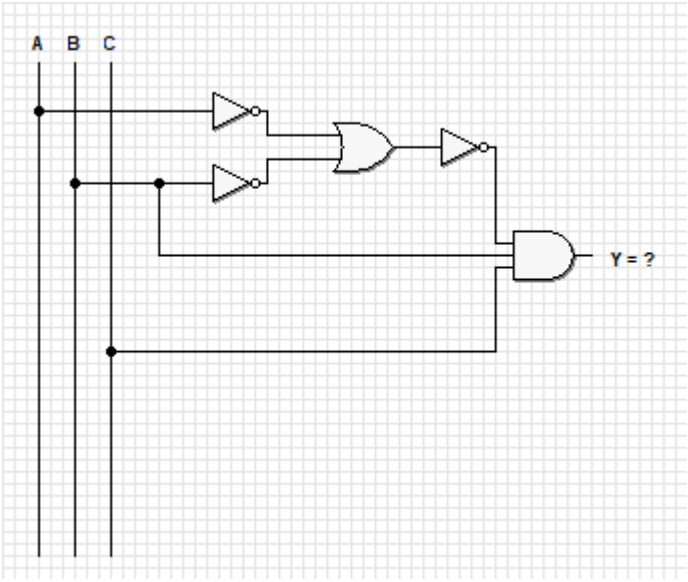
(22.)



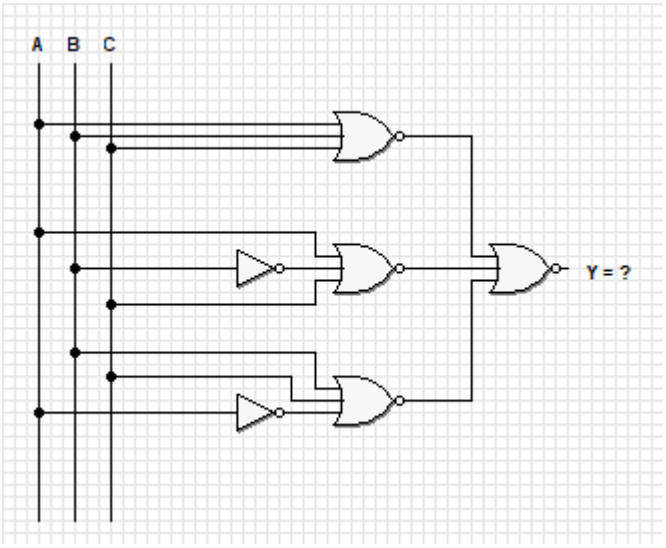
(23.)



(24.)



(25.)



10. EGYSZERŰSÍTÉS IGAZSÁGTÁBLÁZATTAL

1. A GYAKORLAT CÉLJA:

A logikai függvények megadása és egyszerűsítése igazságtáblázat alapján, majd kapcsolástechnikai megvalósításuk NÉV, NAND és NOR funkcionálisan teljes rendszerekben, illetve a működés tesztelése *Deeds (Digital Circuit Simulator)* software és/vagy a *Leybold Didactic* munkapanel és eszközpark segítségével.

2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator) programcsomag

3. ELMÉLETI ALAPOK RÖVIDEN:

A *táblázatos forma* a logikai függvények egy olyan sajátos megadási lehetősége, amelynek során a függő és a független változók valamennyi lehetséges értékét táblázatban adjuk meg egymáshoz rendelésük figyelembevételével. A leggyakrabban használt logikai táblázat az *igazságtáblázat*, amelynél a független változók minden egyes értékkombinációjához megadjuk az adott szabályok által hozzárendelhető függő változó logikai értékét, a táblázat oszlopainak számát az összes logikai (függő + független) változó száma, a sorok számát pedig a független változók lehetséges kombinációinak száma (2^n, ahol n független logikai változók száma) adja meg. A legegyszerűbb kitöltési mód az, amikor a legjobboldalibb oszlopot tekintjük a függő változók oszlopának, jobbról balra haladva a többi oszlopot a független változóknak tartjuk fent és azért, hogy az összes lehetséges értékkombinációt lefedjük, az a kitöltési szabály, hogy a logikai 0-sok és 1-esek olyan frekvenciával váltakoznak az oszlopokban, mint a 2^k, ahol $k = 0, 1, 2 \ldots$ az adott oszlop helyének távolsága a függő változók oszlopától.

a 3 független változós és 1 függő változós igazságtáblázat esetén:

"távolság"	2	1	0	a függő
"frekvencia"	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$	változó
	a független változók oszlopai			oszlopa

Tehát a táblázat független változós része kitöltve így fog kinézni:

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

A függvény "kiolvasásához" megkeressük a függő változó logikai 1-es értékeit, majd az azoknak megfelelő függő változók ÉS kombinációi között VAGY kapcsolatot képezünk.

4. A GYAKORLATI MUNKA MENETE:

(a) olvassuk ki az alábbi igazságtáblázatokból a logikai függvényeket
(b) felhasználva a Boole-féle algebra tételeit és azonosságait, amennyiben az lehetséges, egyszerűsítsük a kapott logikai kifejezéseket
(c) tervezzük meg NÉV, NAND és/vagy NOR funkcionálisan teljes rendszerben az egyszerűsített kifejezést megvalósító logikai kapcsolást
(d) készítsük el a logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével
(e) a rendelkezésünkre álló eszközparkokkal készítsük el a logikai kapcsolások működőképes logikai áramköreit
(f) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

Igazságtáblázatok:

(1.)

D	**C**	**B**	**A**	**Y**
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	0

(2.)

A	**B**	**C**	**Y**
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

(3.)

A	**B**	**C**	**D**	**Y**
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	0

(4.)

A	**B**	**C**	**Y**
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

(5.)

A	**B**	**C**	**Y**
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

(6.)

A	**B**	**C**	**Y**
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

(7.)

D	**C**	**B**	**A**	**Y**
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

(8.)

A	**B**	**C**	**D**	**Y**
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

(9.)

A	**B**	**C**	**Y**
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

(10.)

A	**B**	**C**	**Y**
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

(11.)

A	**B**	**C**	**Y**
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

(12.)

A	**B**	**C**	**Y**
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

(13.)

A	**B**	**C**	**D**	**Y**
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

(14.)

A	**B**	**C**	**Y**
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

(15.)

A	**B**	**C**	**Y**
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

(16.)

A	**B**	**C**	**Y**
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

11. EGYSZERŰSÍTÉS KARNAUGH-VEITCH TÁBLÁVAL

1. A GYAKORLAT CÉLJA:

A logikai függvények megadása és egyszerűsítése Karnaugh-Veitch diagram alapján, majd kapcsolástechnikai megvalósításuk NÉV, NAND és NOR funkcionálisan teljes rendszerekben, illetve a működés tesztelése *Deeds (Digital Circuit Simulator)* software és/vagy a *Leybold Didactic* munkapanel és eszközpark segítségével.

2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator) programcsomag

3. ELMÉLETI ALAPOK RÖVIDEN:

A logikai függvények algebrai alak és Boole axiómák alapján történő egyszerűsítésének helyessége és sikere nagyban attól függ, hogy a számításokat végző személy milyen gyakorlattal vagy figyelemmel és koncentráló képességekkel rendelkezik. Egy másik komoly hátrány az, hogy az algebrai műveletvégzések nem, vagy csak igen nehezen tudják kezelni a nem teljesen definiált hálózatokat és a közömbös állapotokat.

Ezért szükségessé vált egy olyan egyszerűsítési módszer kidolgozása, amely a szubjektív emberi tényezőket kiküszöböli, ugyanakkor teljes mértékben automatizálható és használható lesz a nem teljesen specifikált hálózatok esetében is. A szakmai gyakorlat szerint egy olyan módszer vált be, amely átrendezi az igazságtáblázat alakját, szomszédos mintermeket (implikánsokat) keres és azokat összevonja az egyszerűsítés során.

Az említett módszer egy *grafikus egyszerűsítési módszer*, Edward W. Veitch gondolta ki 1952-ben, majd Maurice Karnaugh fejlesztette tovább 1953-ban – ezért ma legtöbbször *Karnaugh-Veitch módszer*ként emlegetjük és használjuk. Igen egyszerűen és elegánsan alkalmazható két, három vagy négy logikai változóig, öt és hat változó esetén már nagyon ritkán használjuk. Hatnál több logikai változó esetén már nem célszerű használni.

Tulajdonképpen egy olyan sík- vagy térbeli alakzatról (diagramról, táblázatról) van szó, amely az igazságtáblázat célszerűen átalakított változata és négyzetek (cellák) sokaságából áll. Amennyiben n logikai változóról van szó, a diagramnak $N = 2^n$ cellája lesz, minden egyes cella egy szabályos termet képvisel (min- vagy max termet), az egymás mellett levő cellákban olyan termek vannak, amelyek a Gray kód elve alapján egy és csakis egy változó logikai értékében különböznek egymástól.

A *Veitch-diagram* esetén a cellák és a termek kapcsolatát a termekből képzett bináris szám decimális értékével és az egyes változók igaz értékéhez tartozó oszlopok és sorok megjelölésével (tehát súlyozással) adjuk meg.

A *Karnaugh-tábla* számára a cellák és a termek kapcsolatát a diagram oszlopainak és sorainak bináris kódolásával adjuk meg.

Karnaugh-tábla 2 változóra:

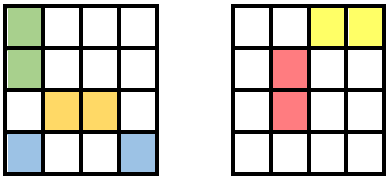
B / A	0	1
0	$\overline{A} \cdot \overline{B}$	$\overline{A} \cdot B$
1	$A \cdot \overline{B}$	$A \cdot B$

Veitch-diagram 2 változóra:

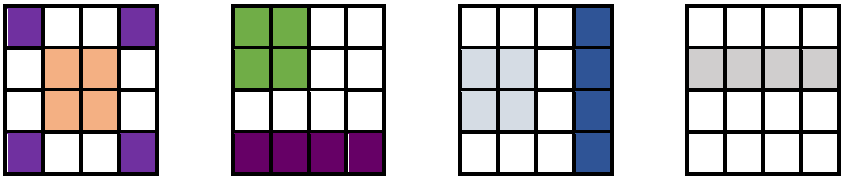
		B	
		0	1
A	2	3	

Ha a logikai 1-es 2 szomszéd cellában van, akkor azok egy *pár*t alkotnak és az értékváltás miatt 1 logikai változó fog majd „kiesni” (eltűnni, leegyszerűsödni). A 4 szomszédos cellában levő logikai 1-esek ú.n. *kvád*ot alkotnak és az említett értékváltás miatt 2 logikai változó „esik” majd ki. Amennyiben egy *oktett*et fedezünk fel (logikai 1-esek 8 szomszédos cellában) az értékváltás eredménye 3 logikai változó leegyszerűsödéséhez fog elvezetni.

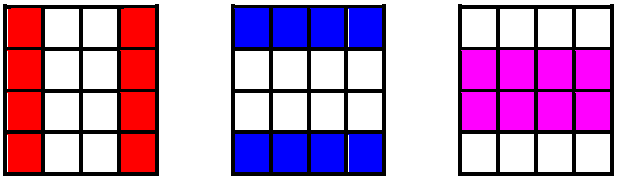
Néhány lehetséges változatot az alábbiakban szemléltetünk:



PÁR = 2 szomszéd cella ⇔ 1 változó esett ki



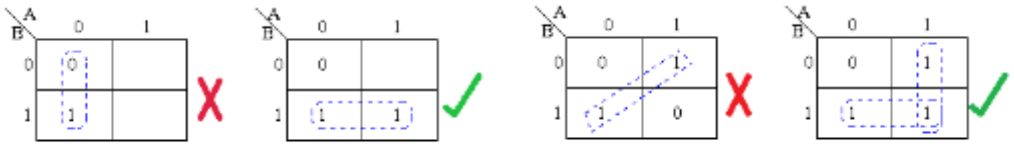
KVÁD = 4 szomszéd cella ⇔ 2 változó esett ki



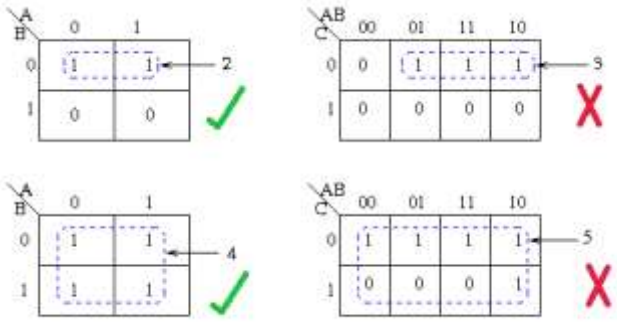
OKTETT = 8 szomszéd cella ⇔ 3 változó esett ki

A diagramok egyszerűsítésének szabályai:

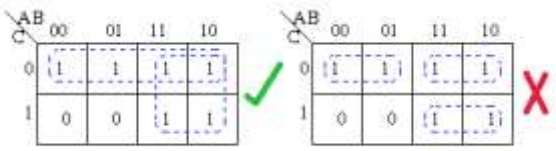
(a) A csoportosításban csak szomszédok vehetnek részt. A szomszédos cellákban csak logikai 1-esek lehetnek. A cellák csak akkor számítanak szomszédoknak, ha van közös élük.



(b) Egy csoportban csak 2^n (n = 0, 1, 2, …) cella nem lehet. Páratlan számú cellával nem alkothatunk csoportokat.

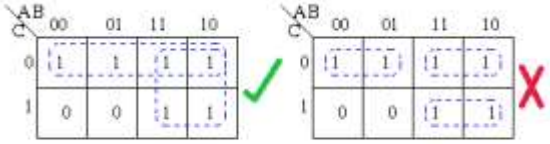


(c) Egy csoport a lehető legnagyobb kell legyen.

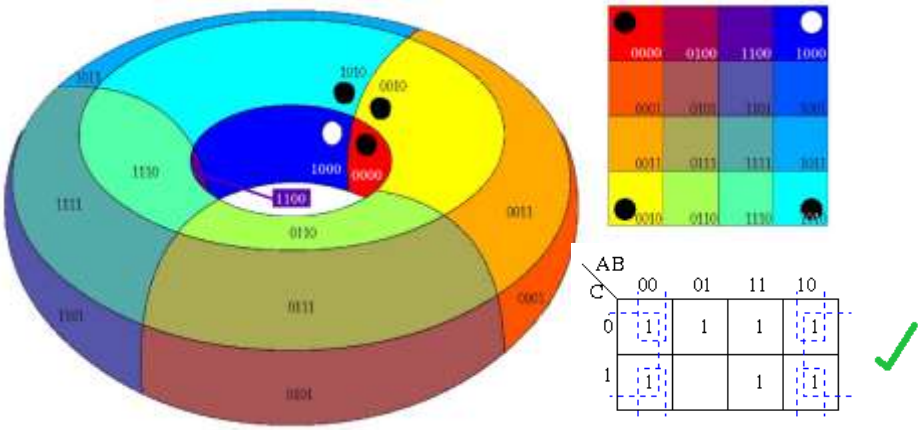


(d) Minden logikai 1-es legalább egy cellában kell legyen.

(e) A csoportok fedhetik egymást. Arra kell törekedni, hogy a kialakított csoportok száma a lehető legkisebb legyen.



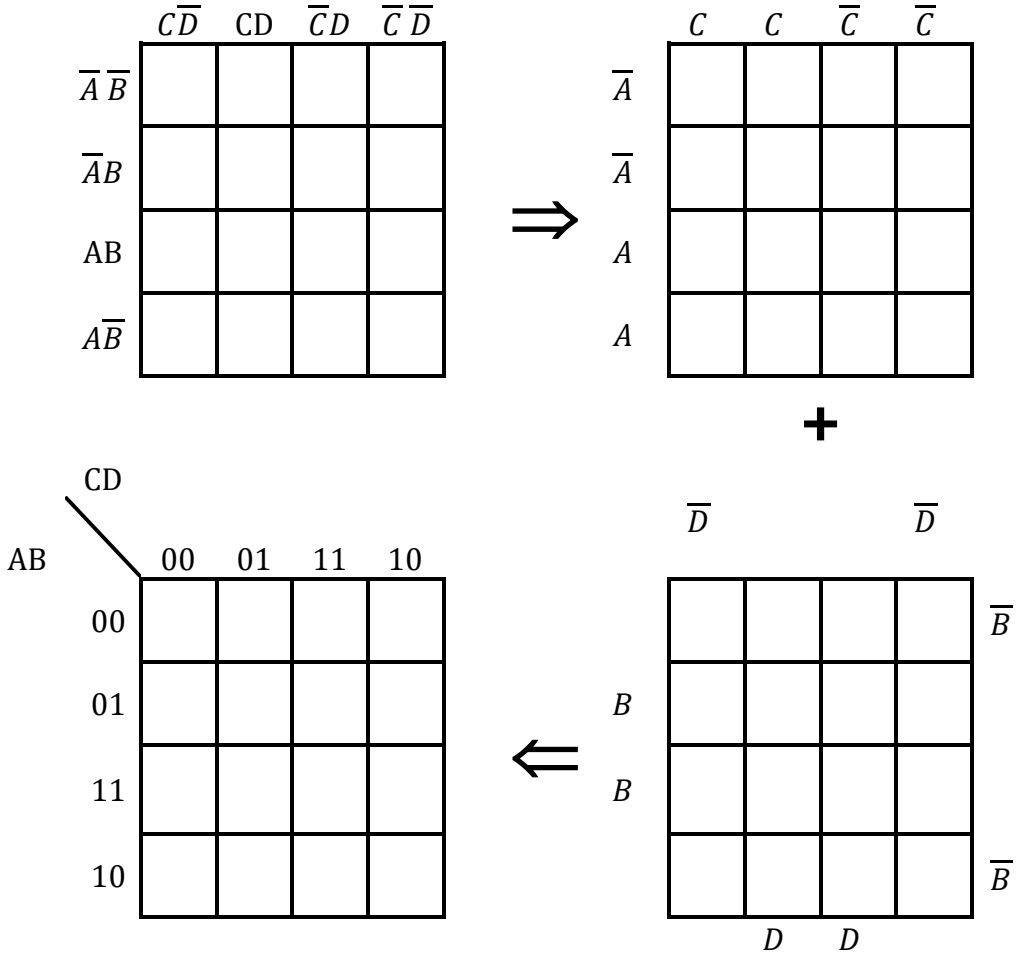
(f) A tábla 3D-s formájában „lyukas” típusú, amerikai fánk (*doughnut*) kinézetű, tehát a peremeken és a sarkokon levő cellák is szomszédok lesznek!



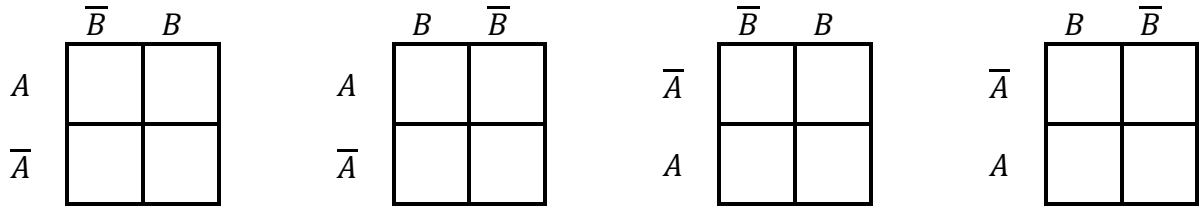
Lásd még: https://en.wikipedia.org/wiki/File:Karnaugh_map_torus.svg
https://en.wikipedia.org/wiki/File:Torus_from_rectangle.gif

A szakmai gyakorlatban célszerű összevonni az egyébként egymással ekvivalens diagramokat és mindenikből megtartani azt, ami az ábrázolásnál, illetve a kitöltésnél előnyösebb.

Karnaugh és Veitch diagramok ekvivalenciájának és kölcsönös kapcsolata 4 változó esetén:

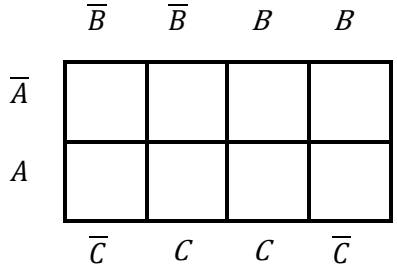


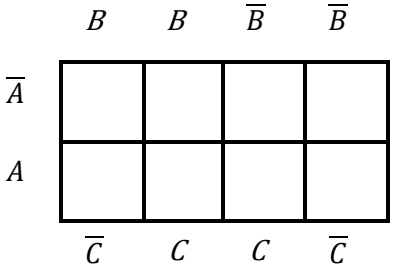
A *2 változós Karnaugh-Veitch tábla* lehetséges szemléltetési és kitöltési változatai:

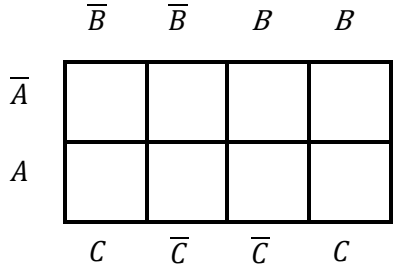


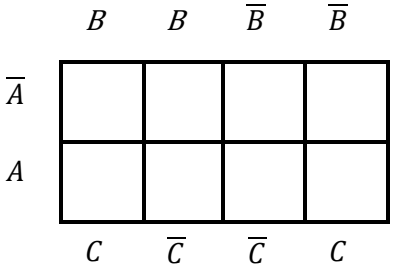
Természetesen megadhatjuk az *egyváltozós diagram*okat is, de ezeknek nincs semmiféle egyszerűsítési hasznuk, csupán tudománytörténeti jelentőséggel bírnak:

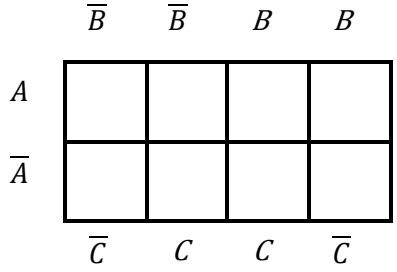
3 változó esetén a diagramok lehetséges kitöltési változatai:

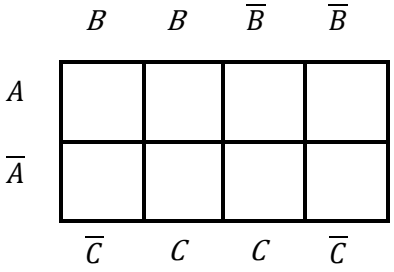


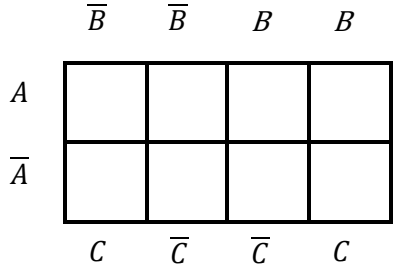


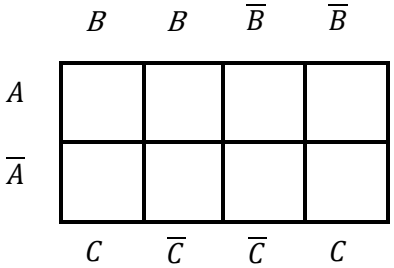




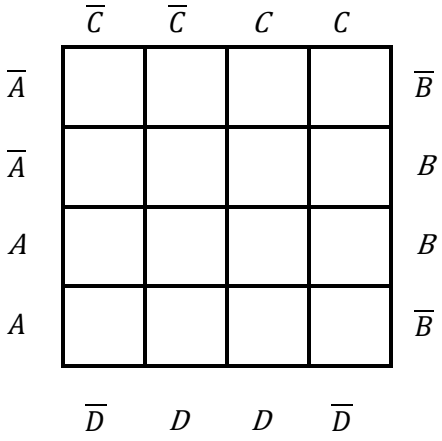


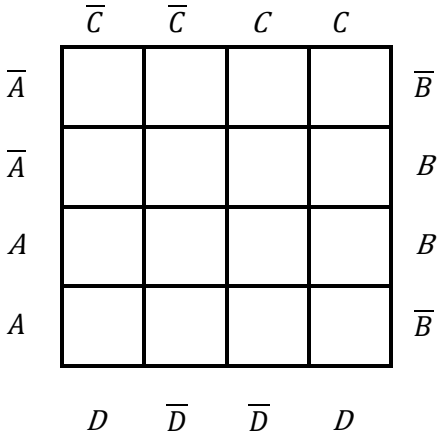


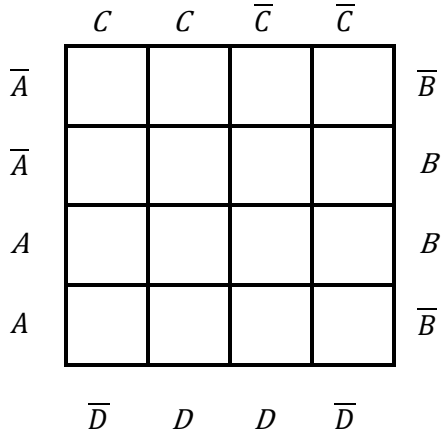


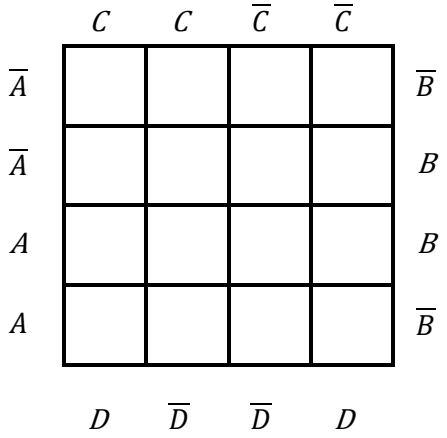


A teljesség igénye nélkül, néhány lehetséges elrendezés 4 változó esetén:



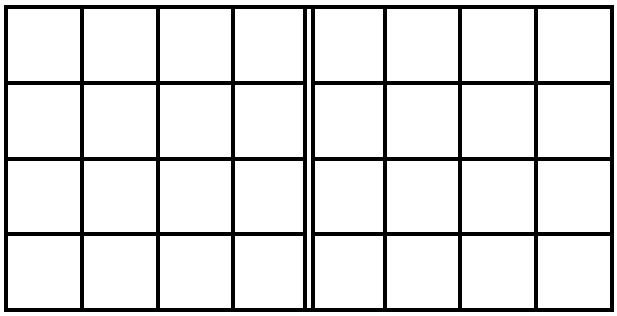


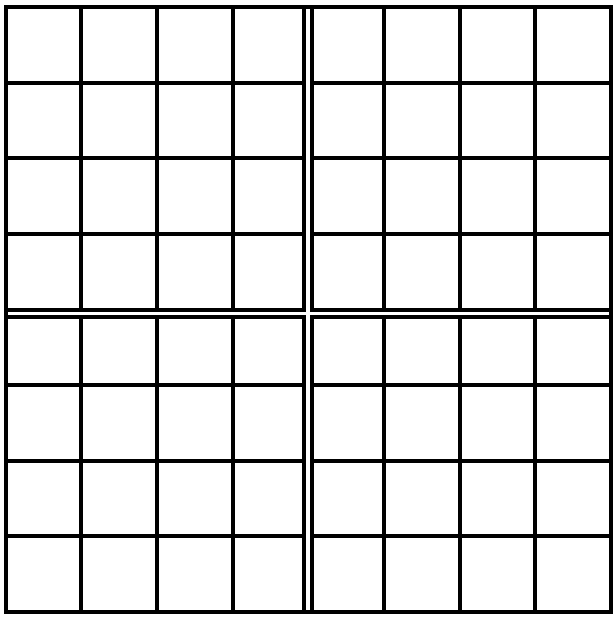




Az öt és hat változós logikai függvényeknél már sokkal bonyolultabb, de még kezelhető/megoldható helyzetbe kerülünk, de a peremezés is megnehezíti a munkát.

Az ötváltozós Karnaugh tábla nem más, mint 2 egymás mellé ragasztott "négyes", a hatváltozós Karnaugh tábla megadására ellenben már két lehetőségünk is van: vagy két egymás alá ragasztott "ötös"-t, vagy négy darab összeragasztott "négyes"-t kell használnunk.





Természetesen úgy a kitöltés, mint az összevonás igen nagy figyelmet igénylő, nehézkés tevékenység.

4. A GYAKORLATI MUNKA MENETE:

(a) olvassuk ki az alábbi diagramokból a logikai függvények algebrai alakját
(b) felhasználva a Boole-féle algebra tételeit és azonosságait, amennyiben az lehetséges, egyszerűsítsük a kapott logikai kifejezéseket
(c) tervezzük meg NÉV, NAND és/vagy NOR funkcionálisan teljes rendszerben az egyszerűsített kifejezést megvalósító logikai kapcsolást

(d) készítsük el a logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével
(e) a rendelkezésünkre álló eszközparkokkal készítsük el a logikai kapcsolások működőképes logikai áramköreit
(f) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

Egyszerűsítendő Karnaugh-Veitch diagramok:

(1.)

	$\overline{C}$	$\overline{C}$	C	C	
$\overline{A}$	1	0	1	1	$\overline{B}$
$\overline{A}$	1	1	0	0	B
A	1	1	0	0	B
A	1	0	0	0	$\overline{B}$
	$\overline{D}$	D	D	$\overline{D}$	

(2.)

	$\overline{C}$	$\overline{C}$	C	C	
$\overline{A}$	0	0	0	0	$\overline{B}$
$\overline{A}$	0	1	0	0	B
A	0	1	1	0	B
A	0	0	0	0	$\overline{B}$
	$\overline{D}$	D	D	$\overline{D}$	

(3.)

	$\overline{C}$	$\overline{C}$	C	C	
$\overline{A}$	1	1	1	1	$\overline{B}$
$\overline{A}$	0	0	0	0	B
A	0	0	1	0	B
A	0	1	1	1	$\overline{B}$
	$\overline{D}$	D	D	$\overline{D}$	

(4.)

	$\overline{C}$	$\overline{C}$	C	C	
$\overline{A}$	0	1	0	0	$\overline{B}$
$\overline{A}$	0	1	1	0	B
A	0	1	1	0	B
A	0	0	0	0	$\overline{B}$
	$\overline{D}$	D	D	$\overline{D}$	

(5.)

	$\overline{C}$	$\overline{C}$	C	C	
$\overline{A}$	1	0	0	1	$\overline{B}$
$\overline{A}$	1	1	0	1	B
A	1	1	0	0	B
A	1	0	0	1	$\overline{B}$
	$\overline{D}$	D	D	$\overline{D}$	

(6.)

	$\overline{C}$	$\overline{C}$	C	C	
$\overline{A}$	1	1	0	1	$\overline{B}$
$\overline{A}$	1	1	0	1	B
A	1	1	0	1	B
A	1	1	1	1	$\overline{B}$
	$\overline{D}$	D	D	$\overline{D}$	

(7.)

	$\overline{C}$	$\overline{C}$	C	C	
$\overline{A}$	1	1	0	0	$\overline{B}$
$\overline{A}$	1	1	0	0	B
A	0	0	0	0	B
A	1	0	0	0	$\overline{B}$
	$\overline{D}$	D	D	$\overline{D}$	

(8.)

	$\overline{C}$	$\overline{C}$	C	C	
$\overline{A}$	0	1	0	0	$\overline{B}$
$\overline{A}$	0	1	1	1	B
A	0	0	0	1	B
A	1	1	0	1	$\overline{B}$
	$\overline{D}$	D	D	$\overline{D}$	

(9.)

	$\overline{B}$	$\overline{B}$	B	B
$\overline{A}$	0	0	0	1
A	1	0	1	0
	$\overline{C}$	C	C	$\overline{C}$

(10.)

	$\overline{B}$	$\overline{B}$	B	B
$\overline{A}$	1	1	0	1
A	1	1	0	1
	$\overline{C}$	C	C	$\overline{C}$

(11.)

	$\overline{B}$	$\overline{B}$	B	B
$\overline{A}$	0	1	0	1
A	1	0	0	1
	$\overline{C}$	C	C	$\overline{C}$

(12.)

	$\overline{B}$	$\overline{B}$	B	B
$\overline{A}$	0	1	0	0
A	1	1	1	1
	$\overline{C}$	C	C	$\overline{C}$

(13.)

	$\overline{B}$	$\overline{B}$	B	B
$\overline{A}$	1	0	1	0
A	1	1	0	1
	$\overline{C}$	C	C	$\overline{C}$

12. KÜLÖNLEGES LOGIKAI FÜGGVÉNYEK

1. A GYAKORLAT CÉLJA:

Kombinációs logikai hálózatok kapcsolástechnikai megvalósítása NÉV, NAND és NOR funkcionálisan teljes rendszerekben, illetve a működés tesztelése *Deeds (Digital Circuit Simulator)* software és/vagy a *Leybold Didactic* munkapanel és eszközpark segítségével.

2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator) programcsomag

3. ELMÉLETI ALAPOK RÖVIDEN:

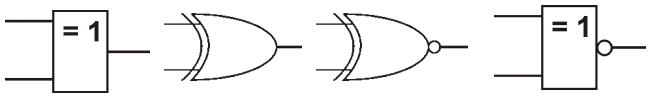
Azt a logikai függvényt melynek logikai értéke csak akkor IGAZ, ha a független (bemeneti) változok különböznek *KIZÁRÓ VAGY* (XOR), illetve *ANTIVALENCIA* néven ismerjük.

Azt a logikai függvényt melynek logikai értéke csak akkor IGAZ, ha a független (bemeneti) változok azonosak *KIZÁRÓ-NEM-VAGY* (XNOR), illetve *EKVIVALENCIA* vagy *MEGENGEDŐ ÉS* néven ismerjük.

Bár igen gyakran a logikai alapfüggvények közé szoktak sorolni őket, de a gyakorlatban inkább kombinációs logikai hálózatoknak felelnek meg és nem tekinthetők alapfüggvényeknek.

A logikai műveletek algebrai alakja és igazságtáblázata, illetve az őket megvalósító elektronikus kapcsolás áramköri (kapu) rajzjelei a következő:

bemenetek		*kimenet*	
		XOR	*XNOR*
A	B	$A \oplus B$	$A \odot B$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1



XOR XNOR

4. A GYAKORLATI MUNKA MENETE:

(a) határozzuk meg a két logikai függvény algebrai alakját
(b) létezik-e valamilyen logikai kapcsolat a két függvény között?
(c) tervezzük meg NÉV, NAND és/vagy NOR funkcionálisan teljes rendszerben a függvények kifejezését megvalósító logikai kapcsolásokat
(d) tervezzük meg a két logikai függvényt megvalósító logikai áramkört mechanikai kapcsolók segítségével
(e) hasonlítsuk össze az ekvivalencia műveletét megvalósító mechanikai kapcsolást a háztartási villanyszerelések esetén használatos kétállású kapcsoló, vagy keresztkapcsoló esetével (azaz ki-be kapcsolom ugyanazt a fényforrást két különböző helyiségből, két különböző kapcsolóval)
(f) általánosítsuk a két logikai függvényt kettőnél több logikai változóra, adjuk meg annak algebrai kifejezését és igazságtáblázatát
(g) készítsük el a logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével
(h) a rendelkezésünkre álló eszközparkokkal készítsük el a logikai kapcsolások működőképes logikai áramköreit
(i) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

13. EGY LOGIKAI TALÁLÓS KÉRDÉS

1. A GYAKORLAT CÉLJA:

A logikai kapuáramkörök által okozott késleltetési idők együttes hatásának vizsgálata.

2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- NI Multisim szimulációs program
- oszcilloszkóp és mérőszondák

3. ELMÉLETI ALAPOK RÖVIDEN:

A *logikai tagadás* (*negáció*) egyváltozós logikai művelet. Hatásként egy független logikai változóhoz hozzárendel egy olyan függő logikai változót, amely az adott változó ellentétje. A műveletet megvalósító logikai kapu neve *inverter* (*negátor*), rajzjelét és a logikai művelet igazságtáblázatát az alábbiakban adjuk meg.

jelölés/olvasat:

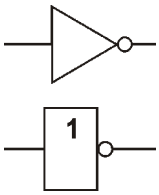
$\overline{A}$

" NON A "

igazságtáblázat:

bemenet	*kimenet*
A	$\overline{A}$
0	1
1	0

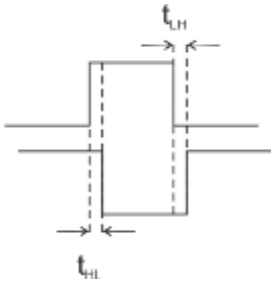
rajzjelek:



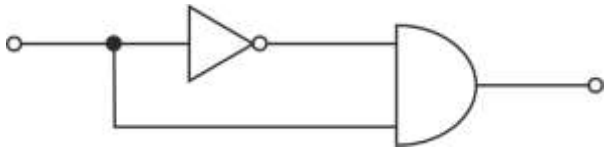
Amikor a logikai műveleteket analitikusan végezzük el, kimarad egy igen fontos gyakorlati tény, ami a műveletet megvalósító kapu(k) felépítő elektronikájából következik: a logikai kapukra kapcsolt logikai jel nem azonnal, hanem késleltetve, azaz egy igen rövid időtartam elteltével jut el csak a kimenetre.

A *késleltetési idő* a bemenő jel logikai állapotának megváltozása és a kimenő jel logikai állapotának megváltozása között eltelt idő. Mivel az 0 ↗ 1 és a 1 ↘ 0 logikai állapot ugrásokhoz más és más késleltetési idő tartozik (és legtöbbször nem tudjuk előre megmondani a felfutó és a lefutó élek számát) ... célszerű az átlagos késleltetési idő-vel számolni. Ennek számítási elve:

$$t_{delay} = \frac{t_{HL} + t_{HL}}{2}$$

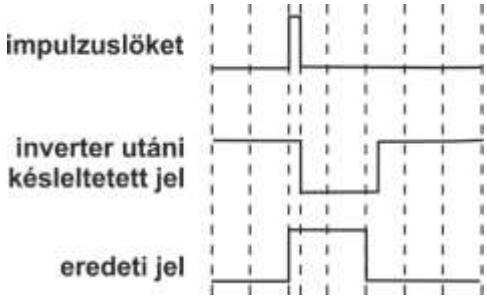


Figyeljük meg az alábbi logikai kapcsolás:



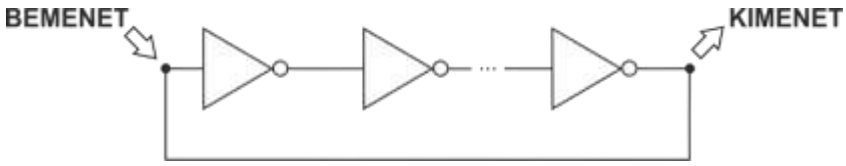
Ha nem vesszük figyelembe az *INVERTER* késleltetési idejét (ami elvileg *ns* nagyságrendű), úgy tűnik, hogy a bemutatott logikai áramkör esetén az *ÉS* kapu kimenetén mindig logikai 0 lenne!

A valóságban, az *INVERTER* okozta késleltetés miatt, az *ÉS* kapu bemenetére jutó két jel között időbeli eltolódás lesz! Ez egy impulzuslöketet fog eredményezni a kimeneten és a szekvenciális logikai hálózatok tanulmányozásánál bír gyakorlati fontossággal.



4. A GYAKORLATI MUNKA MENETE:

(a) készítsünk egy olyan logikai kapcsolást, ahol *páratlan* számú invertert kapcsolunk sorba és az utolsó kapu kimenetét visszakötjük az első bemenetére:



(b) működőképes-e egy ilyen logikai hálózat? – indokoljuk meg röviden válaszunk (magyarázzuk el miért nem vagy igen)!

(c) demonstráljuk érveinket az *NI Multisim 14.2* software-csomag segítségével elkészített szimulációval

(d) demonstráljuk érveinket a rendelkezésünkre álló kísérleti eszközökkel és eszközparkkal

(e) próbáljunk nevet találni ennek a kapcsolásnak!

(f) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

Figyelem!

- próbáljuk megbecsülni egy inverter késleltetési idejét
- az eszközparki lehetőségek függvényében, dolgozzunk több páratlan számú inverterrel (3, 5, 7, 9 stb.) és nézzük meg milyen hatással van az inverterek száma a megfigyelt eseményekre
- vizsgáljuk meg, hogy mi történik, ha az inverterek száma páros!

14. MULTIPLEXELÉS ÉS DEMULTIMPLEXELÉS

1. A GYAKORLAT CÉLJA:

A kombinációs logikai hálózatok kapcsolástechnikai megvalósítása NÉV, NAND és NOR funkcionálisan teljes rendszerekben, illetve a működés tesztelése *Deeds (Digital Circuit Simulator)* software és/vagy a *Leybold Didactic* munkapanel és eszközpark segítségével.

2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator) programcsomag

3. ELMÉLETI ALAPOK RÖVIDEN:

A logikai hálózat olyan, több be- és kimenettel rendelkező komplexebb kapcsolás, amely egy bonyolultabb logikai feladatot old meg.

Köztudott, hogy a kombinációs hálózatok kimenetei csak a bemenetek pillanatnyi állapotaitól, illetve azok kombinációitól függnek, egy későbbi logikai eredmény majd csak kizárólag a következő bemeneti értékkombinációktól fog függeni és semmi mástól.

A kombinációs logikai hálózatok kapuáramkörökből épülnek fel, a bemenetekre érkező logikai jelek hatására a kimeneten az eredmény azonnal megjelenik (az esetleges késleltetés csak a kapuáramkörök működési sebességéből adódik) és ennek az eredménynek nincs semmi köze a hálózat által előzőleg produkált eredményekhez.

A kombinációs logikai hálozatok tervezése és megvalósítása viszonylag egyszerű, a bonyolultsági fok egyenesen arányos az adott logikai függvény összetettségével. Hátrányként megemlítendő, hogy ez a hálózat típus nem alkalmas a logikai értékek tárolására és későbbi feldolgozására.

A kombinációs hálózatokkal megoldható logikai feladatok között vannak olyanok is, amelyek nem egyediek, hanem egy olyan igen jól meghatározott műveletként jelentkeznek, amely más feladatokhoz is szükséges lehet. Ezért ezeket a logikai hálozatokat *kombinációs funkcionális áramkör*öknek nevezzük. Mivel a szakmai gyakorlatban igen elterjedtek, akár integrált kivitelben is gyártják őket. A leggyakrabban előforduló funkcionális áramkörök közül megemlíthetők az aritmetikai (összeadó, kivonó, szorzó, összehasonlító – komparátor) és az adat átvivő (nyaláboló, nyalábosztó, kódoló, dekódoló és a kódátalakító) áramkörök.

Gyakran felmerül az igény, hogy különböző jelek állapotát egymás után végig vizsgáljuk egyazon kimeneten, vagy több különböző helyről érkező jelek közül válasszunk ki egyet és azt a kimenetre továbbítsuk.

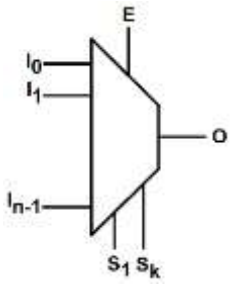
A *multiplexer* (MUX, MX vagy nagyon ritkán MULDEX) vagy *nyalábolóó* egy olyan adatátviteli funkcionális áramkör, amely két vagy több bejövő jelet egy kimenő jellé egyesít úgy, hogy azok később egyértelműen szétválaszthatók (az egyesítés vagy nyalábolás jól meghatározott, úgynevezett *címző*, *kiválasztó* vagy *szelektáló* jelek hatására történik).

Ha n a bejövő jelek száma, akkor a kiválasztó jelek k számára igaz, hogy:

$$k = log_2(n)$$

és a multiplexer szakmai gyakorlatban is bevált megnevezése "*1 az n-ből*".

A multiplexer rajzjele:



ahol: $I_0 \dots I_{n-1} =$ az adatbemenetek (n darab)

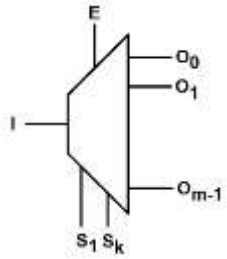
$O =$ az adatkimenet

$S_1 \dots S_k =$ a kiválasztó (címző/szelektáló) bemenetek (k darab)

$E =$ engedélyező bemenet (logikai 1-re vagy logikai 0-ra aktív, annak függvényében, hogy milyen kapukat használunk a kivitelezésre)

Természetesen a vételi oldalon szükség van egy olyan egységre, amely elvégzi a visszaalakítás műveletét. Ez lesz a *demultiplexer* (DEMUX, DMX) vagy *nyalábosztó*. A szakmai gyakorlatban bevált megnevezés "*m-ből az 1*" lesz, ha a kimenetek száma m.

A demultiplexer rajzjele:



ahol: $I =$ az adatbemenet

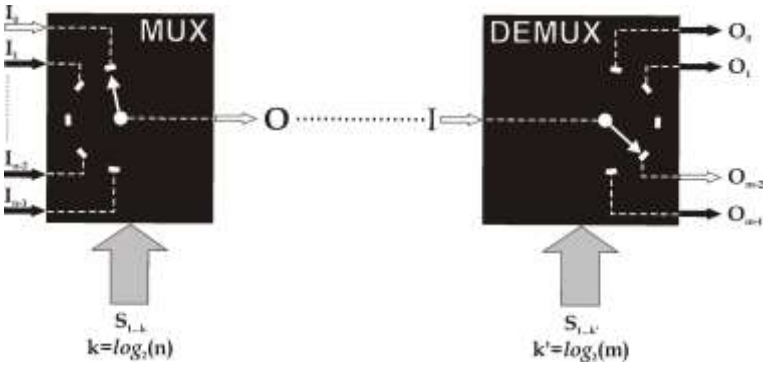
$O_0 \dots O_{n-1} =$ az adatkimenetek (m darab)

$S_1 \dots S_{k'} =$ a kiválasztó (címző/szelektáló) bemenetek (k' darab)

$E =$ engedélyező bemenet (logikai 1-re vagy logikai 0-ra aktív, annak függvényében, hogy milyen kapukat használunk a kivitelezésre)

A multiplexerek és demultiplexerek elvi működését kitűnően szemlélteti egy vasúti pályaudvar forgalmi szabályozása.

Tételezzük fel, hogy az n vágánnyal rendelkező A vasútállomást csak egyetlen egy sínpár köti össze a szomszédos B pályaudvarral, ahol mondjuk m érkezési vágány található. Az A pályaudvarban állomásozó vonatokat egy jól meghatározott logika* szerint választjuk ki és engedjük a kimenő vonalra, ahonnan egy másik jól meghatározott logika** alapján érkezhetnek be a B állomás megfelelő vágányra:



Példák a jól meghatározott szelektáló logikákra 2 adatbemenet vagy 2 adatkimenet esetére (mindkét helyzetben csak $log_2(2) = 1$ kiválasztójel szükséges!):

* az $S = 0$ -ra kiengedjük az I_0 adatot, $S = 1$ -re pedig az I_1 adatot

** az $S = 0$ -ra az I bejövő adatot beengedjük az O_0 -ra, $S = 1$ -re pedig az O_1 -re

4. A GYAKORLATI MUNKA MENETE:

(a) tervezzük meg és valósítsuk meg az "1 a 2-ből", "1 a 4-ből" és "1 a 8-ból" multiplexereket, illetve a "2-ből az 1", "4-ből az 1" és "8-ból az 1" demultiplexereket

(b) határozzuk meg a megfelelő logikai függvények algebrai alakját minden esetben

(c) tervezzük meg NÉV, NAND és/vagy NOR funkcionálisan teljes rendszerben a függvények kifejezését megvalósító logikai kapcsolásokat

(d) készítsük el a logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével

(e) a rendelkezésünkre álló eszközparkokkal készítsük el a logikai kapcsolások működőképes logikai áramköreit

(f) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

15. ARITMETIKAI ÁRAMKÖRÖK

1. A GYAKORLAT CÉLJA:

Az aritmetikai kombinációs funkcionális logikai hálózatok kapcsolástechnikai megvalósítása NÉV, NAND és NOR funkcionálisan teljes rendszerekben, illetve a működés tesztelése *Deeds (Digital Circuit Simulator)* software és/vagy a *Leybold Didactic* munkapanel és eszközpark segítségével.

2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator) programcsomag

3. ELMÉLETI ALAPOK RÖVIDEN:

A logikai hálózat olyan, több be- és kimenettel rendelkező komplexebb kapcsolás, amely egy bonyolultabb logikai feladatot old meg.

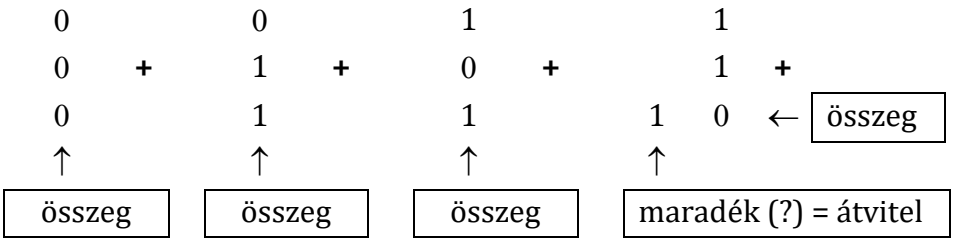
Egyik sajátos fajtája a kombinációs logikai hálózat. Egy ilyen hálózat kimenetén a logikai érték (eredmény) kizárólag csak a bemenetre érkező független logikai változók pillanatnyi logikai érték-kombinációinak eredménye. A kombinációs logikai hálózatok kapuáramkörökből épülnek fel, a bemenetekre érkező logikai jelek hatására a kimeneten az eredmény azonnal megjelenik (az esetleges késleltetés csak a kapuáramkörök működési sebességéből adódik) és ennek az eredménynek nincs semmi köze a hálózat által előzőleg produkált eredményekhez.

A kombinációs logikai hálozatok tervezése és megvalósítása viszonylag egyszerű, a bonyolultsági fok egyenesen arányos az adott logikai függvény összetettségével. Hátrányként megemlítendő, hogy ez a hálózat típus nem alkalmas a logikai értékek tárolására és későbbi feldolgozására.

A kombinációs hálózatokkal megoldható logikai feladatok között vannak olyanok is, amelyek nem egyediek, hanem egy olyan igen jól meghatározott műveletként jelentkeznek, amely más feladatokhoz is szükséges lehet. Ezért ezeket a logikai hálozatokat *kombinációs funkcionális áramkör*öknek nevezzük. Mivel a szakmai gyakorlatban igen elterjedtek, akár integrált kivitelben is gyártják őket. A leggyakrabban előforduló funkcionális áramkörök közül megemlíthetők az aritmetikai (összeadó, kivonó, szorzó, összehasonlító – komparátor) és az adat átvivő (nyalaboló, nyalabosztó, kódoló, dekódoló és a kódátalakító) áramkörök.

Azokat a digitális áramköröket, amelyek számtani műveletek elvégzésére alkalmasak *aritmetikai áramkörök*nek nevezzük. Az aritmetikai áramkörök bemenetét képező számokat bináris kódban szokták megadni és természetesen az eredmény is ugyanabban a kódban adódik.

Megfigyelhető, hogy két egybites szám összeadása okozhat "fejtörést" mivel maradék jelenhet meg, amit nem szeretnénk elveszíteni:



A fenti műveleteknek megfeleltethető igazságtáblázat:

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$\Rightarrow$ $C = A \cdot B$ (maradék)

$\Downarrow$

$S = A \oplus B$ (összeg)

Az igazságtáblázat a legegyszerűbb aritmetikai áramkörhöz vezet el, ezt *félösszeadó*nak nevezzük. Ez két operandus összeadására, az eredmény (összeg) és az átvitel (maradék) képzésére alkalmas, de nem tudja figyelembe venni az előző helyértékről származó átvitelt – ezért csak a legkisebb helyértéken alkalmazható! Amikor figyelembe kell venni az előző helyértékről származó átvitelt is úgynevezett *teljes összeadó*t kell használni. Ez egy hárombemenetű áramkör lesz (a két összeadandó bit és az előzőhelyértéken keletkezett átvitel), igazságtáblázata pedig a következő módon alakul:

C_{in}	A	B	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$\Rightarrow$ $((A \oplus B) \cdot C_{in}) + (A \cdot B)$ (maradék)

$\Uparrow$ az előző átvitel

$\Downarrow$

$(A \oplus B) \oplus C_{in}$ (összeg)

Egy másik fontos aritmetikai áramkör a *digitális komparátor*. Ennek feladata az, hogy két n bites abszolútértékes ábrázolású szám között jelezze a viszonyt (>, <, =). Egy komparátornak jellegzetesen 2 n bites adatbemenete ($A_{n-1} \dots A_0$ és $B_{n-1} \dots B_0$) van, amihez 3 kimenet társul (mindenik viszonynak egy-egy), valamint a 3 komparációs relációnak megfeleltetett bemenetek, amelyek a további láncszerű (kaszkádos) tovább-bővítést, összekapcsolást teszik lehetővé.

Az 1 bites összehasonlító két 1 bites kifejezést hasonlít össze. A feladatnak megfelelő igazságtáblázat:

A	B	A = B	A > B	A < B
0	**0**	**1**	0	0
0	**1**	0	0	**1**
1	**0**	0	**1**	0
1	**1**	**1**	0	0

amelyből már könnyedén kiolvashatók a 3 összehasonlítási viszonyoknak megfeleltethető logikai függvények.

Több bites kifejezések esetén az azonos helyi értékkel rendelkező biteket szoktuk összehasonlítani.

4. A GYAKORLATI MUNKA MENETE:

(a) tervezzük meg NÉV, NAND és/vagy NOR funkcionálisan teljes rendszerben a félösszeadó és a teljes összeadó függvények kifejezését megvalósító logikai kapcsolásokat
(b) tervezzünk egy négybites összeadót a megfelelő számú teljes összeadó láncba fűzésével
(c) az 1 bit logikai összehasonlítására szolgáló logikai komparátor igazságtáblázata alapján, tervezzük meg NÉV, NAND és NOR funkcionálisan teljes rendszerekben az összehasonlítást megvalósító kombinációs logikai hálózatot
(d) tervezzük meg NÉV, NAND és/vagy NOR funkcionálisan teljes rendszerben a 2 és a 4 bites szavak logikai összehasonlítására szolgáló logikai kapcsolást
(e) készítsük el a logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével
(f) a rendelkezésünkre álló eszközparkokkal készítsük el a logikai kapcsolások működőképes logikai áramköreit
(g) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

16. KÓDÁTALAKÍTÓK

1. A GYAKORLAT CÉLJA:

A kódátalakító kombinációs funkcionális logikai hálózatok kapcsolástechnikai megvalósítása NÉV, NAND és NOR funkcionálisan teljes rendszerekben, illetve a működés tesztelése *Deeds (Digital Circuit Simulator)* software és/vagy a *Leybold Didactic* munkapanel és eszközpark segítségével.

2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator) programcsomag

3. ELMÉLETI ALAPOK RÖVIDEN:

A *kód* egy információ, adott szabályok szerinti szimbólumokkal történő kifejezése. Az a folyamatot, amelynek során két szimbólumrendszert egymásnak rendelünk *kódolás*nak nevezzük. Annak ellenére, hogy a kód "megfejtését" (*dekódolás*t vagy *visszakódolás*t) a kódolási folyamat fordítottjának tekintjük, a két logikai művelet között nincs lényeges különbség, a dekódolás is egyfajta kódolásnak tekinthető.

A kódoló vagy kódátalakító olyan funkcionális egységnek fogható fel, amely a bementére érkező, bizonyos kódrendszerbeli információt a szabályoknak megfelelően egy másik kódrendszerbeli, de a bemenetivel azonos információt hordozó kódszavakká alakítja.

Talán az egyik legegyszerűbb példa decimális számrendszerben felírt számok binárissá alakítása és annak fordított folyamata, a visszaalakítás:

Decimális számrendszer:													**Kettes számrendszer:**			
X_0	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9				Y_3	Y_2	Y_1	Y_0
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**				$\mathbf{2^3}$	$\mathbf{2^2}$	$\mathbf{2^1}$	$\mathbf{2^0}$
1	0	0	0	0	0	0	0	0	0	=	**0**	⇔	0	0	0	0
0	1	0	0	0	0	0	0	0	0	=	**1**	⇔	0	0	0	1
0	0	1	0	0	0	0	0	0	0	=	**2**	⇔	0	0	1	0
0	0	0	1	0	0	0	0	0	0	=	**3**	⇔	0	0	1	1
0	0	0	0	1	0	0	0	0	0	=	**4**	⇔	0	1	0	0
0	0	0	0	0	1	0	0	0	0	=	**5**	⇔	0	1	0	1
0	0	0	0	0	0	1	0	0	0	=	**6**	⇔	0	1	1	0
0	0	0	0	0	0	0	1	0	0	=	**7**	⇔	0	1	1	1
0	0	0	0	0	0	0	0	1	0	=	**8**	⇔	1	0	0	0
0	0	0	0	0	0	0	0	0	1	=	**9**	⇔	1	0	0	1

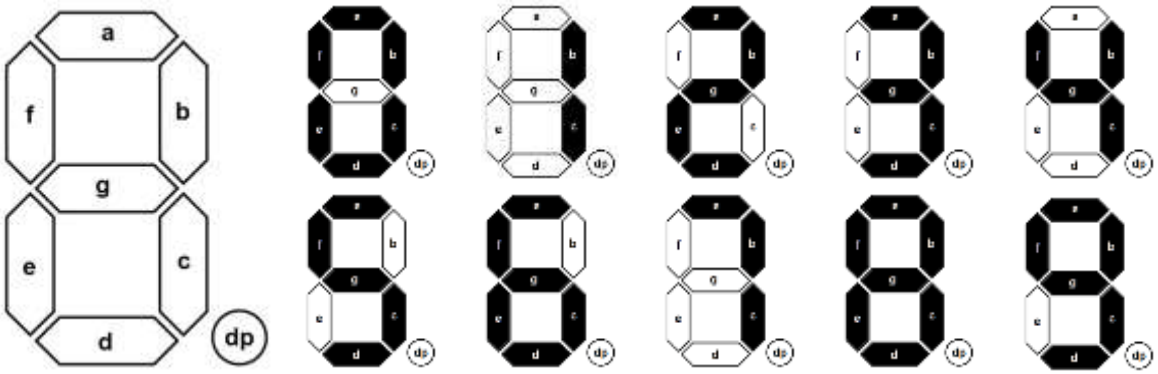
Egy másik igen fontos kódrendszer a *Gray kód*, amelynek legjellemzőbb tulajdonsága, hogy két sorrendszomszédos kódszó mindig csak egyetlen kód-elemnél térhet el egymástól. Legelterjedtebb felhasználási területe a forgó mozgást végző mechanikai rendszerek elfordulási szögének meghatározása.

Egy 4-bit-es Gray kód 16 kimeneti állapotot írhat le, az így meghatározható elfordulási szög pedig 360°/16 = 22.5° lesz.

Decimális		**Bináris**					**Gray**			
0	⇔	0	0	0	0	⇔	0	0	0	0
1	⇔	0	0	0	1	⇔	0	0	0	1
2	⇔	0	0	1	0	⇔	0	0	1	1
3	⇔	0	0	1	1	⇔	0	0	1	0
4	⇔	0	1	0	0	⇔	0	1	1	0
5	⇔	0	1	0	1	⇔	0	1	1	1
6	⇔	0	1	1	0	⇔	0	1	0	1
7	⇔	0	1	1	1	⇔	0	1	0	0
8	⇔	1	0	0	0	⇔	1	1	0	0
9	⇔	1	0	0	1	⇔	1	1	0	1

A digitális elektronika (és nemcsak) egyik legelterjedtebb kijelző egysége a 7 szegmenses világítódiódás kijelző.

A 7 fénykibocsátó diódát (*LED*-et) magába foglaló egység a 0 – 9 értéktartományban található tizedes számrendszer egységeit jeleníti meg (olykor még létezik egy nyolcadik LED is, amely a tizedesvesszőnek felel meg).

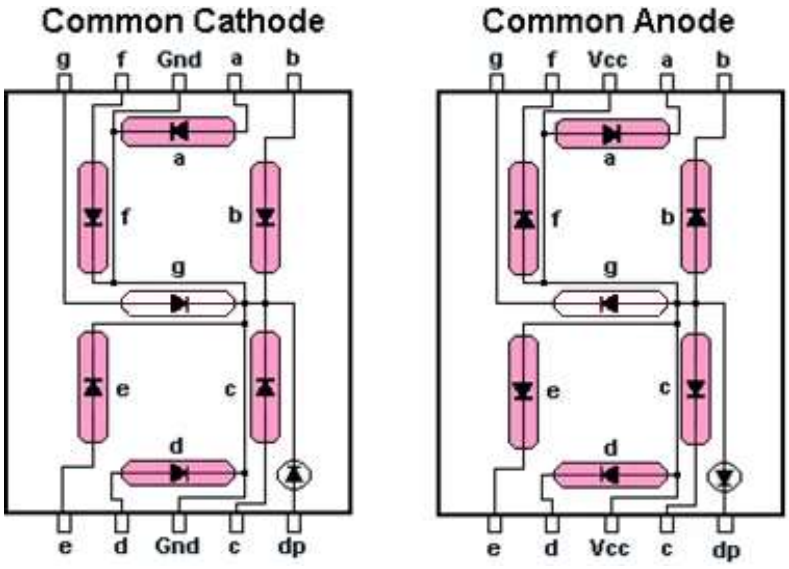


Jól látható, hogy melyik tízes számrendszerű számnak, milyen világító LED kombináció felel meg. A köztes bináris kódok a könnyebb átalakítást segítik elő.

A 7 szegmens kijelző egységek kétfajta kivitelezésben érhetők el, annak függvényében, hogy a LED-ek melyik kivezetése képezi a közös pontot. Ennek értelmében megkülönböztethető a közös anódú (angolul *common anode*) és a és közös katódú (angolul *common cathode*) kialakítás.

	bináris kód:					**közös katódú kivitel:**							**közös anódú kivitel:**						
	D	**C**	**B**	**A**		**a**	**b**	**c**	**d**	**e**	**f**	**g**	**a**	**b**	**c**	**d**	**e**	**f**	**g**
0	0	0	0	0	⇔	1	1	1	1	1	1	0	0	0	0	0	0	0	1
1	0	0	0	1	⇔	0	1	1	0	0	0	0	1	0	0	1	1	1	1
2	0	0	1	0	⇔	1	1	0	1	1	0	1	0	0	1	0	0	1	0
3	0	0	1	1	⇔	1	1	1	1	0	0	1	0	0	0	0	1	1	0
4	0	1	0	0	⇔	0	1	1	0	0	1	1	1	0	0	1	1	0	0
5	0	1	0	1	⇔	1	0	1	1	0	1	1	0	1	0	0	1	0	0
6	0	1	1	0	⇔	1	0	1	1	1	1	1	0	1	0	0	0	0	0
7	0	1	1	1	⇔	1	1	1	0	0	0	0	0	0	0	1	1	1	1
8	1	0	0	0	⇔	1	1	1	1	1	1	1	0	0	0	0	0	0	0
9	1	0	0	1	⇔	1	1	1	1	0	1	1	0	0	0	0	1	0	0

A két kivitelezés szemléltetése és a 0 számjegy kijelenítése:



Forrás: http://microcontroller-project.com

4. A GYAKORLATI MUNKA MENETE:

(a) határozzuk meg a decimális számrendszerben felírt számok binárissá alakítását leíró logikai függvény algebrai alakját (binárisan kifejezett decimális számjegyek)

(b) határozzuk meg a bináris számrendszerben felírt számok decimálissá alakítását leíró logikai függvény algebrai alakját (decimálisan kódolt bináris számjegyek)

(c) határozzuk meg a Gray kódot megvalósító logikai függvény algebrai alakját

(d) határozzuk meg a közös anódú 7 szegmenses kijelző vezérlését megvalósító logikai függvényt
(e) határozzuk meg a közös katódú 7 szegmenses kijelző vezérlését megvalósító logikai függvényt
(f) tervezzük meg NÉV, NAND és/vagy NOR funkcionálisan teljes rendszerben a függvények kifejezését megvalósító logikai kapcsolásokat
(g) készítsük el a logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével
(h) a rendelkezésünkre álló eszközparkokkal készítsük el a logikai kapcsolások működőképes logikai áramköreit
(i) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

17. ELEMI TÁROLÓK FELÉPÍTÉSE ÉS MŰKÖDÉSE

1. A GYAKORLAT CÉLJA:

Az elemi tárolók kapcsolástechnikai megvalósítása NÉV, NAND és NOR funkcionálisan teljes rendszerekben, illetve a működés tesztelése *Deeds (Digital Circuit Simulator)* software és/vagy a *Leybold Didactic* munkapanel és eszközpark segítségével.

2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator) programcsomag

3. ELMÉLETI ALAPOK RÖVIDEN:

A logikai hálózat olyan, több be- és kimenettel rendelkező komplexebb kapcsolás, amely egy bonyolultabb logikai feladatot old meg. Egyik sajátos fajtája a sorrendi vagy szekvenciális logikai hálózat, melynek működési elve számottevően különbözik a kombinációs logikai hálózatokétól.

Köztudott, hogy a kombinációs hálózatok kimenetei csak a bemenetek pillanatnyi állapotaitól, illetve azok kombinációitól függnek, egy későbbi logikai eredmény majd csak kizárólag a következő bemeneti értékkombinációktól fog függeni és semmi mástól. Ezzel szemben nagyon sok gyakorlati esetben az adott digitális áramkör működése függhet az előzményektől, az időtől, és az események bekövetkezési sorrendjétől. Ezek a plusz befolyások szükségessé teszik, hogy a hálózat valamilyen formában emlékezzen előéletére és azt valahogyan visszajuttassa a beérkező független logikai változókhoz, hogy majd együtt szolgáltassák az új kimeneti eredmény.

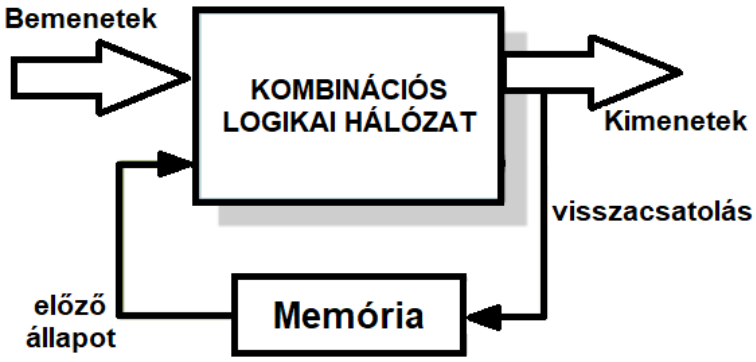
Azokat a logikai hálózatok, amelyek kimeneti jelei nemcsak a pillanatnyi bemeneti jelkombinációtól függenek, hanem attól is, hogy korábban milyen bemeneti jelkombinációk voltak (azaz mi volt a kimeneten előzőleg) sorrendi logikai hálózatoknak nevezzük.

Tehát az ilyen hálózat kimenetén a logikai érték (eredmény) nemcsak a bemenetre érkező független logikai változók pillanatnyi logikai érték-kombinációinak eredménye, hanem annak is, hogy milyen eredmény volt azelőtt a kimeneten.

Ha egy kombinációs logikai hálózat kimenetét visszavezetjük a bemenetre valamilyen tároló (memoráló) egység segítségével, a mindenkori kimenet aktuális állapota a bemenő és visszacsatolt jelek hatására bármelyik pillanatban megváltozhat - ezért ezt a szekvenciális logikai hálózatot aszinkron sorrendi hálózat-nak nevezzük.

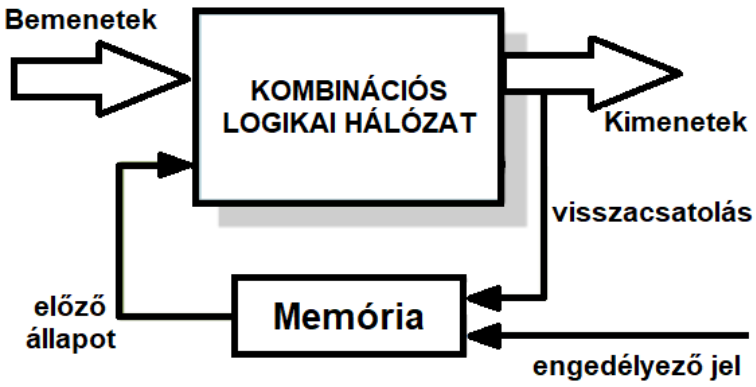
Ezek a hálózatok esemény vezéreltnek tekinthetők, hiszen a kimeneti állapotváltás a bemeneti változók változási eseményeit követik bármely időpillanatban.

Az aszinkron sorrendi logikai hálózatok vázlatos elvi felépítését az alábbi tömbvázlat szemlélteti:



Ha a visszacsatolt jelt csak bizonyos feltételek teljesítésekor, vagy csak jól meghatározott időközönként engedjük vissza a bemenetekre (például valamilyen külső órajel hatására, mindegyik periódusban csak egyetlen egyszer) akkor a kimenetek állapotváltozásai is csak ilyen ütemben történhetnek majd meg, vagyis a hálózatunk működését a külső vezérlés fogja diktálni (vele kell „szinkronizálni" vagy „szinkronban lenni"). Ezért ezt a szekvenciális logikai hálózat típust időzített vagy szinkron sorrendi hálózat-nak nevezzük.

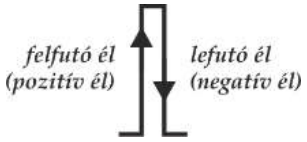
Az szinkron sorrendi logikai hálózatok vázlatos elvi felépítését az alábbi tömbvázlat szemlélteti:



A későbbiekben látni fogjuk, hogy az engedélyező jel lehet egy adott logikai szint (időben állandó logikai érték, azaz 0 vagy 1), illetve egy órajel (angolul *clock*, rövidítve CLK) amelyet általában egy kristályoszcillátor által szolgáltatott négyszögjel biztosít. Ez utóbbi esetében igen fontos az, hogy mikor történjen a vezérlés: az alacsony/magas („felfutó") vagy magas/alacsony („lefutó") szintek közötti változás (billenés) hatására. Ezért szokták még az ilyen sorrendi logikai hálózatokat szint vezérelt, illetve élvezérelt működésűnek nevezni.

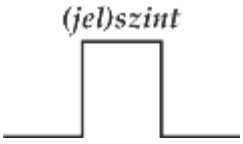
Az alábbiakban összehasonlítás céljából illusztráljuk a szintvezérlést és az élvezérlést, és egyidejűleg szemléltetjük a jelölés módjukat is:

a pozitív élvezérlés jelölése

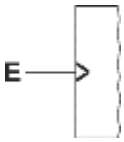


az órajel és vezérlő élei

a negatív élvezérlés jelölése



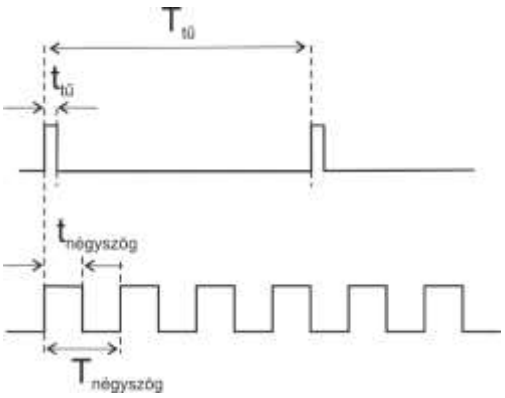
a vezérlő szint



a szintvezérlés jelölése

A digitális elektronikában használatos *impulzus* fogalma egy olyan feszültséget jelent, amelynek értéke két nyugalmi állapot között ugrásszerűen változik, vagyis két szélsőérték (0 V és 5 V) között változtatja az amplitúdóját.

A vezérléshez szükséges órajel egy nagyon rövid ideig tartó impulzus, tűimpulzusnak vagy impulzuslöketnek is nevezik – összehasonlítva egy négyszögimpulzussal, jól láthatók jellegzetességei:

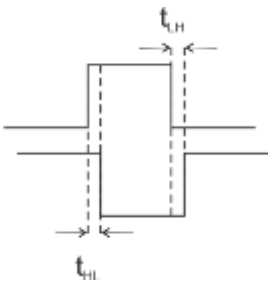


A fenti ábrán szemléltetett esetben a leglátványosabb különböztetőjel a *kitöltési tényező* (azaz az impulzusidő és a periódusidő aránya) amely a négyszögjelnél 50 %, míg a tűimpulzusnál jóval kisebb – meg kell jegyezni, hogy a mérvadó MINDIG az impulzusidő! Az órajelek impulzusideje igen kicsi, de attól még lehet 50 %-os a kitöltési tényező.

A szinkron hálózatok vezéreltetése miatt feltevődik a kérdés hogyan állítható elő egy szintvezérlő jel? Elvileg egyszerűen, a kitöltési tényező növelésével és a frekvencia egyidejű csökkentésével. Órajelt is elég egyszerűen állíthatunk elő, felhasználva azt a tényt, hogy a logikai kapukra kapcsolt jel késleltetve jut a kimenetre (ezért is lehet például gyűrűs oszcillátort készíteni páratlan számú invertert használva).

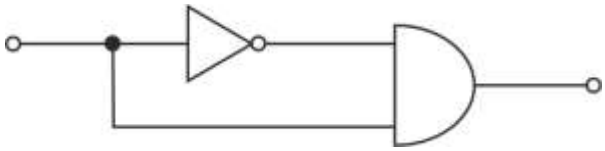
A késleltetési idő a bemenő jelszint megváltozása és a kimenő jel logikai állapotának megváltozása között eltelt idő – mivel legtöbbször az 1 ↘ 0 és a 0 ↗ 1 ugrásokhoz más más késleltetési idő tartozik, és nem lehet előre megmondani a felfutó és a lefutó élek számát ... célszerű az *átlagos késleltetési idő*-vel számolni:

$$t_{delay} = \frac{t_{HL} + t_{HL}}{2}$$



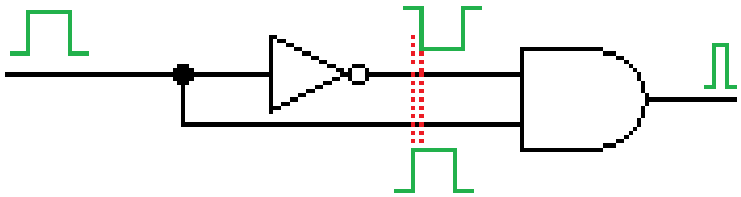
Mindezeket figyelembe véve, a továbbiakban tárgyaljuk vázlatosan az élvezérléshez szükséges órajel-impulzus és él detektálásának lehetőségeit.

Tekintsük a következő egyszerű kombinációs hálózatot:

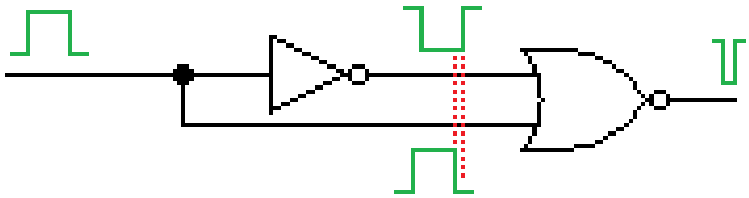


Ha nem vesszük figyelembe az inverter késleltetési idejét (ami elvileg *ns* nagyságrendű), úgy tűnik, hogy az ÉS kapu kimenetén mindig logikai 0 lenne.

A valóságban, a késleltetés miatt az ÉS bemenetére jutó két jel egy, a felfutó pozitív élre érzékeny impulzuslöketet fog eredményezni a kimeneten:



Ha az ÉS kaput egy NOR kapuval helyettesítjük, akkor gyakorlatilag az eredmény egy lefutó negatív élre érzékeny impulzuslöket lesz:



Az 1 bit-nyi információ (logikai 0 vagy logikai 1) elraktározását az úgynevezett elemi tárolók végzik, a több bit (byte) elraktározására már több tervszerűen összekapcsolt elemi tárolóra van szükség – ezek együttesen egy regisztert alkotnak.

A digitális elektronikában kétfajta elemi tárolóval találkozunk, ezek a flip-flop és a latch (a közöttük levő különbség az átbillenés kiváltásának módjában rejlik).

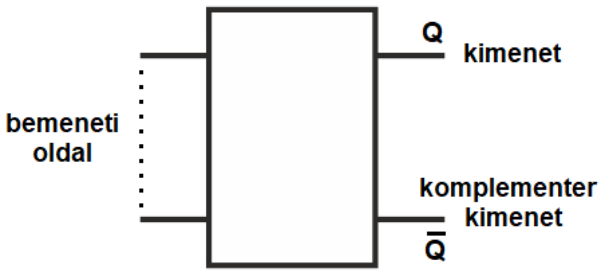
A billenőkörökről tudjuk, hogy azok olyan pozitívan visszacsatolt digitális áramkörök, amelyek kimeneti feszültsége nem folytonosan változik, hanem csak két jól meghatározott értéket vehet fel – a lehetséges két érték közötti átváltás (billenés) megtörténhet spontán módon vagy kiváltható egy külső, vezérlő jel hatására – a szakirodalom három különböző fajta billenőkört ismert (bistabil, monostabil, astabil). Ezek közül a bistabil billenőkör az, amelynek két stabil állapota van, a közöttük váló átváltás (billenés) egy külső jel hatására következik be. Ez a kapcsolás képes megvalósítani az 1 bit információ tárolását.

A flip-flopok esetében a billenés élvezérelt (*edge triggered*) lesz, azaz egy órajel felfutó (pozitív) vagy lefutó (negatív) élén történik, vagyis ahányszor a vezérlő órajel (clock, CLK) logikai 1-ről logikai 0-ra ugrik (vagy fordítva) billenés történik. Ezzel szemben a latch esetében a billenés szintvezérelt (*level triggered*), azaz akkor történik, amikor a vezérlő jel a logikai 1 szinten tartózkodik és úgy is marad amíg az azon a logikai szinten tartózkodik.

Minden tárolóra három üzemmód jellemző, ezek a SET (a logikai 1 beírása), a RESET (a logikai 0 beírása, azaz a logikai 1 kitörlése) és a STORE (tárolás, memorálás - az előző állapot megtartása – legyen az akár 0, akár 1).

A tárolók egyik igen érdekes működési sajátossága az, hogy a kimenet komplementere is mindig rendelkezésünkre áll.

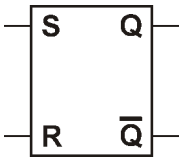
A tároló rajzjele egy, a rövidebb oldalán álló téglalap, amelynek a baloldali részén vannak a bemenetek (a logikai változók és az órajel vagy vezérlőszint számára).



A téglalap belsejébe, a megfelelő bement mellé fel szokták tűntetni az adott szerepnek megfelelő betűjel. Ezek a betűk szokták meghatározni az adott tároló típusát és biztosítanak annak megkülönböztető elnevezést.
Ugyanakkor létezhetnek olyan tárolók is amelyek a felsorolt bemenetek mellett aszinkron bemenetekkel is rendelkeznek. Ezeket a „rövid” oldalakon szokás feltűntetni.

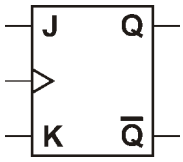
A szakirodalom négy elemi tárolót tart számon, ezek rajzjele és működésüket meghatározó igazságtáblázatukat az alábbiakban szemléltetjük (az aszinkron SR kivételével mindegyik tároló órajel vezérelt):

SR



S	R	Q_{n+1}
0	0	Q_n*
1	0	1
0	1	0
1	1	**

JK

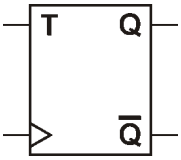


J	K	Q_{n+1}
0	0	Q_n
1	0	1
0	1	0
1	1	$\overline{Q_n}$

* kezdeti állapot

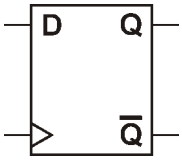
** nem megengedett állapot

T



T	Q_{n+1}
0	Q_n
1	$\overline{Q_n}$

D



D	Q_{n+1}
0	0
1	1

4. A GYAKORLATI MUNKA MENETE:

(a) a megadott igazságtáblázatok alapján határozzuk meg az elemi tárolók működését definiáló logikai függvényeket

(b) tervezzük meg NÉV, NAND és/vagy NOR funkcionálisan teljes rendszerben az elemi tárolók kifejezését megvalósító logikai kapcsolásokat

(c) készítsük el a logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével

(d) a rendelkezésünkre álló eszközparkkal készítsük el a logikai kapcsolások működőképes logikai áramköreit

(e) tervezzük meg JK tárolók segítségével a T és a D tárolókat

(f) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

18. AZ ÁTLÁTSZÓSÁG ÉS A MESTER-SZOLGA ELV

1. A GYAKORLAT CÉLJA:

Az átlátszóság fogalmának és a mester-szolga elvnek az elméleti, illetve gyakorlati megértése.

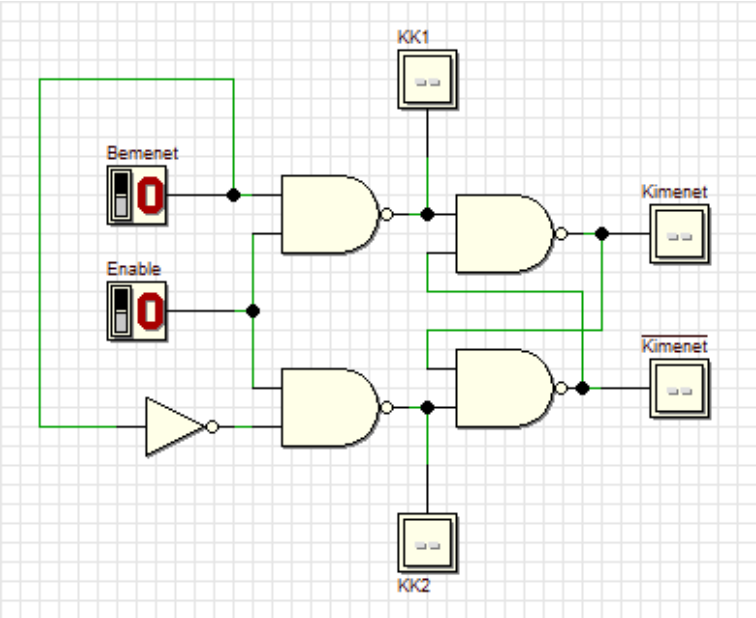
2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator) programcsomag

3. ELMÉLETI ALAPOK RÖVIDEN:

A bistabil áramkörök működésének lefolyása alatt létezhet egy olyan időintervallum amikor az áramkör ÁTLÁTSZÓ, azaz a bemenet(ek) és a kimenet(ek) között közvetlen kapcsolat van – vagyis a bemenet(ek)re adott változások azonnal kijutnak a kimenet(ek)re!

Jó szemléltető példa erre az alábbi ábrán látható logikai kapcsolás, ahol a két köztes kijelző (KK1 és KK2 csak az áttekinthetőséget, a könnyebb nyomon követes célját szolgálják.

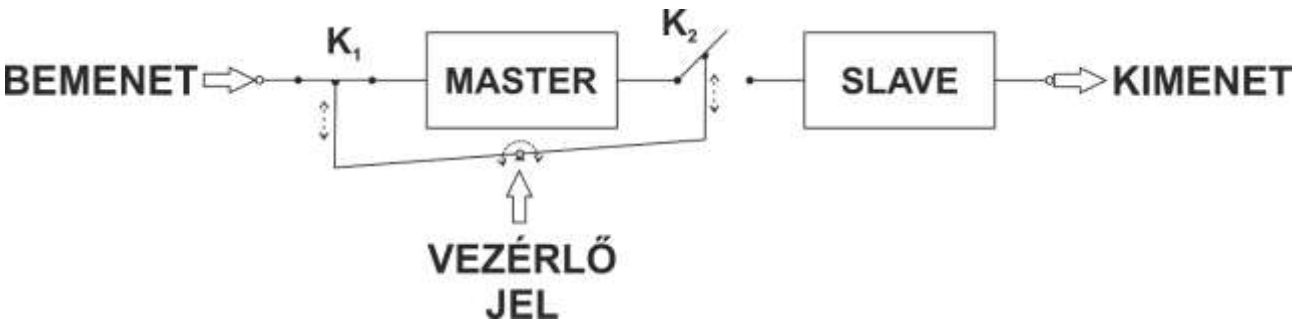


Azonnal megfigyelhető, hogy az áramkör csak akkor működik, ha ezt az engedélyező jel lehetővé teszi, azaz ENABLE = logikai 1. Ahogy a bementi jel logikai értéket vált … az azonnal megtörténik a kimeneten is … ez az úgynevezett átlátszóság, ami a logikai feladatok elvégzése során nem mindig jó dolog – így felmerül az igény olyan tároló előállítására, amely a beírási folyamat alatt nem lesz átlátszó.

A feladatot olyan kétfokozatú tárolók oldják meg, amelyek egymással hierarchikus vezérlési viszonyban állnak. Ezt a kölcsönös kapcsolatot tükrözve a szakirodalom a MESTER–SZOLGA elv elnevezést használja, ami az angol MASTER–SLAVE viszonylag sikertelen fordításának, magyarításának az eredménye. Megjegyzendő az, hogy leginkább az ÚR–RABSZOLGA

megnevezés tükrözné a legjobban hierarchikus vezérlési viszonyban levő, két altárolóból álló kapcsolás működési elvét.

A mester-szolga elv tömbvázlatos szemléltetése az alábbi ábra segítségével történik:



A „vezérlő jel" ellenfázisban működteti a két (K_1 és K_2) kapcsolót, azaz amikor egyik nyit … a másik zárni fog.

Ha K_1 zár a bemeneten levő jel beíródik a MASTER elnevezésű áramkörbe, de mivel a K_2 kapcsoló nyitva van, nem tud a SLAVE áramkörbe is átjutni és azután esetleg a kimenetre is. Tehát a kimenet nem tudja azonnal követni a bemenet változását.

Amikor a vezérlő jel hatására K_1 kinyit már nem lehet többet a MASTER-be beírni, de mivel K_2 bezáródik, a MASTER-be beírt információ átíródik a SLAVE-be és közvetlenül a kimenetre jut. Ha a vezérlő jel miatt K_2 nyit és K_1 zár, a SLAVE-be beírt jel a kimeneten lesz mindaddig amíg a vezérlő jel lehetővé nem teszi a MASTER „új" jelének megérkezését.

A szakmai gyakorlat az ellenvezérlést INVERTER-rel, a kapcsolást pedig ÉS kapukkal oldja meg.

4. A GYAKORLATI MUNKA MENETE:

(a) vizsgáljuk meg az átlátszóságot szemléltető kapcsolás működését a Deeds (Digital Circuit Simulator) software segítségével
(b) tervezzük meg NÉV, NAND és/vagy NOR funkcionálisan teljes rendszerben az *SR mester-szolga* és a *JK mester-szolga* elveket szemléltető logikai kapcsolásokat
(c) készítsük el a logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével
(d) a rendelkezésünkre álló eszközparkkal készítsük el a logikai kapcsolások működőképes logikai áramköreit
(e) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

19. DIGITÁLIS FREKVENCIAOSZTÁS

1. A GYAKORLAT CÉLJA:

A digitális frekvenciaosztás fogalmának elméleti, illetve gyakorlati megértése.

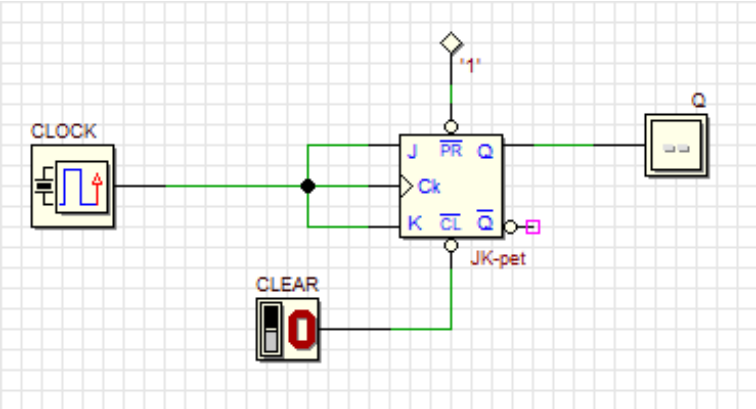
2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator) programcsomag

3. ELMÉLETI ALAPOK RÖVIDEN:

Könnyen megfigyelhető, hogy egy élvezérelt bistabil áramkör esetén a kimeneten megjelenő jel frekvenciája a bemeneti jel frekvenciájának a fele. Ez azért van, mert a kimenetre jutó állapotváltozás mindig csak a felfutó vagy lefutó élnél történik meg.

A frekvenciafelezés elvét az alábbi ábra szemlélteti:



4. A GYAKORLATI MUNKA MENETE:

(a) bizonyítsuk be, hogy csak 2 hatványaival oszthatjuk a bemeneti jel frekvenciáját, a hatványkitevő pedig az alkalmazott bistabilok számával lesz egyenlő

(b) tervezzük meg NÉV, NAND és/vagy NOR funkcionálisan teljes rendszerben a frekvencianegyedelést és a frekvencianyolcadolást szemléltető logikai kapcsolásokat

(c) készítsük el a logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével

(d) a rendelkezésünkre álló eszközparkkal készítsük el a logikai kapcsolások működőképes logikai áramköreit

(e) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

20. REGISZTEREK FELÉPÍTÉSE ÉS MŰKÖDÉSE

1. A GYAKORLAT CÉLJA:

A több bit tárolását ellátó funkcionális sorrendi logikai hálózatok kapcsolástechnikai megvalósítása és tanulmányozása *Deeds (Digital Circuit Simulator)* software és/vagy a *Leybold Didactic* munkapanel és eszközpark segítségével.

2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator) programcsomag

3. ELMÉLETI ALAPOK RÖVIDEN:

Az 1 bit-nyi információ (logikai 0 vagy logikai 1) elraktározását az úgynevezett elemi tárolók végzik, a több bit (byte) elraktározására már több tervszerűen összekapcsolt elemi tárolóra van szükség – ezek együttesen egy *regiszter*-t alkotnak.

A regisztert közös vezérléssel rendelkező összekapcsolt bistabil billenőáramkörök alkotják, az összekapcsolt bistabilok száma meghatározza a beírható bitek számát vagy a regiszter kapacitását – arra használjuk őket, hogy a beírt információt eltárolják, illetve annak relatív helyzetét a tárolórendszerben megváltoztassák

Azaz információbevitel és az információkiolvasás szempontjából négy fajta regisztert különböztetünk meg, ezek:

a) SISO (*Serial Input Serial Output*), azaz soros beírás és soros kiolvasás – ebben az esetben a regiszter első és utolsó flip-flopjához lehet hozzáférni, vagyis az elsőbe beírunk és az utolsóból fogunk kiolvasni, de közben az információt el kell vinni (léptetni) a beíró flip-floptól a kiolvasó flip-flopig, ezért ezek az úgynevezett léptető regiszter-ek

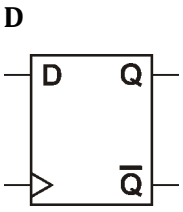
b) SIPO (*Serial Input Paralel Output*), azaz soros beírás és párhuzamos kiolvasás – természetesen a SIPO rendszer esetén az információ kiolvasható minden tároló után is, ez párhuzamos kimenetelt feltételez

c) PISO (*Paralel Input Serial Output*), azaz párhuzamos beírás és soros kiolvasás – ez is egyfajta léptető regiszter, tulajdonképpen a tárolandó információ egyszerre érkezik párhuzamosan, de a kiolvasás az utolsó tároló után történik, a bitek egymásutáni léptetésével

d) PIPO (*Paralel Input Paralel Output*), azaz párhuzamos beírás és párhuzamos kiolvasás – párhuzamos beírásnál és kiolvasásnál az információt a regiszter minden flip-flopjába egyszerre írják be és egyszerre olvassák ki, ezért ezek csak tárolásra alkalmasak és átmeneti tároló- vagy közbenső tároló (puffer) regisztereknek nevezzük őket

A regiszterek lekönnyebben a D típusú elemi tárolók segítségével valósíthatók meg. Egy ilyen elemi tároló legfontosabb jellemzője az, hogy a bemenetére érkező bitet az órajel engedélyezése után változatlanul a kimenetre juttatja.

Rajzjelét és igazságtáblázatát az alábbiakban példáztatjuk:



D	Q_{n+1}
0	0
1	1

4. A GYAKORLATI MUNKA MENETE:

(a) tervezzük meg D tárolók segítségével a SISO, SIPO, PISO és PIPO típusú regisztereket
(b) készítsük el a megtervezett logikai kapcsolások működőképes szimulációját a Deeds (Digital Circuit Simulator) software segítségével
(c) a rendelkezésünkre álló eszközparkkal készítsük el a logikai kapcsolások működőképes logikai áramköreit
(d) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

21. SZÁMLÁLÓK FELÉPÍTÉSE ÉS MŰKÖDÉSE

1. A GYAKORLAT CÉLJA:

A számlálás feladatát ellátó funkcionális sorrendi logikai hálózatok kapcsolástechnikai megvalósítása és tanulmányozása *Deeds (Digital Circuit Simulator)* software és/vagy a *Leybold Didactic* munkapanel és eszközpark segítségével.

2. SZÜKSÉGES ESZKÖZTÁR:

- Leybold Didactic munkapanel, eszközpark és csatlakozók
- Deeds (Digital Circuit Simulator) programcsomag

3. ELMÉLETI ALAPOK RÖVIDEN:

Elméletileg minden olyan digitális áramkör, amelynél valamilyen egyértelmű kapcsolat van a beérkező órajel impulzusok száma és a kimeneti változók állapotváltozása között, számlálónak tekinthető.

A bevált szakmai gyakorlat szerint egy számláló feladata az, hogy az órajel bemenetre (CLK) érkező impulzusokat megszámolja, és a következő órajel impulzus beérkezéséig a számolás eredményét eltárolja és kijelezze. Ezért szükségszerűen egy számlálónak legalább annyi egymástól különböző állapottal kell rendelkezni, mint amennyi a megszámlálandó órajelimpulzusok számának maximális értéke.

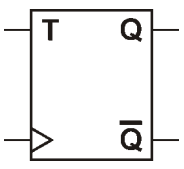
Miután ezt az értéket elérte, a további beérkező impulzusok hatására a számlálás alapállapotból kezdődik újra – tehát egy ciklikus működésre kell számítani (ha a maximálisnál kisebb értékig kívánunk számolni akkor a megfelelő érték elérésekor törlés/reset funkcióval kell visszaugrani az alapállapothoz).

Azt az órajel számot, ami után a kezdeti vagy bármely pillanatnyi állapot ismétlődik a számláló *moduló* értékének nevezzük. Ez az érték fogja majd megmutatni azt, hogy az adott számláló hány különféle állapotot tud felvenni.

Mivel számlálás közben tárolni (memorálni) is kell, a számlálók legtipikusabb alapáramköre a T típusú elemi tároló lesz.

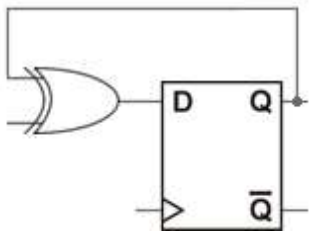
A T elemi tároló rajzjelét és igazságtáblázatát az alábbiakban példáztatjuk:

T



T	Q_{n+1}
0	Q_n
1	$\overline{Q_n}$

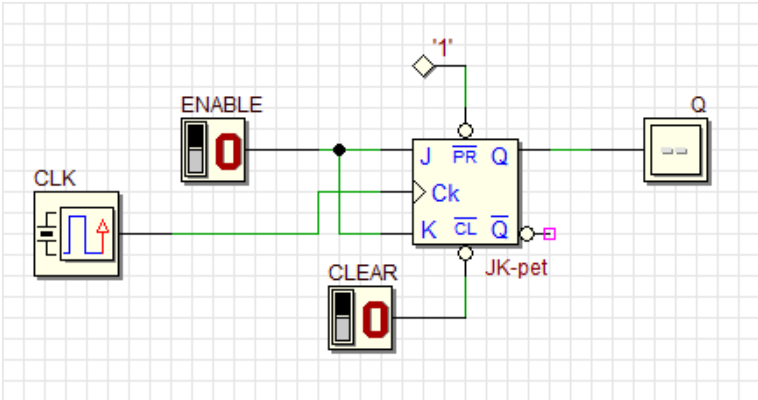
Természetesen egy T tároló kialakítható JK tárolóból is (lásd a megfelelő laboratóriumi gyakorlatot) vagy D tárolóból, XOR kapu segítségével az alábbi módon:



Mivel minden bit tárolására egy-egy tárolóra van szükség, az *n*-bites számlálóhoz *n* tárolóra van szükségünk és az 2^n állapotot vehet fel (tehát modulusa 2^n lesz), illetve 2^n -1 értékig számol.

Például a 2 bites számláló, 2 T típusú tárolóval, $2^2 = 4$ állapotot vehet fel (modulusa 4) és 0-tól 3-ig számol, illetve jelez ki.

A JK-ból képzett T típusú tárolóval megvalósított 1 bites számláló kapcsolását az alábbi ábra szemlélteti



Az engedélyező jel esetén, minden órajelre ellenkező állapotba billen á, de mivel csak két megkülönböztethető állapota van, ezért csupán két órajelimpulzus egyértelmű megszámlálására lesz alkalmas, a CLEAR lenullázza a kezdeti állapotot.

A szakmai gyakorlat igen sok számláló típust tart számon, a rendelkezésünkre álló osztályozási kritériumok is sokfélék lehetnek.

A rendszerezett áttekinthetőség kedvéért az alábbiakban egy összehasonlító táblázatban szemléltetjük a legfontosabb kritériumokat, illetve a számláló típusokat.

Kritérium	**Típus**	**Jellemző**
belső felépítés	aszinkron	az órajelek csak a legkisebb helyi értéket képviselő tárolót vezérlik, a többi az órajelt egymástól fogja kapni és a billenések nem azonos időpontban fognak történni
	szinkron	az órajelek a számláló összes tárolóját egyszerre vezérlik, ezért a billenések azonos időpontban fognak történni
kijelzési kód	bináris	az eltárolt impulzusszám kódolása bináris számrendszerben történik
	BCD	a tárolt impulzusszám kódolása (általában a 8421 kódú) decimális számrendszerben történik
számlálási irány vagy sorrend	előre (felfele)	az eltárolt impulzusszám növekvő sorrendű
	hátra (lefele)	az eltárolt impulzusszám csökkenő sorrendű
	reverzibilis (fel-le)	a számlálás iránya megfordítható

Mivel a számlálás egy igen fontos logikai feladat, a számlálók is funkcionális típusú sorrendi logikai áramköröknek számítanak. Integrált áramköri formában hátrafele számláló áramköröket nem szoktak előállítani – ezt a technológiát kizárólag az előre és a reverzibilis módon működő számlálók kivitelezésére használják.

4. A GYAKORLATI MUNKA MENETE:

(a) tervezzük meg T tárolók segítségével 4 bites aszinkron és szinkron előre- illetve hátra számláló logikai áramköröket
(b) készítsük el a megtervezett logikai kapcsolások működőképes szimulációját a Deeds (*Digital Circuit Simulator*) software segítségével
(c) a rendelkezésünkre álló eszközparkkal készítsük el a logikai kapcsolások működőképes logikai áramköreit
(d) készítsünk részletes és kimerítő kiértékelő jelentést munkánkról

22. VÁLOGATOTT KÖNYVÉSZET ÉS WEBOGRÁFIA

Könyvek, tankönyvek, jegyzetek

Buzás G. - *Bevezetés a digitális elektronikába*, Erdélyi Tankönyvtanács, Ábel kiadó 2008
Buzás G., Simon A. - *Az analóg és digitális elektronika alapjai*, Ábel kiadó, 2001
Hegyesi L., Kovács Cs. - *Digitális Elektronika*, General Press kiadó 2010
Kovács Cs. - *A digitális elektronika alapjai*, General Press kiadó 2007
R. J. Tocci - *Digital Systems: Principles and Applications*, Prentice Hall, 2010
S. D. Anghel - *Bazele electronicii analogice şi digitale*, Presa Universitară Clujeană, 2007
Simon A., Tunyagi A. R.: *Elektronika Laboratóriumi Praktikum 1.*, Presa Universitară Clujeană, 2021
Zombori B. - *Digitális elektronika*, Tankönyvmester kiadó 2006

Honlapok – Elméleti információk:

https://tudasbazis.sulinet.hu/hu/szakkepzes/elektronika-elektrotechnika/digitalis-alaparamkorok
http://centroszet.hu/tananyag/digit_aramkorok/index.html
https://www.inf.u-szeged.hu/~tanacs/oktatas/szamitogep_architekturak_gd/index.html
http://www.inf.u-szeged.hu/~mingesz/tananyagok/digitalis-technika-tananyagok/
https://www.digitalelectronicsdeeds.com/learningmaterials/labtopics.html
https://learn.electronics-course.com/
https://www.nutsvolts.com/magazine/article/understanding_digital_buffer_gate_and_ic_circuits_part_1
https://www.nutsvolts.com/magazine/article/understanding_digital_buffer_gate_and_ic_circuits_part_2
https://www.nutsvolts.com/magazine/article/understanding_digital_buffer_gate_and_ic_circuits_part_3
https://www.nutsvolts.com/magazine/article/understanding_digital_buffer_gate_and_ic_circuits_part_4
https://www.nutsvolts.com/magazine/article/understanding_digital_buffer_gate_and_ic_circuits_part_5

Honlapok – Karnaugh táblás számítógépes programok:

https://circuitverse.org/simulator
http://k-map.sourceforge.net/
https://www.mathematik.uni-marburg.de/~thormae/lectures/ti1/code/karnaughmap/
https://www.charlie-coleman.com/experiments/kmap/
http://www.32x8.com/index.html
https://www.softpedia.com/get/Others/Home-Education/KarnaughMap.shtml
https://www.softpedia.com/get/Others/Home-Education/Boole-Deusto.shtml
https://download.cnet.com/Karnaugh-NET/3000-2053_4-10517216.html
https://code.google.com/archive/p/bmin/

Honlapok – Minimalista logikai szimulátor:

https://sebastian.itch.io/digital-logic-sim

Honlapok – Digitális elektronika szimulációs programcsomag:

https://www.digitalelectronicsdeeds.com/index.html

Honlapok – Hasznos információk:

Elektronikai alkatelem adatlapok gyűjteménye (https://atom.ubbcluj.ro/alpar/datasheets/)
Elektronikus eszközök használati útmutatói (https://atom.ubbcluj.ro/alpar/manuals/)
Elektronikus kapcsolások online megrajzolása (https://www.circuit-diagram.org/editor/)

ISBN gen. 978-606-37-1254-8
ISBN vol. 2 978-606-37-1688-1