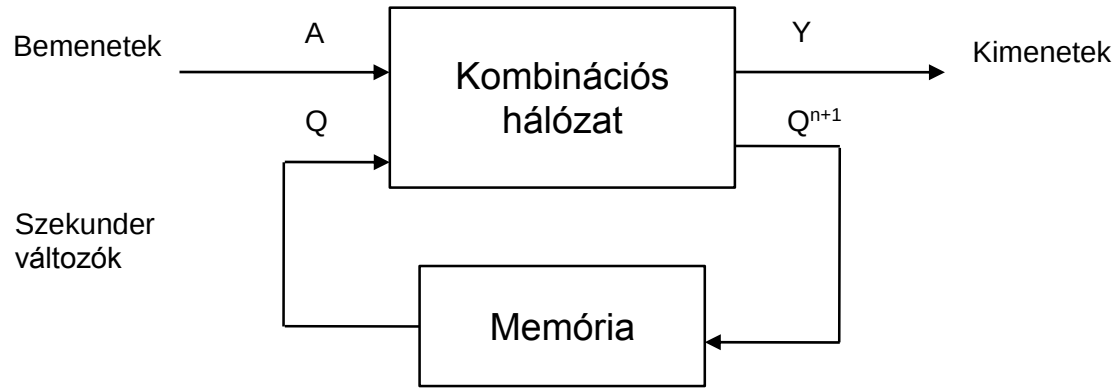
6. hét Szinkron hálózatok tervezése és viszgálata

6.1. Bevezetés

A szinkron sorrendi hálózatok kapcsán a korábbiakban leszögeztük, hogy a hálózat az alábbi módon épül fel:



1. ábra A sorrendi hálózat

Ebből a felépítésből az is következik, hogy a hálózat az alábbi függvények segítségével írható fel. A kimeneti kombinációt előállító leképezést az alábbi alakban is definiálhatjuk:

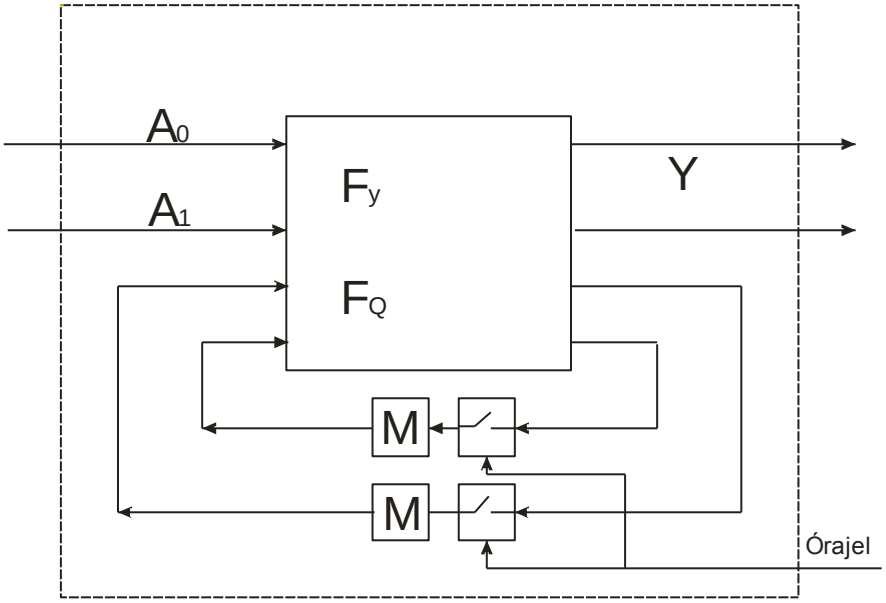
$$f_{\vec{y}} = (A, Q) \Rightarrow Y$$

Ekkor azonban a bemeneti változók csak látszólag nem befolyásolja mert az

$$f_{Q^{n+1}} = (A, Q) \Rightarrow Q^{n+1}$$

leképezés Q-t ezúttal is A-tól függően fogja előállítani.

Tekintsük az alábbi rendszert:



2. ábra A visszacsatoló ágak periodikus nyitásának modellje

A visszacsatoló ágakban jelképesen olyan kapcsolókat ábrázolunk, amelyek periodikusan ismétlődő négyszögimpulzusok (azaz az órajel) hatására létrehozzák, ill. megszüntetik a visszacsatolást. Az egyes kapcsolók után rajzolt M jelű elemekről tételezzük fel, hogy kimeneteiken azt az értéket jelenítik meg, amely a kapcsoló zárásainak pillanatában a bemenetükre jutott. Tételezzük fel továbbá azt is, hogy ezt a kimeneti értéket mindaddig fenntartják, amíg egy újabb kapcsolózárás be nem következik. Ezért az M jelű elemek kimeneti értéke a kapcsolók nyitásakor, vagyis a visszakapcsoló ágak megszakítása alatt nem változik. Az M elemek tehát memória tulajdonságúak is.

Az órajel logikai 1 szintjének időtartama alatt a visszacsatoló ágak zártak, 0 esetén pedig nyitottak.

Induljunk ki abból, hogy két órajel között a hálózat éppen nyugalomban van. Ekkor a visszacsatoló ágak nyitottak, s az éppen jelenlévő A és az előző óraimpulzus hatására a bemenetre jutó Q alakítja ki az Y-t és a Q^{n+1}- et. Ha ezek után megérkezett az óraimpulzus, akkor ennek hatására az éppen fennálló Q kombináció a bemenetre jut,az M jelű elemek közvetítésével. Az így kialakuló új Q kombináció az éppen aktuális A –val együtt új Y és Q kombinációt hoz létre. Ez mindaddig ismétlődik minden órajel hatására, amíg a rendszer stabil állapotú nem lesz. Ha a mindenkori A megváltoztatásával mindig megvárjuk a stabil állapot kialakulását, akkor a hálózat aszinkron jellegű lesz. Az előzőekhez képest azonban mégis van annyi különbség, hogy hogy az instabil állapotok

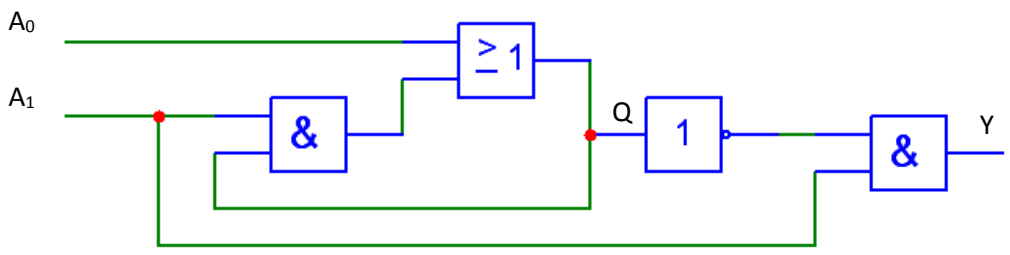
felbukkanása nem véletlenszerű, hanem mindig az órajel ütemezésében következnek be egymás után. Ezért már fennállási időtartamuk sem véletlenszerű, hanem pontosan az órajel periódusideje. Következésképpen a hálózat működési sebességét az órajel frekvenciája korlátozza, hiszen két óraimpulzus között a visszahatások meg vannak szüntetve. Az így működő hálózatokat ütemezett aszinkron hálózatoknak nevezzük. Ennek a hálózattípusnak nagy hátránya a sebességcsökkenés, előnye viszont, hogy az órajel periódusideje rögzíti az instabil állapotok fennállási idejét.

A hálózat tehát egy kombinációs hálózatból áll, melynek nem csak be és kimeneti értékait ismerjük, mint primer változókat, hanem a hálózatban a belső állapotok is fontosak (Q) , melyeket szekunder változóként kezelünk.. Korábban láttuk, miképpen írható le a hálózat állapottábla és állapotgráf segítségével, valamit azt is, hogy a vezérlési tábla megszerkesztésével hogyan tudjuk egy folyamat egymást követő lépéseit lekövetni. Az előző fejezetben volt egy példahálózatunk, melynek elemeit megtanultuk e két felírási móddal létrehozni.

6.1.1. Az állapottábla

Az állapottábla nem más, mint a sorrendi hálózatok működésének leírása táblázatos formában. A táblázatban fel kell tüntetnünk a bemeneti és kimeneti változókat, valamint a szekunder változók n-edik és n+1.- dik értékét. Két bemenet esetén még a belső állapot értékével, mint bemenettel kell számolnunk, valamint a kimenet (ek) mellett „belső kimenet tulajdonképpen ”a szekunder változók következő értéke is. A táblázatnak természetesen 2 bemenet és 1 belső változó esetén nyolc sora lesz (mert három független változónak számít).

pl.



3. ábra A példahálózat ki - és bemenetei

A **Q_{n+1}-gyel jelölve a belső változó aktuális, és Q-val az előző értékét**, logikai függvénykapcsolat írható fel a közbenső változóra és a kimenetre is.

$$Q_{n+1} = A_0 + A_1 Q$$

$$Y = A_1 \cdot \bar{Q}$$

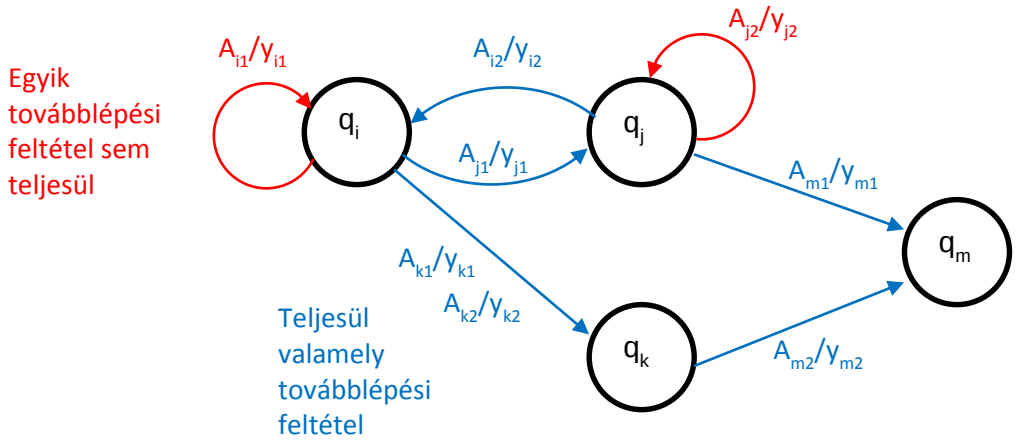
Q	A_0	A_1	Q_{n+1}	Y
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	1	1
1	0	0	0	0
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

4. ábra Állapottábla

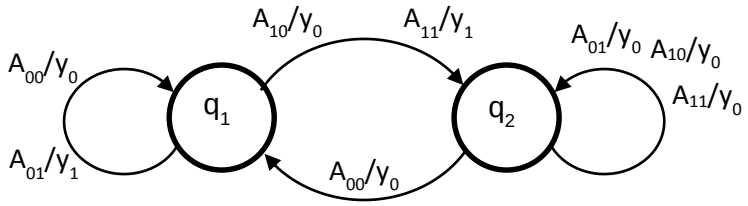
6.1.2. Az állapotgráf

Az állapotgráf a rendszer egy grafikus szemléltetési módja. A szemléltetés oly módon történik, hogy a sorrendi hálózat belső állapotait a gráf csomópontjai szemléltetik. (Az ábrán q-val jelölve, hogy a különböző belső állapotok megkülönböztethetők legyenek. A csomópontokat összekötő irányított élek (nyilak) az egyik állapotból a másikba történő átmenetet reprezentálják. Az éleken az átmenetet előidéző bemeneti A kombináció szerepel. Emellett az y kimeneti értékeket is gyakran fel szokás tüntetni

Ha nem teljesül semmilyen továbblépési feltétel, a rendszer marad az előző állapotban. Előfordul, hogy több feltétel is kielégítheti a továbblépés feltételét. Fontos, hogy egy állapotból visszafelé, egy előző állapotba is lehetséges állapotátmenet.

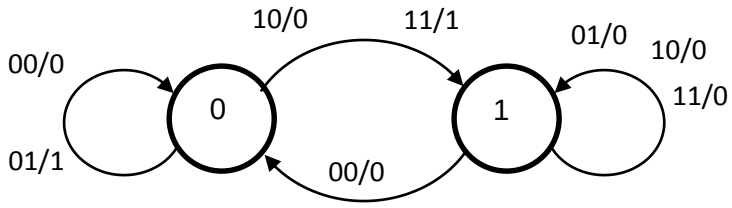


Mivel ennek a hálózatnak két belső állapota van, ezért gráfként két csomóponttal írható fel.



5. ábra A rendszer gráfja

Ha a megfelelő értékeket behelyettesítjük:



6. ábra A rendszer állapotgráfja

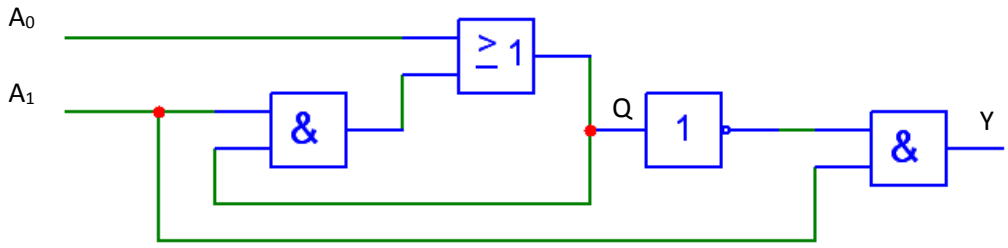
6.1.3. A vezérlési táblázat

A vezérlési táblázat az állapottábla célszerűen átalakított formája, ahol az oszlopok a bemenő jelek, a sorok pedig a késleltetés után előállt visszacsatolt jelek.

A cellákba a bemenő jel hatására keletkező Q_{n+1} jelet írjuk, majd a stabil állapotokat bekarikázzuk.

- Ahol $Q = Q_{n+1}$, ott nincs állapotváltozás a visszacsatoló hurokban.
- Ahol $Q \neq Q_{n+1}$, ott instabil állapot lép fel, , mert állapotváltozás zajlik a visszacsatoló hurokban. Tulajdonképpen ilyenkor a jel még nem ért át a késleltetőn.

6.1.3.1. Egy szekunder változós vezérlési tábla



7. ábra A példahálózat ki - és bemenetei

kiindulás: $Q = A_0 = A_1 = 0$

bemeneti szekvencia:

01,

11,

01,

00

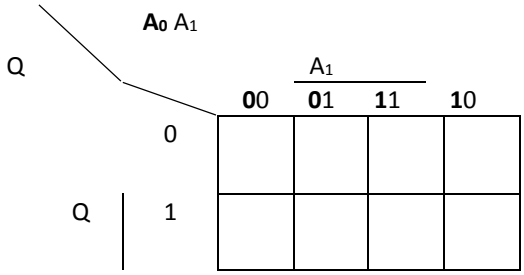
A rendszer állapottáblája a következő módon állítható elő:

i	Q	A_0	A_1	Q_{n+1}	Y
0	0	0	0	0	0
1	0	0	1	0	1
2	0	1	0	1	0
3	0	1	1	1	1
4	1	0	0	0	0
5	1	0	1	1	0
6	1	1	0	1	0
7	1	1	1	1	0

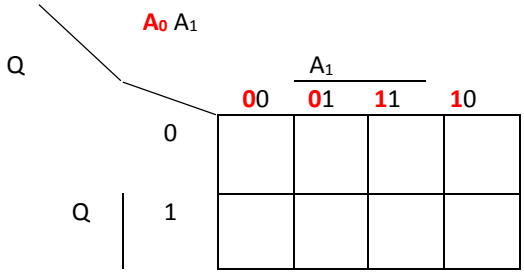
8. ábra Állapottábla

Ezt a korábbiakból már ismerjük, de ez alkalommal megszámoztuk a sorokat. Vegyük észre, hogy ez a számozás ismét nem önkényes: i értéke decimálisan pontosan annyi, amennyit Q és A_0 valamint A_1 változók binárisan adnak.

A rendszer működését le tudjuk írni az állapottábla alapján egy egyszerű táblával, melyet vezérlési táblázatnak nevezünk, mert a sorrendi hálózat működési elvét írja le, sorban lekövethetők rajta a hálózat egymást követő állapotai. A leírás hasonló a korábban tanult Karnaugh- tábla módszerhez, DE nem teljesen azonos vele!

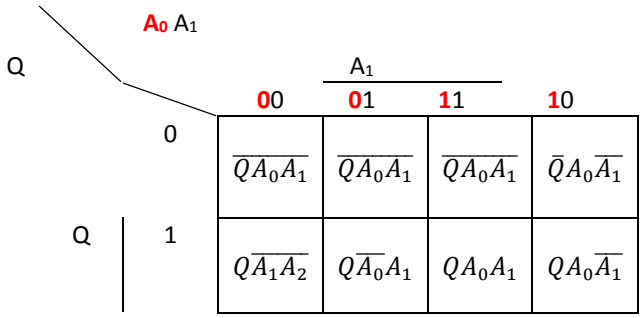


9. ábra Három változós vezérlési tábla

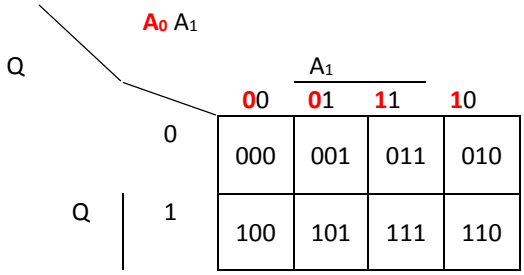


10. ábra Három változós (A_0 elemei kiemelve) vezérlési tábla

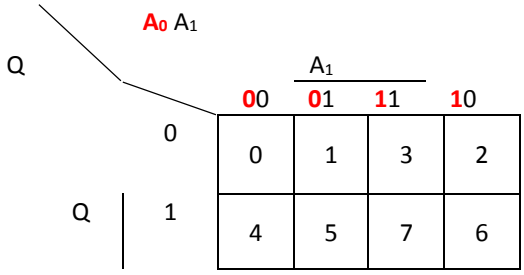
Természetesen a cellák ebben az esetben is feltölthetők a változók megfelelő értékeivel, akár betűkről, akár számokról van szó.



11. ábra Három változós (A_0 elemei kiemelve) vezérlési tábla értékeinek számítás



12. ábra Három változós (A_0 elemei kiemelve) vezérlési tábla bináris értékekkel



13. ábra Három változós (A_0 elemei kiemelve) vezérlési tábla cellaértékei

Természetesen a fenti hálózatot is felírhatjuk ilyen vezérlési táblába az alábbi módon:

- Az állapottáblából kiválasztjuk azokat a sorokat, ahol Q értéke 1 lesz a folyamat során.
- Ezt vezetjük be a vezérlési táblába.
- Meghatározzuk a stabil állapotokat. (Minden olyan állapot stabil, ahol $Q_n = Q_{n+1}$)
- Kiolvassuk a vezérlés folyamatát.

	Q	A_0	A_1	Q_{n+1}	Y
0	0	0	0	0	0
1	0	0	1	0	1
2	**0**	**1**	**0**	**1**	0
3	**0**	**1**	**1**	**1**	1
4	1	0	0	0	0
5	**1**	**0**	**1**	**1**	0
6	**1**	**1**	**0**	**1**	0
7	**1**	**1**	**1**	**1**	0

14. ábra Az állapottáblából kiválasztott értékek

Q \ A_0 A_1	00	01	11	10
0	0	0	1	1
1	0	1	1	1

15. ábra Az kitöltött vezérlési tábla

Q \ A_0 A_1	00	01	11	10
0	**0**	**0**	1	1
1	0	1	1	1

16. ábra A stabil állapotok (sötétvörös)

A stabil állapotokat az fogja jelenteni, ahol

$$Q_n = Q_{n+1}$$

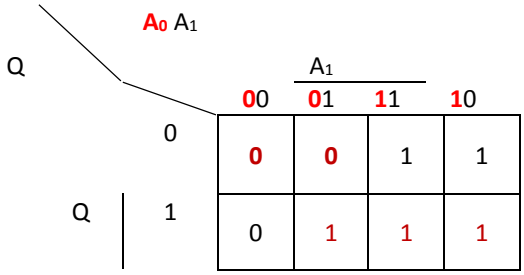
tehát adott esetben 0, 1, 5, 6, 7 értékeknél.

A rendszer működésének kiolvasása pedig az állapottábla figyelembevételével zajlik:

i	Q	A_0	A_1	Q_{n+1}	Y
0	0	0	0	0	0
1	0	0	1	0	1
2	0	1	0	1	0
3	0	1	1	1	1
4	1	0	0	0	0
5	1	0	1	1	0
6	1	1	0	1	0
7	1	1	1	1	0

17. ábra Az állapottábla

A kiolvasás során érdemes a táblát az állapottáblával együtt figyelni.



18. ábra Az állapottábla kiolvasása

A következőkben fontos azt is tudnunk, hogy hogyan épül fel a rendszer másik eleme: a memória. A rendszer memóriája: ún. tárolókkal oldható meg.

6.2. A tárolók definíciója

A tárolóegység, memóriaegység tehát tároló elemekből épül fel. A tároló feladata: információ tárolás. Fontos, hogy ***egy tároló elem 1 bit információt képes tárolni.***

A tároló alapelemek közül az egyik legfontosabb csoport az ún.

Kétállapotú (bistabil) billenő elemek (Flip-Flopok)

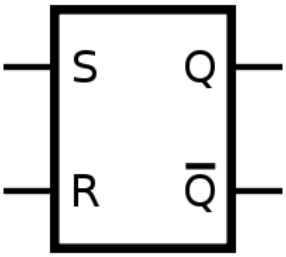
A digitális hálózati elemek közül flip-flopnak (bistabil multivibrátor) nevezik azokat, amelyek egyidejűleg tudják fogadni a következő bemenetet, és szolgáltatni az aktuális kimenetet, így egyszerű memóriaelemként is használhatóak. Vezérelhetőek több órajellel, egy órajel felfutó és lefutó élével, vagy logikai kapukkal. Előnyös tulajdonságuk, hogy két állapot közötti átmenetkor nem válnak átlátszóvá. Gyakran a kimeneteik negáltját is szolgáltatják. Mindaddig megtartják előző állapotukat míg külső jel ennek megváltoztatására nem kényszeríti.

A flip-flopoknak több fajtája létezik, így beszélhetünk:

- SR (Set-Reset) flip-flopról
- D (Data) flip-flopról
- DG (Data-Gate) flip-flopról
- T (toggle) flip-flopról
- JK flip-flopról

6.3. A tárolók fajtái

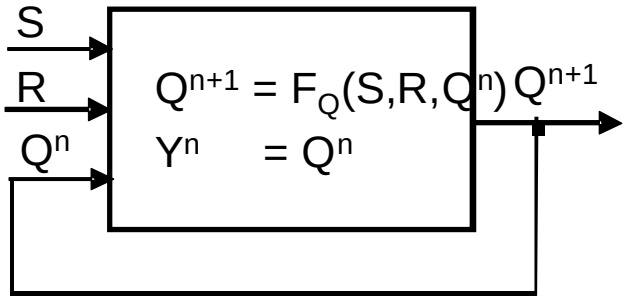
6.3.1. Az S-R flip-flop



19. ábra Az R -S flip- flop

Az S-R flip-flopnak egy beállító (Set), és egy törlő (Reset) bemenete van. Az egyik legegyszerűbb flip-flopnak tekinthető, bár alapvetően tároló. A két bemenet egyidejű felemelését tiltani szokták, mivel ez instabil állapotot idézne elő (ld. versenyhelyzet).

6.3.1.1. Az RS tároló modellje



20. ábra Az RS tároló felírása modellel

6.3.1.2. Az RS tárolót leíró függvény

A modell alapján a rendszer függvényét is fel tudjuk írni.

$$Q^{n+1} = S + \bar{R}Q^n$$

$$\bar{Q}^{n+1} = R + \bar{S}\bar{Q}^n$$

6.3.1.3. Az RS tároló állapottáblája

A rendszer felírását elvégezhetjük állapottábla segítségével. Így jól nyomon követhető a működés is.

- Az S(Set) bemenetre adott „1”-es a kimenetet „1”-be állítja
- Az R(Reset) bemenetre adott „1”-es a kimenetet „0”-ba állítja

R	S	Q^{n+1}
0	0	Q^n
0	1	1
1	0	0
1	1	X

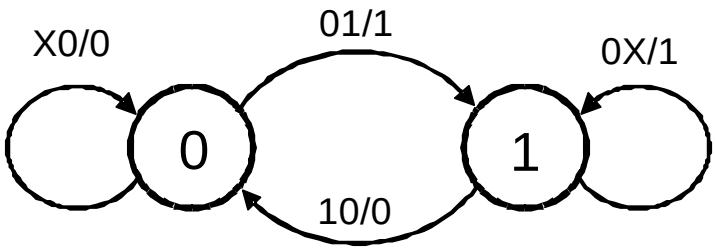
21. ábra A rendszer állapottáblája, hogy jól látszon a FLIP _ FLOP

Jól látszik a fenti állapottáblán a rendszer flip-flopozása is, mert így válik megfigyelhetővé, hogy vagy egyik, vagy másik bemenet lehet 1 . A tároló azon kívül, hogy memóriája van, kapcsolóként is működik.

R	S	Q^n	Q^{n+1}	
0	0	0	0	Változatlan
0	0	1	1	
0	1	0	1	Beírás
0	1	1	1	
1	0	0	0	Törlés
1	0	1	0	
1	1	0	X	Tiltott
1	1	1	X	

22. ábra Az RS tároló állapotáblája

Állapot gráffal történő felírás is elvégezhető természetesen, a már ismert szabályok felhasználásával.



23. ábra Az állapotgráf

6.3.1.4. Az RS tároló vezérlési táblája

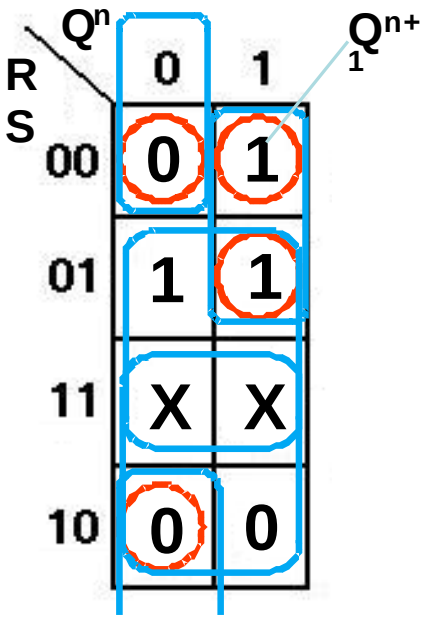
Ebben a rendszerben nincs versenyfutás vagy oszcilláció, tehát az aszinkron működés is stabilnak tekinthető. Vannak viszont érdektelen (Don't care állapotok).

Az állapottáblát Karnaugh-táblának tekintve, Qn+1-re elvégezve az összevonásokat az egyszerűsített logikai függvény:

$$Q^{n+1} = S + \bar{R}Q^n$$

$$\bar{Q}^{n+1} = R + \bar{S}\bar{Q}^n$$

	0	1
00		
01		
11		
10		



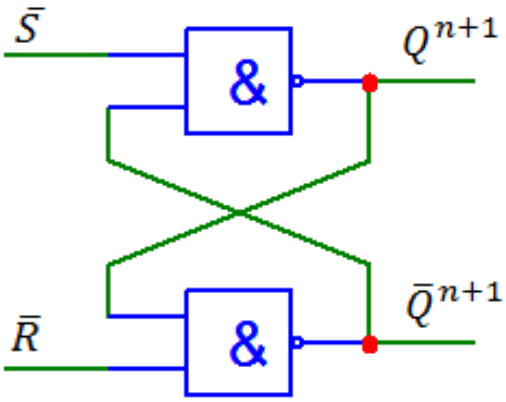
24. ábra Az RS tároló vezérlő táblája

6.3.1.5. Szinkron RS tároló (Filp-flop) megvalósítása

A Qn+1-et és Qn+1 -et megvalósító kombinációs hálózat logikai függvénye. A hálózat NAND kapus megvalósítása a következőképpen hozható létre. Itt fontos szempont, hogy így csak NAND kapukat kell használni.

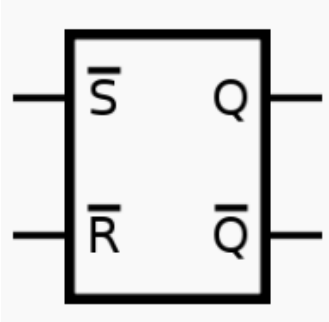
$$Q^{n+1} = S + \bar{R}Q^n = \overline{\bar{S} \cdot \overline{\bar{R}Q^n}}$$

$$\bar{Q}^{n+1} = R + \bar{S}\bar{Q}^n = \overline{\bar{R} \cdot \overline{\bar{S}\bar{Q}^n}}$$

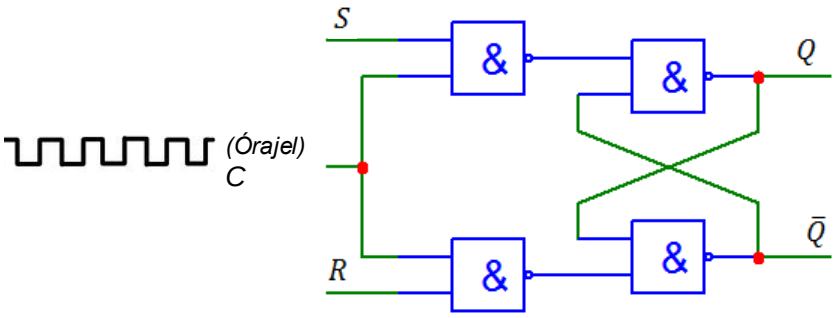


25. ábra Az RS tároló megvalósítása

A megvalósítás során a fentebb leírt két egyenletet kell figyelembe vennünk. Ugyanakkor az is fontos, hogy az R és S bemenetek hatása a szinkronjel (órajel) megérkezésekor érvényesüljön, ezért még két AND kapu felhasználásával a rendszerre rákötjük az órajelet is.



26. ábra A RS tároló NAND kapus megvalósítása

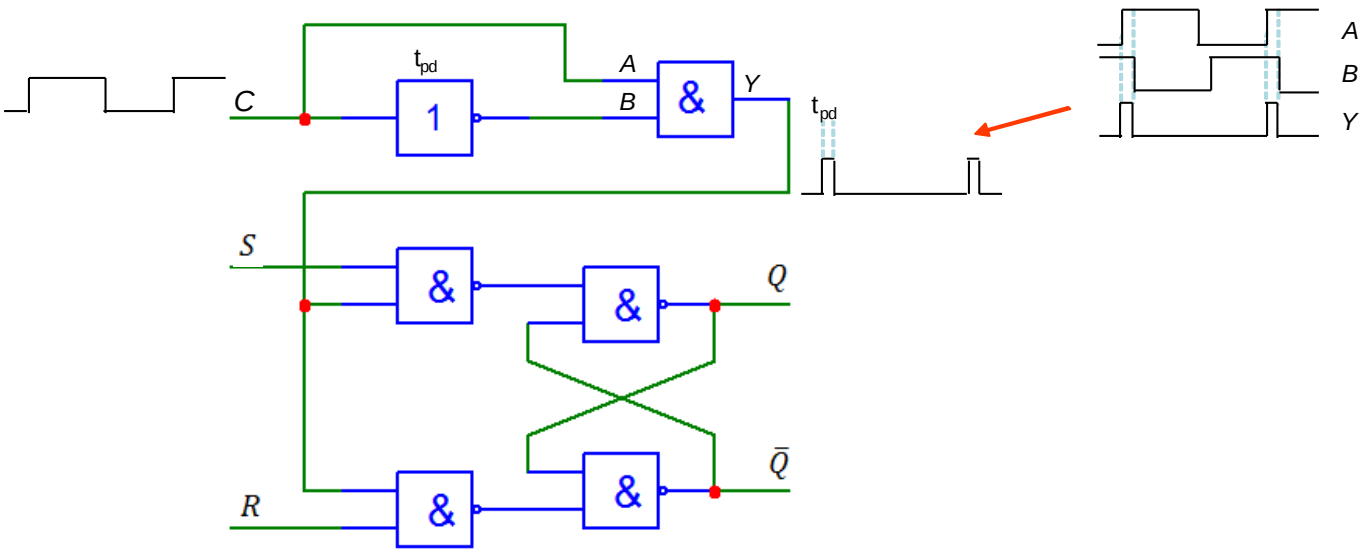


27. ábra A tárolóra rákötjük az órajelet is.

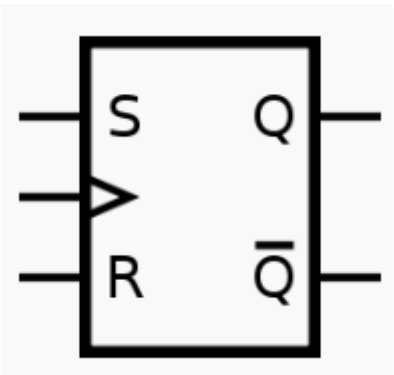
A rendszer statikus vezérlés/ szint vezérlés esetén csak akkor fog átbillenni, ha az órajel 1 értékű. Ez a megoldás nem használható szinkron hálózat építésére mert ún. „átlátszó”. Az órajel „1” értékénél az esetleges többszöri változás a bemeneten a kimenetet is többször átbillentheti, és ez tovább is terjed a flip-flopon keresztül. Ezért ún. dinamikus élvezérlést alkalmazunk, melynek során nem engedjük folyamatosan az órajel „1” értéke alatt hatni a bemeneteket csak egy rövid időre, amíg a tároló át tud billeni, ez után elvesszük a beíró (óra) jelet. Ezt úgy valósítjuk meg, hogy lerövidítjük az órajel „1” értékét, azaz szándékosan hazárdos órajel formáló hálózatot „csinálunk”.

Viszont ez idő alatt az ilyen elemekből felépített hálózat teljes egésze aszinkron módon viselkedne.

Ez szinkron hálózatban nem megengedhető, mert ott egy szinkron jel csak egy változásra adhat lehetőséget.

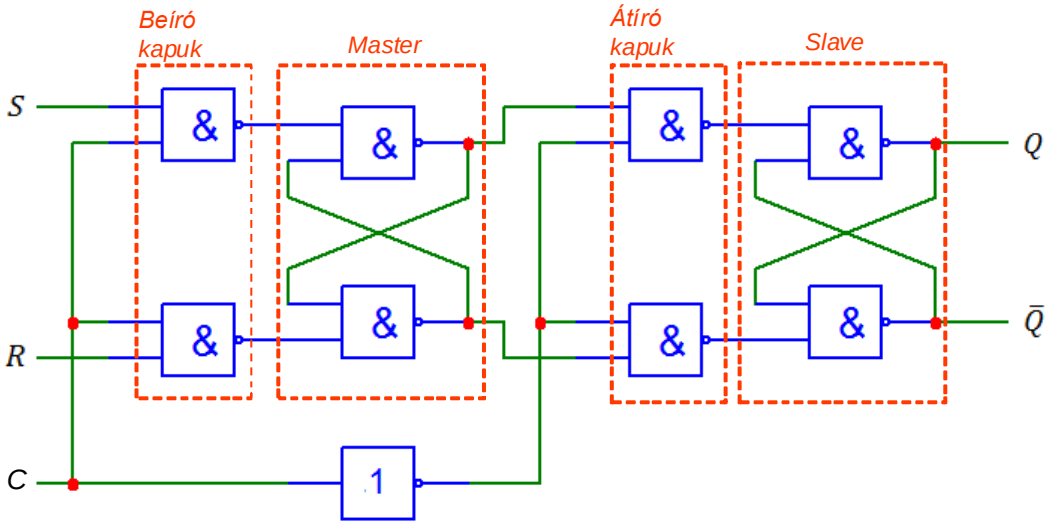


28. ábra Az órajel rövidítése



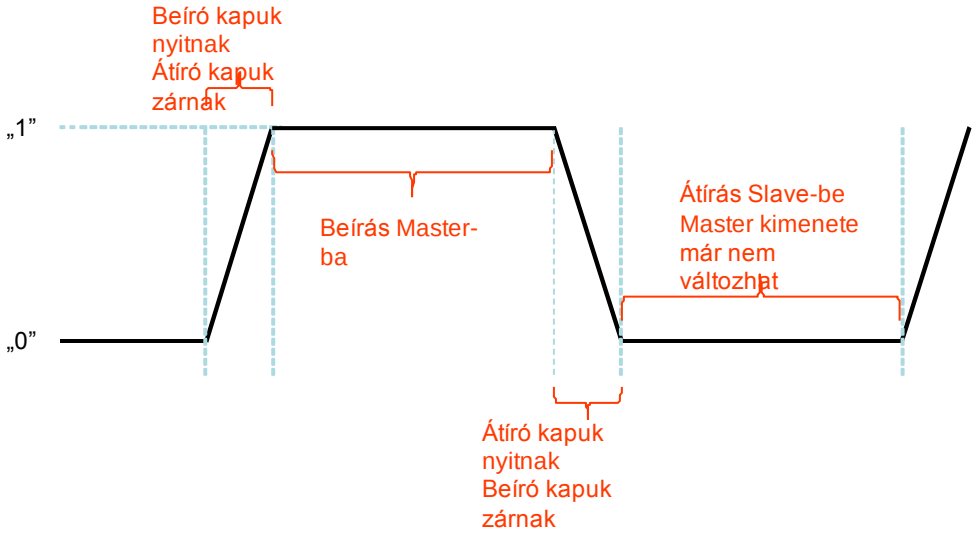
29. ábra RS tároló - órajel rövidítés funkcióval

A megvalósítás során ún. Kétfokozatú tároló (Master-Slave flip-flop) létrehozása az ideális, mert a Master-be írás alatt lehet tranzien,s de az átírás előtt már lecseng, és ezért tiltás alatt Master kimenete állandó.



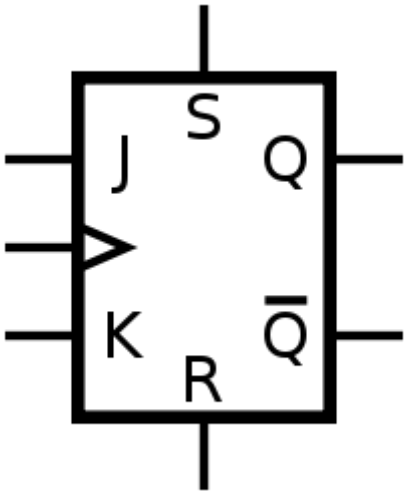
30. ábra Az RS tároló Master - Slave megvalósítása

A Master – Slave kétfokozatú tároló megvalósítás lényege, hogy kétfokozatú tároló a Master-be írás alatt lehet tranziens, de az átírás előtt már lecseng, és az átírás alatt Master kimenete állandó



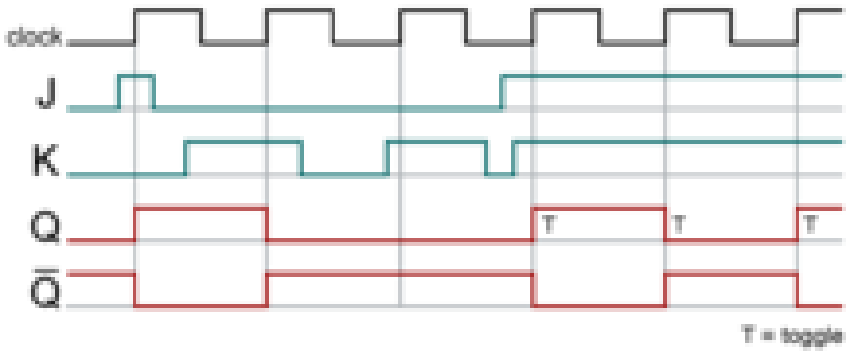
31. ábra Master - Slave kétfokozatú tároló működési elve

6.3.1. A JK flip-flop



32. ábra A JK flip-flop jele

Ha egy SR tároló mindkét bemenetének magas szintűre állítása esetén azt szeretnénk, hogy a kimenetet negálja, akkor – az instabil állapotok kiküszöbölése céljából – egy D flip-flopot teszünk a JK tárolónkra. Az ábrán látható JK flip-flopon egy beállító és egy törlő bemenet is van, ezeket a kezdeti állapotuk beállítására lehet használni. (Általában nem Set és Reset szoktak lenni, hanem Preset és Clear.)



33. ábra A JK flip-flop időzítési diagramja

6.3.1.1. A JK tárolót leíró függvény

A rendszert tehát az alábbi függvénnyel írhatjuk le:

$$Q^{n+1} = J\bar{Q}^n + \bar{K}Q^n$$

6.3.1.2. A JK tároló állapottáblája

A tároló állapottáblája jól megmutatja, hogy itt nincs tiltott bemeneti kombináció, s ezzel ki lehet küszöbölni az előző tároló hibáját. Két egyes esetén visszatér, egy belső állapotártéket kapunk

K	J	Q^{n+1}
0	0	Q^n
0	1	1
1	0	0
1	1	Q^n

34. ábra A JK tároló állapottábla

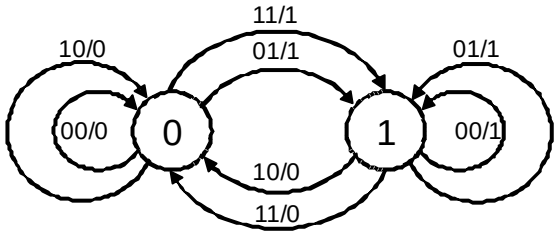
K	J	Q^n	Q^{n+1}	
0	0	0	0	Változatlan
0	0	1	1	
0	1	0	1	Beírás
0	1	1	1	
1	0	0	0	Törlés
1	0	1	0	
1	1	0	1	Billentés
1	1	1	0	

35. ábra A JKtároló állapottáblái (funkciói)

A rendszer 3 funkcióval rendelkezik tudjuk törölni, adatot beírni, és természetesen át tudjuk billenteni egyik állapotból a másikba.

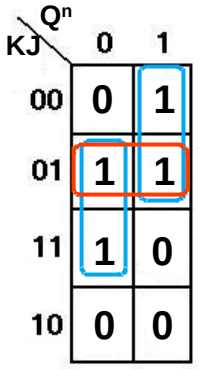
6.3.1.3. A JK tároló állapotgráfja

A tároló állapotgráfja ugyanezt egy kicsit más módon írja le.



36. ábra A JK tároló állapotgráfja

6.3.1.4. A JK tároló vezérlő táblája



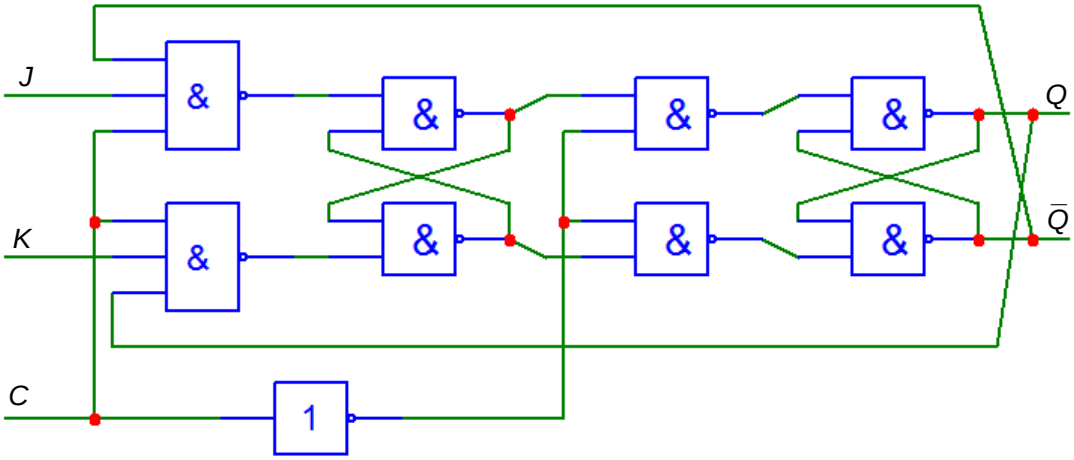
37. ábra A JK tároló vezérlőzáblája

A tároló vezérlőtáblája felírható az állapottábla alapján. Így az egyszerűsítések után megkaphatjuk a rendszer függvényét:

$$Q^{n+1} = J\bar{Q}^n + \bar{K}Q^n$$

6.3.1.5. A JK tároló megvalósítása

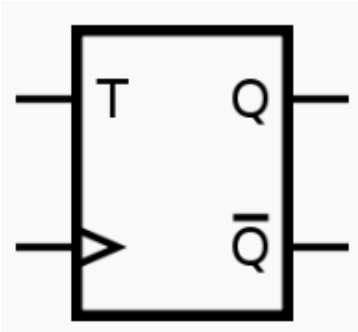
A JK tárolót is meg lehet valósítani kétfokozatú (Master-Slave) megvalósítás segítségével. Ez a megoldás tulajdonképpen RS tárolóból történő megvalósítást jelent, külön visszacsatolásokkal. Így a rendszer működése során a Master-ba írást az előző állapot is vezérli a visszacsatoláson keresztül.



38. ábra A JK tároló megvalósítása RS tárolók segítségével

6.3.1. A T flip-flop

A tároló egy bemenettel és két kimenettel rendelkezik.



39. ábra A T tároló

6.3.1.1. A T tárolót leíró függvény

A T flip-flop magas szintű bemenet esetén a kimenetét negálja. Ezért az őt leíró függvény:

$$Q^{n+1} = T\bar{Q}^n + \bar{T}Q^n$$

6.3.1.2. A T tároló állapottáblája

A rendszernek csak egy bemenete van, ezért az állapottábla felírása is egyszerű.

T	Q^{n+1}
0	Q_n
1	$\overline{Q_n}$

40. ábra A T tároló állapottáblája

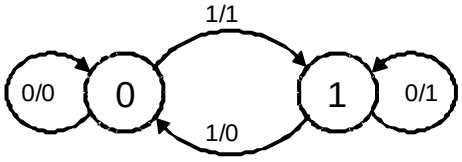
T	Q^n	Q^{n+1}	
0	0	0	Változatlan
0	1	1	
1	0	1	Billentés
1	1	0	

41. ábra A T tároló funkciói

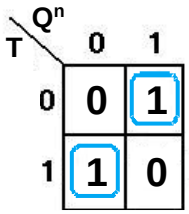
A tároló egyetlen funkcióval bír, megfelelő jelre átbillen.

6.3.1.3. A T tároló állapotgráfja

Ezek alapján az állapotgráf is egyszerűen felírható.



6.3.1.4. A T tároló vezérlő táblája



42. ábra A t tároló vezérlési táblája

A tábla alapján a függvény is felírható.

$$Q^{n+1} = T\bar{Q}^n + \bar{T}Q^n$$

6.3.1.5. A T tároló megvalósítása

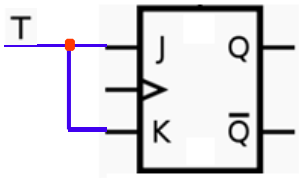
A tárolót JK tárolóval szoktuk megvalósítani,mert e tároló állapottáblájában van két olyan sor , mely leírja ennek a tárolónak a működését. Ez kihasználva a két bemenetre a T értékét kötjük rá.

K	J	Q^{n+1}
0	**0**	**Q^n**
0	1	1
1	0	0
1	**1**	**Q^n**

43. ábra A JK tároló állapottáblájának két sora, mely megvalósítja a T tárolót.

T	Q^{n+1}
0	Q^n
1	Q^n

44. ábra A T tároló állapottáblája

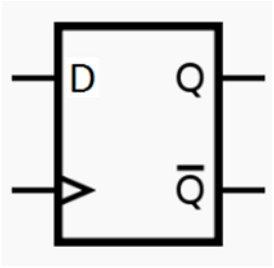


45. ábra a T tároló megvalósításáa JK tárolóval

6.3.2. A D flip-flop

A D flip-flop a legegyszerűbb, 1 bites memóriaelemnek tekinthető. Létezik élvezérelt, és két fázisú órajellel vezérelt típusa is. Alapvetően két D-G tárolóból áll, amelyek master-slave elrendezésűek, tehát az első által fogadott jelet a második – vezérléstől függően – a következő fázisban másolja a kimenetre.

Maga a tároló csak egy bemenettel rendelkezik, és igazából átmeneti információ tárolásra használható.



46. ábra A D tároló

6.3.2.1. A D tárolót leíró függvény

A tároló működését nagyon egyszerű függvény írja le.

$$Q^{n+1} = D$$

6.3.2.2. A D tároló állapottáblája

Az állapottábla alapján elmondható, hogy a tároló a bejövő értéket átmenetileg eltárolja, és adja tovább a kimenetén.

D	Q^{n+1}
0	0
1	1

47. ábra A D tároló állapottáblája

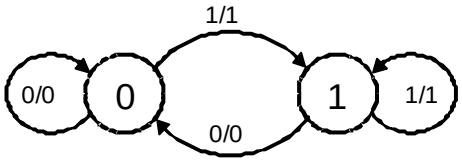
D	Q^n	Q^{n+1}	
0	0	0	Törlés
0	1	0	
1	0	1	Beírás
1	1	1	

48. ábra A D tároló funkciói

A tároló csak írni és törölni képes a funkcióját tekintve.

6.3.2.3. A D tároló állapotgráfja

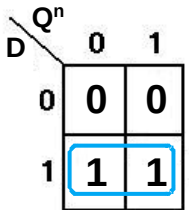
Állapotgráfja ugyanezen feladatokat, állapotokat mutatja meg.



49. ábraA D tároló állapotgráfja

6.3.2.4. A D tároló vezérlő táblája

A vezérlőtábla felírásával és egyszerűsítésével eljutunk az egyszerűbb függvényalakig.



50. ábra A D tároló vezérlóő táblája

6.3.2.5. A D tároló megvalósítása

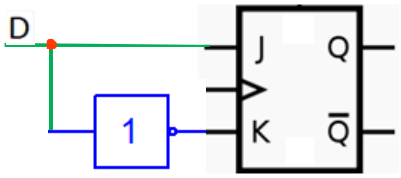
A tárolót JK tárolóval szoktuk megvalósítani,mert e tároló állapottáblájában van két olyan sor , mely leírja ennek a tárolónak a működését. Ez kihasználva a két bemenetre a T értékét kötjük rá.

K	J	Q^{n+1}
0	0	Q^n
0	1	1
1	0	0
1	1	Q^n

51. ábra A JK tároló állapottáblájának két sora, mely megvalósítja a D tárolót.

D	Q^{n+1}
0	0
1	1

52. ábra A D tároló állapottáblája



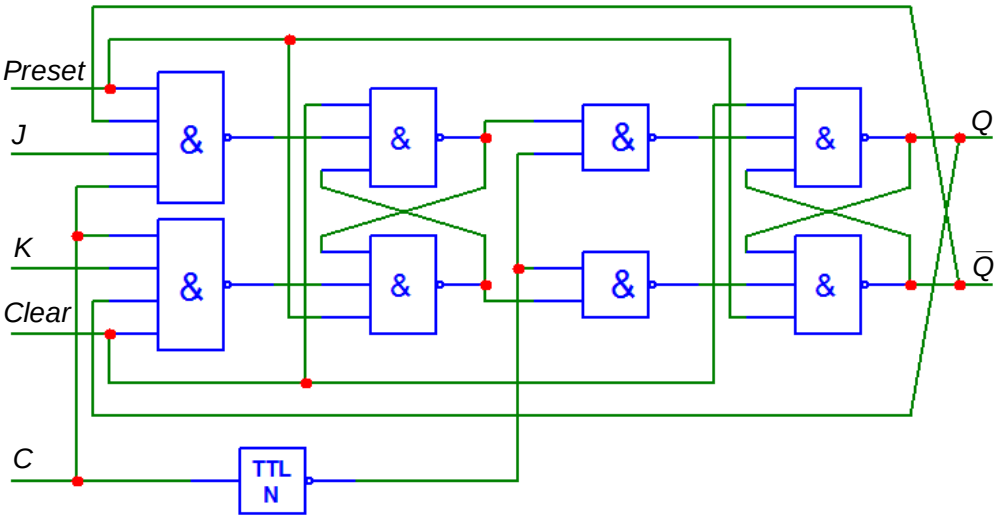
53. ábra a D tároló megvalósításáa JK tárolóval

6.4. Alapállapotba állítás

A berendezések bekapcsolásakor biztosítani kell a stabil, ismert alapállapotot. Mivel általában aszinkron bemenetek állnak rendelkezésre a való életben, ezért erre fokozottan kell figyelni.

A tároló kiindulási állapota lehet „0” vagy „1”. Így a gyakorlatban két plusz bemenetet használunk:

1) Clear (Reset) bemenet - A tároló törlése, „0”-ába állítása
2) Preset (Set) bemenet - A tároló beállítása, „1”-be állítása

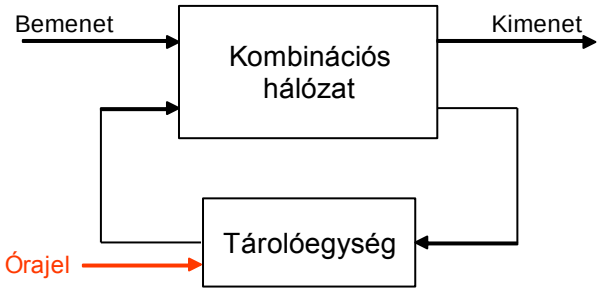


54. ábra A Az alapállapotba állítható hálózat

6.5. A szinkron sorrendi hálózat működése

6.5.1. Bevezetés

A szinkron sorrendi hálózat a múlt órán tanultak alapján úgy épül fel, hogy az alábbi rajzzal kifejezhető:



55. ábra A sorrendi hálózat felépítése

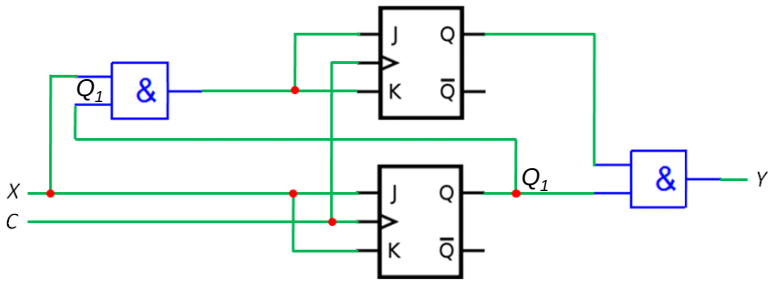
Minden tároló elem két állapotot vehet fel: „0” vagy „1”. Ezért, ha n tárolóelem van, a teljes hálózatnak 2^n állapota lehet. Működés közben ezek közül természetesen nem feltétlenül valósul meg mindegyik hiszen lehetnek tiltott állapotok. Viszont egyik állapotból a másikba csak egy újabb órajel hatására kerülhet a rendszer.

Fontos, hogy a hálózatokban a bemeneti jelek és a tároló elemek tartalma együttesen határozzák meg a következő (Q_{n+1}) állapotot. A tároló elemek pedig az előző órajel hatására létrejött belső (Qn) állapotot tárolják.

6.5.2. Kapcsolási rajz

Legyen egy példahálózatunk, melyben 2 tároló elem (T tárolók)található. Természetesen egy egyszerű bemeneti és kimeneti kombinációs hálózatról legyen szó.

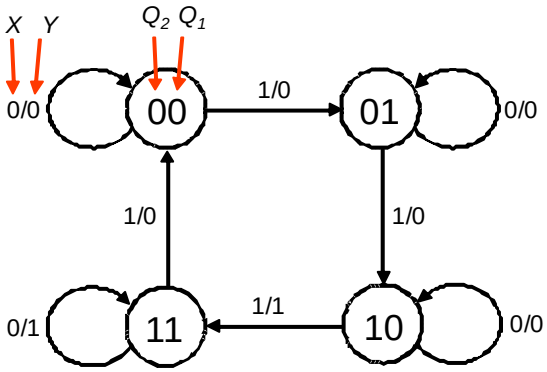
Fontos észrevenni, hogy sorrendi hálózatoknál a tároló elemek és visszacsatolások nehezítik a megértést még ennél a viszonylag egyszerű hálózatnál is. Bonyolultabb esetben átláthatatlanná válhat a kapcsolási rajz. Ezért az összeköttetéseket gyakran összekötő vonalak helyett azonos elnevezéssel helyettesítik.



56. ábra A kapcsolási rajz bonyolultsága

6.5.3. Állapotgráf

A rendszer állapotgráfja a következő:



57. ábra A hálózat állapotgráfja

A rendszer állapotgráfja szemléletes, könnyen áttekinthető. A hálózat két (belső) szekunder változót tartalmaz, melynek négy lehetséges állapota van. A lehetséges állapotokat a tároló elemek kimeneti jelével kódoljuk:

Q_2Q_1 = 00, 01, 10, 11

Az állapotok közül egyik állapot sem tiltott

Ha X = 1, akkor állapotváltozás következik be.

6.5.4. Állapottáblázat

Az állapot gráfból könnyen felírható

Q_2^n	Q_1^n	X	Q_2^{n+1}	Q_1^{n+1}	Y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	1	0	0
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	1

58. ábra A hálózat állapottáblája

6.5.5. Állapotegyenlet

Egy sorrendi hálózat elvi működése két logikai függvénnyel írható le. Az egyik az állapotegyenlet, mely a szekunder változók függvénye, a másik a kimeneti függvény , mely a függő változók függvénye.

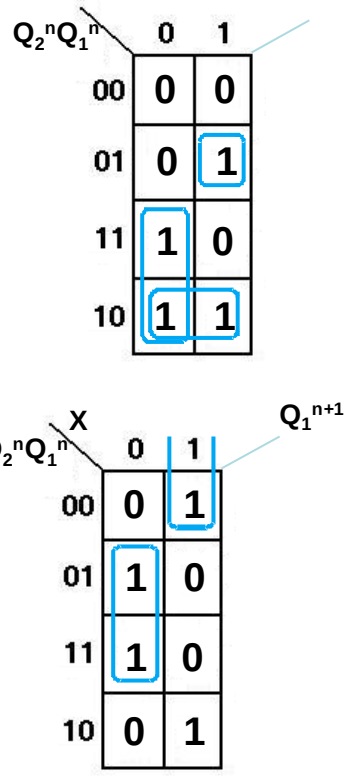
Természetesen annyi állapotegyenlet lesz, ahány szekunder változó (ahány tároló elem), és annyi kimeneti egyenlet, ahány kimenet.

Hátránya ennek a felírási módnak, hogy a hálózat tényleges felépítésére nem ad információt, ezért nem tudjuk, hogy JK, T vagy D tárolóval, NAND, NOR kapuvel került e megvalósításra.

$$Q^{n+1} = F_Q(X^n, Q^n)$$

$$Y^n = F_Y(X^n, Q^n)$$

A megoldáshoz természetesen szükségünk van a vezérlési táblára



59. ábraA rendszer vezérlési táblái

$$Q_1^{n+1} = Q_1^n \cdot \bar{X} + \bar{Q}_1^n \cdot X$$

$$Q_2^{n+1} = \bar{Q}_2^n \cdot Q_1^n \cdot X + \bar{Q}_2^n \cdot \bar{X} + \cdot Q_2^n \cdot \bar{Q}_1^n$$

$$Y = Q_1 \cdot Q_2$$

60. ábra A hálózat egyenletei

6.5.6. VHDL

VHDL (VHSIC Hardware Description Language) - VHSIC : very-high-speed integrated circuits- Egy hardver leíró nyelv. Logikai áramkörök egyszerű szöveges leírására fejlesztették ki (USA 1987)

A logikai áramkörökre jellemző párhuzamosság kezelésére, leírására fejlesztették, ezért konkurens és szekvenciális utasítások halmazából áll.

Fő feladata a

logikai hálózatok modellezése

szimulációja (testbench)

Szintetizálása (hardver megvalósítás).

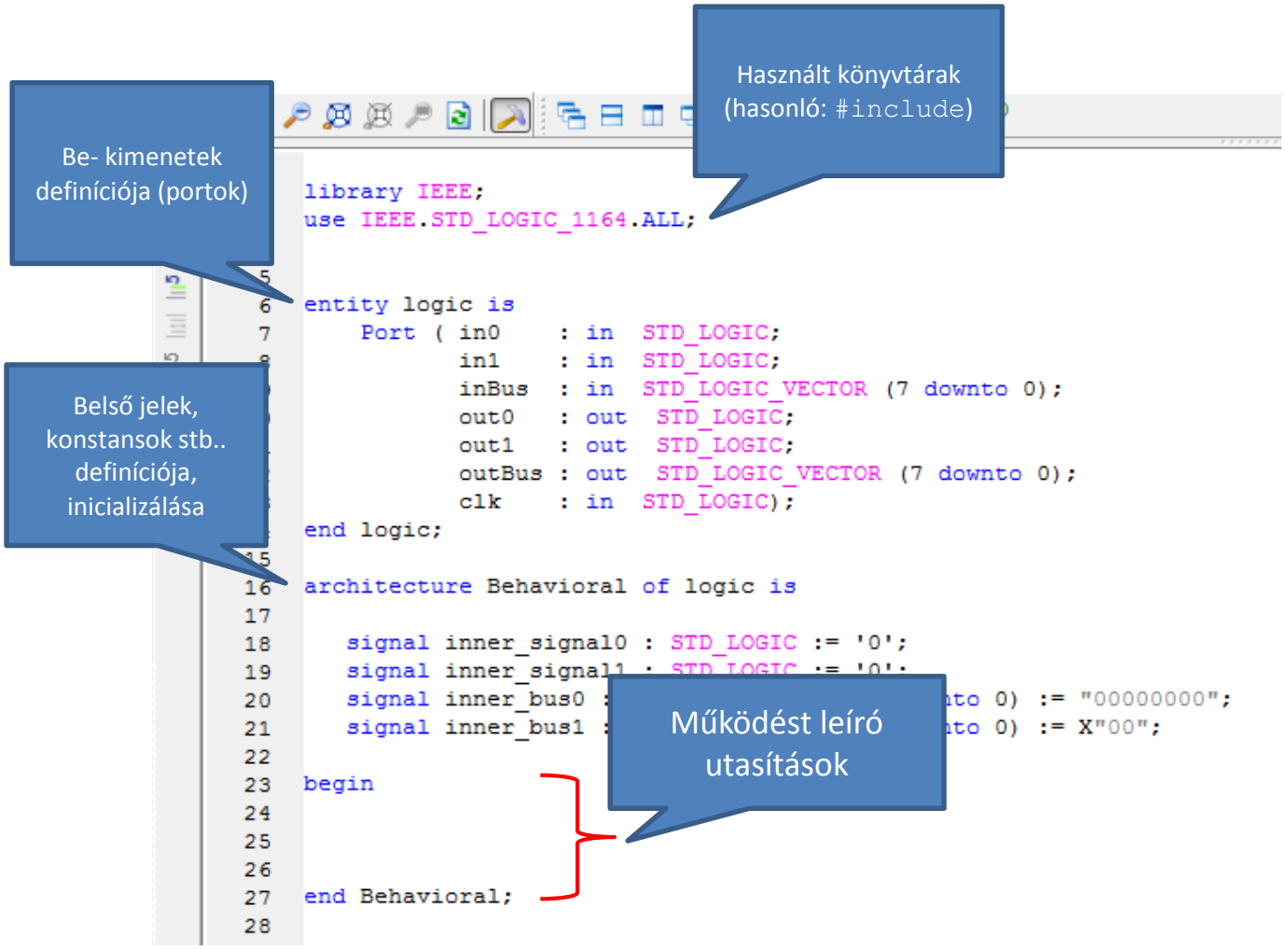
Támogatja az IEE két szabványát:

- IEEE Std 1076-1987
- IEEE Std 1076-1993

A programozási nyelvekhez hasonló a felépítése.

Elsősorban Integrált áramkörök gyártásánál és programozható logikai áramkörök (CPLD, FPGA) fejlsztéséhez használják.

VHDL



61. ábra VHDL kód felépítése

A VHDL kódban megadhatók a rendszer portjai, bemenetek és kimenetek állapotai ((pl. konstansok beállíthatóak, valamint találunk egy működést leíró részt is.

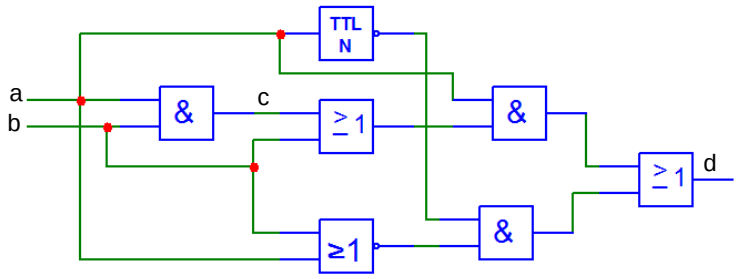
```
signal a : STD_LOGIC;
signal b : STD_LOGIC;
signal c : STD_LOGIC;
signal d : STD_LOGIC;
…
…
begin
   a <= '0';
```

```
    b <= '1';
    c <= a and b;
    d <= c or b  when a = '1' else
         a nor b when a = '0';
…
…
end Behavioral
```

62. ábra Értékadás VHDL kódban - Konkurens utasítások

Konkurens utasítások jellemzője, hogy az utasítások egyszerre hajtódnak végre, a leírás sorrendjétől függetlenül. Ezek az utasítások általában kombinációs hálózatot tudnak leírni.

A fenti kód az alábbi hálózatot írja le:



63. ábra A leírt hálózat

A szekvenciális utasítások leírás sorrendjében hajtódnak végre és velük szekvenciális hálózatot lehet leírni.

pl. ilyen az alábbi D tárolót leíró részlet.

```
signal d : STD_LOGIC;
signal q : STD_LOGIC;
signal qn : STD_LOGIC;
signal reset : STD_LOGIC;
signal clk : STD_LOGIC;
…
begin
…
   qn <= not q;
   process
   begin
     if (reset = '1') then
        q <= '0';
```

```
        elsif (clk`event and clk = '1') then
           q <= d;
        end if;
     end process;
…
end Behavioral;
```

64. ábra A D tárolót kódoló részlet

6.6. Példa

6.6.1.1. Feladat

T tárolók és ÉS –VAGY kombinációs hálózat segítségével tervezze meg és rajzolja fel egy 3 bites szinkron számláló MEALY - MODELL szerinti logikai kapcsolási rajzát, amely a következő sorrendben számlál: 0, 1, 2, 3, 5 . Ezután ismétlődik. A belső állapotokat Q_0, Q_1, Q_2, a tároló bemeneteket pedig T_0, T_1, T_2 szimbólumokkal jelölje.

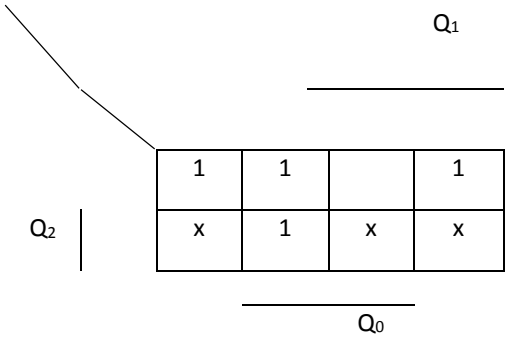
6.6.1.2. Megoldás

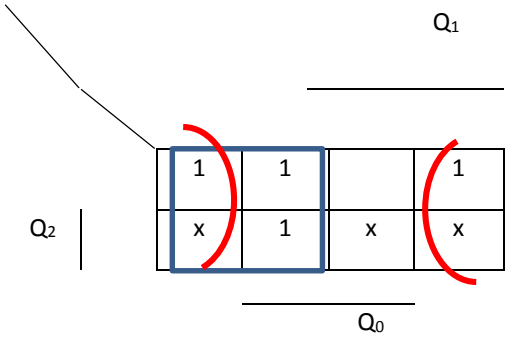
A feladat megoldása során először fel kell írnunk a lehetséges állapotokat, majd az ebből képzett vezérlési táblák segítségével a rendszer egyenleteit, hiszen ekkor tudjuk megrajzolni a hálózatot.

6.6.1.2.1. Állapottábla

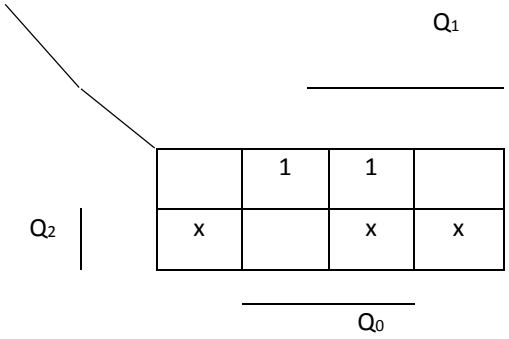
i	n			n+1			T Tárolók		
	Q_2	Q_1	Q_0	Q_2	Q_1	Q_0	T_2	T_1	T_0
0	0	0	0	0	0	1	0	0	1
1	0	0	1	0	1	0	0	1	1
2	0	1	0	0	1	1	0	0	1
3	0	1	1	1	0	1	1	1	0
4	1	0	0	x	x	x	x	x	x
5	1	0	1	0	0	0	1	0	1
6	1	0	0	x	x	x	x	x	x
7	1	1	1	x	x	x	x	x	x

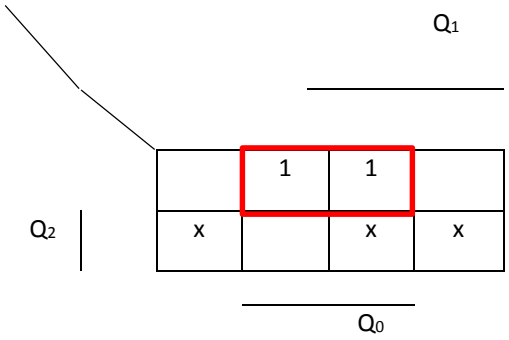
6.6.1.2.2. Vezérlési táblák és Egyenletek



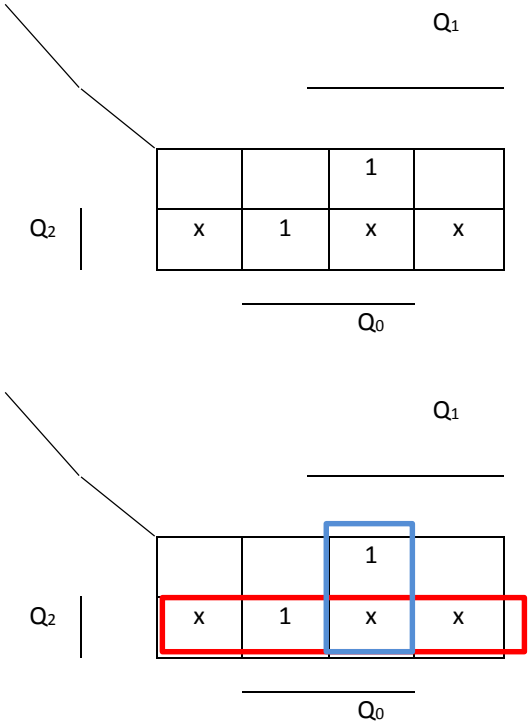


$$T_0 = \overline{Q_0} + \bar{Q}_1$$



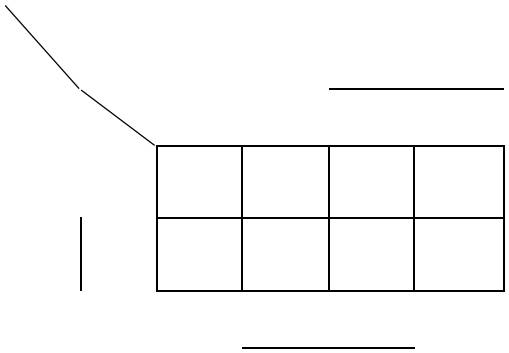


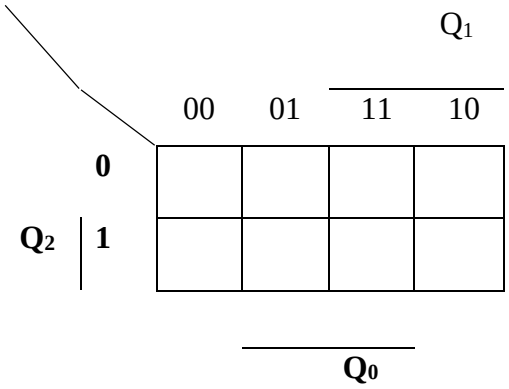
$$T_1 = \overline{Q_2} Q_0$$

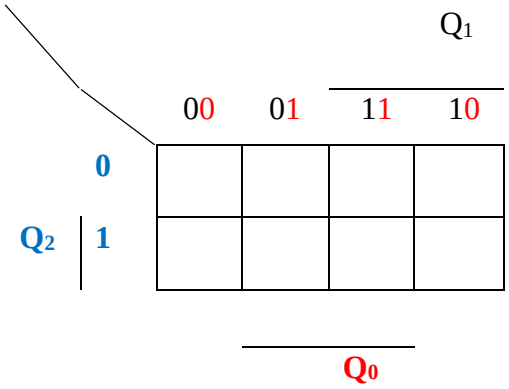


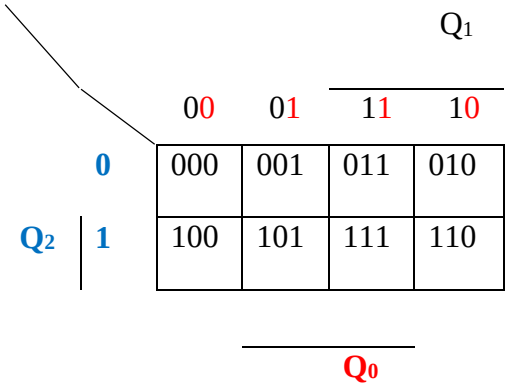
$$T_2 = Q_2 + Q_1 Q_0$$

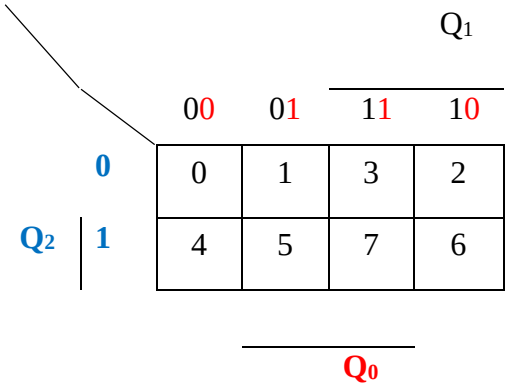
6.7. Feladatmegoldást segítő






Q1
00 01 11 10
0
Q2 1
Q0


Q1
00 01 11 10
0
000 001 011 010
Q2 1
100 101 111 110
Q0


Q1
00 01 11 10
0
0 1 3 2
Q2 1
4 5 7 6
Q0

		Q_1			
		00	01	11	10
Q_2	0	$\overline{Q_2}\overline{Q_1}\overline{Q_0}$	$\overline{Q_2}\overline{Q_1}Q_0$	$\overline{Q_2}Q_1Q_0$	$\overline{Q_2}Q_1\overline{Q_0}$
	1	$Q_2\overline{Q_1}\overline{Q_0}$	$Q_2\overline{Q_1}Q_0$	$Q_2Q_1Q_0$	$Q_2Q_1\overline{Q_0}$
			$\overline{Q_0}$		

6.8. Ellenőrző kérdések

1) Mit nevezünk sorrendi hálózatnak?
2) Milyen fajtái vannak a sorrendi hálózatnak?
3) Milyen sorrendi hálózati felírási lehetőségeket ismer?
4) Definiálja az állapottábla fogalmát!
5) Definiálja az állapotgráf fogalmát!
6) Definiálja a vezérlési tábla fogalmát!
7) Rajzolja fel a sorrendi hálózat Mealy – modell szerinti ábrázolását! tüntesse fel a hozzá kapcsolódó függvényeket is!
8) Definiálja a tároló fogalmát!
9) Melyek a tárolók főbb jellemzői?
10) Nevezze meg a tárolók fajtáit!
11) Jellemezze az RS tárolót az összes tanult felírási móddal!
12) Jellemezze az JK tárolót az összes tanult felírási móddal!
13) Jellemezze az T tárolót az összes tanult felírási móddal!
14) Jellemezze az D tárolót az összes tanult felírási móddal!
15) Hogyan lehet megvalósítani az RS tárolót? Valamennyi tanult lehetőséget ismertesse!
16) Hogyan lehet megvalósítani az JK tárolót? Valamennyi tanult lehetőséget ismertesse!
17) Hogyan lehet megvalósítani az T tárolót? Valamennyi tanult lehetőséget ismertesse!
18) Hogyan lehet megvalósítani az D tárolót? Valamennyi tanult lehetőséget ismertesse!
19) Hogyan állíthatunk egy tárolót alapállapotba?
20) Mi lehet a fő probléma a kapcsolási rajzos ábrázolás során? Hogyan szoktuk ezt orvosolni?
21) Hogyan határozható meg az állapotegyenlet?
22) Melyek a VHDL kód részei?
23) Tekintse át a tárolós feladatot. Melyek a feladatmegoldás lépései? (részletesen)

6.9. Feladatok

1) T tárolók és ÉS –VAGY kombinációs hálózat segítségével tervezze meg és rajzolja fel egy 3 bites szinkron számláló MEALY - MODELL szerinti logikai kapcsolási rajzát, amely a következő sorrendben számlál: 0, 1, 2, 3, 5 . Ezután ismétlődik. A belső állapotokat Q_0, Q_1, Q_2, a tároló bemeneteket pedig T_0, T_1, T_2 szimbólumokkal jelölje.
2) T tárolók és ÉS –VAGY kombinációs hálózat segítségével tervezze meg és rajzolja fel egy 3 bites szinkron számláló MEALY - MODELL szerinti logikai kapcsolási rajzát, amely a következő sorrendben számlál: 0, 1, 2, 3 . Ezután ismétlődik. A belső állapotokat Q_0, Q_1, Q_2, a tároló bemeneteket pedig T_0, T_1, T_2 szimbólumokkal jelölje.
3) T tárolók és ÉS –VAGY kombinációs hálózat segítségével tervezze meg és rajzolja fel egy 3 bites szinkron számláló MEALY - MODELL szerinti logikai kapcsolási rajzát, amely a következő sorrendben számlál: 0, 1, 2, 3, 4, 5 . Ezután ismétlődik. A belső állapotokat Q_0, Q_1, Q_2, a tároló bemeneteket pedig T_0, T_1, T_2 szimbólumokkal jelölje.
4) T tárolók és ÉS –VAGY kombinációs hálózat segítségével tervezze meg és rajzolja fel egy 3 bites szinkron számláló MEALY - MODELL szerinti logikai kapcsolási rajzát, amely a következő sorrendben számlál: 0, 1, 3, 5 . Ezután ismétlődik. A belső állapotokat Q_0, Q_1, Q_2, a tároló bemeneteket pedig T_0, T_1, T_2 szimbólumokkal jelölje.
5) T tárolók és ÉS –VAGY kombinációs hálózat segítségével tervezze meg és rajzolja fel egy 3 bites szinkron számláló MEALY - MODELL szerinti logikai kapcsolási rajzát, amely a következő sorrendben számlál: 0, 1, 3, 5, 6. Ezután ismétlődik. A belső állapotokat Q_0, Q_1, Q_2, a tároló bemeneteket pedig T_0, T_1, T_2 szimbólumokkal jelölje.
6) T tárolók és ÉS –VAGY kombinációs hálózat segítségével tervezze meg és rajzolja fel egy 3 bites szinkron számláló MEALY - MODELL szerinti logikai kapcsolási rajzát, amely a következő sorrendben számlál: 0, 1, 2, 3, 5, 6, 7 . Ezután ismétlődik. A belső állapotokat Q_0, Q_1, Q_2, a tároló bemeneteket pedig T_0, T_1, T_2 szimbólumokkal jelölje.
7) T tárolók és ÉS –VAGY kombinációs hálózat segítségével tervezze meg és rajzolja fel egy 3 bites szinkron számláló MEALY - MODELL szerinti logikai kapcsolási rajzát, amely a

következő sorrendben számlál: 1, 2, 3, 5 . Ezután ismétlődik. A belső állapotokat Q_0, Q_1, Q_2, a tároló bemeneteket pedig T_0, T_1, T_2 szimbólumokkal jelölje.

8) T tárolók és ÉS –VAGY kombinációs hálózat segítségével tervezze meg és rajzolja fel egy 3 bites szinkron számláló MEALY - MODELL szerinti logikai kapcsolási rajzát, amely a következő sorrendben számlál: 0, 1, 2, 5, 6 . Ezután ismétlődik. A belső állapotokat Q_0, Q_1, Q_2, a tároló bemeneteket pedig T_0, T_1, T_2 szimbólumokkal jelölje.
9) T tárolók és ÉS –VAGY kombinációs hálózat segítségével tervezze meg és rajzolja fel egy 3 bites szinkron számláló MEALY - MODELL szerinti logikai kapcsolási rajzát, amely a következő sorrendben számlál: 0, 1, 5, 6, 7 . Ezután ismétlődik. A belső állapotokat Q_0, Q_1, Q_2, a tároló bemeneteket pedig T_0, T_1, T_2 szimbólumokkal jelölje.
10) T tárolók és ÉS –VAGY kombinációs hálózat segítségével tervezze meg és rajzolja fel egy 3 bites szinkron számláló MEALY - MODELL szerinti logikai kapcsolási rajzát, amely a következő sorrendben számlál: 0, 3, 5, 6, 7 . Ezután ismétlődik. A belső állapotokat Q_0, Q_1, Q_2, a tároló bemeneteket pedig T_0, T_1, T_2 szimbólumokkal jelölje.

6.10. Irodalom

Kóré László: Digitális elektronika I. (BMF 1121)

Zsom Gyula: Digitális technika I. (Műszaki Könyvkiadó, Budapest, 2000, KVK 49-273/I, ISBN 963 6 1786 6)

Zsom Gyula: Digitális technika II. (Műszaki Könyvkiadó, Budapest, 2000, KVK 49-273/II, ISBN 963 16 1787 4)

Arató Péter: Logikai rendszerek tervezése (Tankönyvkiadó, Budapest, 1990, Műegyetemi Kiadó 2004, 55013)

Zalotay Péter: Digitális technika (http://www.kobakbt.hu/jegyzet/DigitHW.pdf)

Rőmer Mária: Digitális rendszerek áramkörei (Műszaki Könyvkiadó, Budapest, 1989, KVK 49-223)

Rőmer Mária: Digitális technika példatár (KKMF 1105, Budapest 1999)

Matijevics István: Digitális Technika Interaktív példatár (ISBN 978-963-279-528-7 Szegedi Tudományegyetem)

http://www.inf.u–szeged.hu/projectdirs/digipeldatar/digitalis_peldatar.html