Az elektronikus hálózatok csoportjai:

(jelfeldolgozás alapján)

- ***analóg hálózatok***
  - *a jelek értékkészlete (elméletileg) végtelen nagy*
  - *a jelek értelmezési tartománya folytonos eloszlású*

- ***digitális hálózatok***
  - *a jelek értékkészlete véges*
  - *a jelek eloszlása diszkrét*

Széchenyi István Egyetem

A logikai értékek és műveletek

„Kétértékes" rendszerek:

- Állítások:

  IGAZ, HAMIS

- Bináris számrendszer:

  1, 0

- Kapcsolók:

  bekapcsolva, megszakítva

'·' („ÉS")	'+' („VAGY")	(„NEGÁLT")
$0 \cdot 0 = 0$	$0 + 0 = 0$	$\overline{0} = 1$
$0 \cdot 1 = 0$	$0 + 1 = 1$	$\overline{1} = 0$
$1 \cdot 0 = 0$	$1 + 0 = 1$	
$1 \cdot 1 = 1$	$1 + 1 = 1$	

- *A digitális hálózatokban az információ hordozója a **bit**.*

A kapcsoló algebra azonosságai (1)

Asszociativitás:

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

Kommutativitás:

$$A \bullet B = B \bullet A$$
$$A + B = B + A$$

Disztributivitás:

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

Ellentmondás törvénye:

$$A \bullet \bar{A} = 0$$

A kapcsoló algebra azonosságai (2)

Elnyelési törvények 1.	$A + (A \bullet B) = A$ $A \bullet (A + B) = A$
Elnyelési törvények 2.	$A + (\bar{A} \bullet B) = A + B$ $A \bullet (\bar{A} + B) = A \bullet B$
Elnyelési törvények 3.	$A + 1 = 1$ $A \bullet 0 = 0$
Harmadik kizárásának törvénye:	$A + \bar{A} = 1$

A kapcsoló algebra azonosságai (3)

Identitások törvénye:

$$A + 0 = A$$
$$A \bullet 1 = A$$

Idempotencia törvénye:

$$A + A = A$$
$$A \bullet A = A$$

Involúció törvénye:

$$\bar{\bar{A}} = A$$

De-Morgan azonosságok:

$$\overline{A + B} = \bar{A} \bullet \bar{B}$$
$$\overline{A \bullet B} = \bar{A} + \bar{B}$$

Széchenyi István Egyetem

KOMBINÁCIÓS HÁLÓZATOK

A kombinációs hálózat fekete-doboz modellje:

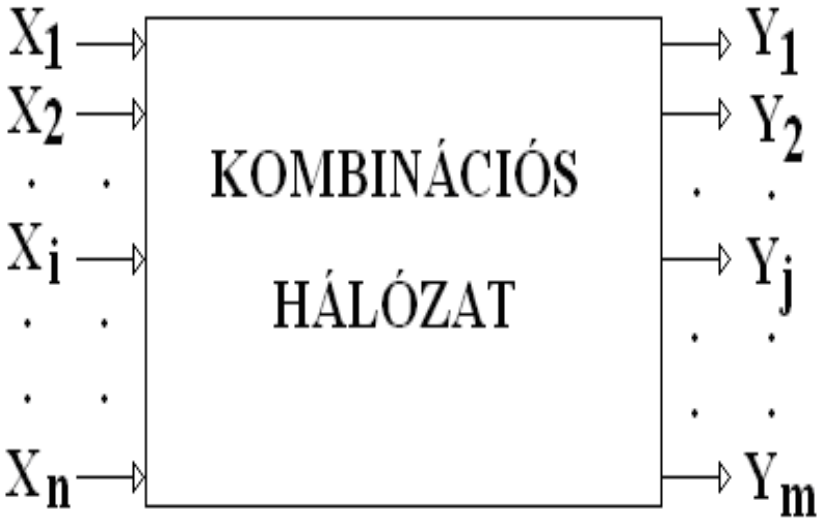
X1. . . .Xn :
bemenetek,

logikai változók

Y1. . . .Ym :

kimenetek,

logikai változók



A kombinációs hálózat definíciója:

a kombinációs hálózat viselkedésének legfontosabb sajátossága, hogy egy meghatározott bemeneti kombináció ismételt rákapcsolásaira - a tranziens idő eltelte után - mindig ugyanazt a kimeneti kombinációt szolgáltatja, függetlenül attól, hogy az adott bemeneti kombináció két rákapcsolása között milyen más bemeneti kombinációkat kapcsoltunk a hálózatra.

Széchenyi István Egyetem

Kombinációs hálózat definiálása táblázattal

Példa:

három bemenet **:**

X1, X2, X3

két kimenet**:**

Y1, Y2

Bemeneti variációk			Kimeneti variációk	
X1	X2	X3	Y1	Y2
0	0	0	0	1
0	0	1	1	0
0	1	0	0	1
0	1	1	1	1
1	0	0	1	1
1	0	1	0	1
1	1	0	1	1
1	1	1	1	0

Kombinációs hálózatok specifikációs mélysége

- ***teljesen specifikált („ts”)***:

  minden bemeneti variációra minden kimenet értéke elő van írva

- ***nem teljesen specifikált („nts”)*:**

  van olyan bemeneti variáció, ahol egy kimeneti változó értéke közömbös

Logikai függvények megadási módjai:

- *igazságtáblázattal*
- *algebrai kifejezéssel*
- *elvi (grafikus) logikai vázlattal*

- ***logikai függvény megadása igazságtáblával:***

Példa: egy teljesen specifikált („ts”), egykimenetű kombinációs hálózat igazságtáblázata

bemenetek:

A, B, C

kimenet:

F

igazságtábla

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

- ***logikai függvény megadása algebrai kifejezéssel (1):***

A minterm definíciója:

azokat a logikai szorzatokat (termeket), amelyekben a függvény valamennyi változója szerepel, mintermeknek nevezzük. A logikai függvény megadásának ezt a módját, azaz azon mintermek összegét, amelyekhez a függvény „1”-et rendel, mintermes kanonikus normál alaknak nevezzük.

- ***logikai függvény megadása algebrai kifejezéssel (2):***

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

A hálózat mintermes kanonikus normál alakja:

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

A szorzat változója

- ponált, ha a bemeneti variációban ‚1' szerepel az oszlopában,
- negált, ha a bemeneti variációban ‚0' szerepel az oszlopában

$$F(x_1, x_2, \ldots x_n) = \sum_{i=0}^{2^n - 1} v_i \cdot m_i$$

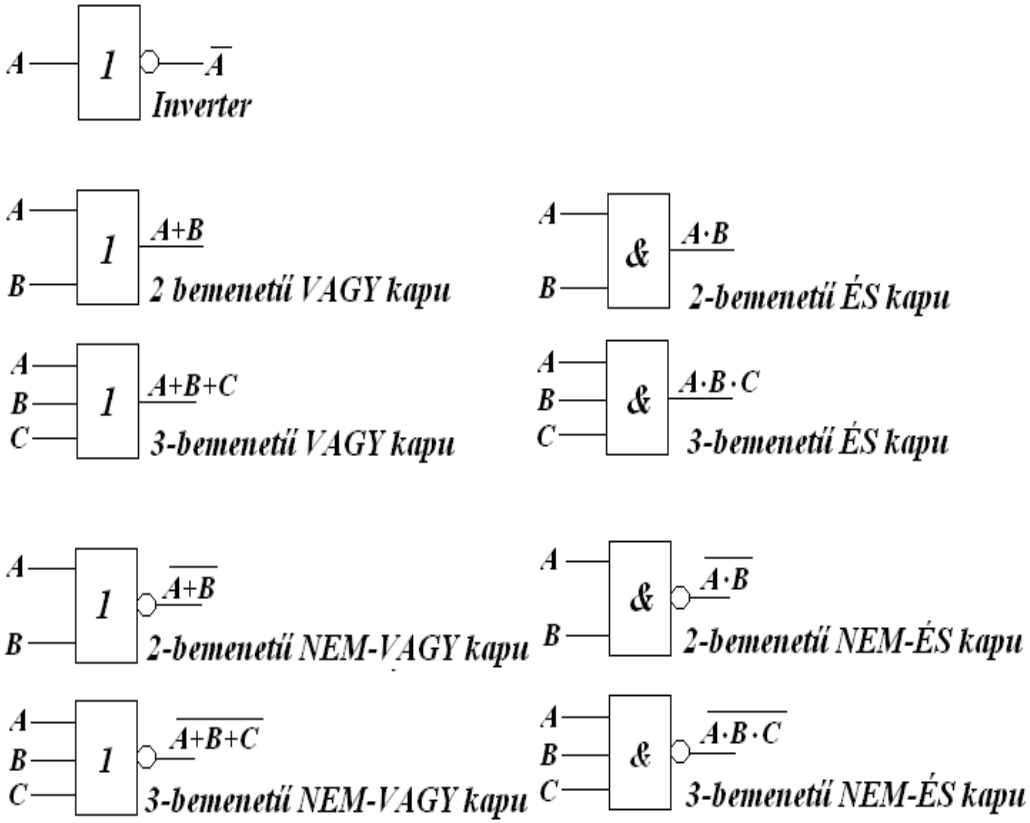
v_i : a két bináris érték egyike, 0 vagy 1

m_i : az igazság-tábla i.sorához rendelt logikai szorzat, amely
- minden változót tartalmaz
- kifejezi a sorhoz tartozó bemeneti kombináció 1-es értékét

$$F(x_1, x_2, \ldots x_n) = \sum_{v_i = 1} 1 \cdot m_j + \sum_{v_i = 0} 0 \cdot m_k = \sum_{v_i = 1} 1 \cdot m_j = \sum_{v_i = 1} m_j$$

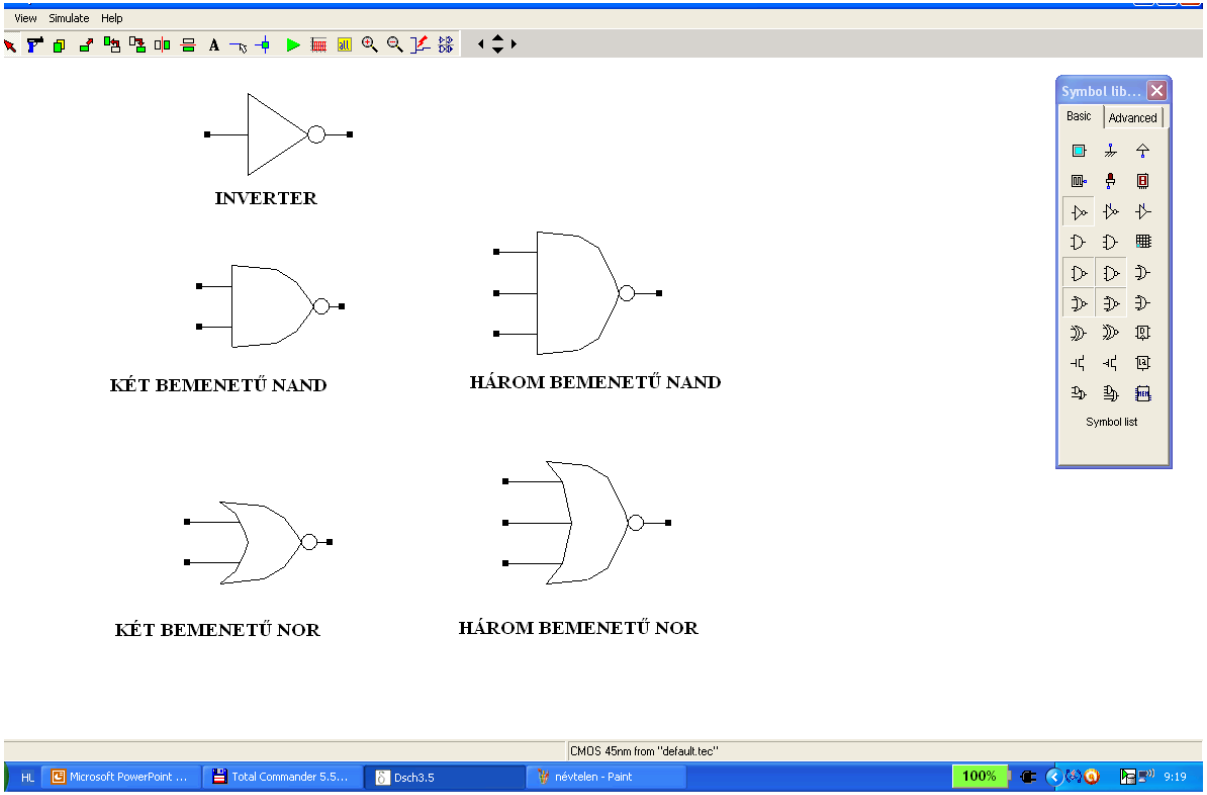
- ***logikai függvény megadása grafikus logikai vázlattal (1):***

A leggyakrabban használt grafikus logikai szimbólumok (európai szabvány):



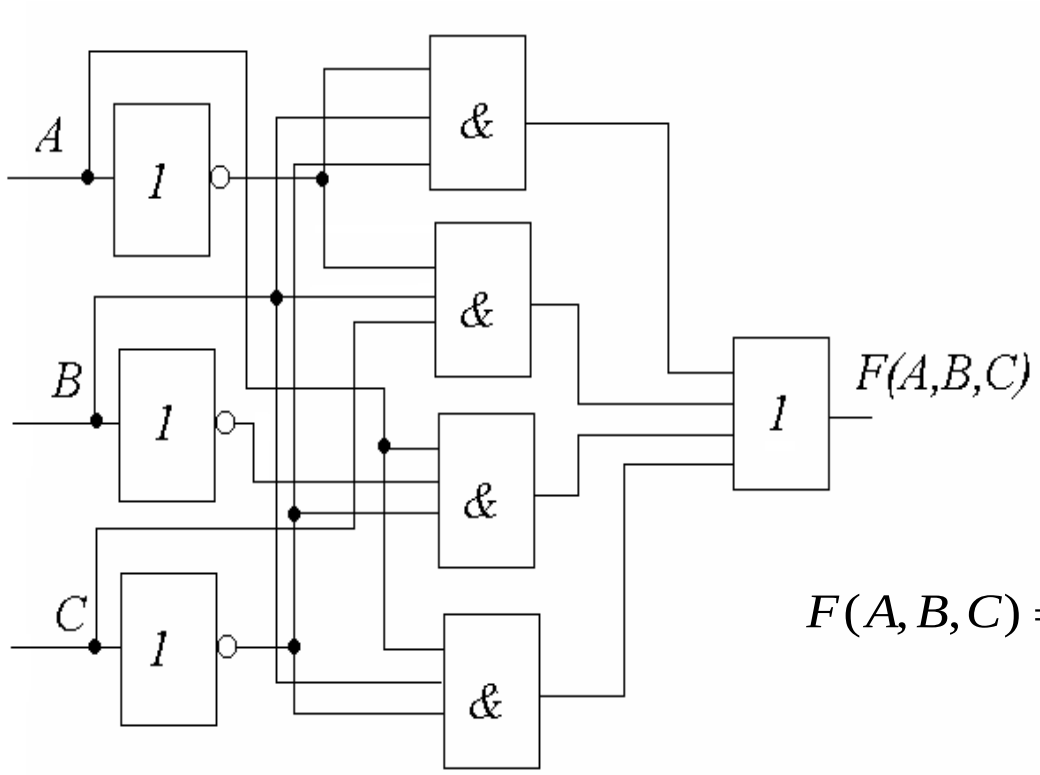
- ***logikai függvény megadása grafikus logikai vázlattal (2):***

A DSCH szimulátorban használt grafikus logikai szimbólumok (IEEE szabvány):



- ***logikai függvény megadása grafikus logikai vázlattal (3):***

A mintapélda grafikus logikai szimbólumokkal megadott vázlata:



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

Nevezetes kétváltozós logikai függvények (1)

BEM. VÁLT.		FÜGGVÉNYÉRTÉKEK															
x1	x2	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Nevezetes kétváltozós logikai függvények (2)

- **’0’**(F_0) **és ’1’**(F_{15}) **generátor**
- **’ÉS’**(F_1) **és ’NEM-ÉS’**(F_{14}) **függvény, illetve kapu**
- **’VAGY’**(F_7) **és ’NEM-VAGY’**(F_8) **függvény, illetve kapu**
- **ANTIVALENCIA**(F_6) **és EKVIVALENCIA**(F_9) **függvény, illetve kapu**
- **INHIBÍCIÓ**(F_2)
- **IMPLIKÁCIÓ**(F_{13})

Logikai függvények egyszerűsítésének módszerei

- *egyszerűsítés algebrai módszerrel*
- *Quine módszere*
- *Karnaugh táblás módszer*

Széchenyi István Egyetem

Logikai függvények egyszerűsítésének módszerei: algebrai módszer (1)

Példa:

három bemenet :

A, B, C

egy kimenet:

F

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

Logikai függvények egyszerűsítésének módszerei: algebrai módszer (2)

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

A mintermes kanonikus normál függvényalak egyszerűsítése algebrai módszerrel a disztributivitás, valamint „a harmadik kizárásának törvénye” alapján:

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

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

Logikai függvények egyszerűsítésének módszerei: Quine módszer (1)

Fogalmak

- ***prímimplikáns:*** olyan term, amelyből nem hagyható el több változó (nem egyszerűsíthető tovább)
- ***megkülönböztetett minterm:*** olyan minterm, amelyet csak egy prímimplikáns fed le
- ***lényeges prímimplikáns:*** olyan prímimplikáns, amely legalább egy megkülönböztetett mintermet tartalmaz

Széchenyi István Egyetem

Logikai függvények egyszerűsítésének módszerei: Quine módszer (2)

Példa:

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

1.lépés:
egyszerűsítés

mintermek	*I.*	*II.*	*III.*
2	$\overline{A}B\overline{C}$ ✓	(2, 3) $\overline{A}B$	
3	$\overline{A}BC$ ✓	(2, 6) $B\overline{C}$	
4	$A\overline{B}\overline{C}$ ✓	(4, 6) $A\overline{C}$	
6	$AB\overline{C}$ ✓		

Széchenyi István Egyetem

Logikai függvények egyszerűsítésének módszerei: Quine módszer (3)

2.lépés:
feltétlenül szükséges prímimplikánsok kiválasztása

Prímimplikáns táblázat

		2	3	4	6
(2, 3)	$\boxed{\overline{A}B}$	⊛	⊛		
(2, 6)	$B\overline{C}$	*			*
(4, 6)	$\boxed{A\overline{C}}$			⊛	⊛

Logikai függvények egyszerűsítésének módszerei: Quine módszer (4)

3.lépés:
redundáns prímimplikáns(ok) kiválasztása és elhagyása, a végleges (egyszerűsített) függvényalak felírása

- redundáns prímimplikáns: $B\overline{C}$
- irredundáns prímimplikáns: $\overline{A}B,\quad A\overline{C}$

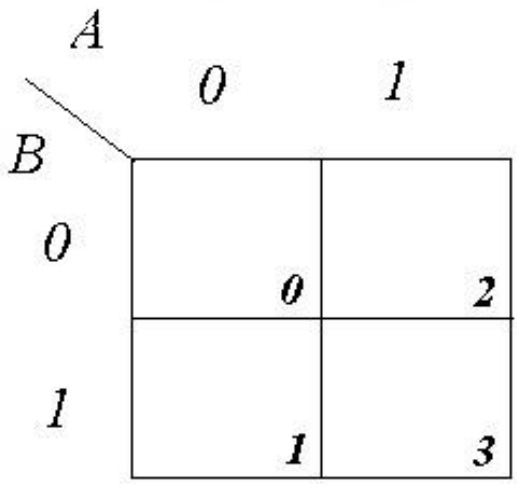
A végleges (legegyszerűbb) függvényalak:

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

Széchenyi István Egyetem

Logikai függvények egyszerűsítésének módszerei: Karnaugh táblás módszer (1)

- kétváltozós Karnaugh tábla:



Logikai függvények egyszerűsítésének módszerei: Karnaugh táblás módszer (2)

- háromváltozós Karnaugh tábla:

A	0	0	1	1
B	0	1	1	0
C				
0	0	2	6	4
1	1	3	7	5

Széchenyi István Egyetem

Logikai függvények egyszerűsítésének módszerei: Karnaugh táblás módszer (3)

- négyváltozós Karnaugh tábla:

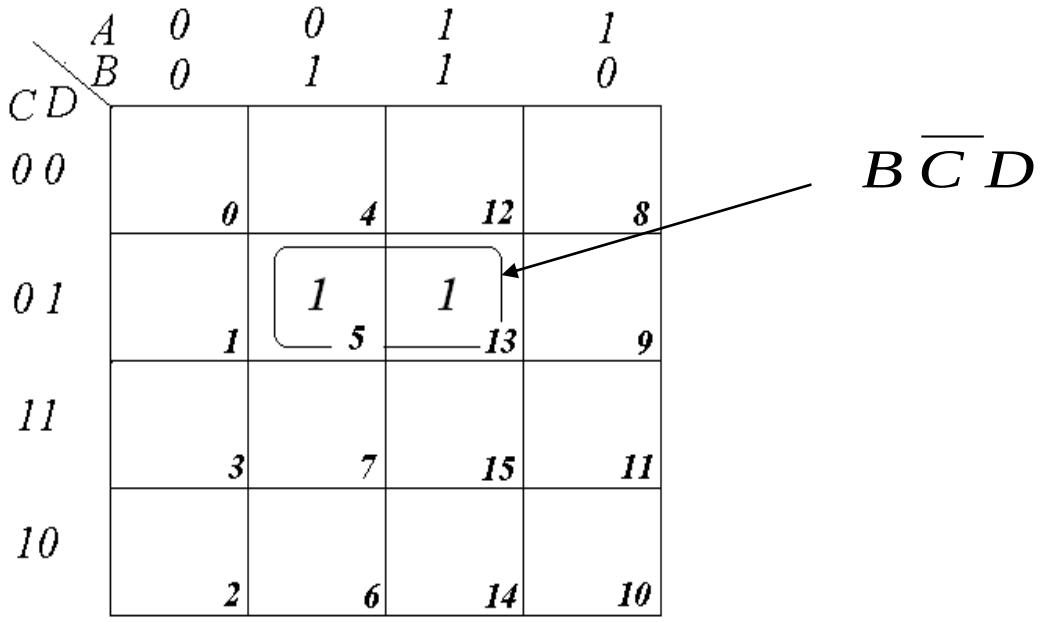
C D \ A B	0 0	0 1	1 1	1 0
0 0	0	4	12	8
0 1	1	5	13	9
1 1	3	7	15	11
1 0	2	6	14	10

Széchenyi István Egyetem

Logikai függvények egyszerűsítésének módszerei: Karnaugh táblás módszer (4)

Egyszerűsítés a Karnaugh táblán: szomszédos termek összevonása

Példa1.:

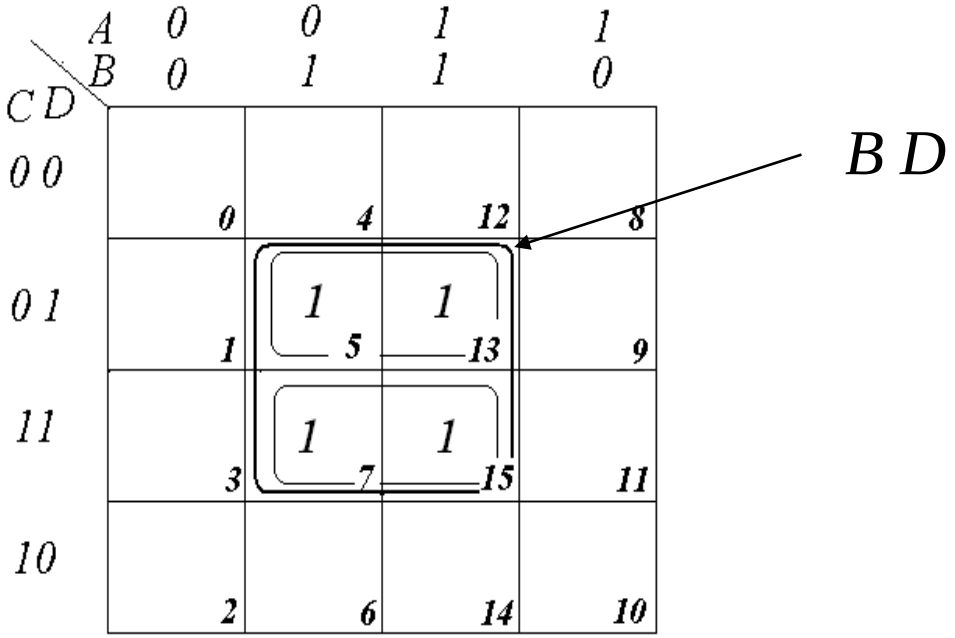


Széchenyi István Egyetem

Logikai függvények egyszerűsítésének módszerei: Karnaugh táblás módszer (5)

Egyszerűsítés a Karnaugh táblán: szomszédos termek összevonása

Példa2.:

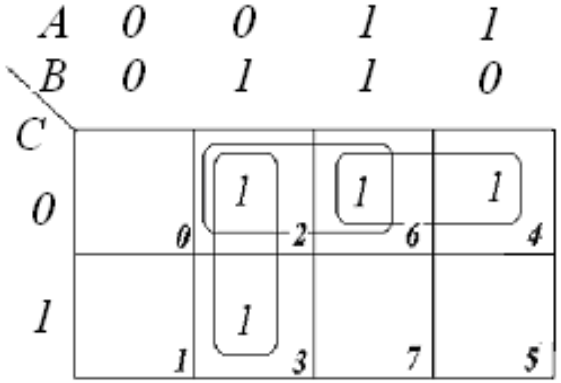


Széchenyi István Egyetem

Logikai függvények egyszerűsítésének módszerei: Karnaugh táblás módszer (6)

Teljesen specifikált hálózatok egyszerűsítése a Karnaugh táblán: szomszédos termek összevonása, prímimplikáns képzés

Példa:



prímimplikánsok:

$\overline{A}B,\quad B\overline{C},\quad A\overline{C}$

redundáns prímimplikáns

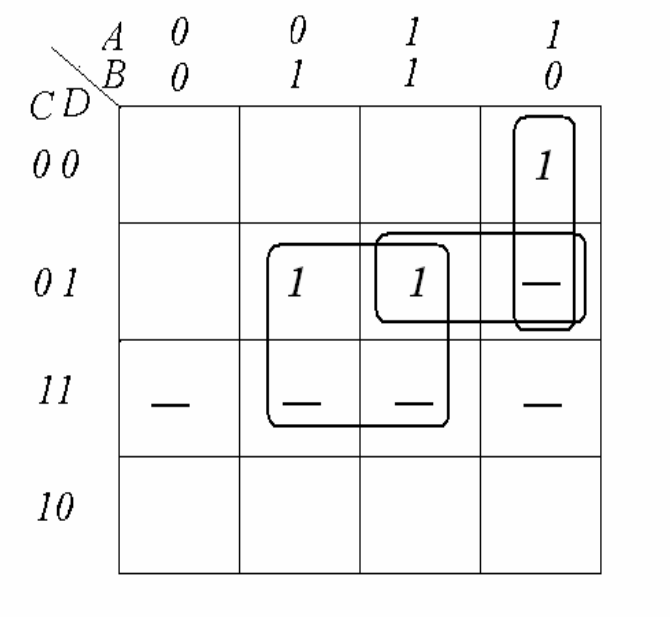
$$F(A,B,C) = \overline{A}B + A\overline{C}$$

Széchenyi István Egyetem

Logikai függvények egyszerűsítésének módszerei: Karnaugh táblás módszer (7)

Nem teljesen specifikált hálózatok egyszerűsítése a Karnaugh táblán: szomszédos termek összevonása, prímimplikáns képzés

Példa:



prímimplikánsok:

$$B\,D, \quad A\overline{C}\,D, \quad A\overline{B}\,\overline{C}$$

redundáns prímimplikáns

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

Egy- és többkimenetű kombinációs hálózatok tervezése

- *Egykimenetű kombinációs hálózatok tervezési lépései:*

1. *igazságtábla és/vagy Karnaugh tábla megadása,*
2. *függvény kiolvasás és egyszerűsítés a Karnaugh táblán (a minimalizálás céljából),*
3. *a rendelkezésre álló logikai építőelemekhez igazodó függvényalak felírás (átalakítás),*
4. *kapuszintű realizáció a rendelkezésre álló logikai építőelemek segítségével.*

Teljesen specifikált, egykimenetű kombinációs hálózatok tervezése - mintapélda

Tervezzük meg „NEM-ÉS” (NAND) kapukkal a következő specifikációval megadott négybemenetű (A,B,C,D) és egykimenetű (F), teljesen specifikált kombinációs hálózatot:

F(’1’) : (2, 4, 5, 6, 9, 10, 11, 12, 13, 14, 15) !

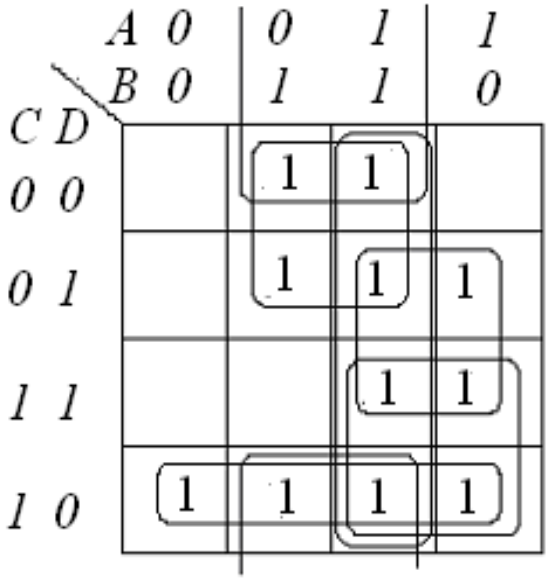
A bemenetek sorrendje: ABCD (MSB!)

Széchenyi István Egyetem

Teljesen specifikált, egykimenetű kombinációs hálózatok tervezése - mintapélda

Megoldás:

1.lépés: a Karnaugh tábla felvétele, és a megadott specifikáció szerinti kitöltése

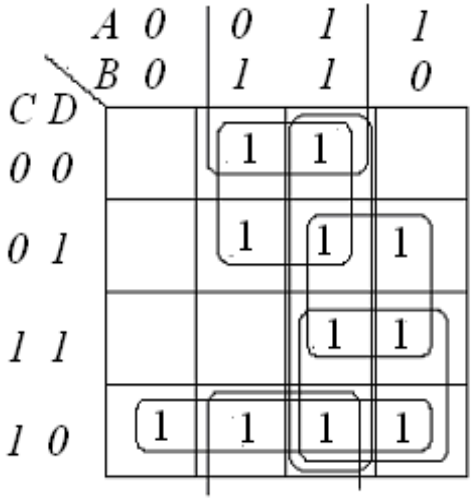


Széchenyi István Egyetem

Teljesen specifikált, egykimenetű kombinációs hálózatok tervezése - mintapélda

Megoldás:

2.lépés: függvény kiolvasás és egyszerűsítés a Karnaugh táblán (a minimalizálás céljából),



prímimplikánsok:

$$B\overline{D},\quad B\overline{C},\quad AD,\quad AC,\quad C\overline{D},\quad AB$$

irredundáns, szükséges és elégséges lefedés:

$$B\overline{C},\quad AD,\quad C\overline{D}$$

Széchenyi István Egyetem

Teljesen specifikált, egykimenetű kombinációs hálózatok tervezése - mintapélda

Megoldás:

3.lépés: a rendelkezésre álló logikai építőelemekhez igazodó (mintermes)függvényalak felírás (átalakítás)

Az irredundáns lefedés prímimplikánsai:

$$B\overline{C},\quad AD,\quad C\overline{D}$$

$$\downarrow$$

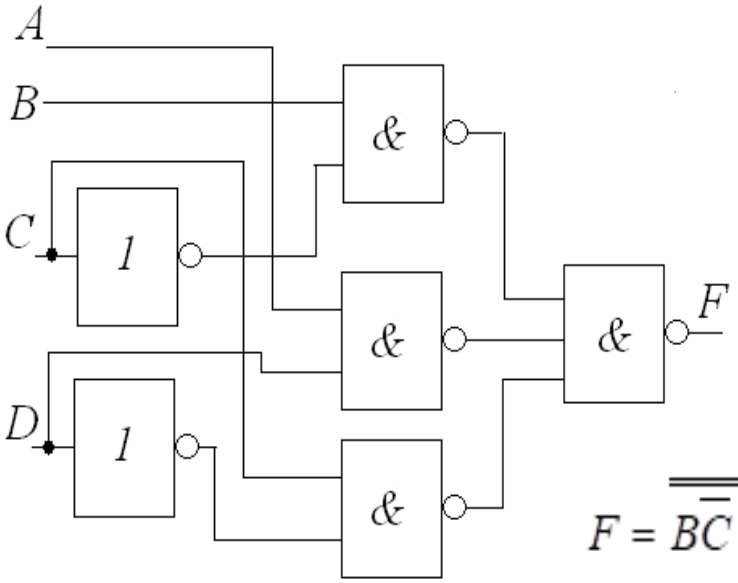
$$F = \overline{\overline{B\overline{C} + AD + C\overline{D}}} = \overline{\overline{B\overline{C}} \cdot \overline{AD} \cdot \overline{C\overline{D}}}$$

Széchenyi István Egyetem

Teljesen specifikált, egykimenetű kombinációs hálózatok tervezése - mintapélda

Megoldás:

4.lépés: kapuszintű realizáció a rendelkezésre álló logikai építőelemek segítségével



$$F = \overline{\overline{B\bar{C}} \cdot \overline{AD} \cdot \overline{C\bar{D}}} = B\bar{C} + AD + C\bar{D}$$

Nem teljesen specifikált, egykimenetű kombinációs hálózatok tervezése – mintapélda1.

Tervezzük meg „NEM-ÉS” (NAND) kapukkal a következő specifikációval megadott négybemenetű (A,B,C,D) és egykimenetű (F), nem teljesen specifikált kombinációs hálózatot:

F(’1’) : (2, 4, 5, 9, 10, 11, 12, 14, 15)

F(’-’): (0, 6, 13)

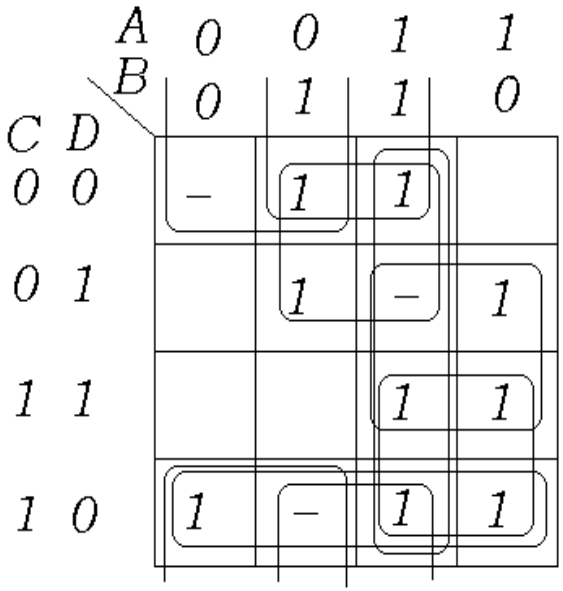
A bemenetek sorrendje: ABCD (MSB!)

Széchenyi István Egyetem

Nem teljesen specifikált, egykimenetű kombinációs hálózatok tervezése – mintapélda1.

Megoldás:

1.lépés: a Karnaugh tábla felvétele, és a megadott specifikáció szerinti kitöltése

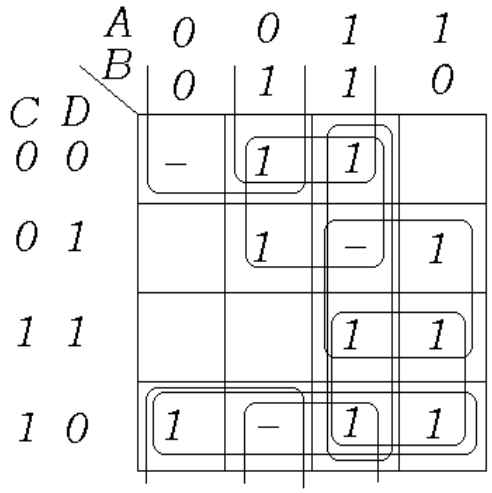


Széchenyi István Egyetem

Nem teljesen specifikált, egykimenetű kombinációs hálózatok tervezése – mintapélda1.

Megoldás:

2.lépés: függvény kiolvasás és egyszerűsítés a Karnaugh táblán (a minimalizálás céljából),



prímimplikánsok:

$\overline{A}\overline{D}$, $B\overline{D}$, $B\overline{C}$, AD, AC, $C\overline{D}$, AB

irredundáns, szükséges és elégséges lefedés:

$B\overline{C}$, AD, $C\overline{D}$

Széchenyi István Egyetem

Nem teljesen specifikált, egykimenetű kombinációs hálózatok tervezése – mintapélda1.

Megoldás:

3.lépés: a rendelkezésre álló logikai építőelemekhez igazodó (mintermes)függvényalak felírás (átalakítás)

Az irredundáns lefedés prímimplikánsai:

$$B\overline{C},\quad AD,\quad C\overline{D}$$

$$\downarrow$$

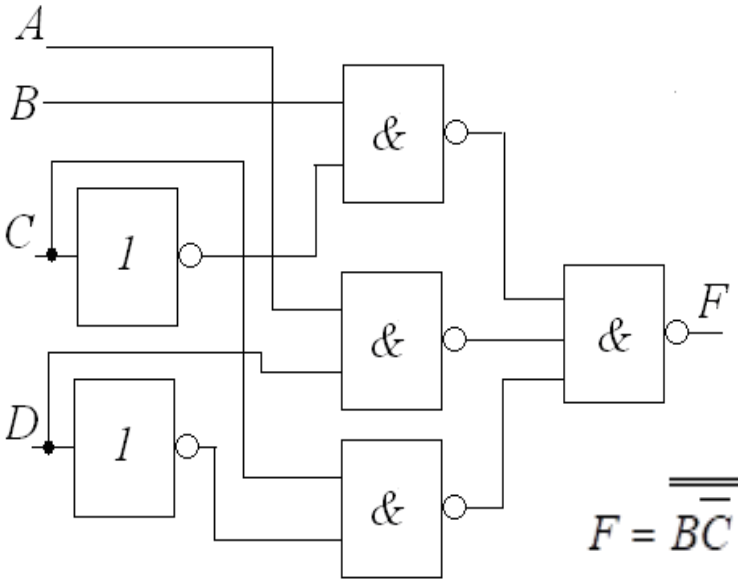
$$F = \overline{\overline{B\overline{C} + AD + C\overline{D}}} = \overline{\overline{B\overline{C}} \cdot \overline{AD} \cdot \overline{C\overline{D}}}$$

Széchenyi István Egyetem

Nem teljesen specifikált, egykimenetű kombinációs hálózatok tervezése – mintapélda1.

Megoldás:

4.lépés: kapuszintű realizáció a rendelkezésre álló logikai építőelemek segítségével

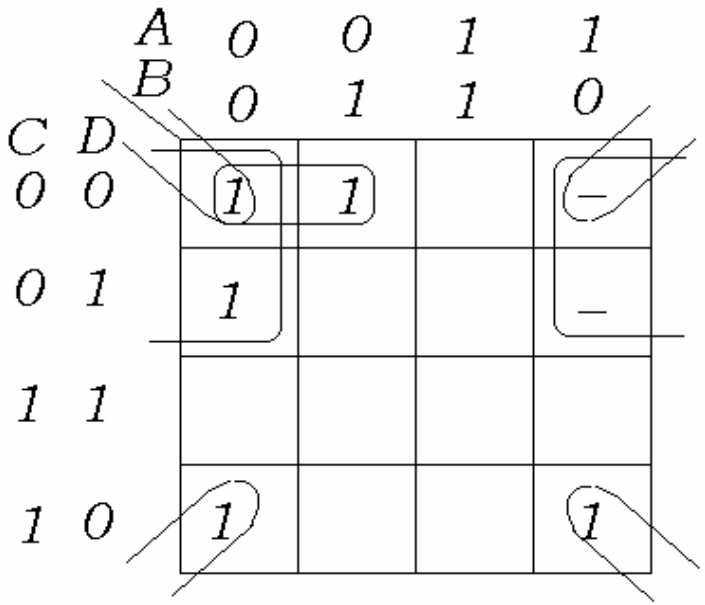


$$F = \overline{\overline{B\bar{C}} \cdot \overline{AD} \cdot \overline{C\bar{D}}} = B\bar{C} + AD + C\bar{D}$$

Széchenyi István Egyetem

Nem teljesen specifikált, egykimenetű kombinációs hálózatok tervezése – mintapélda2.

Példa egy, az alábbi táblázat által specifikált hálózat függvény minimalizálása céljából elvégzett minterm összevonás bemutatására:



Az irredundáns lefedéshez szükséges prímimplikánsok:

$$\overline{C}\,\overline{B}, \quad \overline{A}\,\overline{C}\,\overline{D}, \quad \overline{B}\,\overline{D}$$

- *Többkimenetű kombinációs hálózatok tervezési lépései:*

1. *Karnaugh tábla segítségével felírandó valamennyi kimenethez rendelt függvény prímimplikánsa,*
2. *az összes lehetséges függvény-szorzat prímimplikánsainak megadása,*
3. *az egyes kimeneti függvények mintermjeinek lefedése a saját, más kimenetekhez nem tartozó prímimplikánsokkal, illetve azokkal a maximális közös implikánsokkal, amelyek az adott függvénynek implikánsai*

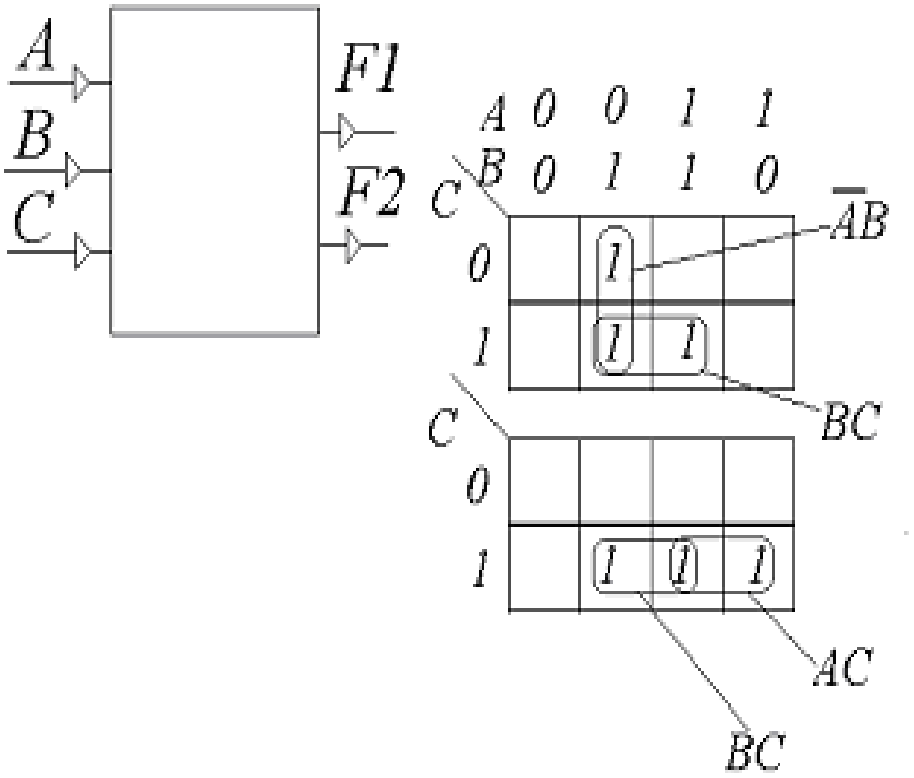
Többkimenetű kombinációs hálózatok tervezése – mintapélda1.

Adott egy három(*A,B,C*) bemenetű és két(*F1, F2*) kimenetű kombinációs hálózat az alábbi Karnaugh táblák szerinti specifikációban.

Tervezzük meg a hálózatot a legegyszerűbb változatban!

A bemenetek sorrendje: ABCD (MSB!)

Többkimenetű kombinációs hálózatok tervezése – mintapélda1.



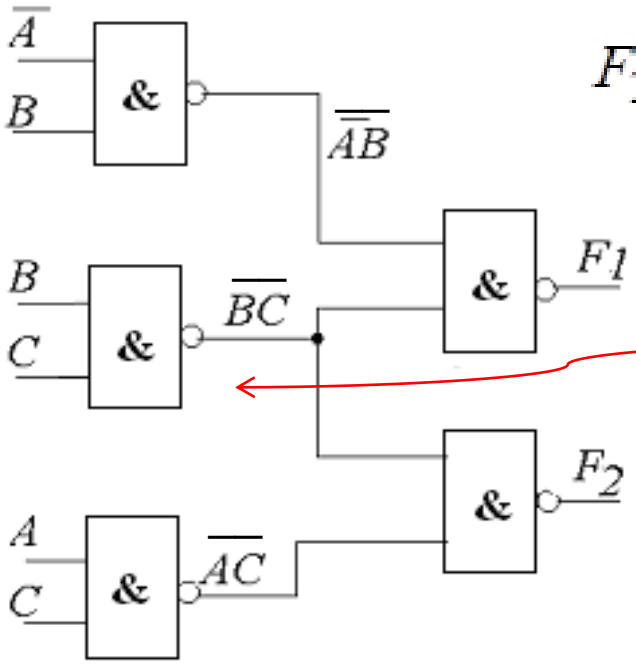
Széchenyi István Egyetem

Többkimenetű kombinációs hálózatok tervezése – mintapélda1.

Megoldás (közös prímplikáns keresés):

$$F_1 = \overline{A}\,B\,\overline{C} + \overline{A}\,B\,C + A\,B\,C = \overline{A}\,B + B\,C$$

$$F_2 = \overline{A}\,B\,C + A\,\overline{B}\,C + A\,B\,C = A\,C + B\,C$$



BC csak egyszer!!!!

Többkimenetű kombinációs hálózatok tervezése – mintapélda2.

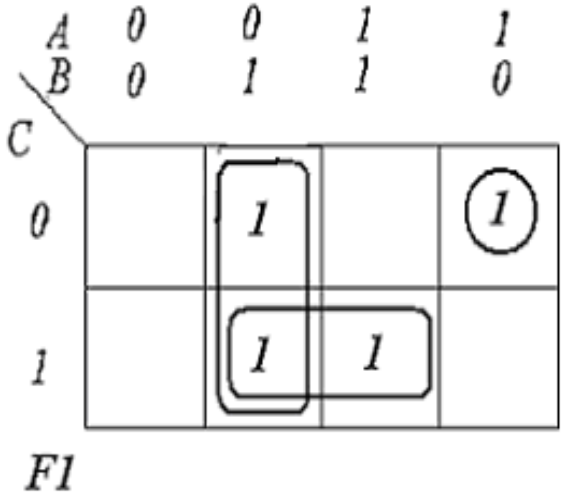
Adott egy három(*A,B,C*) bemenetű és két(*F1, F2*) kimenetű kombinációs hálózat az alábbi Karnaugh táblák szerinti specifikációban!

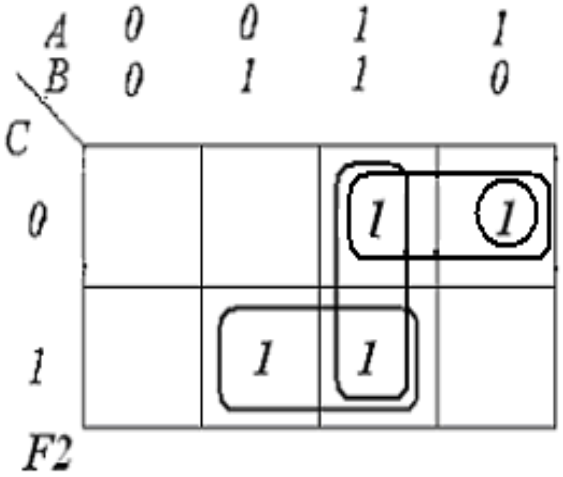
Tervezzük meg a hálózatot a legegyszerűbb változatban!

A bemenetek sorrendje: ABCD (MSB!)

Széchenyi István Egyetem

Többkimenetű kombinációs hálózatok tervezése – mintapélda2.

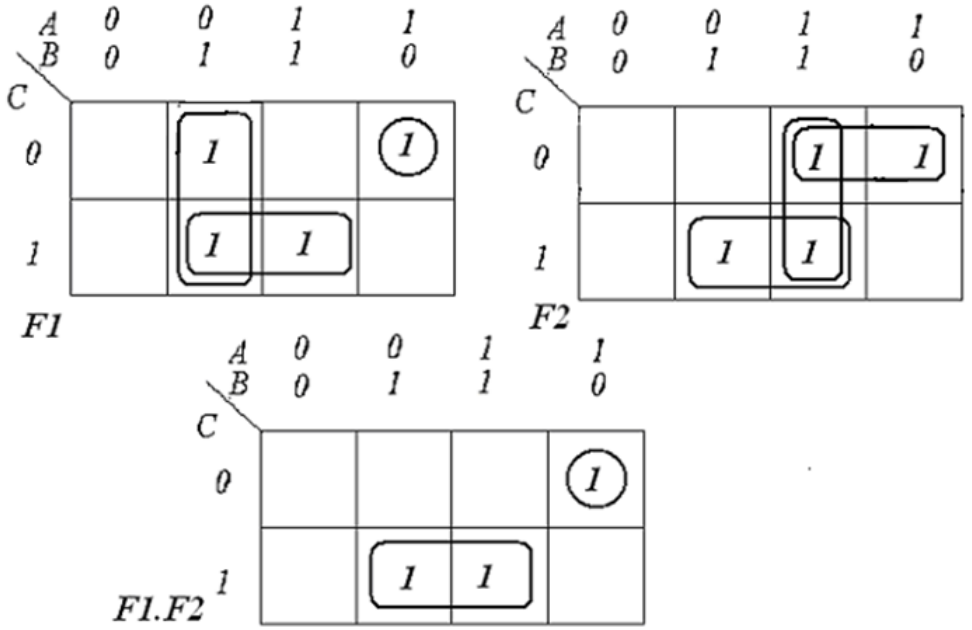




Széchenyi István Egyetem

Többkimenetű kombinációs hálózatok tervezése – mintapélda2.

Megoldás (közös implikáns keresés):



Függvény-szorzat prímimplikánsainak előállítása:

$$F_1: \quad \overline{A}B, \quad BC, \quad A\overline{B}\,\overline{C}$$

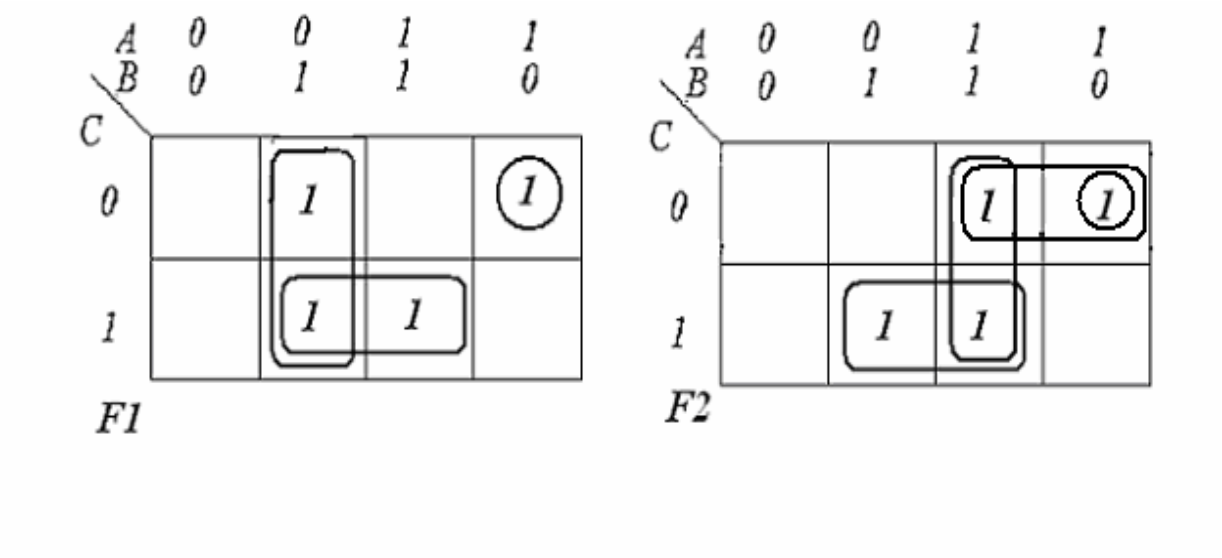
$$F_2: \quad BC, \quad AB, \quad A\overline{C}$$

$$F_1 \cdot F_2: \quad BC, \quad A\overline{B}\,\overline{C}$$

Széchenyi István Egyetem

Többkimenetű kombinációs hálózatok tervezése – mintapélda2.

Megoldás (közös implikáns keresés):



$A\overline{C}$ ***helyett*** $A\overline{B}\overline{C}$!!

Hazárdok kombinációs hálózatokban

A hazárdok azok az eltérések az ideális, késleltetés nélküli hálózatok viselkedésétől, amelyek a logikai kapuk időbeli késleltetéséből adódnak.

- *Típusai*
  - *Statikus hazárdok*
  - *Dinamikus hazárdok*
  - *Funkcionális hazárdok*
  - *Lényeges hazárdok (későbbi fejezet)*

- **Statikus hazárd**

Ha egyetlen bemeneti változó logikai értékének megváltozásakor a kimenet a specifikáció szerint nem változna, de a realizált hálózat kimenetén mégis átmeneti változás zajlik le, statikus hazárdról beszélünk.

❖ *Típusai*

- *‚0'-ás típusú statikus hazárd*
- *‚1'-es típusú statikus hazárd*

‚0'-ás típusú statikus hazárd:

- a specifikált hálózat kimenete a bemeneti változás ellenére alacsony logikai szinten ('0') kell, hogy maradjon, de a realizált hálózat - átmenetileg - egy magas, '1'-es logikai szintű impulzust mutat.

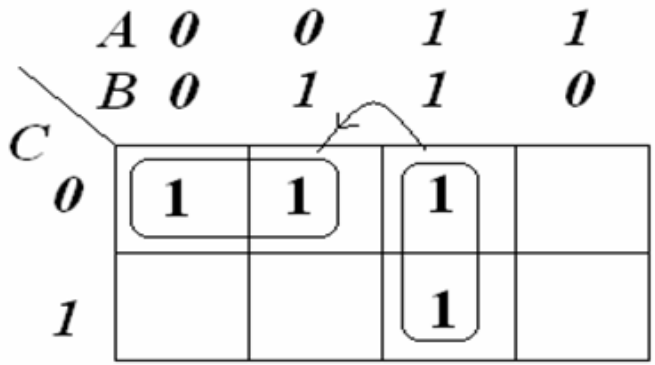
‚1'-es típusú statikus hazárd:

- a specifikált hálózat kimenete a bemeneti változás ellenére magas logikai szinten ('1') kell, hogy maradjon, de a realizált hálózat - átmenetileg - egy alacsony, '0'-ás logikai szintű impulzust mutat.

Statikus hazárd keletkezése (1)

Példa

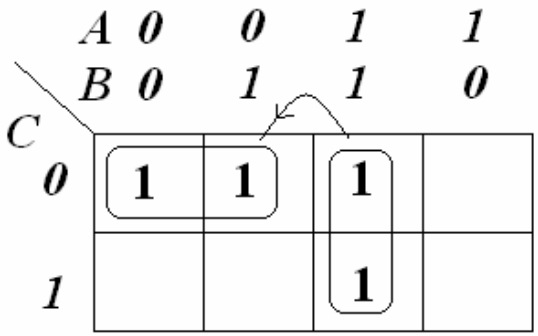
Vizsgáljuk meg az alábbi, grafikus sémával megadott kombinációs hálózat kapuszintű viselkedését az *ABC* bemenetek *1 1 0 → 0 1 0* bitkombináció változása esetén!



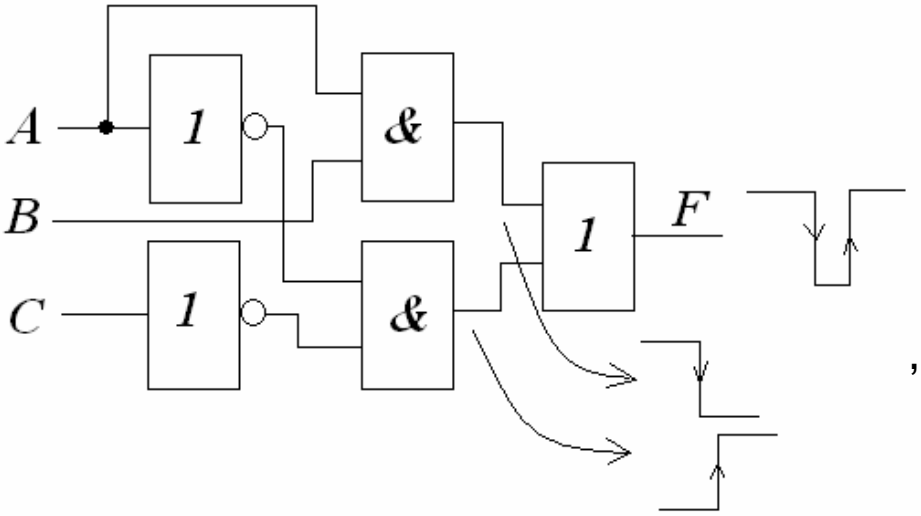
$$F(A,B,C) = \overline{A}\,\overline{C} + AB$$

Széchenyi István Egyetem

Statikus hazárd keletkezése (2)



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

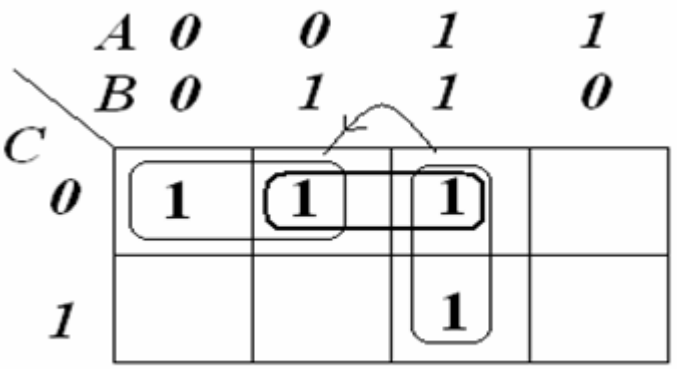


‚1'-es típusú statikus hazárd

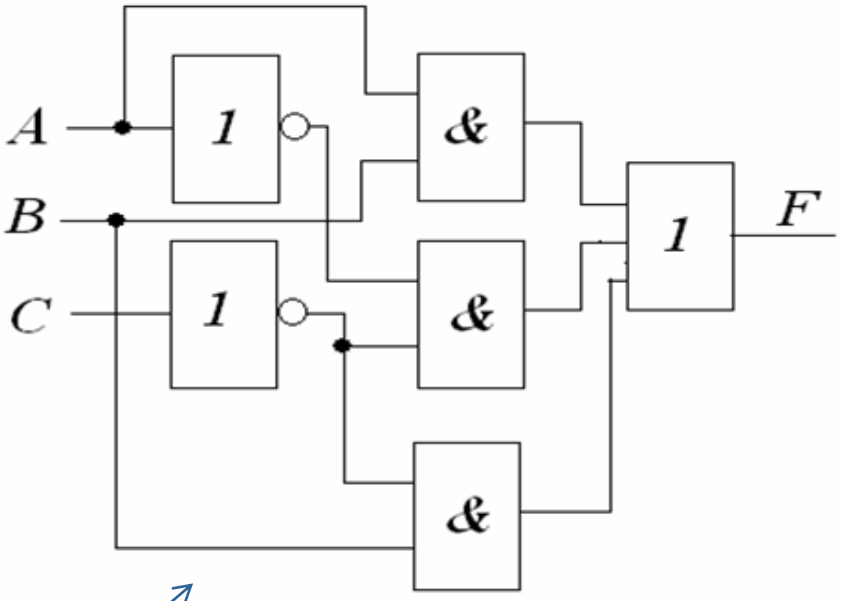
Széchenyi István Egyetem

Statikus hazárd kiküszöbölése

Megoldás: redundáns (‚átfedő') prímimplikáns alkalmazása a kritikus átmenet helyén



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



redundáns term, de megszünteti a hazárdot

- **Dinamikus hazárd:**

  az adott hálózat kimenetének szintet *kell* váltania, de ezt kétszer teszi.
  Kiküszöbölés: a statikus hazárdok megszüntetésével.

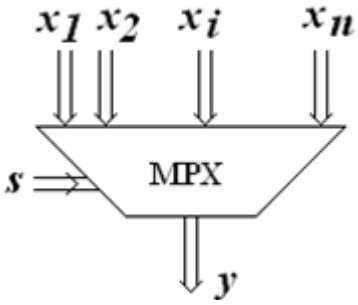
- **Funkcionális hazárd:**

  több bemeneti változó együttes változásakor a kimeneten vagy a specifikációtól eltérő szintváltás, vagy többszörös szintváltás jelentkezik.
  Kiküszöbölés: szinkronizációval.

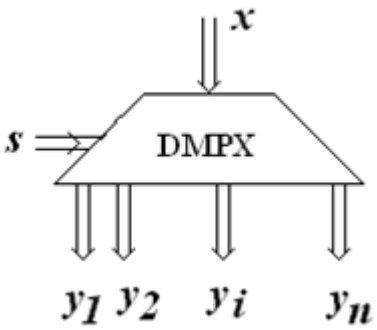
Programozható kombinációs hálózatok (1)

Az adatút kijelölő eszközök, mint programozható kombinációs hálózatok:

- *multiplexerek (MPX,MUX)*



- *demultiplexerek (DMPX,DEMUX)*

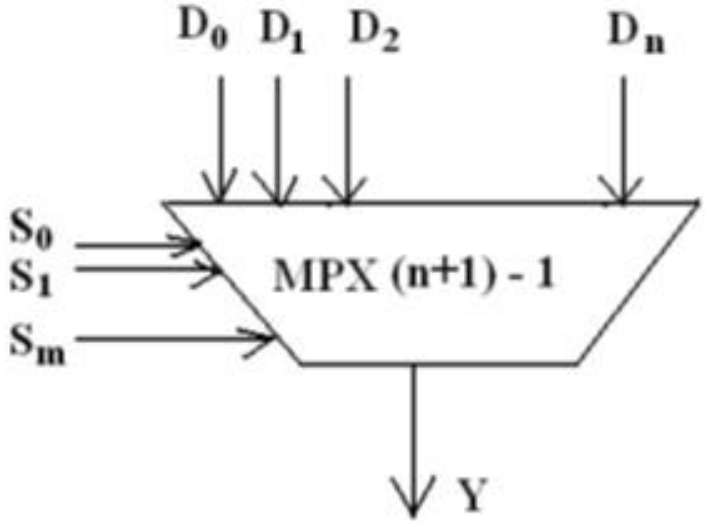


Széchenyi István Egyetem

Programozható kombinációs hálózatok (2):

logikai függvények megvalósítása bit-szervezésű multiplexerekkel

Általános séma:



$n+1$: 2 egész számú hatványa

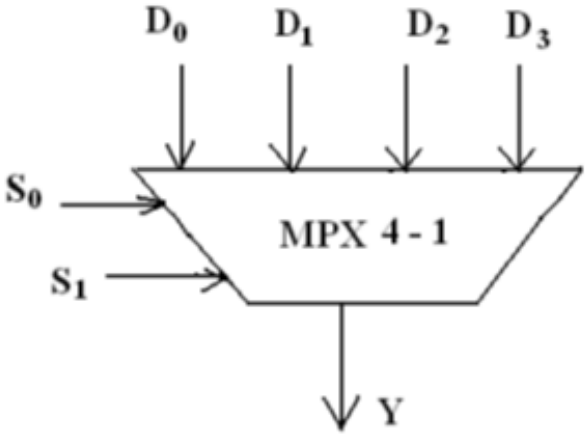
$m = \overset{2}{\smile}\log(n+1)$

Széchenyi István Egyetem

Programozható kombinációs hálózatok (3):

logikai függvények megvalósítása bit-szervezésű multiplexerekkel

Példa: MPX4-1 multiplexer sémája, és a működést leíró igazságtáblája



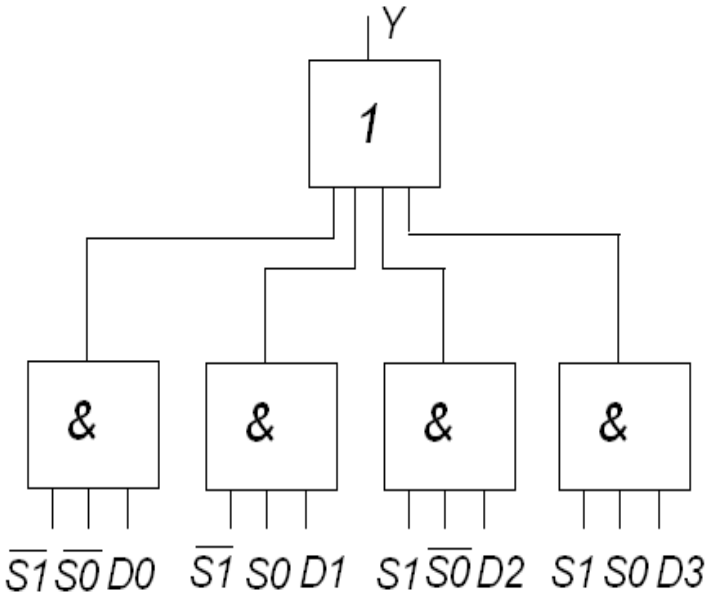
S1	S0	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

Széchenyi István Egyetem

Programozható kombinációs hálózatok (4):

logikai függvények megvalósítása bit-szervezésű multiplexerekkel

Példa: az MPX4-1 multiplexer kapuszintű felépítése



Programozható kombinációs hálózatok (5):

logikai függvények megvalósítása bit-szervezésű multiplexerekkel

A multiplexer, mint programozható egykimenetű kombinációs hálózat:

a függvény mintermjeit a címző bemenetekre adott címek képviselik, és a megcímzett adatbemenetre rá kell kapcsolnunk az adott mintermhez tartozó logikai értéket.

Ezeknek a logikai konstansoknak a bemenetekre való kapcsolását a multiplexer programozásának tekinthetjük.

Programozható kombinációs hálózatok (6):

logikai függvények megvalósítása bit-szervezésű multiplexerekkel

Példa:

Valósítsuk meg multiplexer segítségével (programozásával) az EXOR függvény kapuszintű realizációját!

$$F_{(EXOR)} = \bar{A}B + A\bar{B}$$

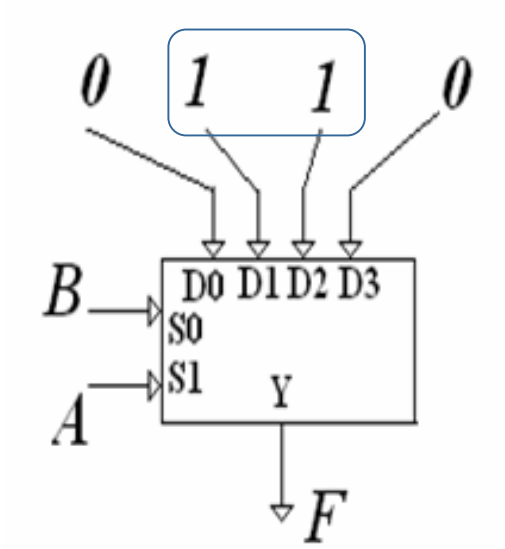
Széchenyi István Egyetem

Programozható kombinációs hálózatok (7):

logikai függvények megvalósítása bit-szervezésű multiplexerekkel

Megoldás:

A	B	F	
0	0	0	(D0)
0	1	1	(D1)
1	0	1	(D2)
1	1	0	(D3)



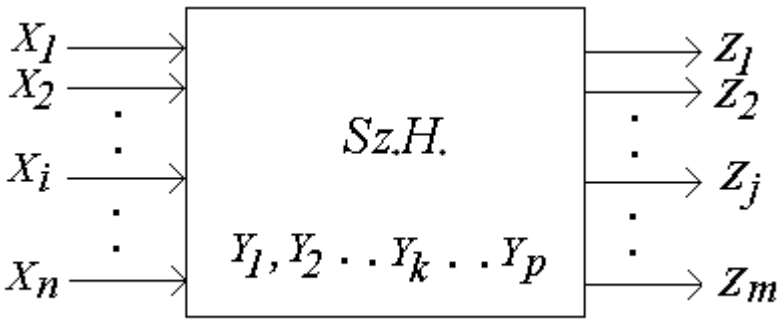
$F_{(EXOR)} = \bar{A}B + A\bar{B}$

Sorrendi (szekvenciális) hálózatok

Definíció: *a sorrendi (szekvenciális) hálózatok legfontosabb tulajdonsága, hogy a kimeneti érték (kombináció) nem csak a pillanatnyi bemeneti értéktől (kombinációtól) függ, hanem a korábbi bemeneti értéktől (kombinációtól), sőt azok sorrendjétől is.*

Széchenyi István Egyetem

A sorrendi hálózatok fekete-doboz modellje (1)



$$Z_1 = f_{z_1}(X_1, \ldots X_i, \ldots X_n, Y_1, \ldots Y_k, \ldots Y_p)$$
$$Z_2 = f_{z2}(X_1, \ldots X_i, \ldots X_n, Y_1, \ldots Y_k, \ldots Y_p)$$
$$\ldots\ldots\ldots\ldots\ldots\ldots$$
$$Z_j = f_{zj}(X_1, \ldots X_i, \ldots X_n, Y_1, \ldots Y_k, \ldots Y_p)$$
$$\ldots\ldots\ldots\ldots\ldots\ldots$$
$$Z_m = f_{zm}(X_1, \ldots X_i, \ldots X_n, Y_1, \ldots Y_k, \ldots Y_p)$$

$$f_z : \mathbf{X} \times \mathbf{Y} \Rightarrow \mathbf{Z}$$
$$f_y : \mathbf{X} \times \mathbf{Y} \Rightarrow \mathbf{Y}$$

$\mathbf{X}$: *a bemeneti kombinációk halmaza*

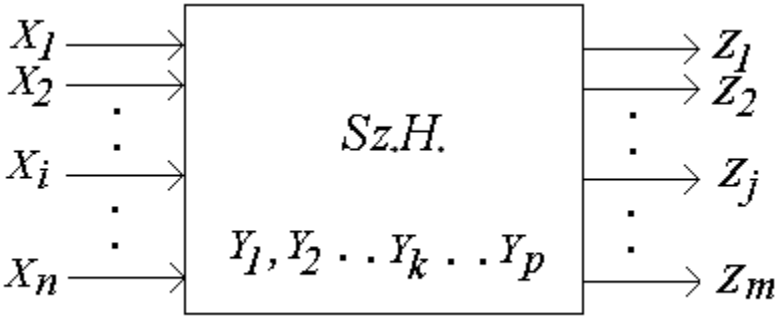
$\mathbf{Z}$: *a kimeneti kombinációk halmaza*

$\mathbf{Y}$: *a szekundér változók halmaza*

$$Y'_1 = f_{y1}(X_1, \ldots X_i, \ldots X_n, Y_1, \ldots Y_k, \ldots Y_p)$$
$$Y'_2 = f_{y2}(X_1, \ldots X_i, \ldots X_n, Y_1, \ldots Y_k, \ldots Y_p)$$
$$\ldots\ldots\ldots\ldots\ldots\ldots$$
$$Y'_k = f_{yk}(X_1, \ldots X_i, \ldots X_n, Y_1, \ldots Y_k, \ldots Y_p)$$
$$\ldots\ldots\ldots\ldots\ldots\ldots$$
$$Y'_p = f_{yp}(X_1, \ldots X_i, \ldots X_n, Y_1, \ldots Y_k, \ldots Y_p)$$

Széchenyi István Egyetem

A sorrendi hálózatok fekete-doboz modellje (2)



$$f_y : (x_i^t, y_k^t) \rightarrow y_l^{t+\Delta t}$$

$$f_z : (x_i^t, y_k^t) \rightarrow z_j^t$$

x_i^t : *egy bemeneti kombinációa t pillanatban*

z_j^t : *egy kimeneti kombinációa t pillanatban*

y_k^t : *egy szekundér változó kombinációa t pillanatban*

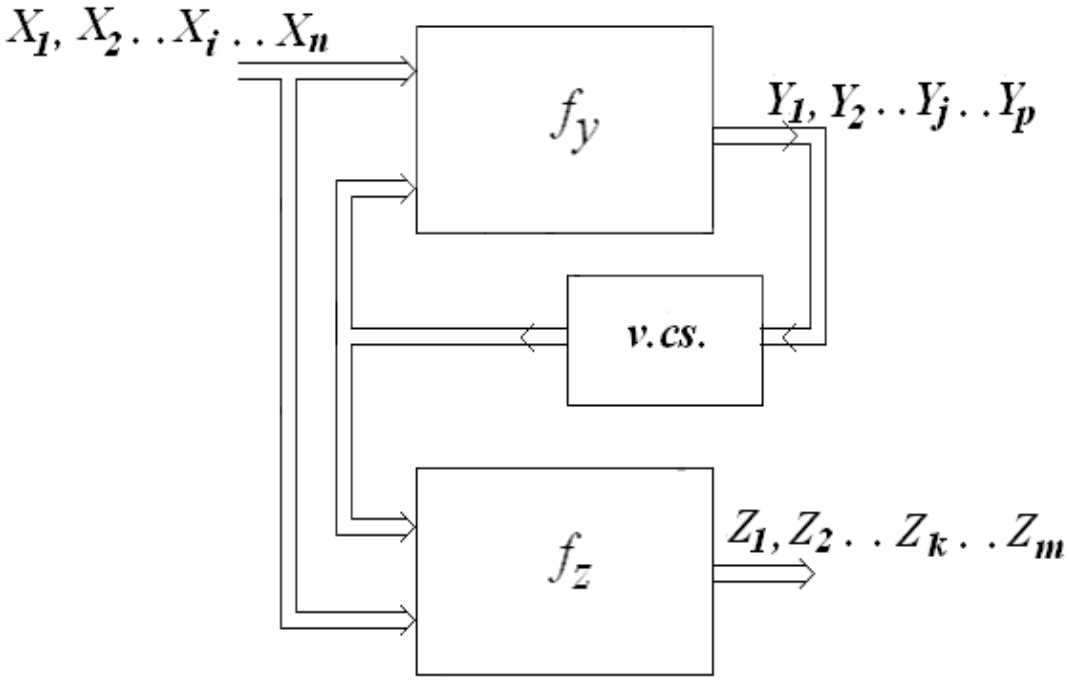
$y_l^{t+\Delta t}$: *egy szekundér változó kombinációa* $(t + \Delta t)$ *pillanatban*

Sorrendi hálózatok strukturális modelljei(1)

- *Mealy-típusú szekvenciális hálózatok*
  - *Szinkron*
  - *Aszinkron*
- *Moore-típusú szekvenciális hálózatok*
  - *Szinkron*
  - *Aszinkron*

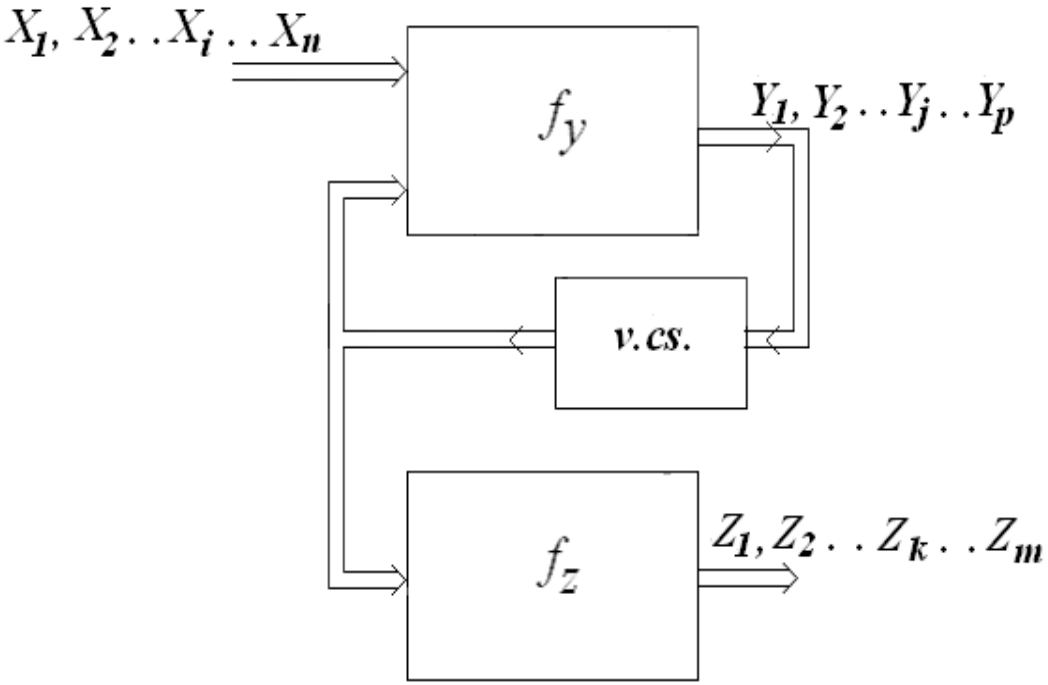
Sorrendi hálózatok strukturális modelljei(2)

Mealy-típusú szekvenciális hálózatok



Sorrendi hálózatok strukturális modelljei(3)

Moore-típusú szekvenciális hálózatok



Sorrendi hálózatok csoportosítása

- *Szinkron szekvenciális hálózatok*
  - *tárolóval („MS”) visszacsatolt szinkron sorrendi hálózatok*

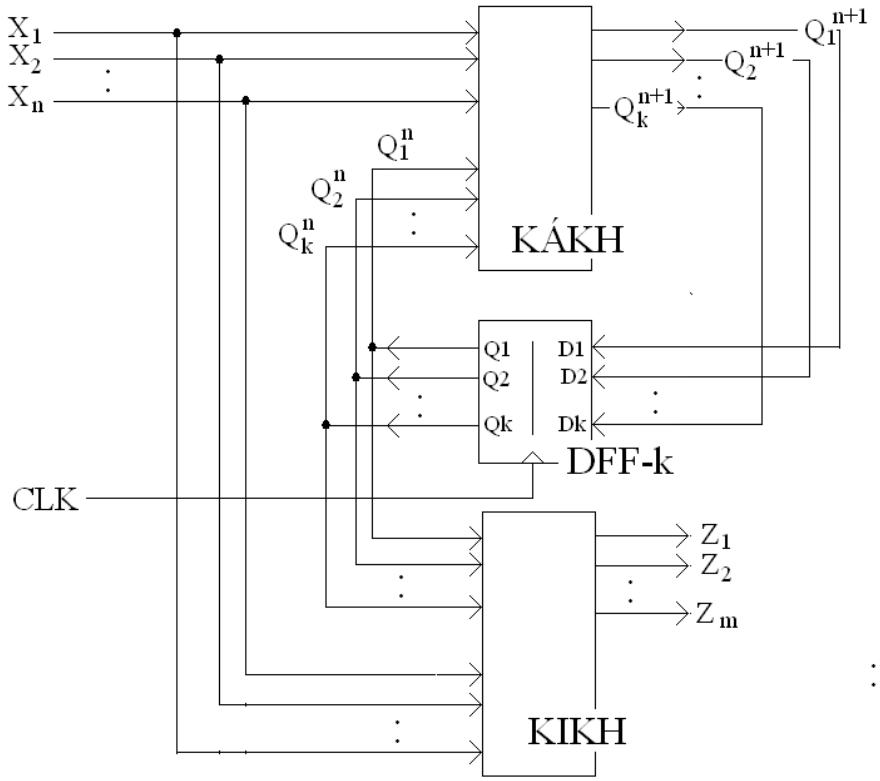
- *Aszinkron szekvenciális hálózatok*
  - *közvetlen visszacsatolásos aszinkron sorrendi hálózatok*
  - *tárolóval visszacsatolt aszinkron sorrendi hálóztok*

Széchenyi István Egyetem

Sorrendi hálózatok felépítése és működése (1)

(általános sémák)

Példa: szinkron MEALY-hálózat, D-MS tárolós visszacsatoló ágakkal



‚KÁKH': következő állapotot előállító kombinációs hálózat

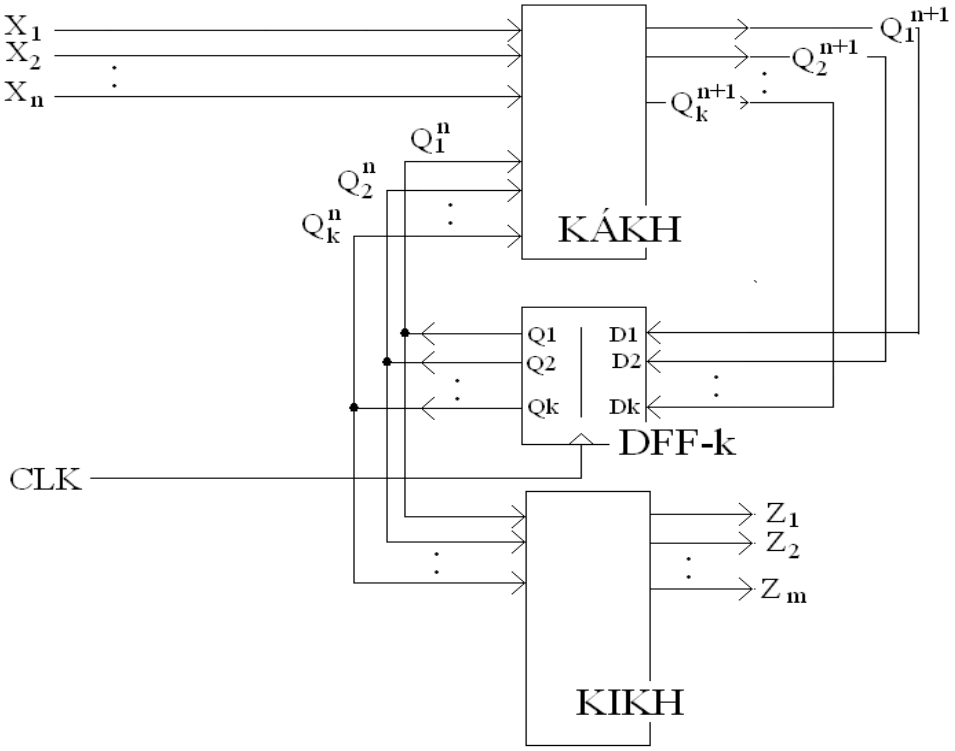
‚KIKH': kimenetet előállító kombinációs hálózat

Széchenyi István Egyetem

Sorrendi hálózatok felépítése és működése (2)

(általános sémák)

Példa: szinkron MOORE-hálózat, D-MS tárolós visszacsatoló ágakkal



‚KÁKH': következő állapotot előállító kombinációs hálózat

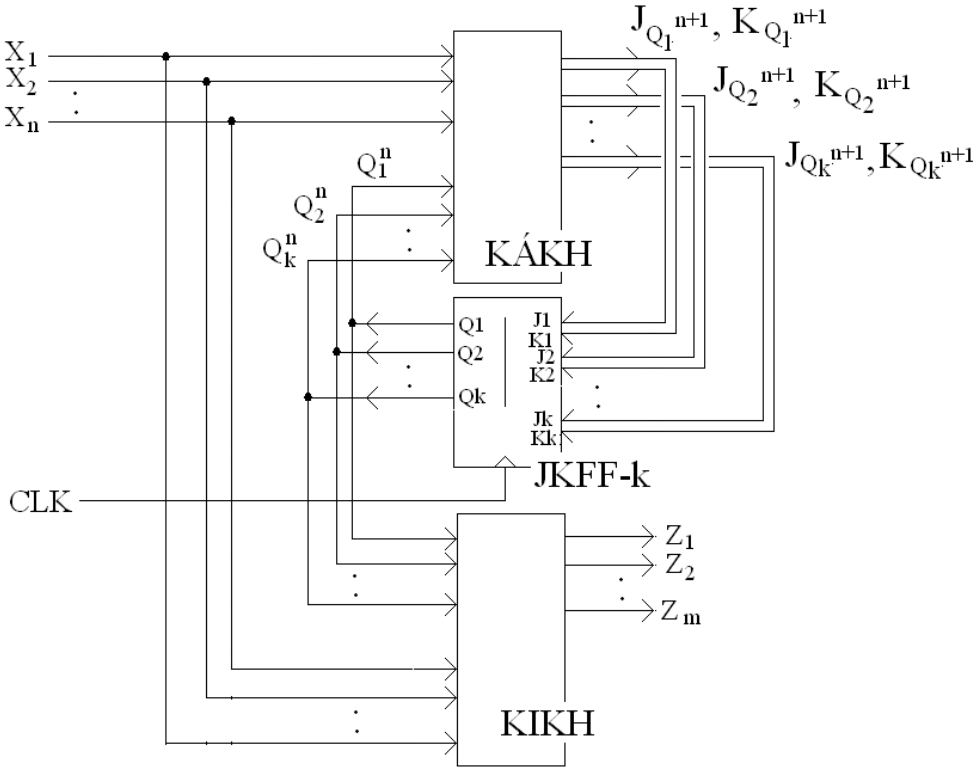
‚KIKH': kimenetet előállító kombinációs hálózat

Széchenyi István Egyetem

Sorrendi hálózatok felépítése és működése (3)

(általános sémák)

Példa: szinkron MEALY-hálózat, JK-MS tárolós visszacsatoló ágakkal



‚KÁKH': következő állapotot előállító kombinációs hálózat

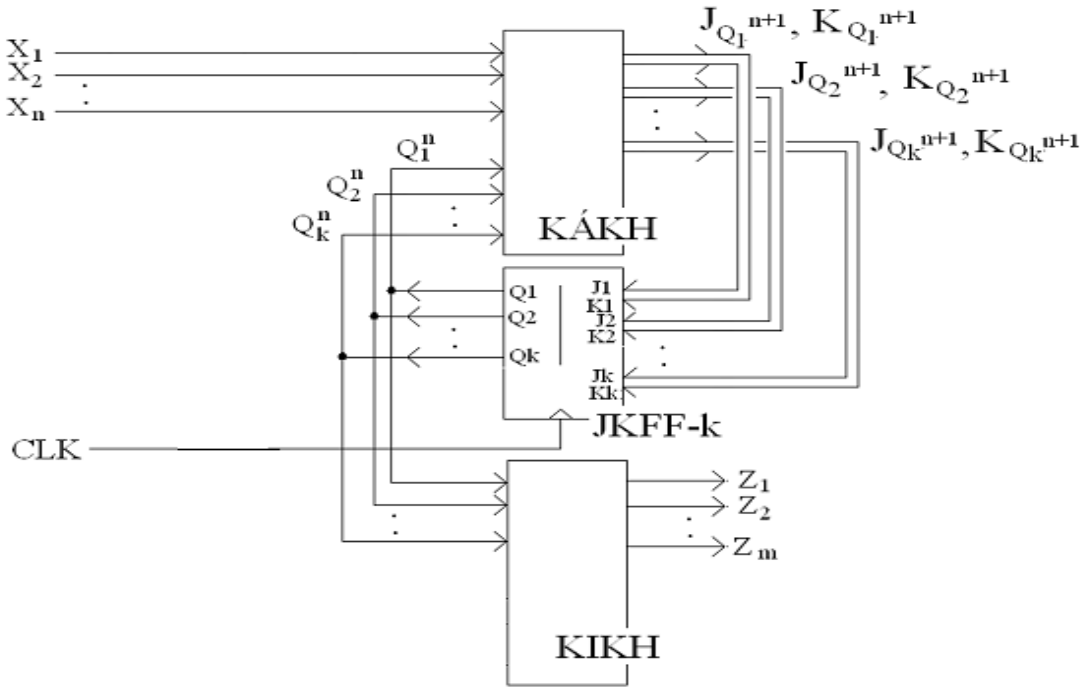
‚KIKH': kimenetet előállító kombinációs hálózat

Széchenyi István Egyetem

Sorrendi hálózatok felépítése és működése (4)

(általános sémák)

Példa: szinkron MOORE-hálózat, JK-MS tárolós visszacsatoló ágakkal



‚KÁKH': következő állapotot előállító kombinációs hálózat

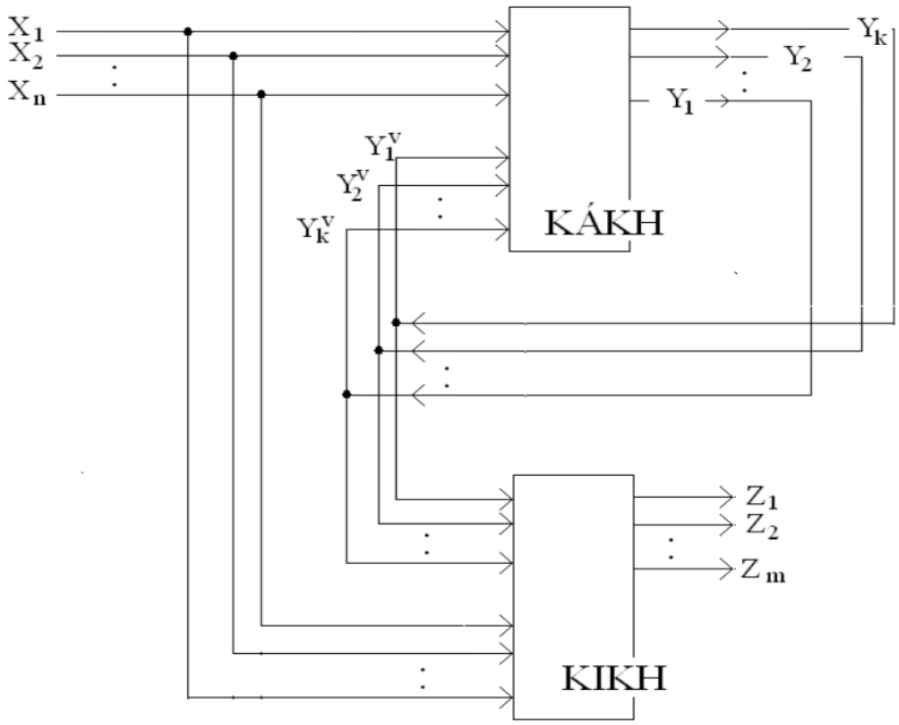
‚KIKH': kimenetet előállító kombinációs hálózat

Széchenyi István Egyetem

Sorrendi hálózatok felépítése és működése (5)

(általános sémák)

Példa: aszinkron MEALY-hálózat, közvetlen visszacsatolásokkal



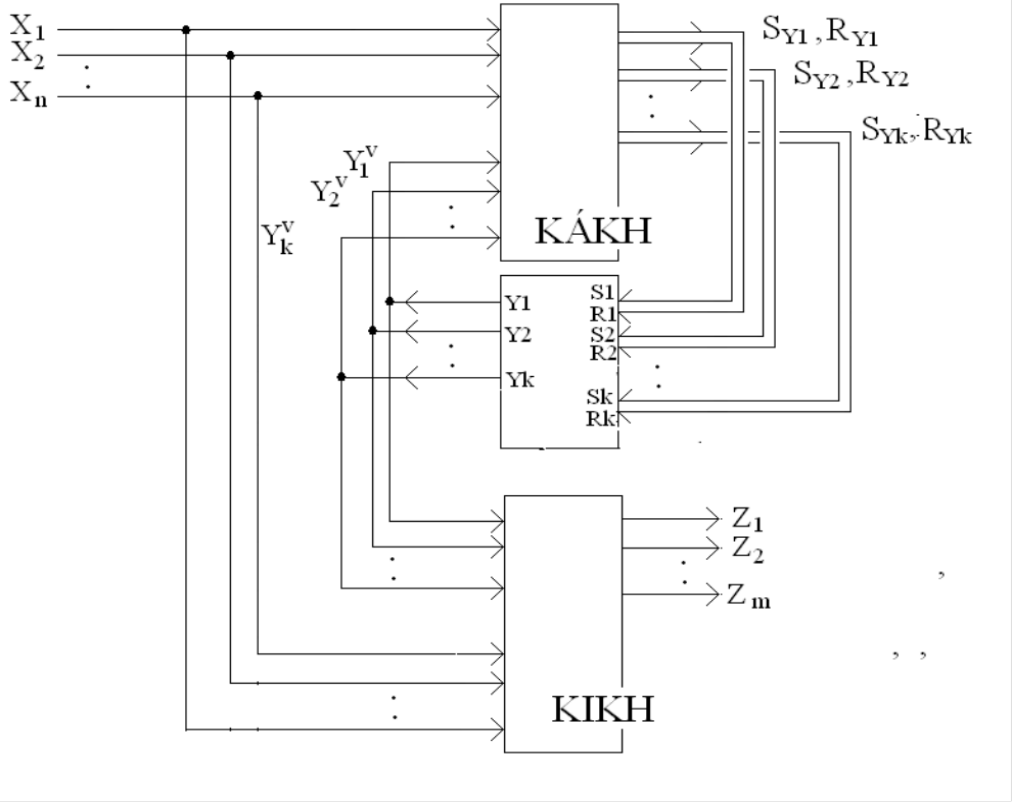
‚KÁKH': következő állapotot előállító kombinációs hálózat

‚KIKH': kimenetet előállító kombinációs hálózat

Sorrendi hálózatok felépítése és működése (6)

(általános sémák)

Példa: aszinkron MEALY-hálózat, SR tárolós visszacsatolásokkal



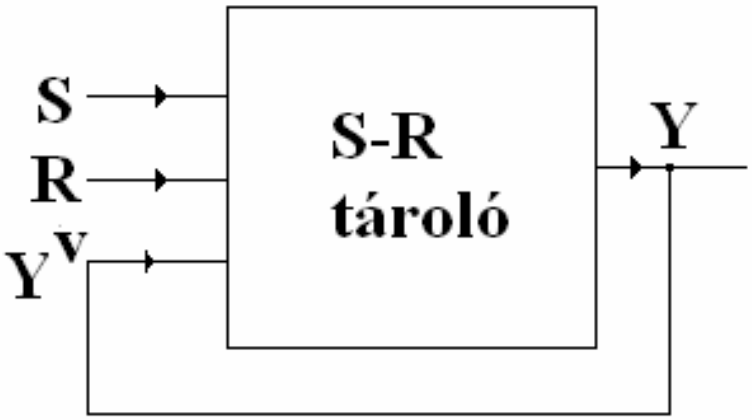
‚KÁKH': következő állapotot előállító kombinációs hálózat

‚KIKH': kimenetet előállító kombinációs hálózat

Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: SR tároló (1)

Általános séma:



- kombinációs hálózat, amelynek kimenete a bemenetre érkezik vissza

Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: SR tároló (2)

egyszerű igazságtábla

S	R	Y
0	0	Y^V
0	1	0
1	0	1
1	1	-

⇒

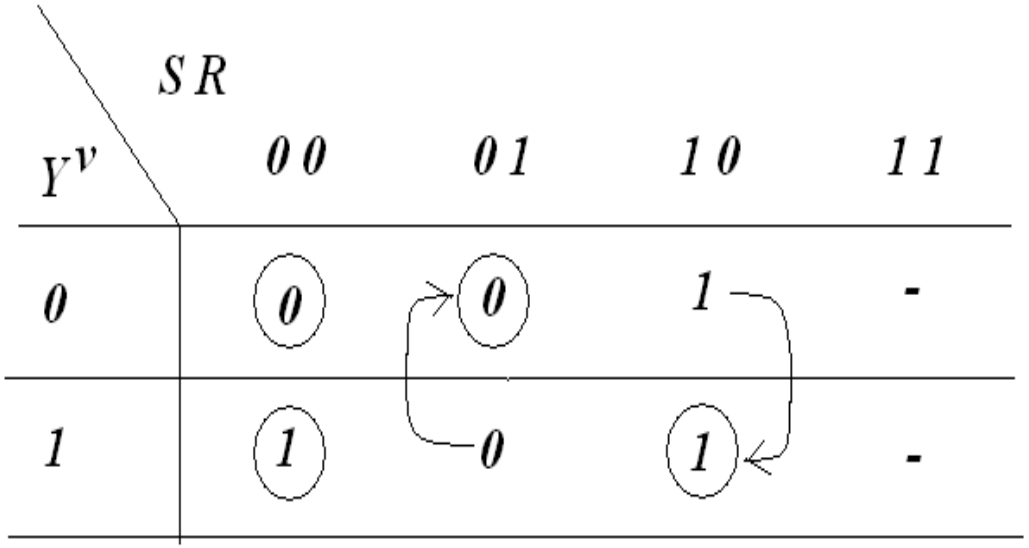
összetett igazságtábla

S	R	Y^V	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	-
1	1	1	-

Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: SR tároló (3)

Az állapot-átmeneti tábla



Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: SR tároló (4)

összetett igazságtábla

S	R	Y^v	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	-
1	1	1	-

⇒

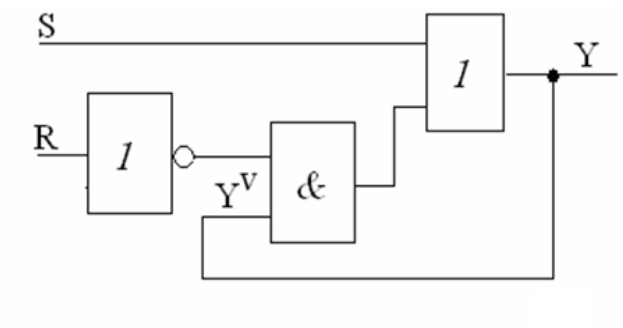
S / R	0 / 0	0 / 1	1 / 1	1 / 0
Y^v = 0	0	0	-	1
Y^v = 1	1	0	-	1

$$Y = S + \overline{R}\,Y^v = \overline{\overline{S + \overline{R}\,Y^v}} = \overline{\overline{S}(\overline{\overline{R}\,Y^v})}$$

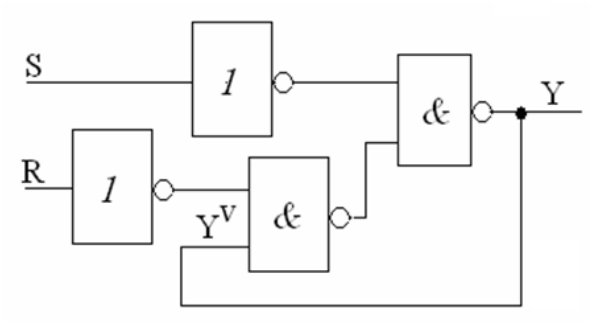
Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: SR tároló (5)

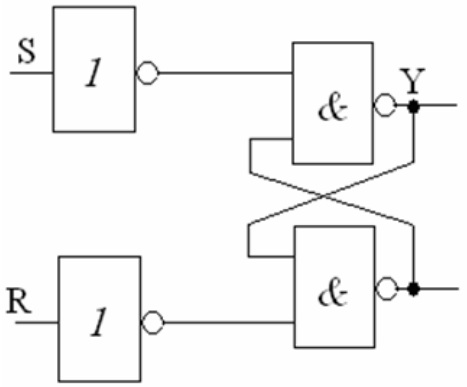
Realizációs megoldások:



- „ÉS-VAGY”



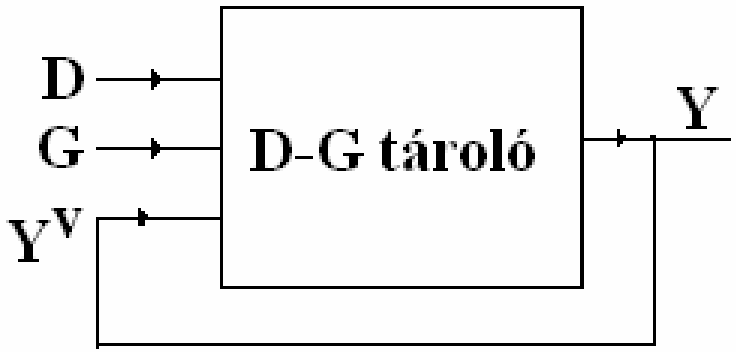
- „NEM-ÉS”



Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: DG tároló (1)

Általános séma:



- kombinációs hálózat, amelynek kimenete a bemenetre érkezik vissza

Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: DG tároló (2)

egyszerű igazságtábla

D	G	Y
0	0	Y^V
0	1	0
1	0	Y^V
1	1	1

⇒

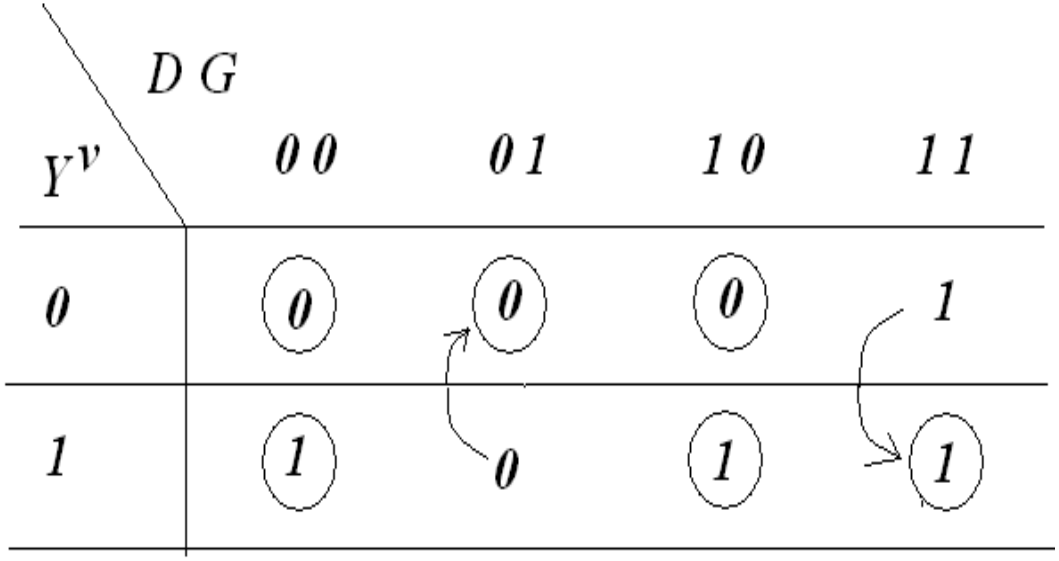
összetett igazságtábla

D	G	Y^V	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	1

Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: DG tároló (3)

Az állapot-átmeneti tábla



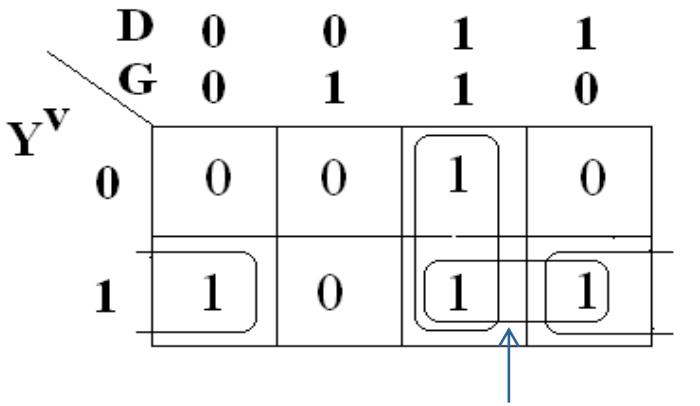
Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: DG tároló (4)

összetett igazságtábla

D	G	Y^v	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	1

⇒



hazárdmentesítés!!

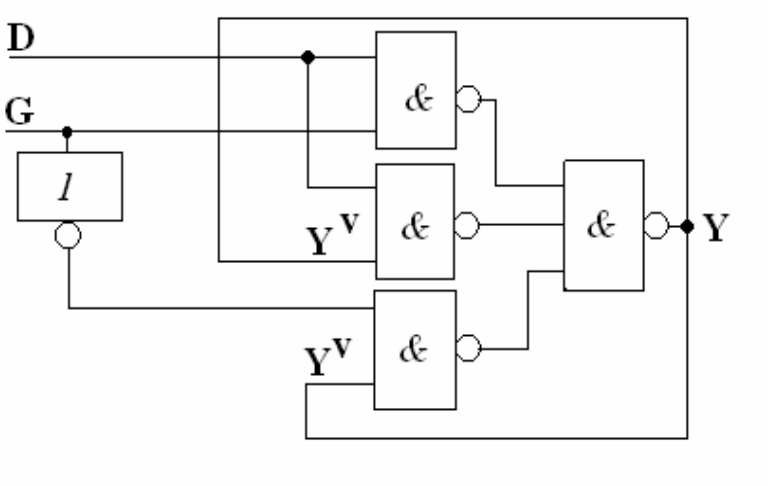
$$Y = D\,G + D\,Y^v + \overline{G}\,Y^v$$

Szabály: visszacsatolt kombinációs hálózattal megvalósított kapcsolást mindig hazárdmentesíteni kell !!!

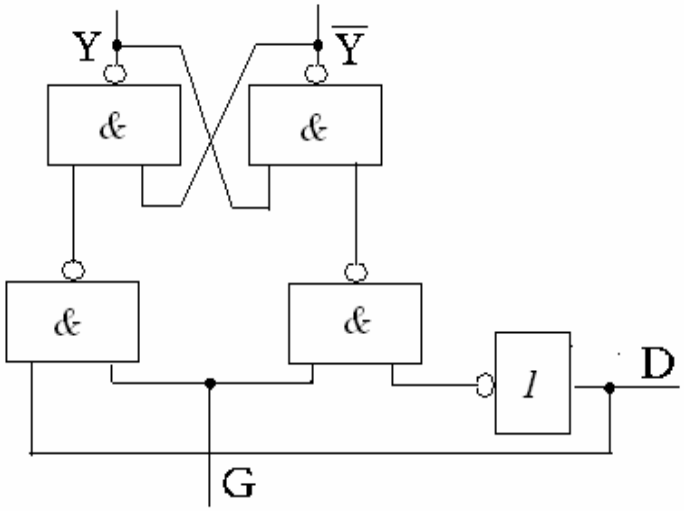
Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: DG tároló (5)

Realizációs megoldások:



- „NAND-NAND”(1)



- „NAND-NAND”(2), SR-ből

Tárolók sorrendi hálózatokban: DG tároló (6)

*A többszörös bemeneti szintváltás szemléltetése **DG** tárolón*

Példa:

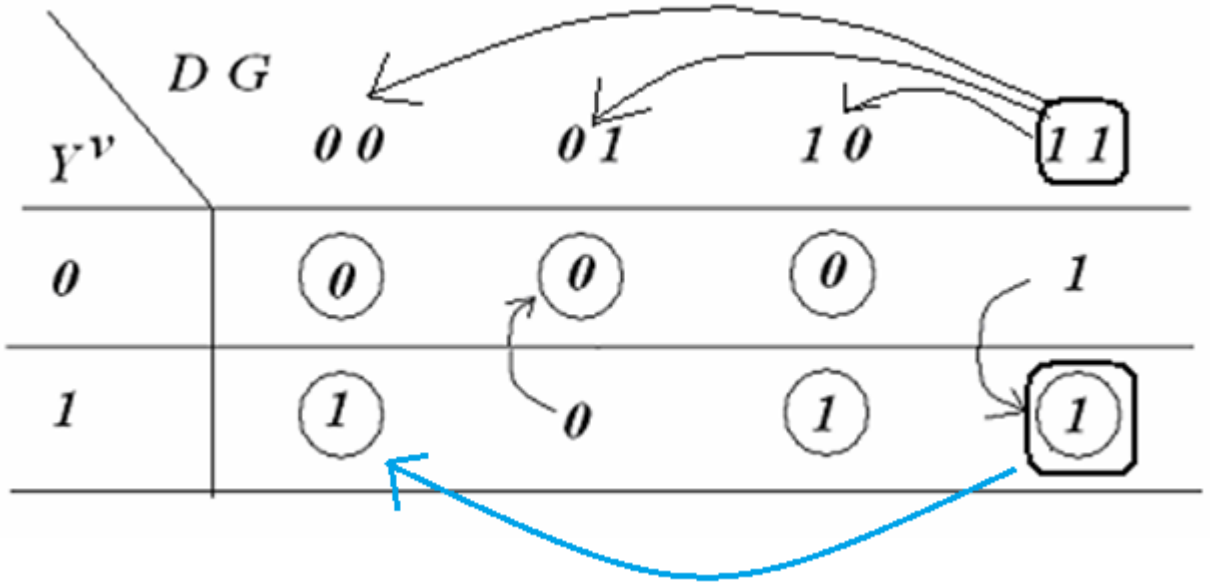
vizsgáljuk meg a következő bemeneti jelváltozást a **DG** tároló esetében az állapot-átmeneti tábla (állapottábla) segítségével!

DGY^v: ’111’ → ’001’

Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: DG tároló (7)

*A többszörös bemeneti szintváltás szemléltetése **DG** tárolón*



DGY^v: ’111’ → ’001’

Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: DG tároló (8)

*A többszörös bemeneti szintváltás szemléltetése **DG** tárolón*

A valóság:

DGY^{v}: ’111’ ~~→~~ ’001’

$\Downarrow$

1. ’111’ → ’101’ → ’001’ vagy

2. ’111’ → ’011’ → ’001’ : ez az átmenet hibás működéshez vezet, mert Y=0-ban stabilizálódik a kimenet!!

Tárolók sorrendi hálózatokban: DG tároló (9)

*A többszörös bemeneti szintváltás szemléltetése **DG** tárolón*

Konklúzió:

A fenti lehetőségnek a kiküszöbölése csak akkor biztosítható, ha megtiltjuk a többszörös bemeneti váltásokat, azaz egyszerre csak egyetlen egy bemeneti jel értéke változhat meg. Ezzel egy olyan általános szabályt is kimondhatunk, mely szerint *visszacsatolt kombinációs hálózatok bemenetei közül egyszerre csak egyet szabad változtatni!*

Tárolók sorrendi hálózatokban: DG tároló (10)

*A **DG** tároló átlátszósága*

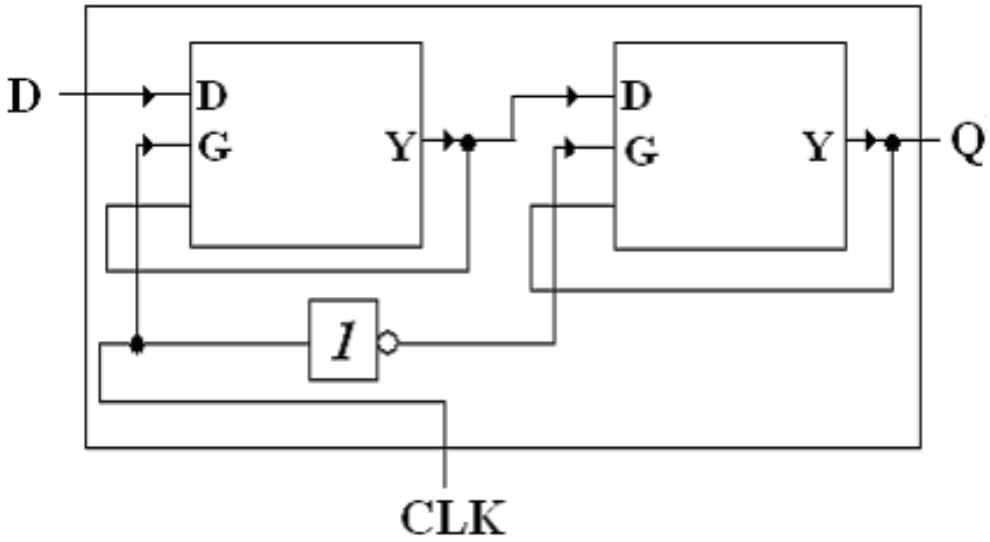
A probléma:
a **DG** tároló sajátossága, hogy a **G**=1 helyzetben a **D**-re adott változások kijutnak a kimenetre. A **G**=1 helyzetben tehát a tároló a **D**-bemenet felől „átlátszó” (transzparens).

Megoldás:
mester-szolga (master-slave) elv alkalmazása

Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: D-MS tároló(1)

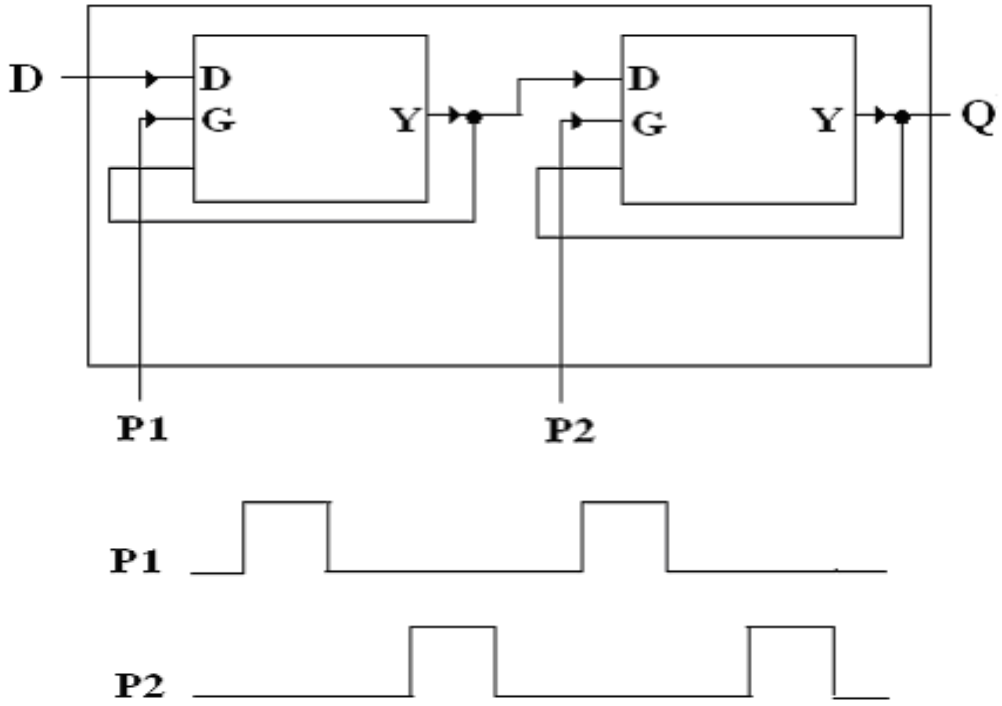
*A **D-MS** tároló megvalósítása **DG** tárolókból a mester-szolga elv alkalmazásával: egyfázisú órajel alkalmazása*



Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: D-MS tároló(2)

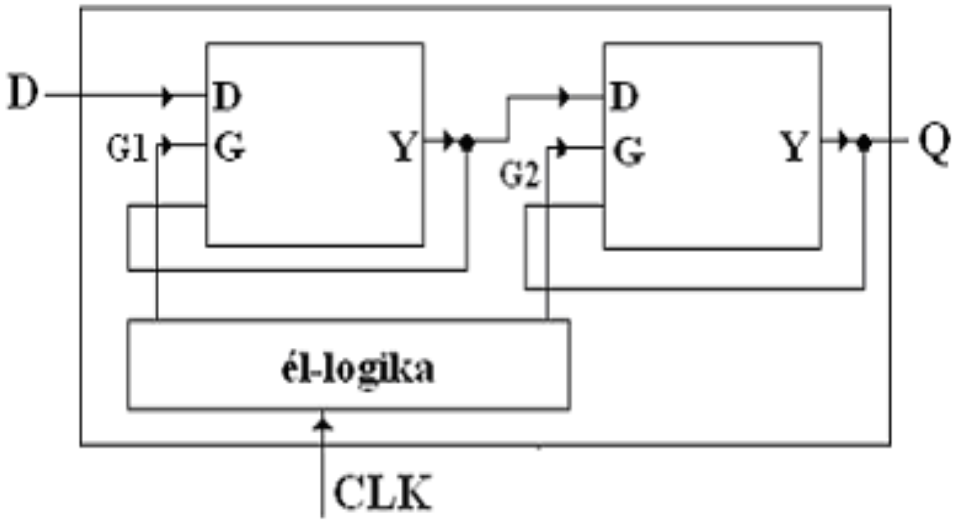
*A **D-MS** tároló megvalósítása **DG** tárolókból a mester-szolga elv alkalmazásával: kétfázisú, ‚nem átlapolt' órajel alkalmazása*

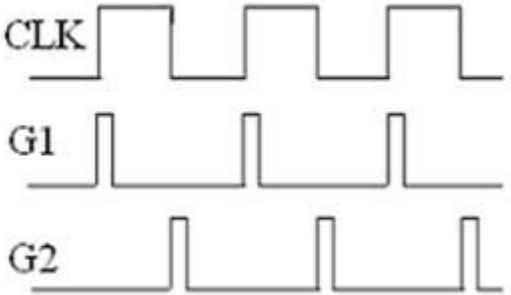


Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: D-MS tároló(3)

*A **D-MS** tároló megvalósítása **DG** tárolókból a mester-szolga elv alkalmazásával: élvezérelt órajel alkalmazása*





Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: D-MS tároló(4)

*A **D-MS** tároló vezérlési táblája:*

D	Q^n	Q^{n+1}	$Q^n \rightarrow Q^{n+1}$		D
0	0	0	0	0	0
0	1	0	0	1	1
1	0	1	1	0	0
1	1	1	1	1	1

$\Rightarrow$

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

Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: JK-MS tároló(1)

egyszerű igazságtábla

J	K	Q^{n+1}
0	0	Q^n
0	1	0
1	0	1
1	1	$\overline{Q^n}$

⇒

összetett igazságtábla

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

Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: JK-MS tároló(2)

*A **JK-MS** flip-flop megvalósítása **D** tárolóval: egy gyakran használt megoldás*

összetett igazságtábla

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

$\Rightarrow$

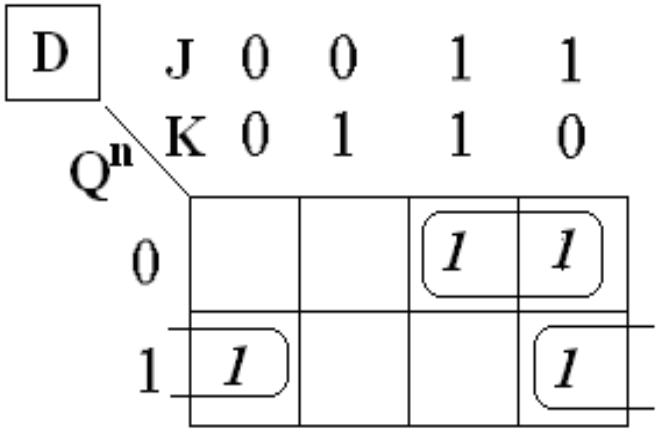
*A **D** bemenet vezérlése*

J	K	Q^n	D
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: JK-MS tároló(3)

*A **JK-MS** flip-flop megvalósítása **D** tárolóval: egy gyakran használt megoldás*



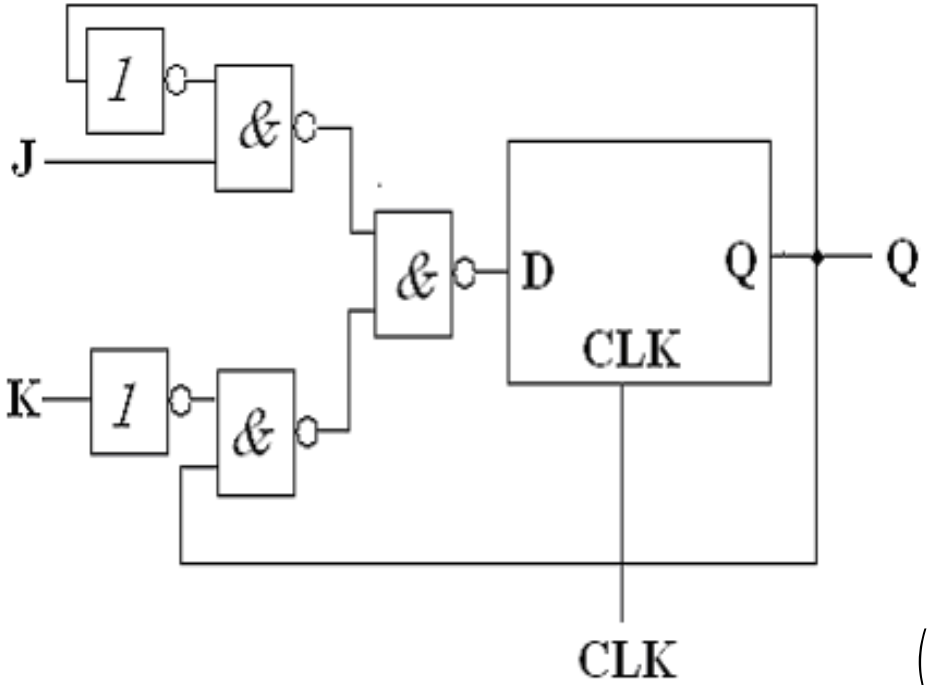
$$D = J\overline{Q^n} + \overline{K}Q^n$$

$$\left(D = \overline{\overline{(J\overline{Q^n})} \bullet \overline{\overline{K}Q^n}}\right)$$

Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: JK-MS tároló(4)

Realizáció:



$$\left(D = \overline{\overline{(J\overline{Q^n})} \bullet \overline{\overline{K}Q^n}}\right)$$

Széchenyi István Egyetem

Tárolók sorrendi hálózatokban: JK-MS tároló(5)

*A **JK-MS** flip-flop vezérlési táblája:*

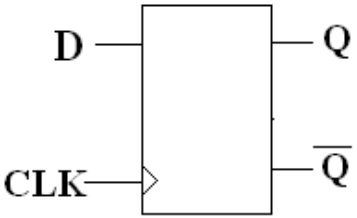
J	K	Q^n	Q^{n+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

$Q^n \rightarrow Q^{n+1}$		J	K
0	0	0	—
0	1	1	—
1	0	—	1
1	1	—	0

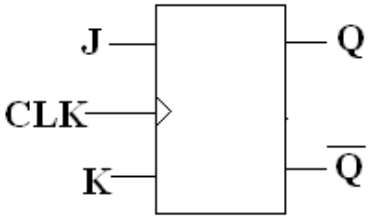
Széchenyi István Egyetem

Flip-flopok segéd-bemenetei és szimbólumaik

Segédbemenetek nélkül:

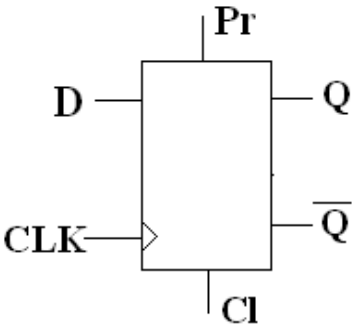


a

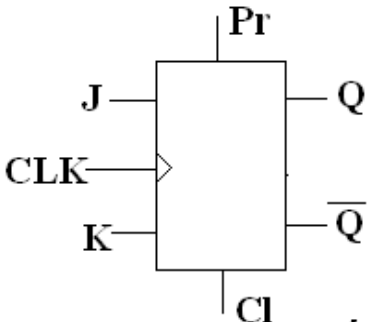


b

Segédbemenetekkel:



c



d

Pr (Preset) : az aktuális állapottól függetlenül ‚1'-be állítja a tárolót(Q)

Cl (Clear) : az aktuális állapottól függetlenül ‚0'-ba állítja a tárolót(Q)

Szinkron sorrendi hálózatok tervezése mintapéldákon keresztül I.

- 1. mintafeladat -

*Adott egy kétbemenetű (X_1, X_2) és egy kimenettel (**Z**) rendelkező szinkron sorrendi hálózat. A hálózat az első X_1=X_2 bemeneti kombinációtól kezdve vizsgálja a bemeneteket, és a **Z** kimenetén jelzi, ha a két bemenet kétszer egymás után azonos logikai szintű. Ha ilyen kombináció sorozat lezajlott, a vizsgálatot újra kezdi. Tervezzük meg a hálózatot **JK-MS** tárolókkal!*

- 1. mintafeladat -

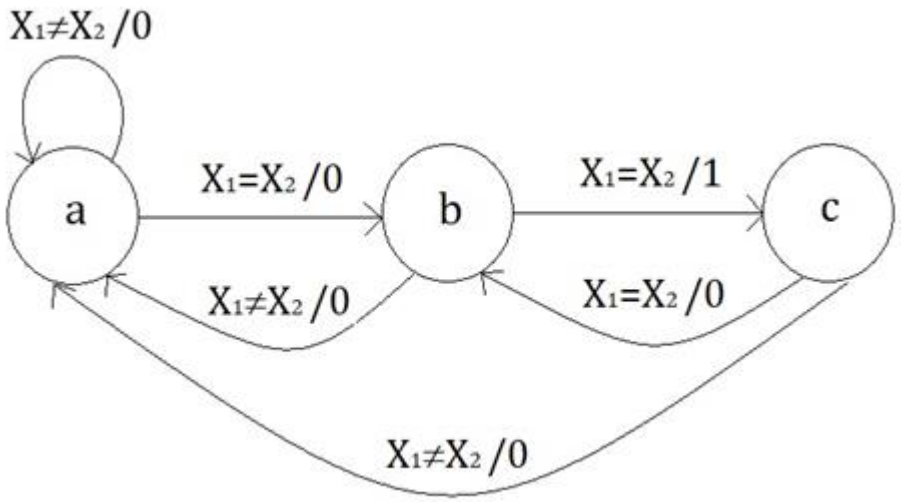
A feladat megoldásához a következő, általános szisztematikus lépéssort alkalmazzuk:

1. *Állapot-átmeneti gráf felrajzolása.*
2. *Előzetes szimbolikus állapottábla felvétele.*
3. *Összevont szimbolikus állapottábla megszerkesztése.*
4. *Kódolt állapottábla elkészítése.*
5. *A specifikációra érvényes vezérlési tábla elkészítése (tárolók használata esetében).*
6. *A szekunder változó(k) és a kimenet(ek) függvényeinek Karnaugh tábla segítségével történő megadása.*
7. *Kezdeti állapotról történő gondoskodás.*
8. *Realizáció logikai kapukkal (és tárolókkal).*

- 1. mintafeladat -

Megoldás 1.*(Mealy-modell):*

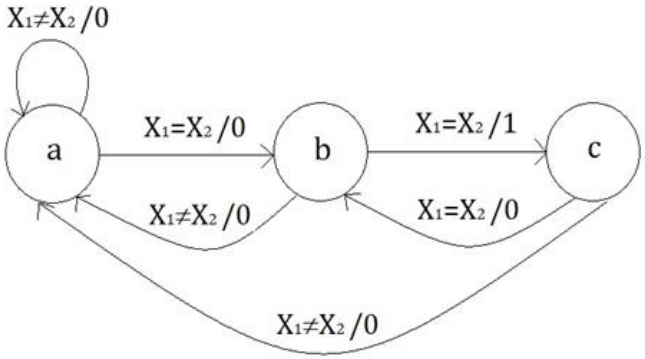
1. Az állapot-átmeneti gráf (állapotgráf) felrajzolása (Mealy):



Széchenyi István Egyetem

- 1. mintafeladat -

2. Az előzetes, szimbolikus állapottábla felvétele (Mealy):



előzetes szimbolikus állapottábla

aktuális állapot	**következő állapot/kimenet**	
	$X_1 \neq X_2$	$X_1 = X_2$
a	$a/0$	$b/0$
b	$a/0$	$c/1$
c	$a/0$	$b/0$

- 1. mintafeladat -

3. Összevont szimbolikus állapottábla megszerkesztése (Mealy):

Az állapot összevonás feltételei:

az előzetes szimbolikus állapottábla két állapotát nem kell megkülönböztetni, ezért azok összevonhatók,

- ***ha bemeneti kombinációnként egyeznek a hozzájuk rendelt kimeneti kombinációk,***
- ***és bemenő kombinációnként ugyanarra a következő állapotra vezetnek.***

Példánkban az ‚*a*' és a ‚*c*' állapotok összevonhatók (‚*ac'* ; ‚*b'*)

- 1. mintafeladat -

- A bemeneti egyszerűsítési lehetőségek kihasználása!

Vegyük észre! ☺ : a két bemenet helyett csak egy bemenetet kell figyelnünk a feladat megoldása során!

$$E = X_1 X_2 + \overline{X_1}\,\overline{X_2}$$

↑

$\{ X_1 = X_2 (00;11)$: exnor = ‚1' $\}$

$\Rightarrow$

összevont szimbolikus állapottábla

aktuális állapot	következő állapot/kimenet	
	$\bar{E}$	E
ac	$ac/0$	$b/0$
b	$ac/0$	$ac/1$

- 1. mintafeladat -

4. Kódolt állapottábla elkészítése (Mealy):

- másodlagos (szekunder) ‚belső' változó (Q) bevezetése az állapotok megkülönböztetésére:

$$ac \rightarrow Q{=}0$$

$$b \rightarrow Q{=}1$$

$\Rightarrow$

kódolt állapottábla

kódolt aktuális állapot, Q^n	kódolt következő állapot/kimenet	
	$\bar{E}$ Q^{n+1}/Z	E Q^{n+1}/Z
0	$0/0$	$1/0$
1	$0/0$	$0/1$

Széchenyi István Egyetem

- 1. mintafeladat -

5. A specifikációra érvényes vezérlési tábla elkészítése (Mealy):

A JK-MS saját vezérlési táblája

$Q^n \rightarrow Q^{n+1}$	J	K
$0 \rightarrow 0$	0	-
$0 \rightarrow 1$	1	-
$1 \rightarrow 0$	-	1
$1 \rightarrow 1$	-	0

kódolt állapottábla

kódolt aktuális állapot, Q^n	kódolt következő állapot/kimenet	
	$\bar{E}$ Q^{n+1}	E Q^{n+1}
0	0/0	1/0
1	0/0	0/1

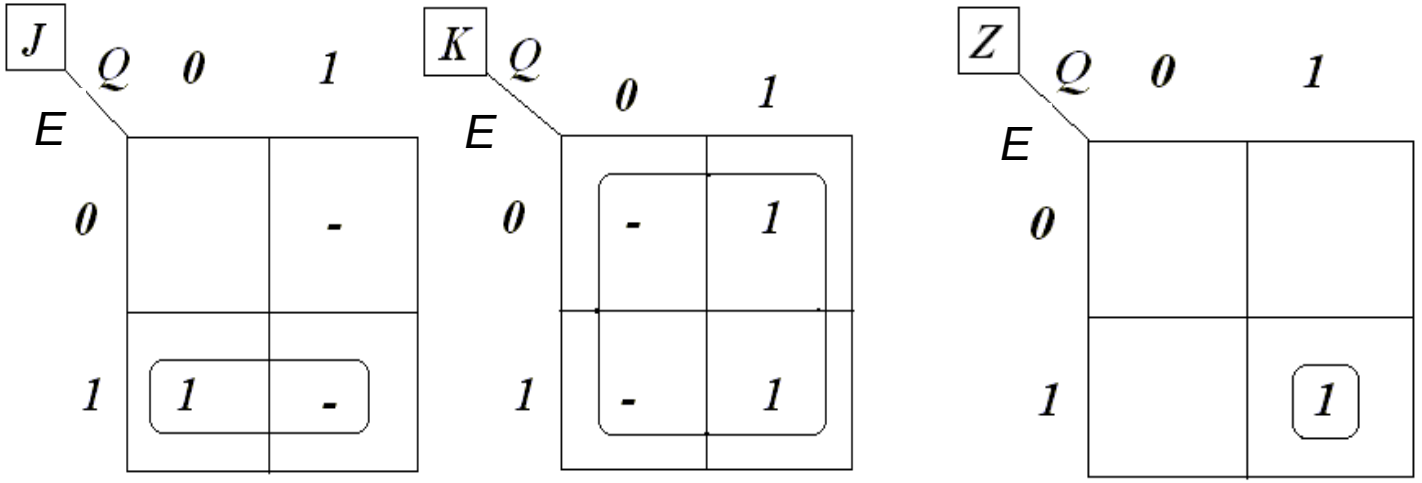
⇒

vezérlési tábla

Q	$\bar{E}$ J K	E J K
0	0 -	1 -
1	- 1	- 1

- 1. mintafeladat -

6. A szekunder változó és a kimenet függvényeinek Karnaugh tábla segítségével történő megadása (Mealy):



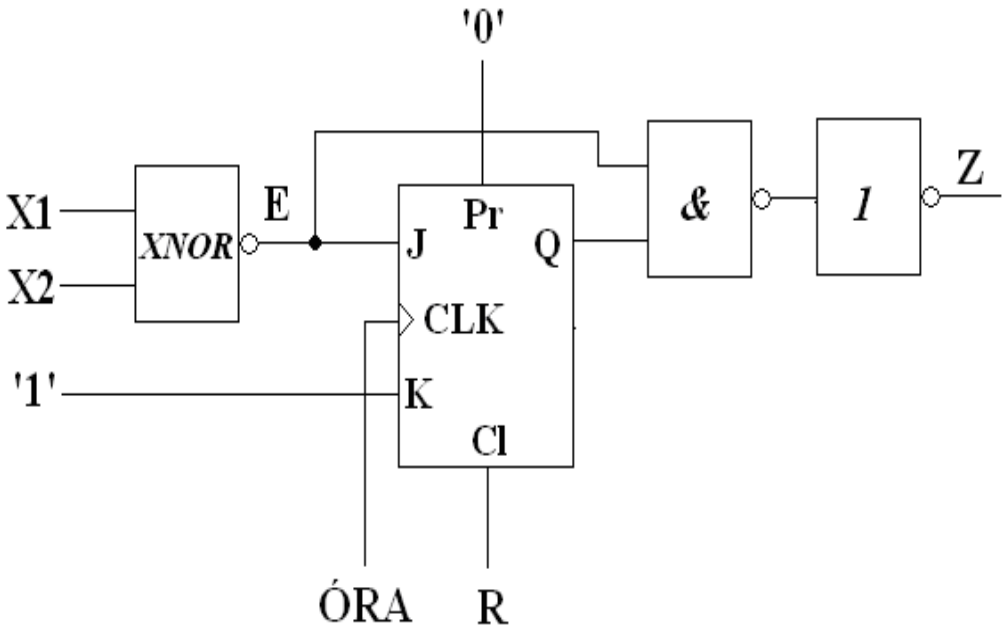
$$J = E, \quad K = 1, \quad Z = Q\,E$$

Széchenyi István Egyetem

- 1. mintafeladat -

7-8. Realizáció (‚NAND'), a kezdeti állapot biztosítása (RESET) (Mealy):

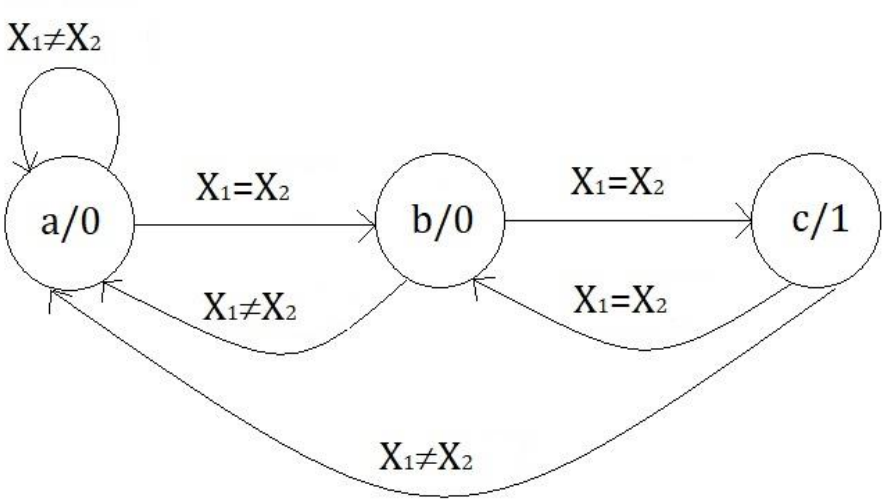
$J = E, \quad K = 1, \quad Z = Q\,E$ (‚NAND'→) $J = E, \quad K = 1, \quad Z = \overline{\overline{QE}}$



- 1. mintafeladat -

Megoldás 2.*(Moore-modell):*

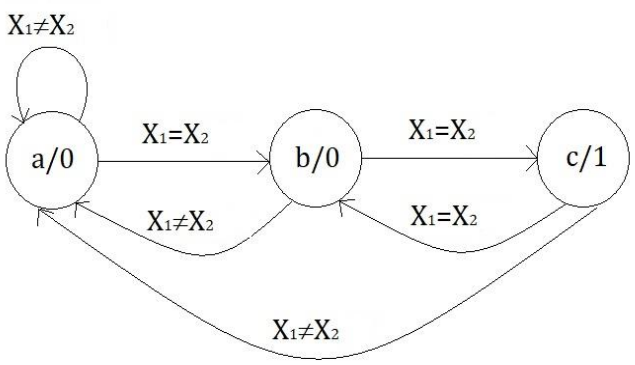
1. Az állapotgráf felrajzolása (Moore):



Széchenyi István Egyetem

- 1. mintafeladat -

2. Az előzetes, szimbolikus állapottábla felvétele (Moore) :



előzetes szimbolikus állapottábla

aktuális állapot/kimenet	**következő állapot**	
	$X_1 \neq X_2$	$X_1 = X_2$
$a/0$	a	b
$b/0$	a	c
$c/1$	a	b

- 1. mintafeladat -

3. Összevont szimbolikus állapottábla megszerkesztése (Moore):

előzetes szimbolikus állapottábla

aktuális állapot/kimenet	következő állapot	
	$X_1 \neq X_2$	$X_1 = X_2$
$a/0$	a	b
$b/0$	a	c
$c/1$	a	b

Az előzetes szimbolikus állapottáblán – az összevonás feltételeit figyelembe véve- ezúttal nem tudunk állapot összevonást végezni. A kódolt állapottáblát így e táblázat alapján kell elkészíteni.

Széchenyi István Egyetem

- 1. mintafeladat -

4. Kódolt állapottábla elkészítése (Moore):

- Szekunder változók bevezetése az állapotok megkülönböztetésére:

egy lehetséges kódkiosztás:

$$a \rightarrow Q_1 Q_2 = 00$$
$$b \rightarrow Q_1 Q_2 = 01$$
$$c \rightarrow Q_1 Q_2 = 10$$

$\Rightarrow$

kódolt állapottábla

kódolt aktuális állapot/kimenet	kódolt következő állapot	
	$\bar{E}$	E
$Q_1^n\ Q_2^n/Z$	$Q_1^{n+1}\ Q_2^{n+1}$	$Q_1^{n+1}\ Q_2^{n+1}$
0 0/0	0 0	0 1
0 1/0	0 0	1 0
1 0/1	0 0	0 1
1 1/-	- -	- -

Széchenyi István Egyetem

- 1. mintafeladat -

5. A specifikációra érvényes vezérlési tábla elkészítése (Moore):

A JK-MS saját vezérlési táblája

$Q^n \rightarrow Q^{n+1}$	J	K
$0 \rightarrow 0$	0	-
$0 \rightarrow 1$	1	-
$1 \rightarrow 0$	-	1
$1 \rightarrow 1$	-	0

kódolt állapottábla

kódolt aktuális állapot/kimenet	kódolt következő állapot	
	$\bar{E}$	E
$Q_1^n\ Q_2^n/Z$	$Q_1^{n+1}\ Q_2^{n+1}$	$Q_1^{n+1}\ Q_2^{n+1}$
$0\ 0/0$	$0\ 0$	$0\ 1$
$0\ 1/0$	$0\ 0$	$1\ 0$
$1\ 0/1$	$0\ 0$	$0\ 1$
$1\ 1/-$	$-\ -$	$-\ -$

$\Downarrow$

$\Rightarrow$

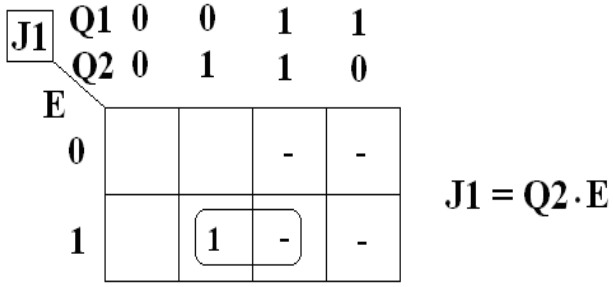
vezérlési tábla

kódolt aktuális állapot/kimenet	kódolt következő állapot			
	$\bar{E}$		E	
$Q_1\ Q_2/Z$	J_1K_1	J_2K_2	J_1K_1	J_2K_2
$0\ 0/0$	$0\ -$	$0\ -$	$0\ -$	$1\ -$
$0\ 1/0$	$0\ -$	$-\ 1$	$1\ -$	$-\ 1$
$1\ 0/1$	$-\ 1$	$0\ -$	$-\ 1$	$1\ -$
$1\ 1/-$	$-\ -$	$-\ -$	$-\ -$	$-\ -$

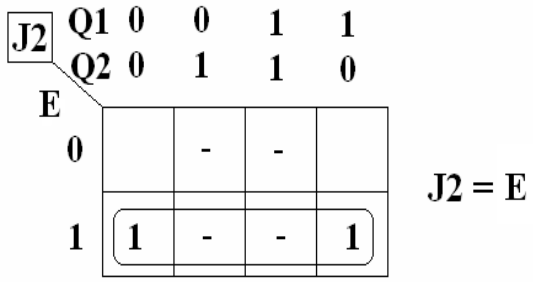
Széchenyi István Egyetem

- 1. mintafeladat -

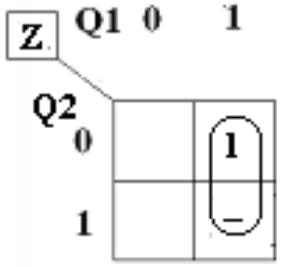
6. A szekunder változók és a kimenet függvényeinek Karnaugh tábla segítségével történő megadása (Moore):



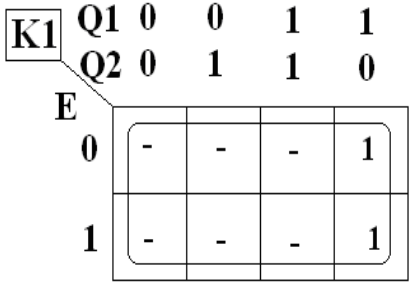
$J1 = Q2 \cdot E$



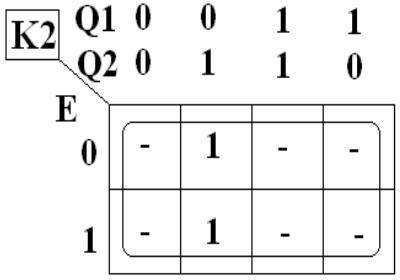
$J2 = E$



$Z = Q_1$



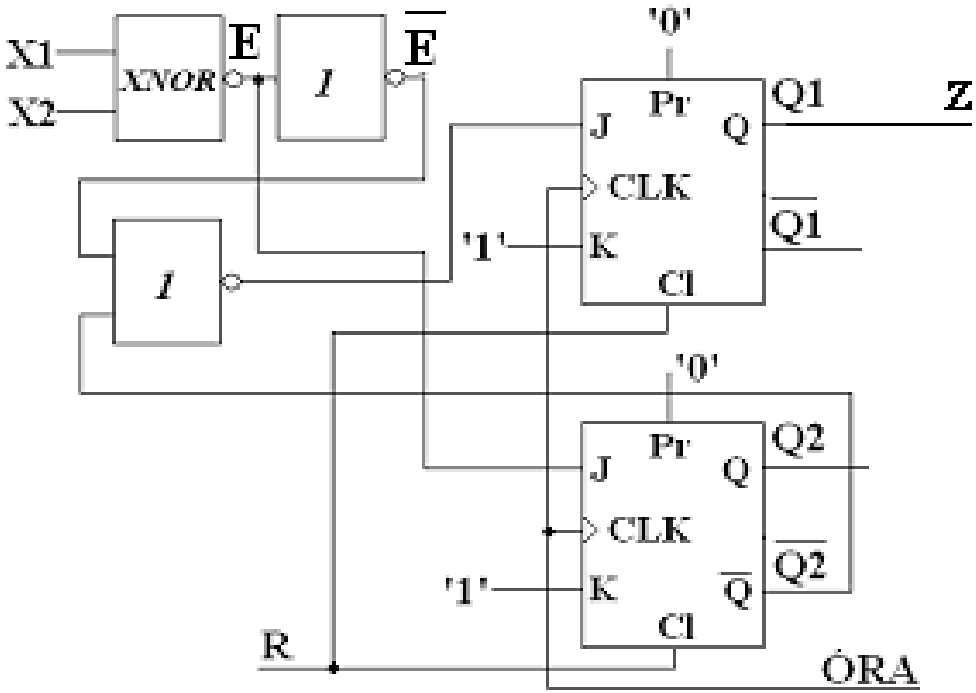
$K1 = 1$



$K2 = 1$

- 1. mintafeladat -

7-8. Realizáció a kezdeti állapot biztosításával (RESET) (Moore):



Szinkron sorrendi hálózatok tervezése mintapéldákon keresztül I.

- 2. mintafeladat -

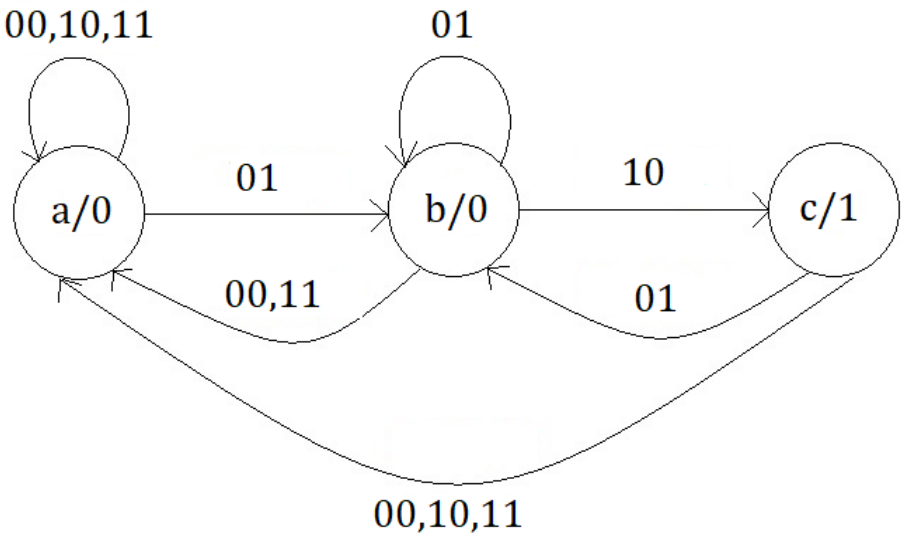
*Tervezzük meg a lehető legegyszerűbb változatban azt a 'preset' és 'clear' bemenetekkel is rendelkező, élvezérelt **D-MS** tárolókkal felépített Moore-típusú szinkron sorrendi hálózatot, amelynek két bemenete (X_1, X_2) és egy kimenete (**Z**) van. A hálózat az órajel felfutása előtt fogadja a bemeneti kombinációkat. A hálózat **Z** kimenete akkor lesz '1', ha a bemenetre az '1 0' kombináció érkezik, de csak abban az esetben, ha az előző óraciklus által fogadott bemeneti bitkombináció '0 1' volt! A kezdeti állapothoz az X_1X_2 = '00' kombináció tartozik.*

Széchenyi István Egyetem

- 2. mintafeladat -

Megoldás 1.*(Moore-modell):*

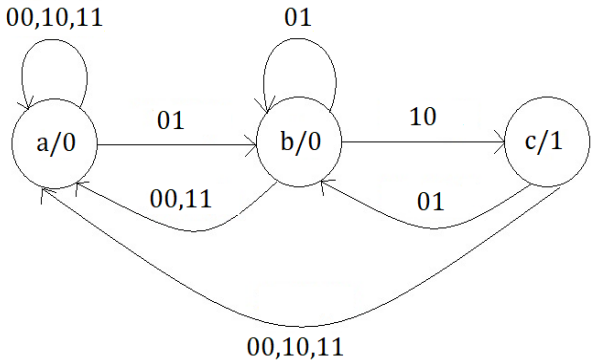
1. Az állapotgráf felrajzolása (Moore):



Széchenyi István Egyetem

- 2. mintafeladat -

2. Az előzetes, szimbolikus állapottábla felvétele (Moore):



előzetes szimbolikus állapottábla

aktuális állapot/ kimenet	következő állapot			
	X_1X_2			
	00	*01*	*10*	*11*
a/0	*a*	*b*	*a*	*a*
b/0	*a*	*b*	*c*	*a*
c/1	*a*	*b*	*a*	*a*

Széchenyi István Egyetem

- 2. mintafeladat -

3. Összevont szimbolikus állapottábla megszerkesztése (Moore):

előzetes szimbolikus állapottábla

aktuális állapot/ kimenet	**következő állapot**			
	X_1X_2			
	00	*01*	*10*	*11*
$a/0$	a	b	a	a
$b/0$	a	b	c	a
$c/1$	a	b	a	a

Az előzetes szimbolikus állapottáblán – az összevonás feltételeit figyelembe véve- nem tudunk állapot összevonást végezni. A kódolt állapottáblát így e táblázat alapján kell elkészíteni.

- 2. mintafeladat -

4. Kódolt állapottábla elkészítése (Moore):

- Szekunder változók bevezetése az állapotok megkülönböztetésére:

egy lehetséges kódkiosztás:

$$a \rightarrow Q_1Q_2=00$$
$$b \rightarrow Q_1Q_2=01$$
$$c \rightarrow Q_1Q_2=10$$

$\Rightarrow$

kódolt állapottábla

kódolt aktuális állapot/kimenet	**kódolt következő állapot**			
	X_1X_2			
	00	*01*	*10*	*11*
$Q_1^nQ_2^n/Z$	$Q_1^{n+1}Q_2^{n+1}$	$Q_1^{n+1}Q_2^{n+1}$	$Q_1^{n+1}Q_2^{n+1}$	$Q_1^{n+1}Q_2^{n+1}$
0 0/0	*0 0*	*0 1*	*0 0*	*0 0*
0 1/0	*0 0*	*0 1*	*1 0*	*0 0*
1 0/1	*0 0*	*0 1*	*0 0*	*0 0*
1 1/-	*- -*	*- -*	*- -*	*- -*

Széchenyi István Egyetem

- 2. mintafeladat -

5. A specifikációra érvényes vezérlési tábla elkészítése (Moore):

A D-MS saját vezérlési táblája

$Q^n \rightarrow Q^{n+1}$	D
$0 \rightarrow 0$	0
$0 \rightarrow 1$	1
$1 \rightarrow 0$	0
$1 \rightarrow 1$	1

kódolt állapottábla

kódolt aktuális állapot/ kimenet	kódolt következő állapot			
	X_1X_2			
	00	01	10	11
$Q_1^nQ_2^n/Z$	$Q_1^{n+1}Q_2^{n+1}$	$Q_1^{n+1}Q_2^{n+1}$	$Q_1^{n+1}Q_2^{n+1}$	$Q_1^{n+1}Q_2^{n+1}$
0 0/0	0 0	0 1	0 0	0 0
0 1/0	0 0	0 1	1 0	0 0
1 0/1	0 0	0 1	0 0	0 0
1 1/-	- -	- -	- -	- -

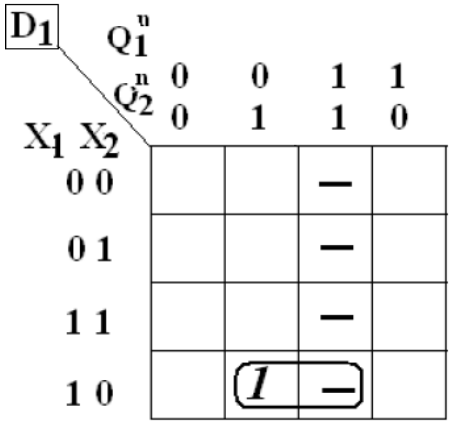
$\Rightarrow$

vezérlési tábla

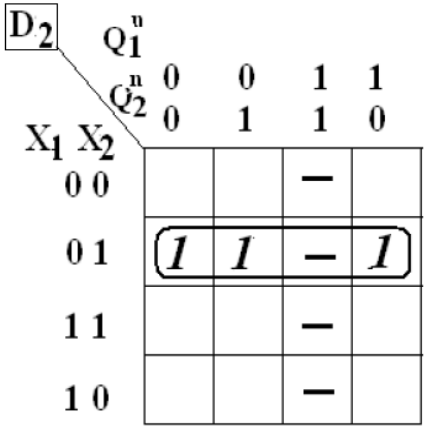
kódolt aktuális állapot/kimenet	X_1X_2			
	00	01	10	11
$Q_1^nQ_2^n/Z$	D_1D_2	D_1D_2	D_1D_2	D_1D_2
0 0/0	0 0	0 1	0 0	0 0
0 1/0	0 0	0 1	1 0	0 0
1 0/1	0 0	0 1	0 0	0 0
1 1/-	- -	- -	- -	- -

- 2. mintafeladat -

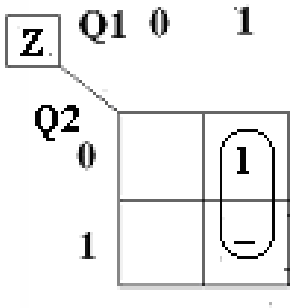
6. A szekunder változók és a kimenet függvényeinek Karnaugh tábla segítségével történő megadása (Moore):



$D1 = Q_2^n \, X1 \, \overline{X2}$



$D2 = \overline{X1} \, X2$

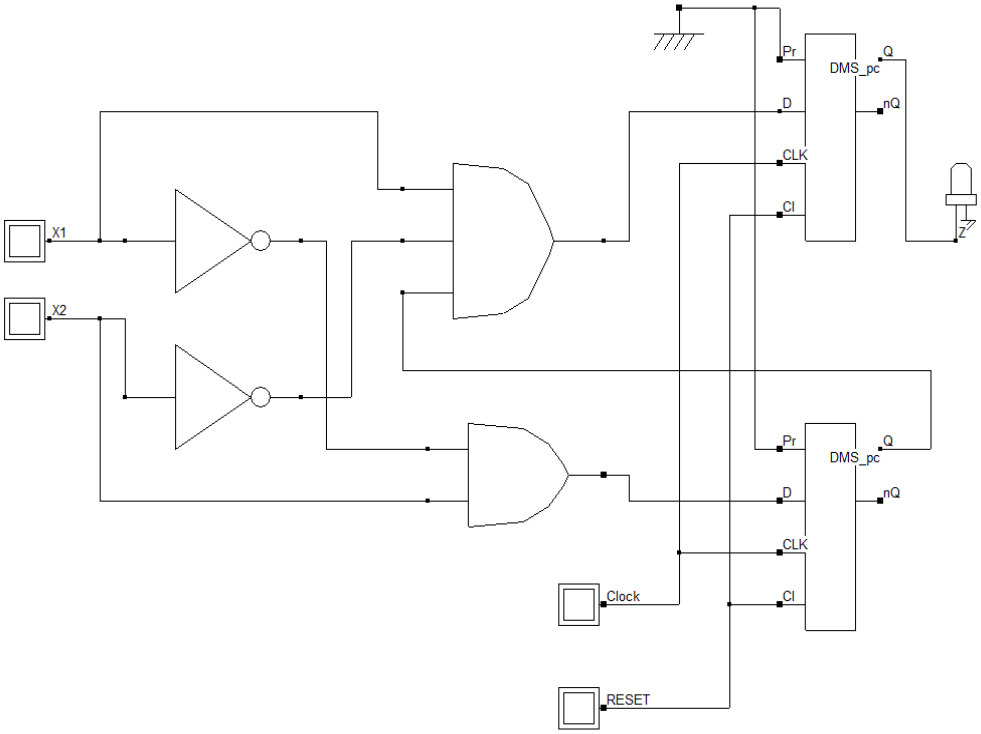


$Z = Q_1$

Széchenyi István Egyetem

- 2. mintafeladat -

7-8. Realizáció, a kezdeti állapot biztosítása (RESET) (Moore):



$$D1 = Q_2^n \, X1 \, \overline{X2}$$

$$D2 = \overline{X1} \, X2$$

$$Z = Q_1$$

A DSCH 3.5 szimulátorban megvalósított áramkör

- 2. mintafeladat -

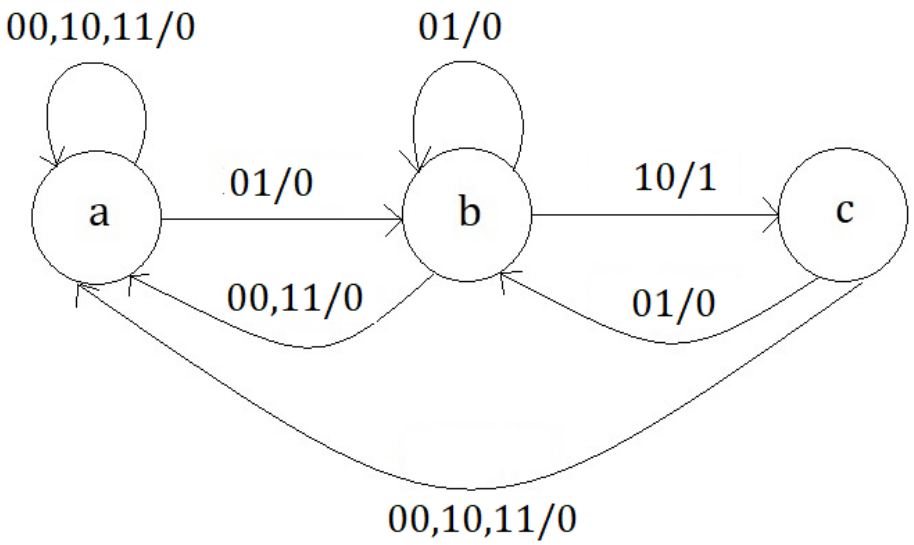
A feladat megoldásának(Moore) DSCH sémája („SZH_eloadasmintafeladat_2_dualishoz”):

http://www.sze.hu/~somi/Digit%e1lis%20h%e1l%f3zatok/DSCH%20s%e9m%e1k%20gyakorl%f3%20feladatokhoz/

- 2. mintafeladat -

Megoldás 2.*(Mealy-modell):*

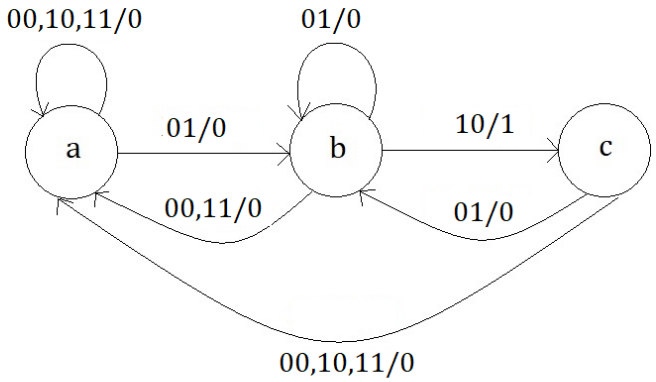
1. Az állapotgráf felrajzolása (Mealy):



Széchenyi István Egyetem

- 2. mintafeladat -

2. Az előzetes, szimbolikus állapottábla felvétele (Mealy):



előzetes szimbolikus állapottábla

aktuális állapot	**következő állapot/kimenet**			
	X_1X_2			
	00	*01*	*10*	*11*
a	$a/0$	$b/0$	$a/0$	$a/0$
b	$a/0$	$b/0$	$c/1$	$a/0$
c	$a/0$	$b/0$	$a/0$	$a/0$

Széchenyi István Egyetem

- 2. mintafeladat -

3. Összevont szimbolikus állapottábla megszerkesztése (Mealy):

előzetes szimbolikus állapottábla

aktuális Állapot	következő állapot/kimenet			
	X_1X_2			
	00	*01*	*10*	*11*
a	$a/0$	$b/0$	$a/0$	$a/0$
b	$a/0$	$b/0$	$c/1$	$a/0$
c	$a/0$	$b/0$	$a/0$	$a/0$

$a,c \rightarrow ac$

$b \rightarrow b$

$\Rightarrow$

összevont szimbolikus állapottábla

aktuális állapot	következő állapot/kimenet			
	X_1X_2			
	00	*01*	*10*	*11*
ac	$ac/0$	$b/0$	$ac/0$	$ac/0$
b	$ac/0$	$b/0$	$ac/1$	$ac/0$

Széchenyi István Egyetem

- 2. mintafeladat -

4. Kódolt állapottábla elkészítése (Mealy):

- Szekunder változó bevezetése az állapotok megkülönböztetésére:

egy lehetséges kódkiosztás:

$$ac \rightarrow Q{=}0$$

$$b \rightarrow Q{=}1$$

$\Rightarrow$

kódolt állapottábla

aktuális állapot, Q	**következő állapot/kimenet**			
	X_1X_2			
	00	*01*	*10*	*11*
0	*0/0*	*1/0*	*0/0*	*0/0*
1	*0/0*	*1/0*	*0/1*	*0/0*

Széchenyi István Egyetem

- 2. mintafeladat -

5. A specifikációra érvényes vezérlési tábla elkészítése (Mealy):

A D-MS saját vezérlési táblája

$Q^n \rightarrow Q^{n+1}$	D
$0 \rightarrow 0$	0
$0 \rightarrow 1$	1
$1 \rightarrow 0$	0
$1 \rightarrow 1$	1

kódolt állapottábla

aktuális állapot, Q	D/Z			
	X_1X_2			
	00	*01*	*10*	*11*
0	*0/0*	*1/0*	*0/0*	*0/0*
1	*0/0*	*1/0*	*0/1*	*0/0*

$\Rightarrow$

vezérlési tábla

aktuális állapot, Q	D/Z			
	X_1X_2			
	00	*01*	*10*	*11*
0	*0/0*	*1/0*	*0/0*	*0/0*
1	*0/0*	*1/0*	*0/1*	*0/0*

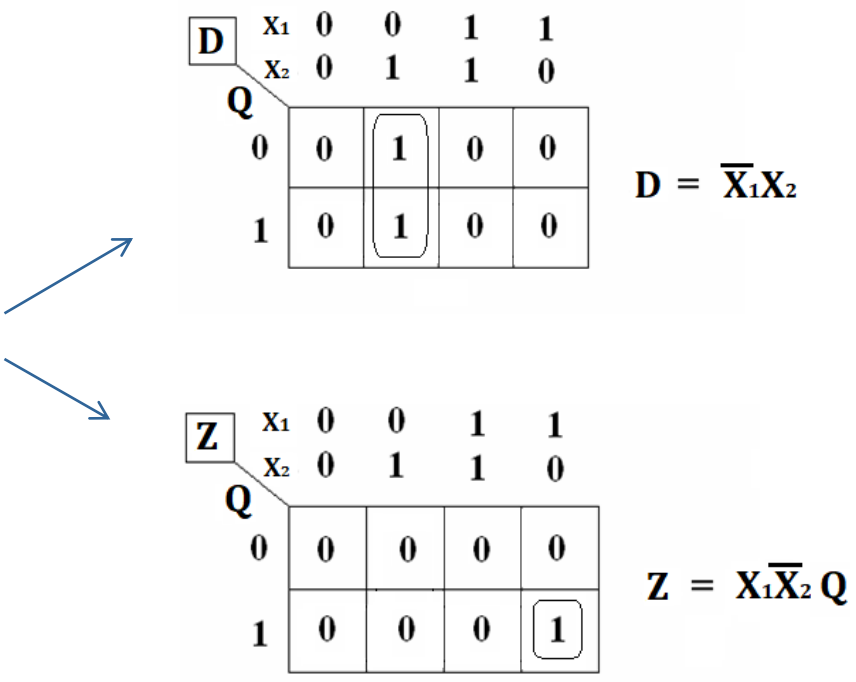
Széchenyi István Egyetem

- 2. mintafeladat -

6. A szekunder változó és a kimenet függvényeinek Karnaugh tábla segítségével történő megadása (Mealy):

kódolt állapottábla

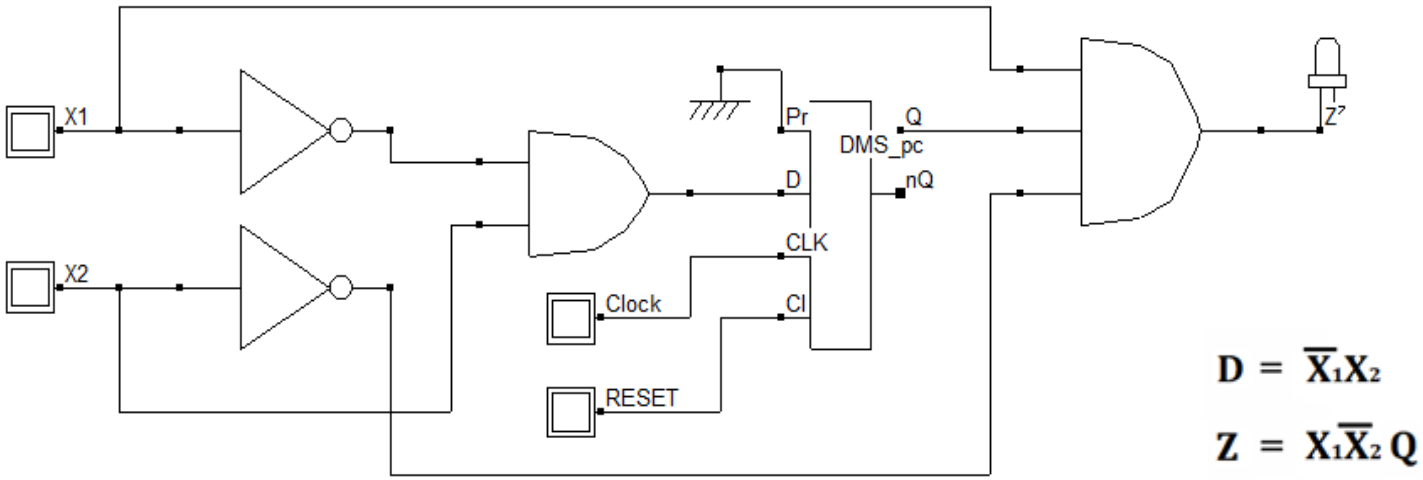
aktuális állapot, Q	D/Z			
	X_1X_2			
	00	*01*	*10*	*11*
0	*0/0*	*1/0*	*0/0*	*0/0*
1	*0/0*	*1/0*	*0/1*	*0/0*



Széchenyi István Egyetem

- 2. mintafeladat -

7-8. Realizáció, a kezdeti állapot biztosítása (RESET) (Mealy):



$$D = \overline{X}_1 X_2$$

$$Z = X_1 \overline{X}_2 Q$$

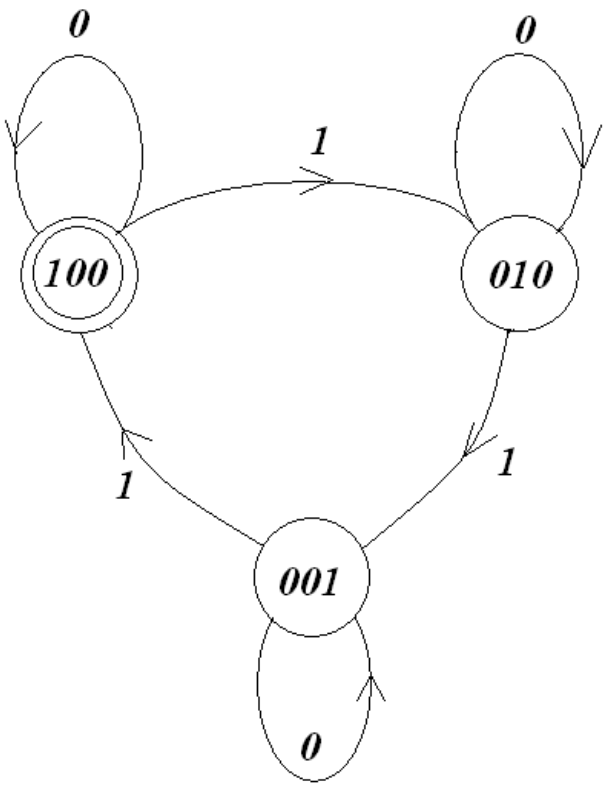
A DSCH 3.5 szimulátorban megvalósított áramkör

Szinkron sorrendi hálózatok tervezése mintapéldákon keresztül I.

- 3. mintafeladat -

*Tervezzük meg szinkron 'preset' és 'clear' bemenetekkel is rendelkező, élvezérelt **D-MS** tárolókkal az ábrán látható állapotgráf szerinti állapot-kimenetű szinkron sorrendi hálózatot a lehető legegyszerűbb változatban! A kezdeti állapotot a gráfon dupla kör jelzi.*

- 3. mintafeladat -



- 3. mintafeladat -

Megoldás 1.:

A hálózat működési specifikációja a kódolt állapotgráffal lett megadva, így a megoldás első lépéseként rögtön elvégezhetjük a kódolt állapottábla felvételét.

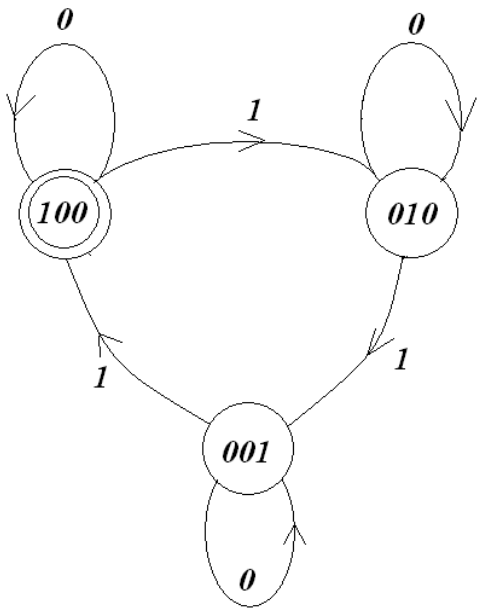
Ugyanakkor az is látható a kódolt állapotgráfból, hogy az adott hálózat Moore-modell szerint működik.

A verbális specifikáció szerint a hálózat állapot-kimenetű: ez azt jelenti, hogy maguk az állapotokhoz tartozó kódok reprezentálják az aktuális kimeneti értékeket.

Széchenyi István Egyetem

- 3. mintafeladat -

1. A kódolt állapottábla felvétele:



⇒

kódolt állapottábla

kódolt aktuális állapot,			kódolt következő állapot					
			X = 0			X = 1		
Q_1^n	Q_2^n	Q_3^n	Q_1^{n+1}	Q_2^{n+1}	Q_3^{n+1}	Q_1^{n+1}	Q_2^{n+1}	Q_3^{n+1}
1	0	0	1	0	0	0	1	0
0	1	0	0	1	0	0	0	1
0	0	1	0	0	1	1	0	0
0	0	0	-	-	-	-	-	-
0	1	1	-	-	-	-	-	-
1	0	1	-	-	-	-	-	-
1	1	0	-	-	-	-	-	-
1	1	1	-	-	-	-	-	-

Széchenyi István Egyetem

- 3. mintafeladat -

2. A specifikációra érvényes vezérlési tábla elkészítése:

A D-MS saját vezérlési táblája

$Q^n \rightarrow Q^{n+1}$	D
$0 \rightarrow 0$	0
$0 \rightarrow 1$	1
$1 \rightarrow 0$	0
$1 \rightarrow 1$	1

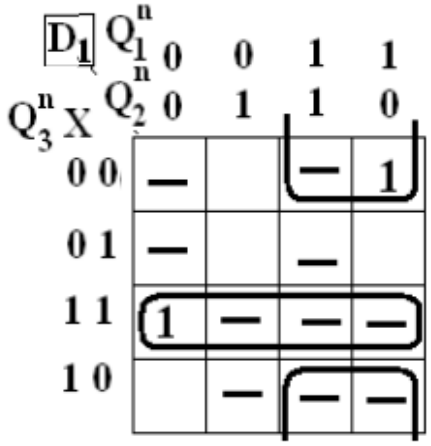
$\Rightarrow$

vezérlési tábla

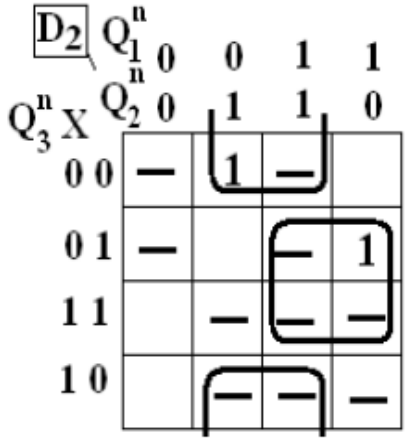
kódolt aktuális állapot,			kódolt következő állapot					
			X = 0			X = 1		
Q_1^n	Q_2^n	Q_3^n	D_1	D_2	D_3	D_1	D_2	D_3
1	0	0	1	0	0	0	1	0
0	1	0	0	1	0	0	0	1
0	0	1	0	0	1	1	0	0
0	0	0	-	-	-	-	-	-
0	1	1	-	-	-	-	-	-
1	0	1	-	-	-	-	-	-
1	1	0	-	-	-	-	-	-
1	1	1	-	-	-	-	-	-

- 3. mintafeladat -

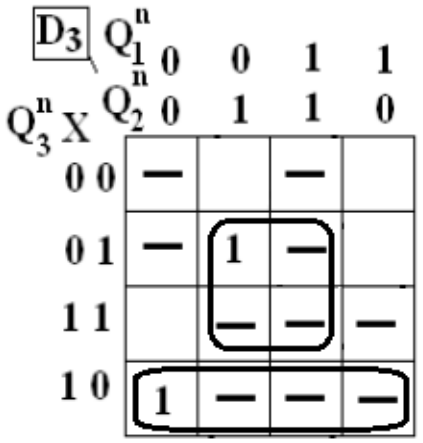
3. A szekunder változók (→kimenetek) függvényeinek Karnaugh tábla segítségével történő megadása:



$$D_1 = Q_1^n \overline{X} + Q_3^n X$$



$$D_2 = Q_2^n \overline{X} + Q_1^n X$$

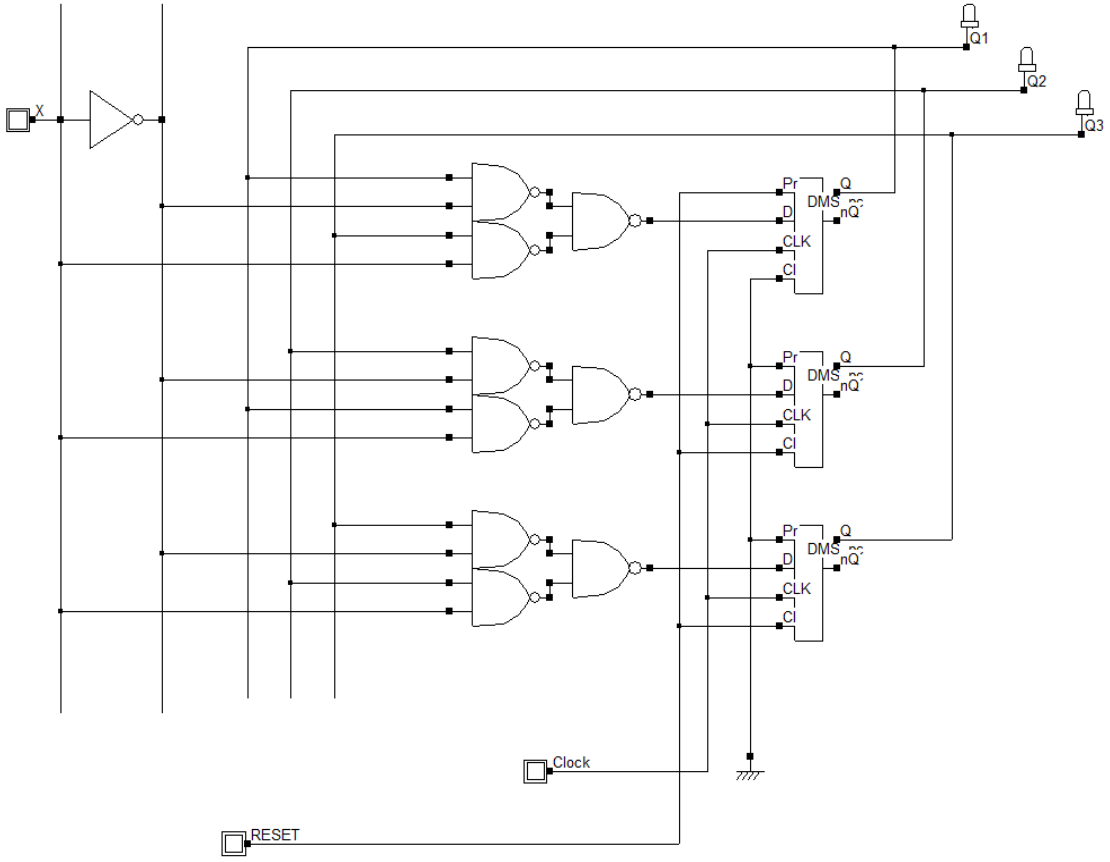


$$D_3 = Q_2^n X + Q_3^n \overline{X}$$

Széchenyi István Egyetem

- 3. mintafeladat -

4. Realizáció(‚NAND'), a kezdeti állapot biztosítása (RESET):



$$D_1 = Q_1^n \overline{X} + Q_3^n X$$

$$D_2 = Q_2^n \overline{X} + Q_1^n X$$

$$D_3 = Q_2^n X + Q_3^n \overline{X}$$

A DSCH 3.5 szimulátorban megvalósított áramkör

- 3. mintafeladat -

A feladat megoldásának DSCH sémája („SZH_eloadasmintafeladat_3_dualishoz”):

http://www.sze.hu/~somi/Digit%e1lis%20h%e1l%f3zatok/DSCH%20s%e9m%e1k%20gyakorl%f3%20feladatokhoz/

- 3. mintafeladat -

***Megoldás 2. (‚1'-es súlyozású kódolási alap)**:*

*A megoldástípus alapja az **‚1'-es súlyozású kódolás**:*
az olyan kódolást, melyben csak egyetlen bit helyén szerepel '1'-es érték (a többi '0'), ‚1'-es súlyú kódolásnak nevezzük.

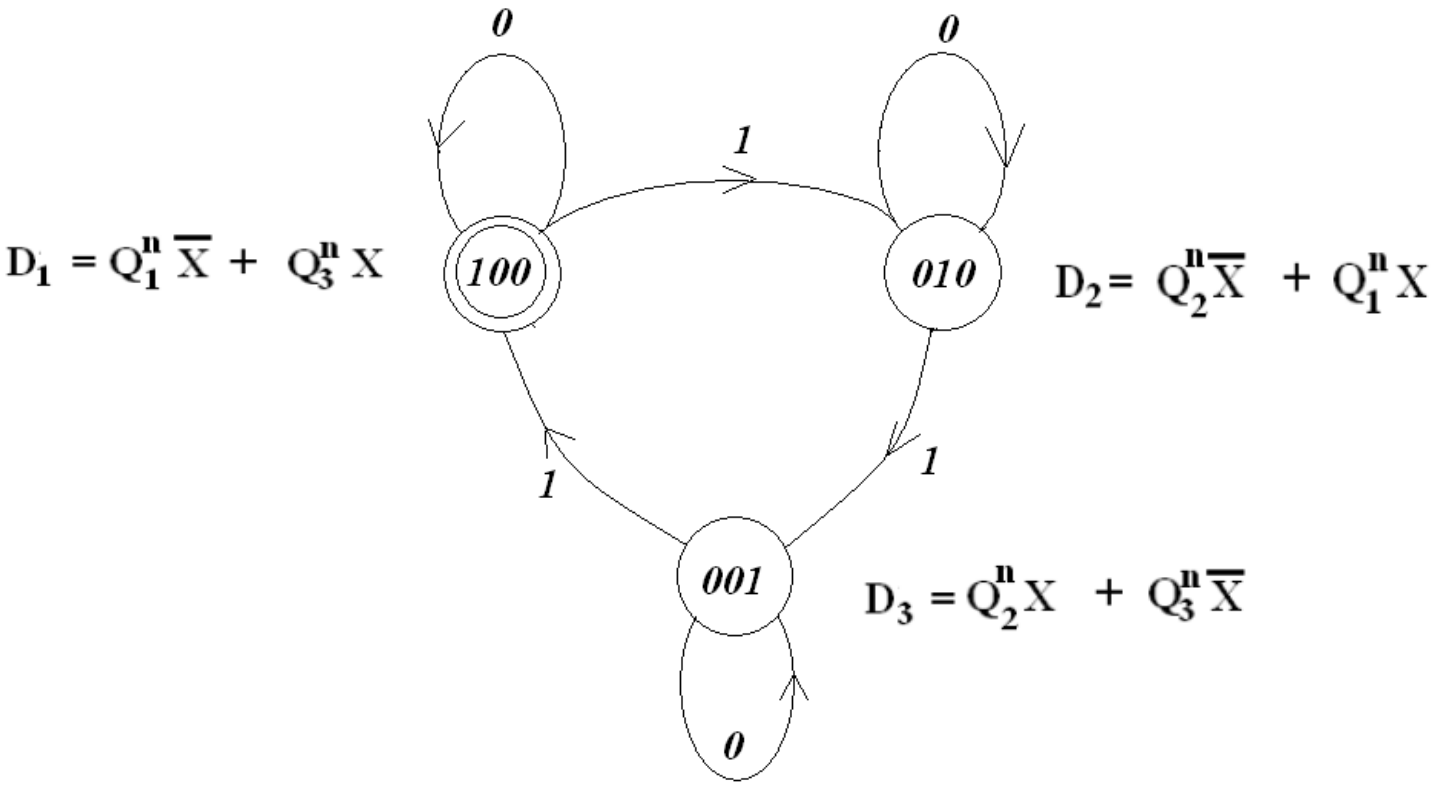
Vegyük észre! ☺ :

a **D** bemenetek vezérlésének megvalósítása ilyenkor rendkívül egyszerű, és az állapotgráf alapján elvégezhető: minden egyes **D** bemenetre akkor és csak akkor kell '1'-et kapcsolni, amikor az általa reprezentált tároló kimenetnek '0'-ról '1'-re kell változnia, azaz amikor az adott flip-flop által reprezentált állapotnak be kell állnia. Ez pedig az állapotgráfból egyértelműen kiolvasható. Az így kapott algebrai függvényalak megegyezik a Karnaugh táblák alapján meghatározottakkal.

Széchenyi István Egyetem

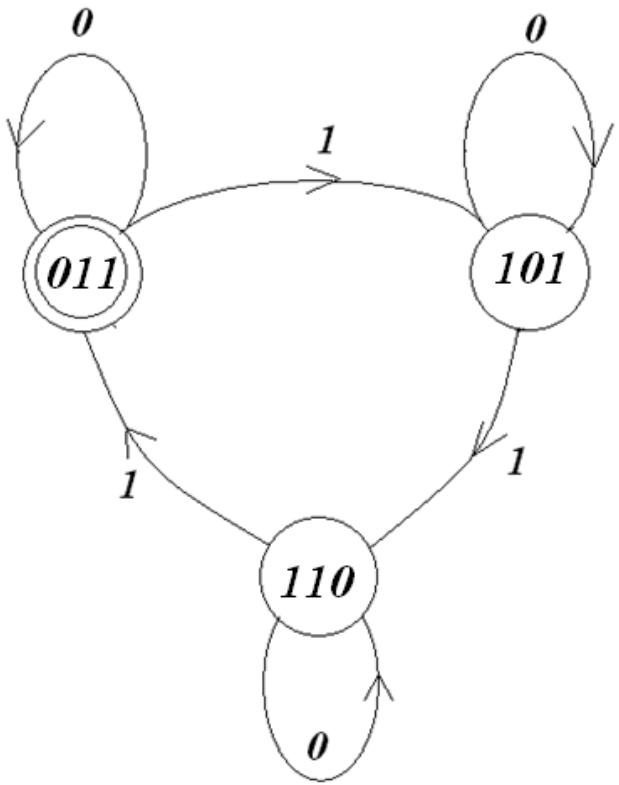
- 3. mintafeladat -

Megoldás 2. (‚1'-es súlyozású kódolási alap):



Széchenyi István Egyetem

Példa egy hasonló, de nem ‚1'-es súlyozású specifikációra:



?

Vegyük észre! ☺ ….

Szinkron sorrendi hálózatok tervezése mintapéldákon keresztül I.

- 4. mintafeladat -

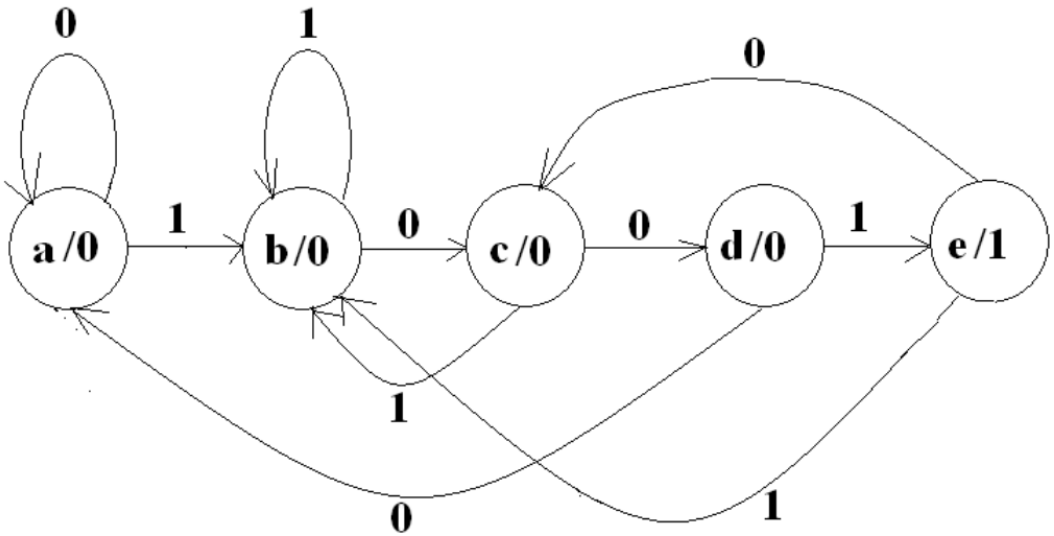
*Tervezzük meg egyes súlyú állapotkóddal, ‚Preset' és ‚Clear' bemenettel is rendelkező D-MS flip-flopokkal azt az egybemenetű (**X**) és egykimenetű (**Z**), Moore-típusú szinkron sorrendi hálózatot, amely **Z**=1 jelzéssel mutatja meg az …-1-0-0-1-… bemeneti sorozat megjelenését egy soros bemeneti szalagon!*

(*Hf.: tervezzük meg a feladatot intuitív módon (‚szabadon') kódolt szekunder változókkal is!)*

- 4. mintafeladat -

Megoldás:

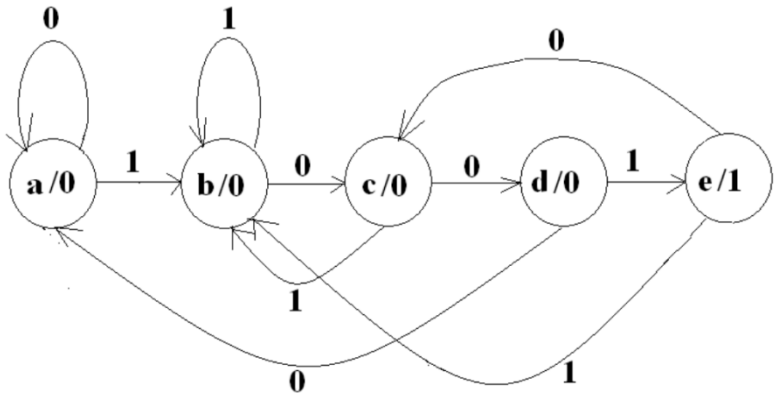
1. Az állapotgráf felrajzolása:



Széchenyi István Egyetem

- 4. mintafeladat -

2. ’1’-es súlyozású kódolás:



⇒

- egy fiktív kódolás:

	Q_1	Q_2	Q_3	Q_4	Q_5
a	1	0	0	0	0
b	0	1	0	0	0
c	0	0	1	0	0
d	0	0	0	1	0
e	0	0	0	0	1

⇒

3. Függvények felírása:

$$D_1 = Q_1^n \overline{X} + Q_4^n \overline{X}$$

$$= (Q_1^n + Q_4^n)\,\overline{X}$$

$$D_2 = Q_1^n X + Q_2^n X + Q_3^n X + Q_5^n X$$

$$= \overline{Q}_4^n X$$

$$D_3 = Q_2^n \overline{X} + Q_5^n \overline{X}$$

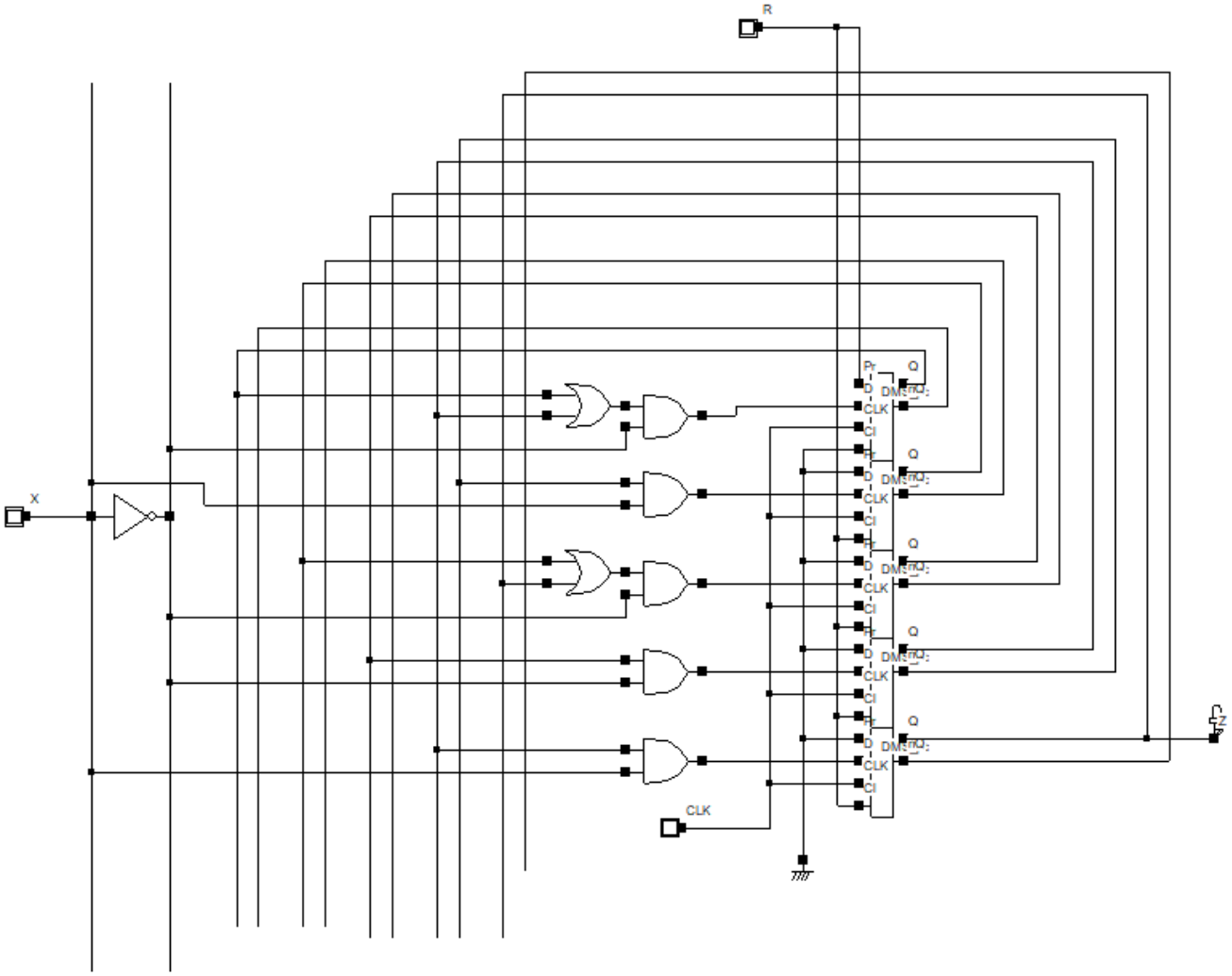
$$(Q_2^n + Q_5^n)\overline{X}$$

$$D_4 = Q_3^n \overline{X}$$

$$D_5 = Q_4^n X$$

- 4. mintafeladat -

4. Realizáció a kezdeti állapot biztosításával (RESET):



$$D_1 = Q_1^n \overline{X} + Q_4^n \overline{X}$$

$$= (Q_1^n + Q_4^n)\,\overline{X}$$

$$D_2 = Q_1^n X + Q_2^n X + Q_3^n X + Q_5^n X$$

$$= \overline{Q_4^n}\, X$$

$$D_3 = Q_2^n \overline{X} + Q_5^n \overline{X}$$

$$(Q_2^n + Q_5^n)\overline{X}$$

$$D_4 = Q_3^n \overline{X}$$

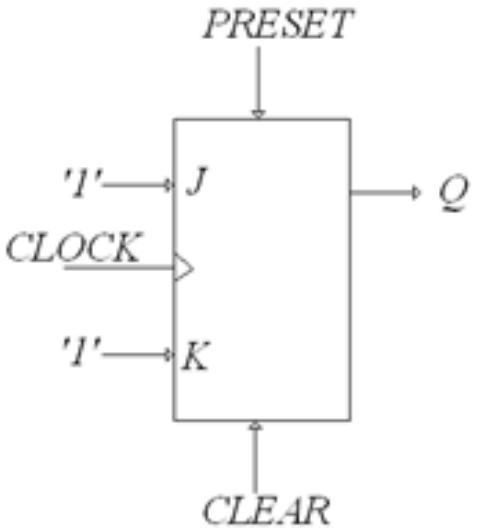
$$D_5 = Q_4^n X$$

Széchenyi István Egyetem

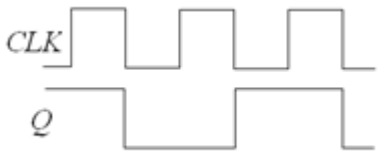
Szinkron sorrendi hálózatok tervezése mintapéldákon keresztül II.:

számlálókkal felépített szinkron sorrendi hálózatok tervezése

*A **JK-MS** tároló, mint a számlálók alapeleme: a kettes osztó funkció*

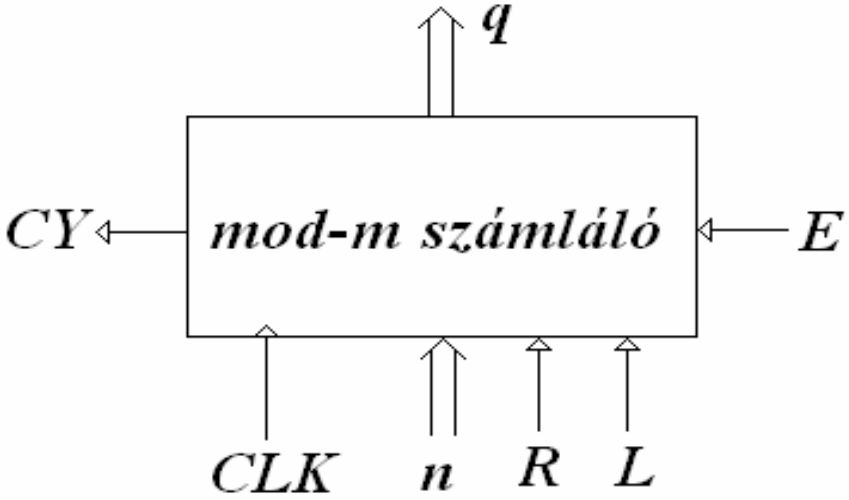


$JK(,11') \rightarrow Q^{n+1}=\overline{Q^n}$



Szinkron számlálók

szinkron *mod-m* számláló általános szimbóluma:



bemenetek:

- R: reset
- L: load
- E: enable

} vezérlőbemenetek, prioritás: R, L, E

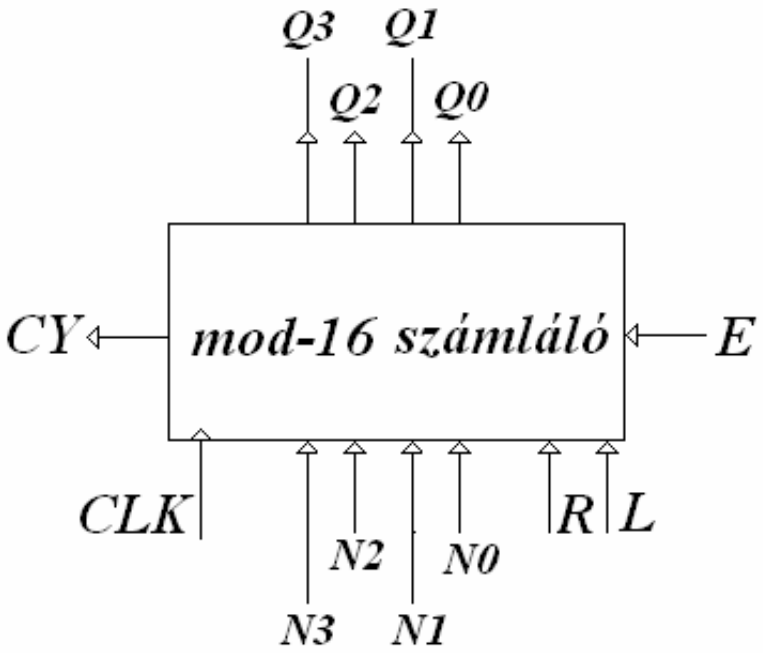
- CLK: clock
- n: bitvektor adatbemenetek

kimenetek:

- q: adatkimenetek(„szám”)
- CY: carry

Szinkron számlálók

Példa: szinkron mod-16 számláló szimbóluma:



bemenetek:

- *R*: reset
- *L*: load
- *E*: enable

} vezérlőbemenetek, prioritás: R, L, E

- *CLK*: clock
- *N3-N0*: bitvektor adatbemenetek

kimenetek:

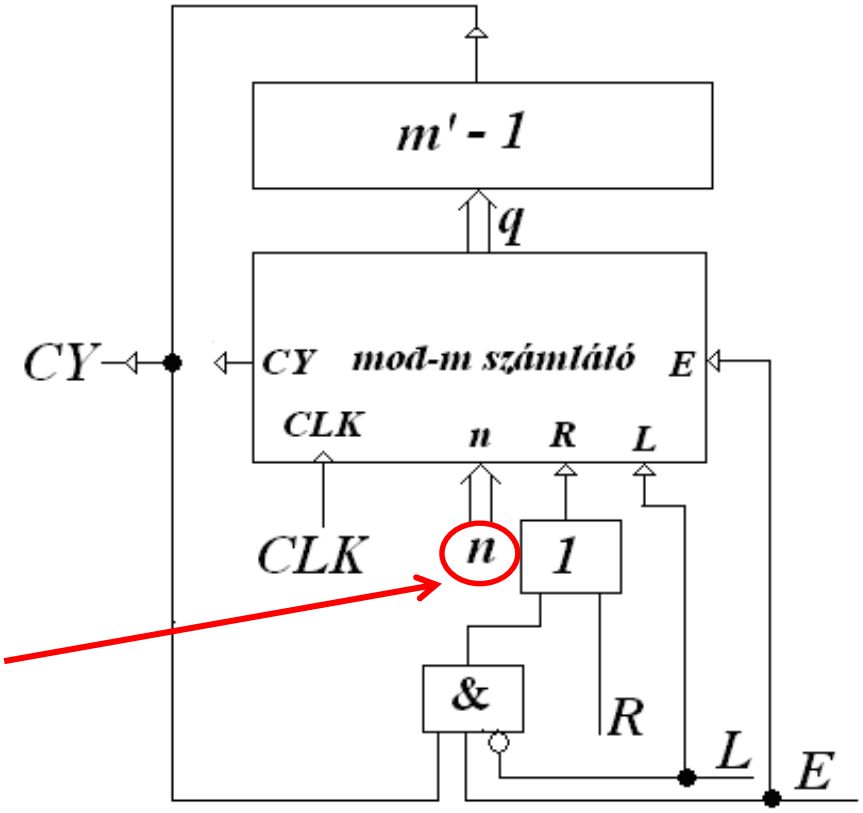
- *Q3-Q0*: adatkimenetek(„szám”)
- *CY*: carry

Szinkron számlálók

Adott modulusú számláló átalakítása más modulusúvá

Feltétel:

$m' < m$

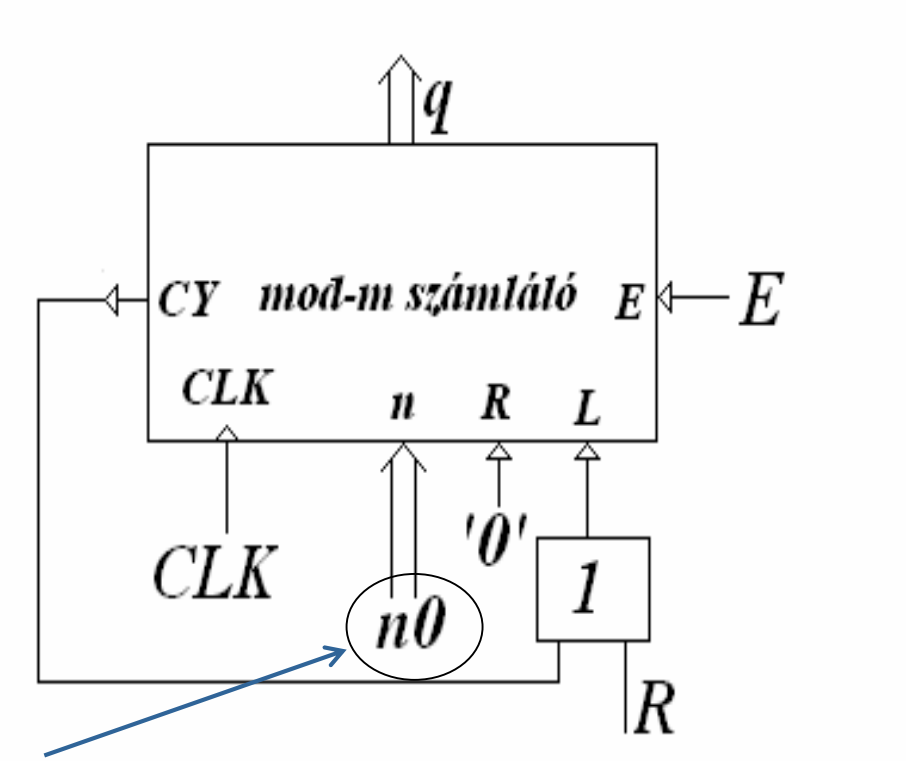


Szabály:

n ≤ m'-1 !!

Szinkron számlálók

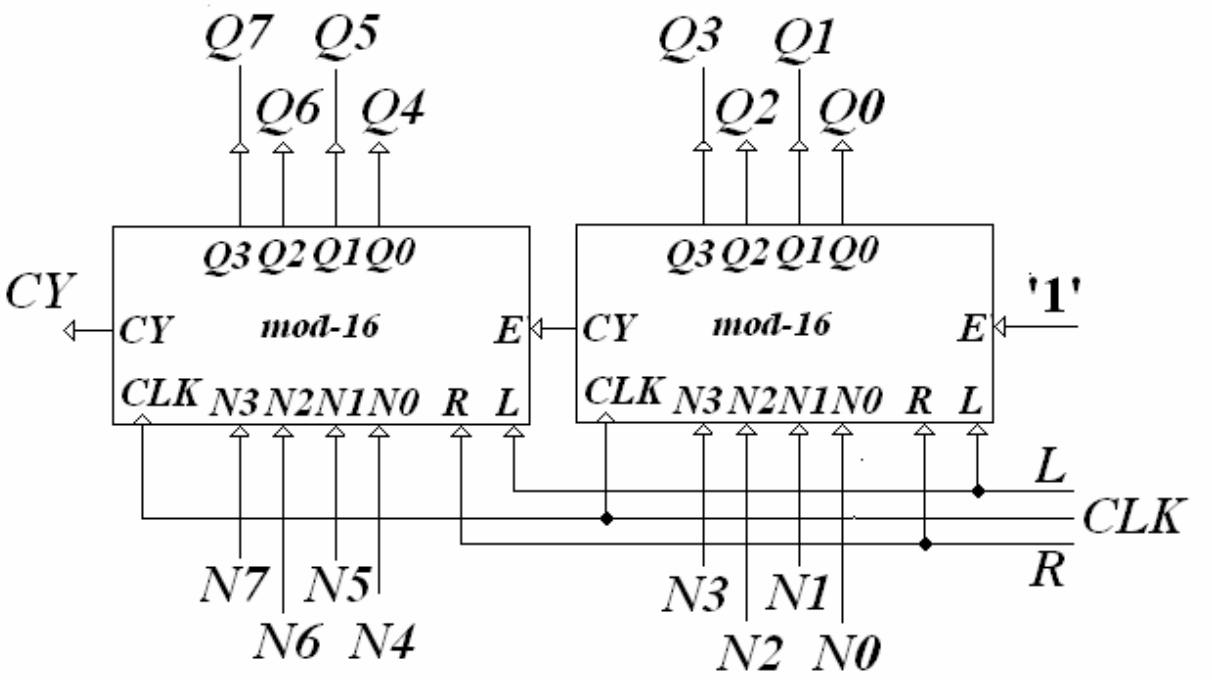
Számláló nullától különböző kezdőértékének beállítása



n0 : „kezdőérték”

Szinkron számlálók

Példa: mod-256-os számláló mod-16 számlálókból



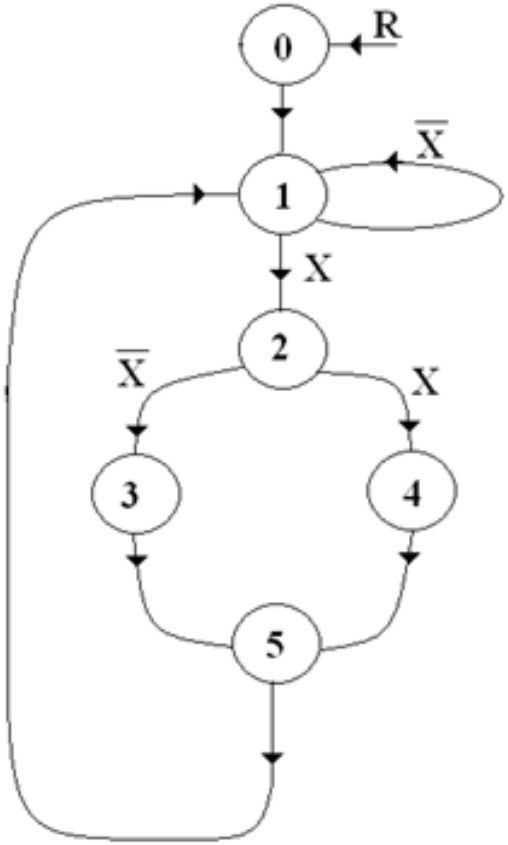
Szinkron sorrendi hálózat tervezése szinkron számlálóval

- 1. mintafeladat -

Valósítsuk meg törölhető (R), betölthető (L) és engedélyezhető (E) számlálóval és multiplexerekkel a mellékelt kódolt állapotgráf szerinti szinkron sorrendi hálózatot!

Széchenyi István Egyetem

- 1. mintafeladat -

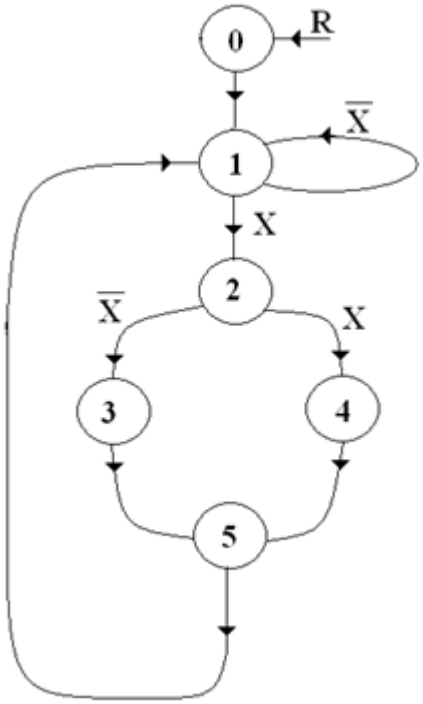


állapot kimenetű kódolt állapotgráf

Széchenyi István Egyetem

- 1. mintafeladat -

Megoldás:



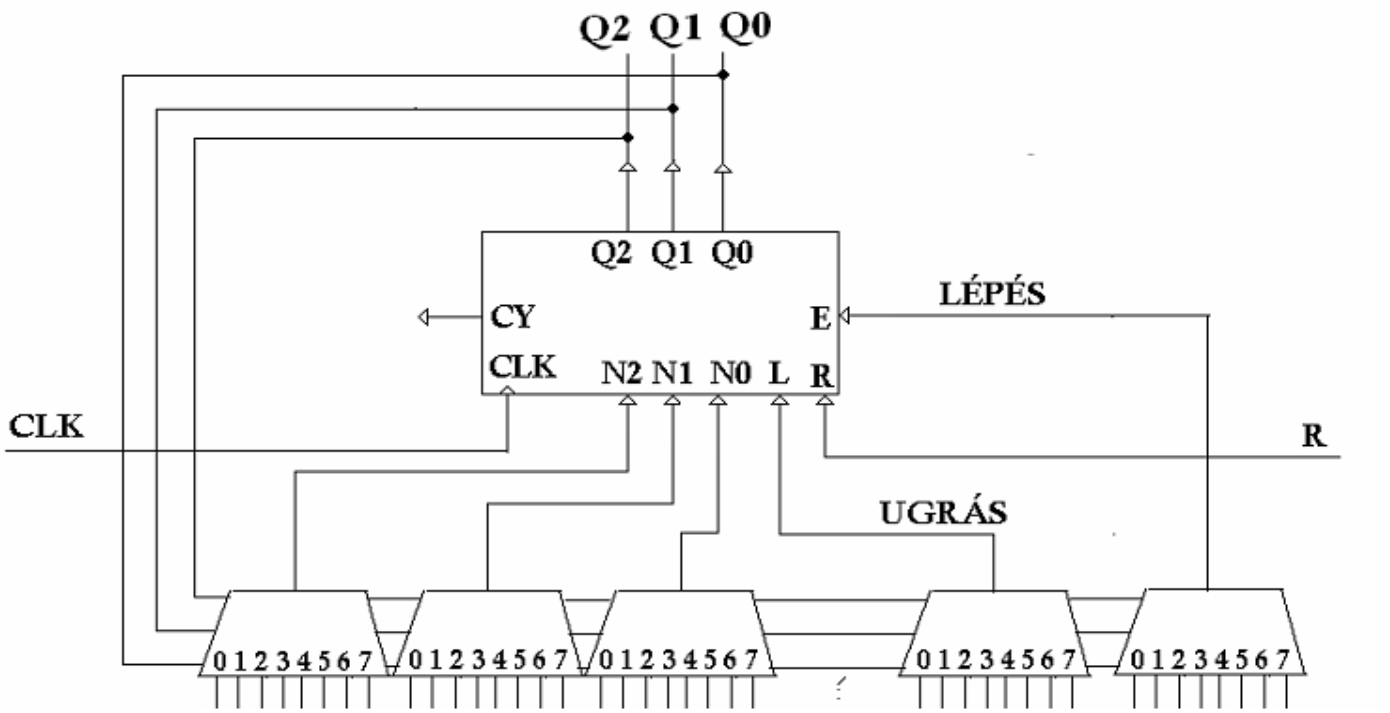
Táblázatok a megvalósításhoz:

LÉPÉS (E=1)		UGRÁS (L=1)		
akt. áll.	feltétel	akt.áll.	feltétel	köv.áll.
0	'1'	2	X	4
1	X	3	'1'	5
2	$\overline{X}$	5	'1'	1
4	'1'			

Széchenyi István Egyetem

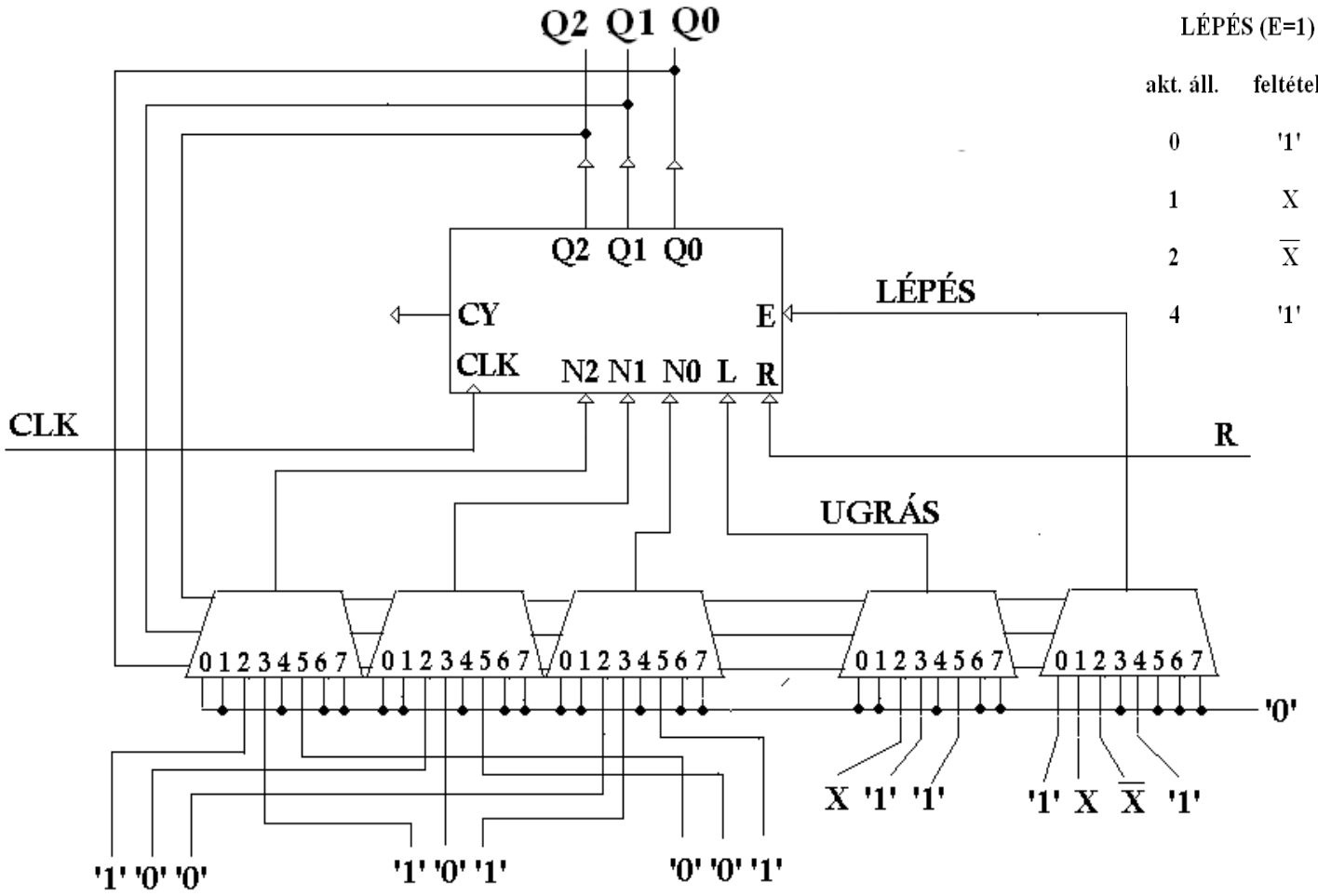
- 1. mintafeladat -

Speciális célarchitektúra szinkron sorrendi hálózatok számlálós megvalósítására



Széchenyi István Egyetem

- 1. mintafeladat -



LÉPÉS (E=1)		UGRÁS (L=1)		
akt. áll.	feltétel	akt.áll.	feltétel	köv.áll.
0	'1'	2	X	4
1	X	3	'1'	5
2	$\overline{X}$	5	'1'	1
4	'1'			

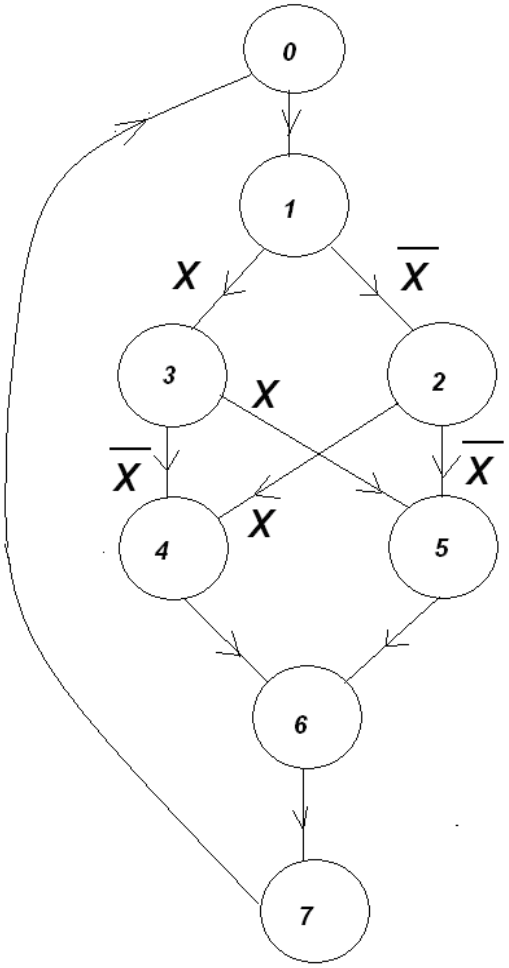
Szinkron sorrendi hálózat tervezése szinkron számlálóval

- 2. mintafeladat -

Valósítsuk meg törölhető (R), betölthető (L) és engedélyezhető (E) számlálóval és multiplexerekkel a mellékelt kódolt állapotgráf szerinti szinkron sorrendi hálózatot!

Széchenyi István Egyetem

- 2. mintafeladat -

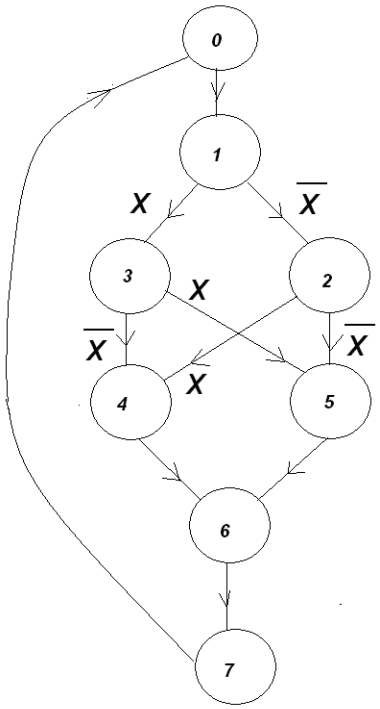


állapot kimenetű kódolt állapotgráf

Széchenyi István Egyetem

- 2. mintafeladat -

Megoldás:



Táblázatok a megvalósításhoz:

lépések		*ugrások*		
akt. áll	*feltétel*	*akt. állapot*	*feltétel*	*köv. állapot*
0	*'1'*	*1*	*X*	*3*
1	$\overline{X}$	*2*	$\overline{X}$	*5*
3	$\overline{X}$	*2*	*X*	*4*
5	*'1'*	*3*	*X*	*5*
6	*'1'*	*4*	*'1'*	*6*
7	*'1'*			

- 2. mintafeladat -

Kettős ugrás az „ugrások” táblában:

lépések		ugrások		
akt. áll	feltétel	akt. állapot	feltétel	köv. állapot
0	'1'	1	X	3
1	$\overline{X}$	2	$\overline{X}$	5
3	$\overline{X}$	2	X	4
5	'1'	3	X	5
6	'1'	4	'1'	6
7	'1'			

Széchenyi István Egyetem

- 2. mintafeladat -

Megoldás a kettős ugrás kezelésére:

akt. állapot	feltétel	köv. állapot
2	$\overline{X}$	5
2	X	4

$\Rightarrow$

segédtáblázat a kettős ugrás lekezelésére

	N2	N1	N0
$\overline{X}$	1	0	1
X	1	0	0

Ha $q = 2$,

$N2(D2) = 1$

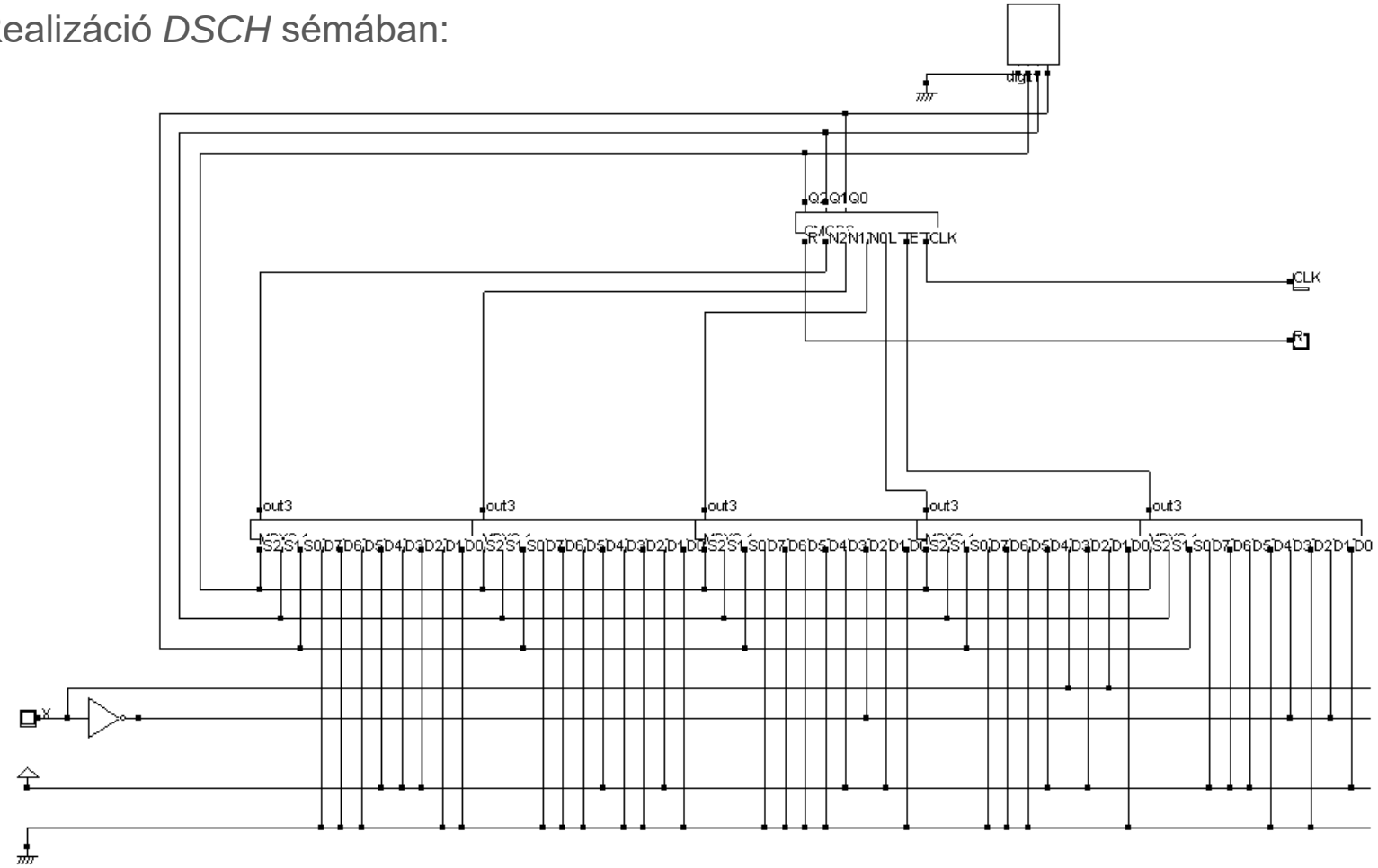
$N1(D2) = 0$

$N0(D2) = \overline{X}$

Széchenyi István Egyetem

- 2. mintafeladat -

Realizáció *DSCH* sémában:



- 2. mintafeladat -

A feladat megoldásának DSCH sémája („SZH_szamlalo_eloadasmintafeladat_2_dualishoz”):

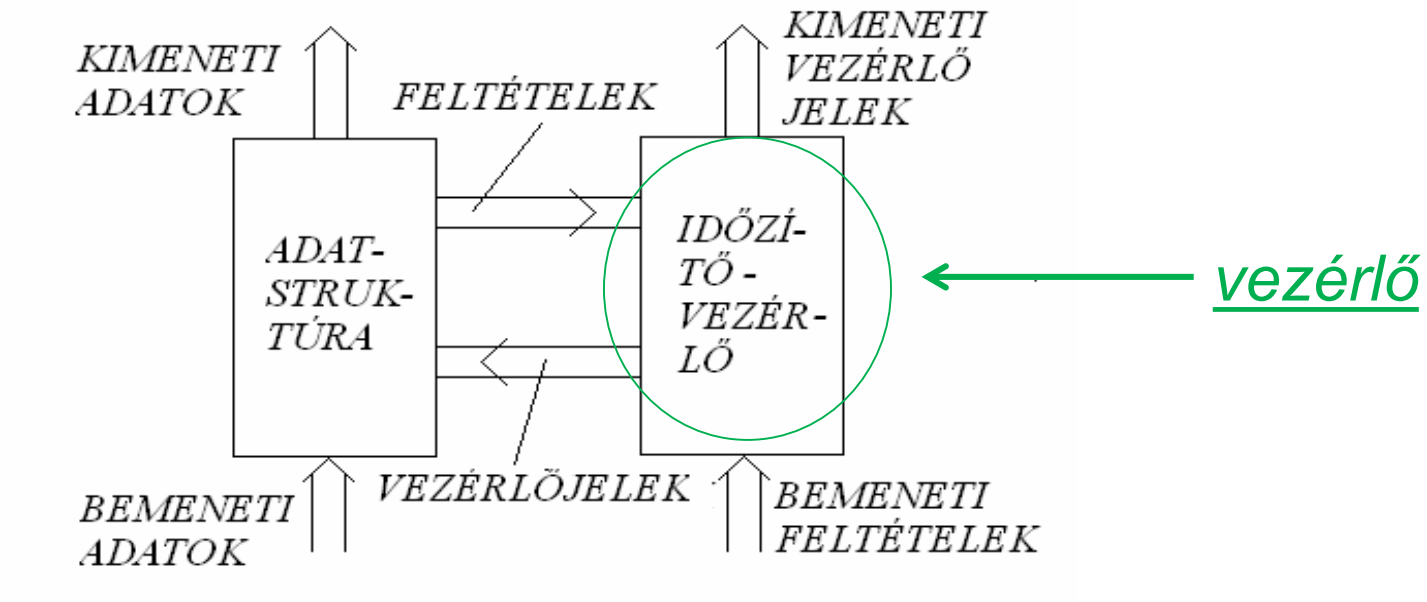
http://www.sze.hu/~somi/Digit%e1lis%20h%e1l%f3zatok/DSCH%20s%e9m%e1k%20gyakorl%f3%20feladatokhoz/

Széchenyi István Egyetem

Szinkron sorrendi hálózat tervezése szinkron számlálóval:

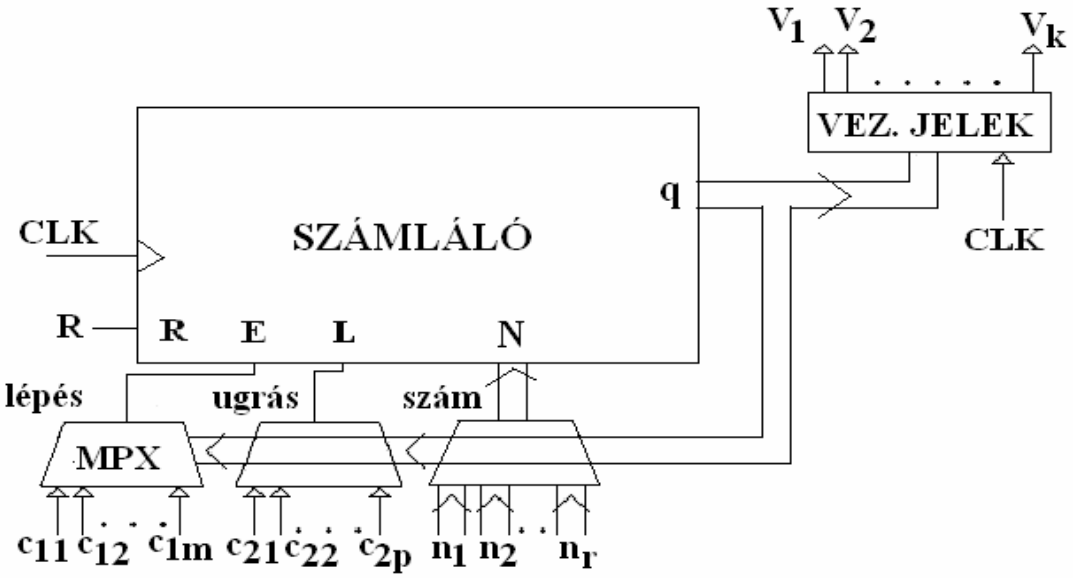
szinkron számlálók alkalmazása időzítő-vezérlő egységekben

A digitális egység felbontása adat- és vezérlő-alegységre :



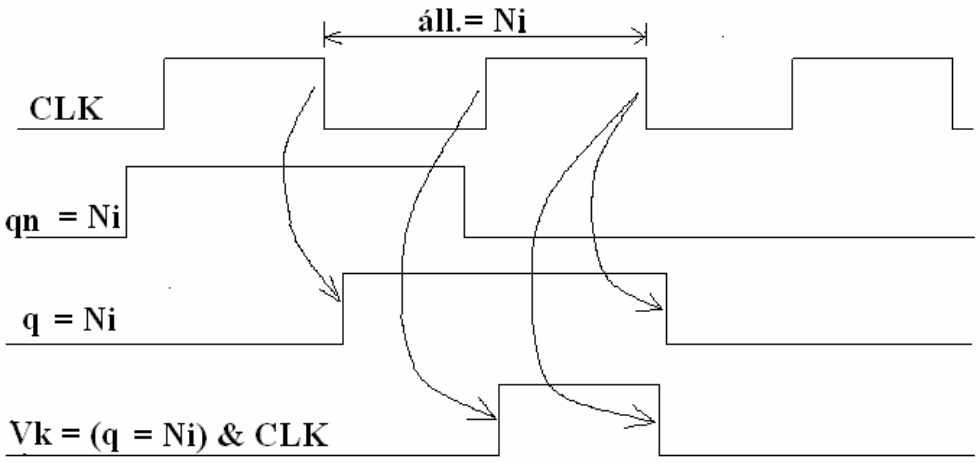
Széchenyi István Egyetem

Számláló-típusú vezérlők



← *a struktúra*

A hazárdmentes vezérlés feltétele:



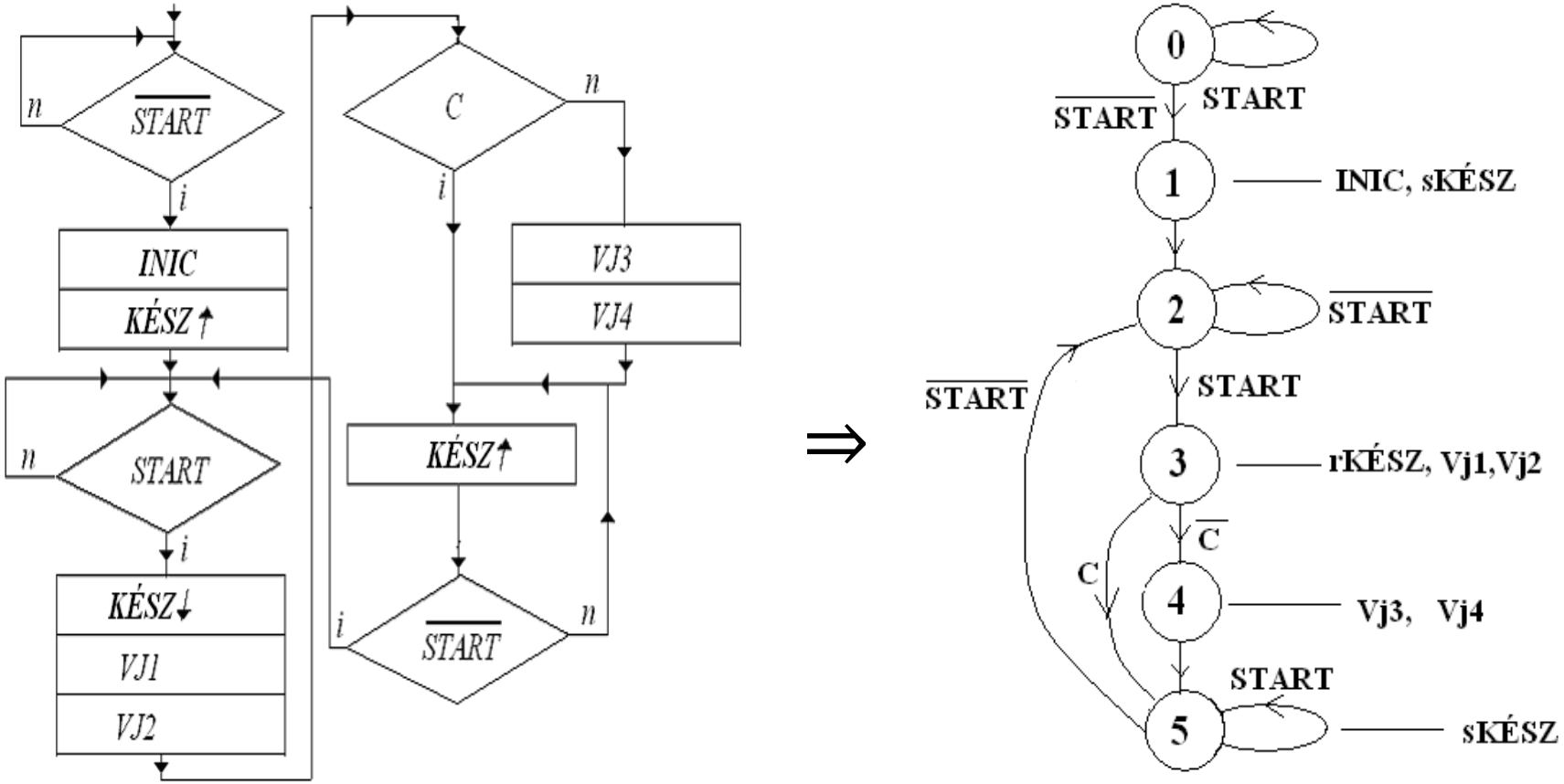
Szinkron sorrendi hálózat tervezése szinkron számlálóval

- 3. mintafeladat -

Tervezzünk időzítő-vezérlő egységet szinkron számláló felhasználásával a mellékelt folyamatábra alapján!

Széchenyi István Egyetem

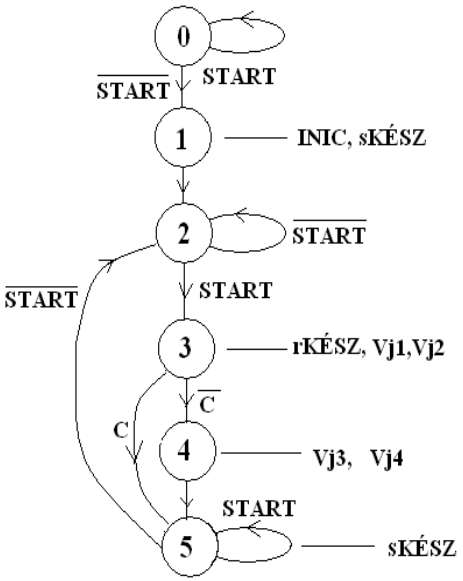
- 3. mintafeladat -



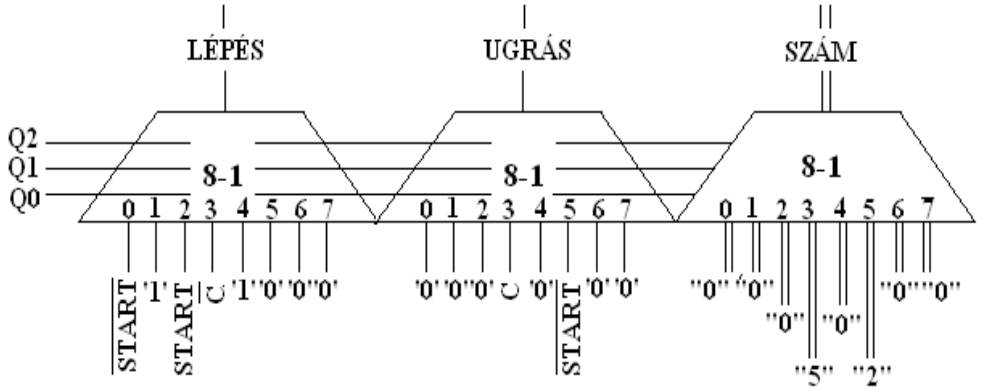
Széchenyi István Egyetem

- 3. mintafeladat -

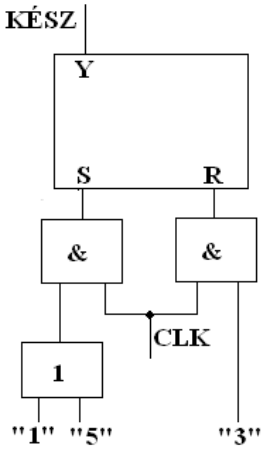
Megoldás:

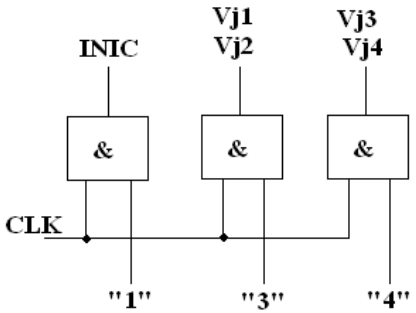


állapot-kimenetű kódolt állapotgráf



A multiplexerek programozása



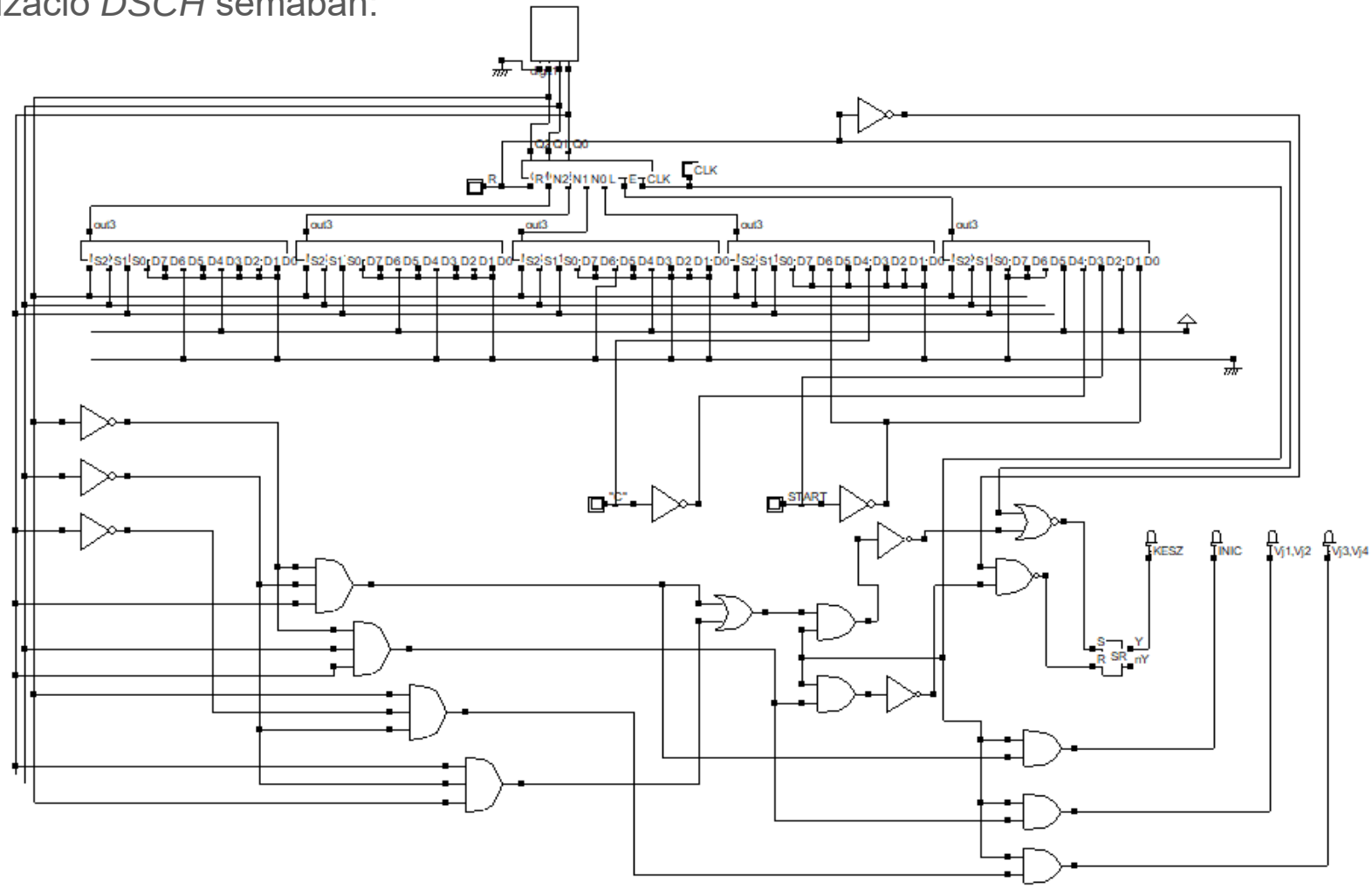


a vezérlőjelek realizációja

Széchenyi István Egyetem

- 3. mintafeladat -

Realizáció *DSCH* sémában:



- 3. mintafeladat -

A feladat megoldásának DSCH sémája („SZH_szamlalo_vezerlo_eloadasmintafeladat_3_dualishoz”):

http://www.sze.hu/~somi/Digit%e1lis%20h%e1l%f3zatok/DSCH%20s%e9m%e1k%20gyakorl%f3%20feladatokhoz/

Aszinkron sorrendi hálózatok tervezése mintapéldákon keresztül

aszinkron szekvenciális hálózat: aszinkron sorrendi hálózatokban nincs szinkronizáló jel (órajel), azaz a következő állapot (*n+1*) értelmezését nem egy órajel ciklus lezajlása után determináljuk. Ezekben a hálózatokban általánosságban azt feltételezhetjük, hogy a bemenet minden egyes változása egy új állapotot generál.

(megjegyzés: az új állapot (*n+1*) nem feltétlenül különbözik (a szekunder változók és a kimenetek tekintetében) az előzőtől (*n*), ezt mindig a specifikáció szerinti - adott időpontbeli - működési feltételek határozzák meg.)

Aszinkron sorrendi hálózatok tervezése mintapéldákon keresztül

- 1. mintafeladat -

*Közvetlenül visszacsatolt kombinációs hálózattal tervezzünk olyan egybemenetű (**X**) és egykimenetű (**Z**) aszinkron sorrendi hálózatot, amelynek kimenetén a szint mindannyiszor ellenkezőjére vált, ahányszor a bemenet magas szintről alacsony szintre vált.*
*Bekapcsolás után a hálózat az **X**=0 bemenetnél **Z** = 0 kimenetet szolgáltasson!*

- 1. mintafeladat -

A feladat megoldásához a következő, általános szisztematikus lépéssort alkalmazzuk:

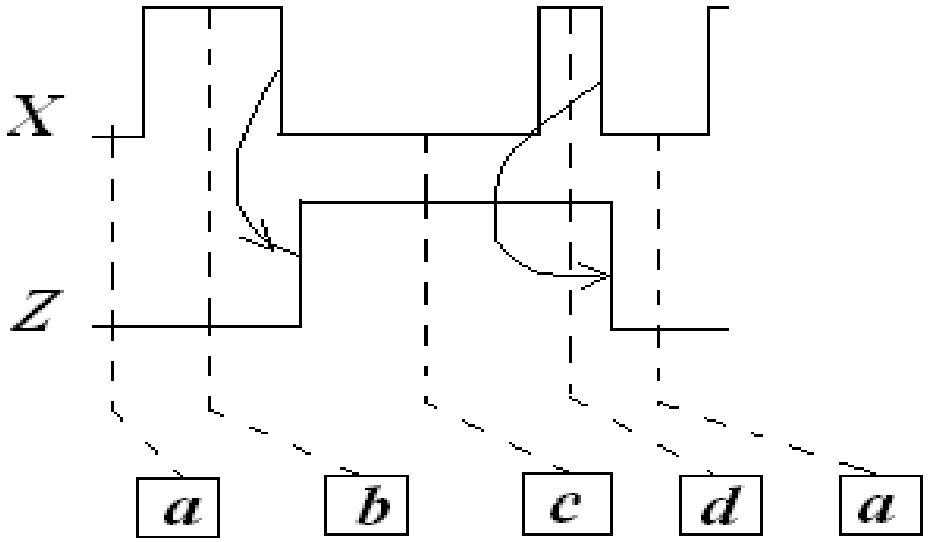
1. *Idődiagram (vagy állapotgráf) felrajzolása.*
2. *Előzetes szimbolikus állapottábla felvétele.*
3. *Összevont szimbolikus állapottábla megszerkesztése.*
4. *Kódolt állapottábla elkészítése a kritikus versenyhelyzet elemzésével.*
5. *- közvetlen visszacsatolt kombinációs hálózat esetében a szekunder változó(k) és a kimenet(ek) függvényeinek Karnaugh tábla segítségével történő statikus hazárdmentesített megadása algebrai alakban;*

   - SR tárolók alkalmazása esetén a vezérlési tábla, valamint a szükséges Karnaugh táblák alapján az SR tároló(k) és a kimenet(ek) statikusan hazárdmentesített algebrai függvényalakjainak megadása.
6. *Kezdeti állapotról történő gondoskodás.*
7. *Lényeges hazárdok vizsgálata (szükség esetén késleltetések beiktatása).*
8. *Realizáció logikai kapukkal (és tárolókkal).*

- 1. mintafeladat -

Megoldás** (Mealy-modell**):

1. lépés: idődiagram(ütemdiagram) felrajzolása



Széchenyi István Egyetem

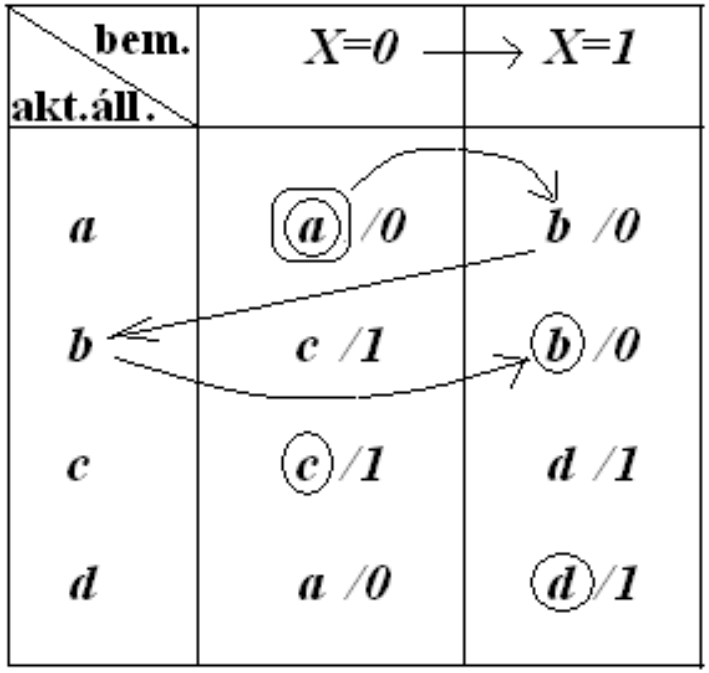
- 1. mintafeladat -

2. lépés: A feladat absztrakt szimbolikus állapottáblájának megszerkesztése

előzetes szimbolikus állapottábla

bem. / akt.áll.	X=0	X=1
a	(a) /0	b /0
b	c /1	(b) /0
c	(c) /1	d /1
d	a /0	(d) /1

Egy stabil-stabil átmenet ($a \rightarrow b$) az állapottáblán:



Széchenyi István Egyetem

- 1. mintafeladat -

3. lépés: összevont szimbolikus állapottábla megszerkesztése

előzetes szimbolikus állapottábla

bem. / akt.áll.	X=0	X=1
a	(a)/0	b /0
b	c /1	(b)/0
c	(c)/1	d /1
d	a /0	(d)/1

Az előzetes szimbolikus állapottáblán – az összevonás feltételeit figyelembe véve- nem tudunk állapot összevonást végezni. A kódolt állapottáblát így e táblázat alapján kell elkészíteni.

Széchenyi István Egyetem

- 1. mintafeladat -

4. lépés: kódolt állapottábla elkészítése, kritikus versenyhelyzet elemzés

egy lehetséges kódkiosztás:

	Y_1	Y_2
a :	0	0
b :	0	1
c :	1	0
d:	1	1

$\Rightarrow$

kódolt állapottábla

kódolt aktuális állapot $Y_1^v Y_2^v$	kódolt következő állapot/kimenet	
	X=0 $Y_1^v Y_2^v$/Z	X=1 $Y_1^v Y_2^v$/Z
0 0	0 0/0	0 1/0
0 1	1 0/1	0 1/0
1 0	1 0/1	1 1/1
1 1	0 0/0	1 1/1

Széchenyi István Egyetem

- 1. mintafeladat -

- kritikus versenyhelyzet elemzése a kódolt állapottáblán

kódolt aktuális állapot $Y_1^v Y_2^v$	kódolt következő állapot/kimenet	
	X=0 $Y_1^v Y_2^v$/Z	X=1 $Y_1^v Y_2^v$/Z
0 0	0 0/0	0 1/0
0 1	1 0/1	0 1/0
1 0	1 0/1	1 1/1
1 1	0 0/0	1 1/1

Egy ideális, ’01’→’10’ stabil-stabil állapot átmenetet

Széchenyi István Egyetem

- 1. mintafeladat -

Valóságos állapot-átmenet: kritikus versenyhelyzet az $Y_1^v Y_2^v$ =*'01'→'10'* állapot átmenet esetében!

kódolt aktuális állapot $Y_1^v Y_2^v$	kódolt következő állapot/kimenet	
	X=0 $Y_1^v Y_2^v$/Z	X=1 $Y_1^v Y_2^v$/Z
0 0	*0 0/0*	*0 1/0*
0 1	*1 0/1*	1. *0 1/0*
1 0	*1 0/1*	2. *1 1/1*
1 1	*0 0/0*	*1 1/1*

<u>*1.eset:*</u> $Y_1^v Y_2^v$ =*'01'→'00'→'10'*

- az *X=1→0* hatására az $Y_1^v Y_2^v = 00$ sorba ugorva a *'00'* kódú állapotban stabilizálódik a hálózatunk:
  ⇒ HIBÁS működés!!

<u>*2.eset:*</u> $Y_1^v Y_2^v$ =*'01'→'11'→'10'*

- az *X=1→0* hatására először az $Y_1^v Y_2^v = 11$ sorba ugorva a *'00'* tranziens kód jelenik meg. Innen továbblépve az $Y_1^v Y_2^v = 00$ sorba ismét a *'00'* kódú állapotban stabilizálódik a hálózatunk:
  ⇒ HIBÁS működés!!

- 1. mintafeladat -

Kritikus versenyhelyzet:

amennyiben egy tranziens állapot kódja egynél több szekunder változó értékében különbözik a kiinduló stabil állapot kódjától, a reális hálózaton - az eltérő jelkésleltetési utak miatt - átmenetileg olyan más tranziens állapotok is jelentkezhetnek az f_y hálózat kimenetén, amelyek stabilizálódhatnak. Ezzel más, a specifikációnak ellentmondó pályára áll az aszinkron hálózat.

*Az ilyen hibalehetőségeket **kritikus versenyhelyzet**eknek nevezzük.*

Széchenyi István Egyetem

- 1. mintafeladat -

A kritikus versenyhelyzet kiküszöbölése:
az állapotkódolás megváltoztatása

egy másik lehetséges kódkiosztás:

	Y_1	Y_2
a :	0	0
b :	0	1
c :	1	1
d:	1	0

$\Rightarrow$

kódolt állapottábla

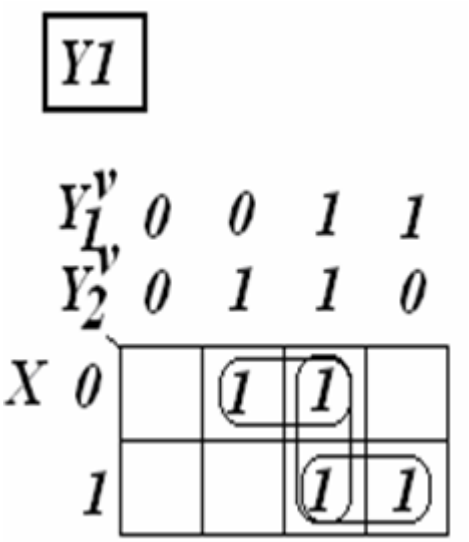
kódolt aktuális állapot $Y_1^v Y_2^v$	kódolt következő állapot/kimenet	
	X=0 $Y_1^v Y_2^v$/Z	X=1 $Y_1^v Y_2^v$/Z
0 0	0 0/0	0 1/0
0 1	1 1/1	0 1/0
1 1	1 1/1	1 0/1
1 0	0 0/0	1 0/1

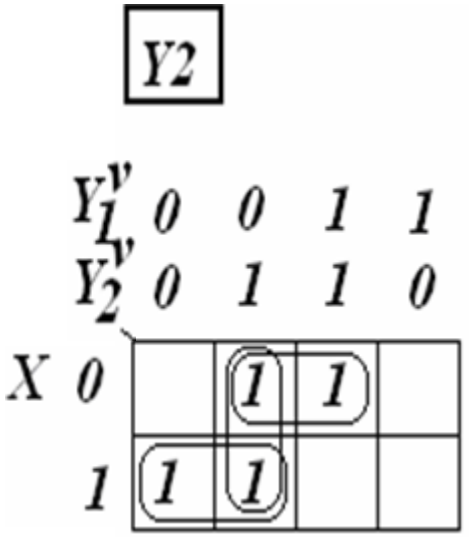
Nincs kritikus versenyhelyzet!

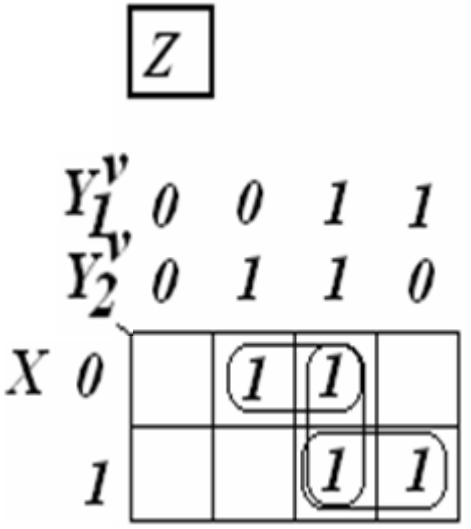
Széchenyi István Egyetem

- 1. mintafeladat -

5. lépés: a szekunder változók és a kimenet függvényeinek Karnaugh tábla segítségével történő megadása (hazárdmentesített függvényalak!!)







$$Y_1 = Y_2^v \overline{X} + Y_1^v X + Y_1^v Y_2^v$$

$$Y_2 = \overline{Y_1^v} X + \overline{Y_1^v} Y_2^v + Y_2^v \overline{X}$$

$$Z = Y_1$$

- 1. mintafeladat -

6. lépés: a kezdeti állapot beállásának ellenőrzése

Az ellenőrzés menete:

a megoldásként kapott függvényekbe, mint egyenletekbe be kell helyettesíteni a kezdeti állapothoz tartozó ismert és előírt bemeneti értéke(ke)t : ha egyértelműek az eredmények, és teljesül az egyenlőség, akkor nem kell külön kezdő állapot beállító jelről (‚RESET') gondoskodni.

Amennyiben azonban a behelyettesítéssel kapott eredmény nem az előírás szerinti kezdő értékeket biztosítja a kimenet(ek) és a szekunder változó(k) tekintetében, illetve nem egyértelmű az eredmény, akkor a következő szabály alapján kell eljárni:

⇓

- 1. mintafeladat -

A kezdeti állapot beállításának általános feltételei(sorrendje):

1. a kezdeti állapot kódját rá kell kényszeríteni az f_y hálózatra, a visszacsatolástól függetlenítve ezeket a bemeneteket. Ezt a helyzetet legalább addig kell fenntartani, amíg az f_y kimenetein kialakul a kezdeti állapot kódja (illetve, ha **SR** tárolókkal történik a visszacsatolás, akkor azok kimenetén alakul ki ez a kód),

2. rá kell kapcsolni a hálózatra azt a bemeneti értéket (kombinációt), amely a kezdeti állapothoz tartozik,

3. végül meg kell szüntetni a kényszerített visszacsatoló ágat (állapotot), és helyre kell állítani az eredeti visszacsatolást, ezzel a hálózat a kezdeti állapotban stabilizálódik.

Széchenyi István Egyetem

- 1. mintafeladat -

A kezdeti állapot beállásának ellenőrzése az 1. mintafeladatban:

$$Y_1 = Y_2^v \overline{X} + Y_1^v X + Y_1^v Y_2^v$$

$$Y_2 = \overline{Y_1^v} X + \overline{Y_1^v} Y_2^v + Y_2^v \overline{X}$$

$$Z = Y_1$$

kezdőállapotban:
$X=0$

$\Rightarrow$

$$0 = Y_2^v \cdot 1 + Y_1^v \cdot 0 + Y_1^v Y_2^v$$

$$0 = \overline{Y_1^v} \cdot 0 + \overline{Y_1^v} Y_2^v + Y_2^v \cdot 1$$

$$0 = Y_1$$

bizonytalan!

Konklúzió:

gondoskodni kell a kezdeti állapot beállításáról! → ‚RESET' (R) jel beiktatása

- 1. mintafeladat -

A ‚RESET' (‚R') jel beiktatásának elve (a feladat specifikációja alapján):

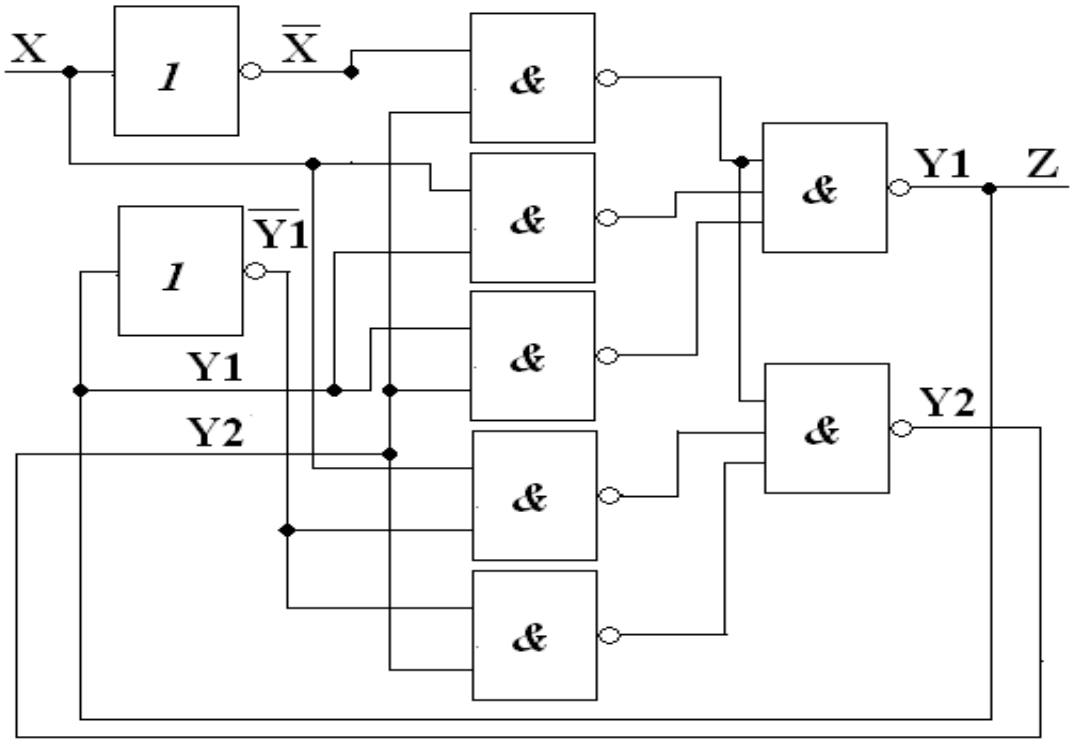
hasítsuk fel a visszacsatolásokat (Y_1^v; Y_2^v), és illesszünk be a visszacsatoló körbe 2 darab kétbemenetű logikát. Az egyik bemenetük közös, a kezdeti állapotba kényszerítő '*R*' (reset) jel, a másik bemenetük a visszacsatolandó szekunder változókra (Y_1^v; Y_2^v) kapcsolandó. A kimeneteket kapcsoljuk az f_y hálózat bemeneteire. Mindkét *R*-logika az *R* = '1' esetben '0'-át ad tovább, ez pedig a kezdeti állapot kódja. Ha *R=0*, a logikák kimenetére a megfelelő szekunder változó kerül, tehát él a visszacsatolás.

Széchenyi István Egyetem

- 1. mintafeladat -

(7. lépés: lényeges hazárdok vizsgálata)
8. lépés: realizáció

NAND realizáció 'RESET' logika nélkül:



$$Y_1 = Y_2^v \overline{X} + Y_1^v X + Y_1^v Y_2^v$$

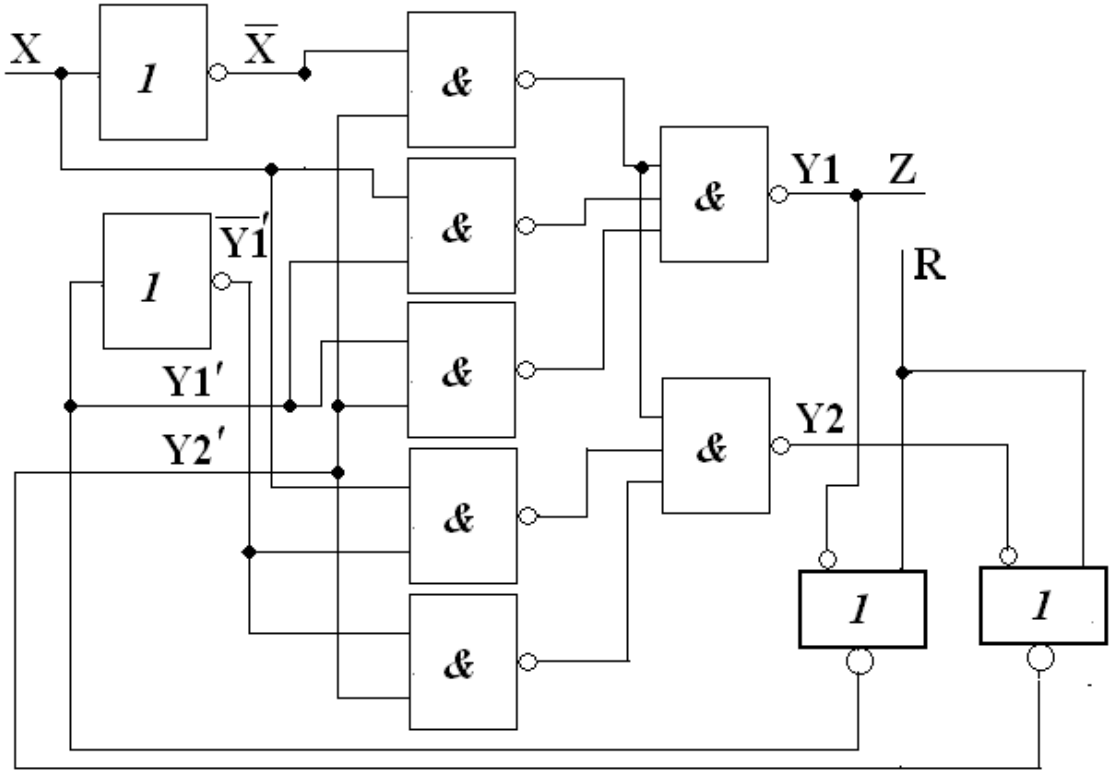
$$Y_2 = \overline{Y_1^v} X + \overline{Y_1^v} Y_2^v + Y_2^v \overline{X}$$

$$Z = Y_1$$

Széchenyi István Egyetem

- 1. mintafeladat -

NAND realizáció 'RESET' logika beiktatásával:



$$Y_1 = Y_2^v \overline{X} + Y_1^v X + Y_1^v Y_2^v$$

$$Y_2 = \overline{Y_1^v} X + \overline{Y_1^v} Y_2^v + Y_2^v \overline{X}$$

$$Z = Y_1$$

Aszinkron sorrendi hálózatok tervezése mintapéldákon keresztül

- 2. mintafeladat -

Tervezzünk kétbemenetű (X_1, X_2) ún. „sorrendi ÉS” áramkört, amelynek ***Z*** *kimenete akkor és csakis akkor ad magas szintet, ha az X_1 bemenet előbb áll ’1’-re, mint az X_2. Kezdeti állapotban az X_1X_2=**00** bemeneti bitkombináció esetén* ***Z=0****.*

A tervezést végezzük el a következő állapotot előállító hálózat közvetlen visszacsatolásával, és ***SR*** *tárolókkal történő visszacsatolással is !*

Széchenyi István Egyetem

- 2. mintafeladat -

Megoldás 1.: *közvetlen visszacsatolás alkalmazása (Mealy-modell szerint)*

(1. lépés: idődiagram vagy állapotgráf felrajzolása – elhagyható)

1. lépés: előzetes szimbolikus állapottábla felvétele a specifikáció alapján

aktuális állapot	következő állapot/kimenet			
	X_1X_2			
	00	01	10	11
a	(a)/0	b/0	c/0	-/-
b	a/0	(b)/0	-/-	d/0
c	a/0	-/-	(c)/0	e/1
d	-/-	b/0	c/0	(d)/0
e	-/-	b/0	c/0	(e)/1

Széchenyi István Egyetem

- 2. mintafeladat -

2. lépés: összevont szimbolikus állapottábla megszerkesztése

előzetes szimbolikus állapottábla

aktuális állapot	következő állapot/kimenet			
	X_1X_2			
	00	01	10	11
a	(a)/0	b/0	c/0	-/-
b	a/0	(b)/0	-/-	d/0
c	a/0	-/-	(c)/0	e/1
d	-/-	b/0	c/0	(d)/0
e	-/-	b/0	c/0	(e)/1

összevonható állapotpárok:

ab, ad, bd, ce

összevont állapotok osztályai:

abd, ce

$$abd \rightarrow \boldsymbol{s_1}$$
$$ce \rightarrow \boldsymbol{s_2}$$

Széchenyi István Egyetem

- 2. mintafeladat -

$abd \rightarrow \boldsymbol{s_1}$

$ce \rightarrow \boldsymbol{s_2}$

előzetes szimbolikus állapottábla

aktuális állapot	következő állapot/kimenet			
	X_1X_2			
	00	01	10	11
a	(a)/0	b/0	c/0	-/-
b	a/0	(b)/0	-/-	d/0
c	a/0	-/-	(c)/0	e/1
d	-/-	b/0	c/0	(d)/0
e	-/-	b/0	c/0	(e)/1

$\Rightarrow$

összevont szimbolikus állapottábla

aktuális állapot,	következő állapot/kimenet			
	X_1X_2			
	00	01	10	11
s_1	(s_1)/0	(s_1)/0	s_2/0	(s_1)/0
s_2	s_1/0	s_1/0	(s_2)/0	(s_2)/1

Széchenyi István Egyetem

- 2. mintafeladat -

3. lépés: kódolt állapottábla elkészítése (kritikus versenyhelyzet elemzés: nem szükséges)

egy lehetséges kódkiosztás:

$$\begin{array}{ll} & Y \\ s_1: & 0 \\ s_2: & 1 \end{array} \Rightarrow$$

kódolt állapottábla

aktuális állapot, Y	következő állapot/kimenet X_1X_2			
	00	01	10	11
0	(0)/0	(0)/0	1/0	(0)/0
1	0/0	0/0	(1)/0	(1)/1

Széchenyi István Egyetem

- 2. mintafeladat -

4. lépés: a szekunder változó és a kimenet függvényeinek Karnaugh tábla segítségével történő megadása

Y

$X1$	0	0	1	1
$X2$	0	1	1	0
Y^v 0				1
1			1	1

$$Y = X_1\overline{X_2} + X_1Y^v$$

Z

$X1$	0	0	1	1
$X2$	0	1	1	0
Y^v 0				
1			1	

$$Z = X_1X_2Y^v$$

Széchenyi István Egyetem

- 2. mintafeladat -

5.lépés: a kezdeti állapot beállásának ellenőrzése

$$Y = X_1\overline{X_2} + X_1Y^v$$

$$Z = X_1X_2Y^v$$

kezdőállapotban:
X_1X_2=00

$\Rightarrow$

$$0(Y) = 0\cdot 1 + 0\cdot Y^v \quad \surd\text{☺}$$

$$0(Z) = 0\cdot 0\cdot Y^v \quad \surd\text{☺}$$

Konklúzió:

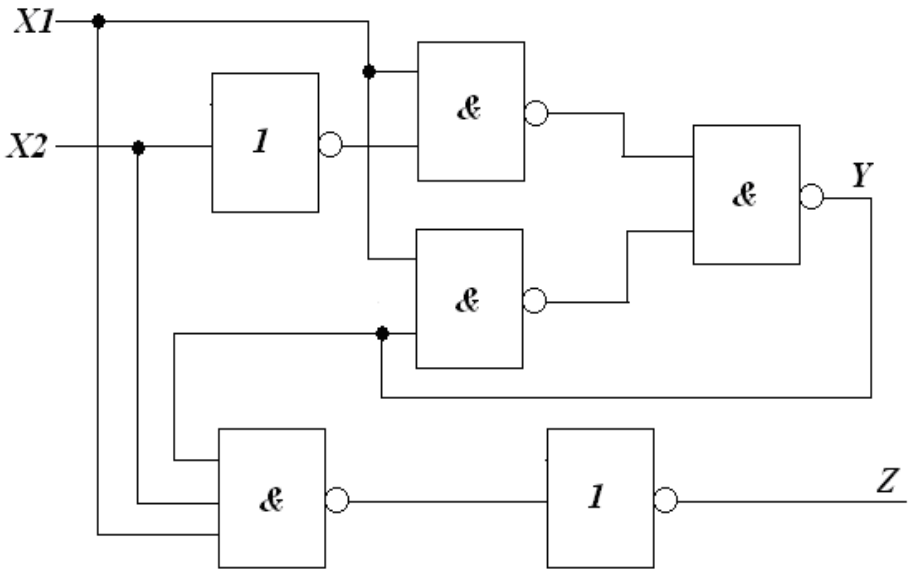
nem kell gondoskodni a kezdeti állapot ’RESET’ jellel történő beállításáról

Széchenyi István Egyetem

- 2. mintafeladat -

(6. lépés: lényeges hazárdok vizsgálata)
7. lépés: realizáció

NAND realizáció ('RESET ' logika nélkül):



$$Y = X_1 \overline{X_2} + X_1 Y^v$$

$$Z = X_1 X_2 Y^v$$

Széchenyi István Egyetem

- 2. mintafeladat -

Megoldás 2.: *SR tárolóval történő visszacsatolás alkalmazása (Mealy-modell szerint)*

1.-3. lépés: megegyezik az előző feladatmegoldás lépéseivel

4. lépés: az alkalmazott SR tároló vezérlésének felírása a kódolt állapottábla alapján

Q: az aktuális állapotot reprezentáló szekunder változó

aktuális állapot, Q	következő állapot/kimenet X_1X_2			
	00	01	10	11
0	0/0	0/0	1/0	0/0
1	0/0	0/0	1/0	1/1

Széchenyi István Egyetem

- 2. mintafeladat -

Az SR tároló saját vezérlési táblája

$Y^V \rightarrow Y$	S	R
0 0	0	–
0 1	1	0
1 0	0	1
1 1	–	0

kódolt állapottábla

aktuális állapot, Q	következő állapot/kimenet X_1X_2 = 00	01	10	11
0	(0)/0	(0)/0	1/0	(0)/0
1	0/0	0/0	(1)/0	(1)/1

↓

vezérlési tábla

kódolt aktuális állapot Q	kódolt következő állapot X_1X_2 = 00	01	10	11
	S R	S R	S R	S R
0	0 -	0 -	1 0	0 -
1	0 1	0 1	- 0	- 0

- 2. mintafeladat -

5. lépés: az SR tároló és a kimenet függvényeinek Karnaugh tábla segítségével történő megadása

S

$X1$	0	0	1	1
$X2$	0	1	1	0
Y^v 0				1
1			–	–

R

$X1$	0	0	1	1
$X2$	0	1	1	0
Y^v 0	–	–	–	
1	1	1		

$$S = X_1 \overline{X_2}$$

$$R = \overline{X_1}$$

$$Z = X_1 X_2 Y^v$$

Z

$X1$	0	0	1	1
$X2$	0	1	1	0
Y^v 0				
1			1	

Széchenyi István Egyetem

- 2. mintafeladat -

6.lépés: a kezdeti állapot beállásának ellenőrzése

kezdeti állapot: Q=0! → szükséges SR vezérlés: $\left\{ \begin{matrix} S=0 \\ R=1 \end{matrix} \right\}$

↓

$$S = X_1 \overline{X_2}$$
$$R = \overline{X_1}$$
$$Z = X_1 X_2 Y^v$$

kezdőállapotban: $X_1X_2=00$ ⇒

$$0(S) = 0 \cdot 1 \quad \surd ☺$$
$$1(R) = 1 \quad \surd ☺$$
$$0(Z) = 0 \cdot 0 \cdot Y^v \quad \surd ☺$$

Konklúzió:

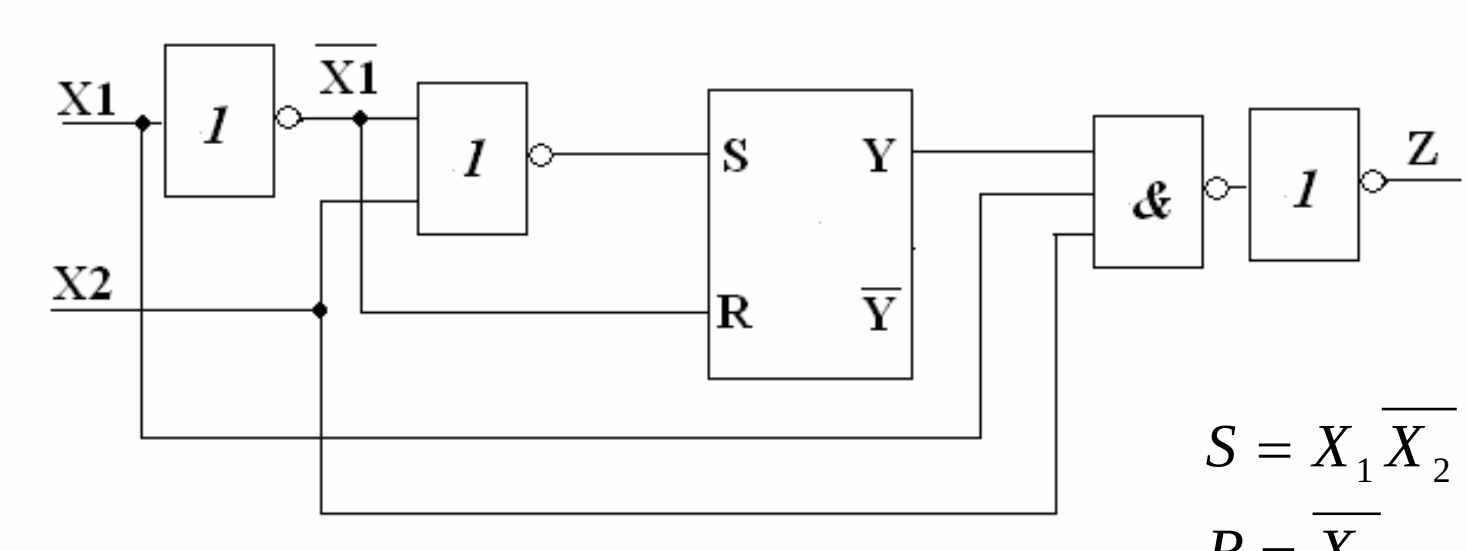
nem kell gondoskodni a kezdeti állapot ’RESET’ jellel történő beállításáról

Széchenyi István Egyetem

- 2. mintafeladat -

(7. lépés: lényeges hazárdok vizsgálata)
8. lépés: realizáció

NAND realizáció ('RESET ' logika nélkül):



$$S = X_1 \overline{X_2}$$

$$R = \overline{X_1}$$

$$Z = X_1 X_2 Y^{\nu}$$

Lényeges hazárdok aszinkron hálózatokban

Az eddigi aszinkron hálózat tervezési példáink megoldása során csak a szekunder változók versengése miatt kialakuló hibákkal, és azok kiküszöbölésével foglalkoztunk. Ez csak akkor tekinthető korrekt eljárásnak, ha garantálni tudjuk azt, hogy a bemeneti jelek változása okozta események a szekunder változók értékeinek megváltozása kezdete előtt már lezajlanak. Ez a feltételezésünk abban is megnyilvánul, hogy amikor az állapottáblán követjük az aszinkron hálózat működését, egyik oszlopról a másikra térünk át, és csak ezután vizsgáljuk a tranzienseket. A valóságban ez a feltételezés nem mindig jogos. *A szekunder változók és egyik bemeneti változó kritikus versenyhelyzete úgynevezett lényeges hazárd veszélyével jár. Ennek kiküszöbölése időkésleltetési manipulációkat igényel.*

Széchenyi István Egyetem

Lényeges hazárdok aszinkron hálózatokban

Példa:

aszinkron szekvenciális hálózat (1.mintapélda) működésének analizálása az állapottáblán:

kódolt aktuális állapot $Y_1^v Y_2^v$	kódolt következő állapot/kimenet	
	X=0 $Y_1^v Y_2^v$/Z	X=1 $Y_1^v Y_2^v$/Z
0 0	(0 0)/0	0 1/0
0 1	1 0/1	(0 1)/0
1 0	(1 0)/1	1 1/1
1 1	0 0/0	(1 1)/1

Sorrendi hálózatok kezdeti állapotának beállítási lehetőségei

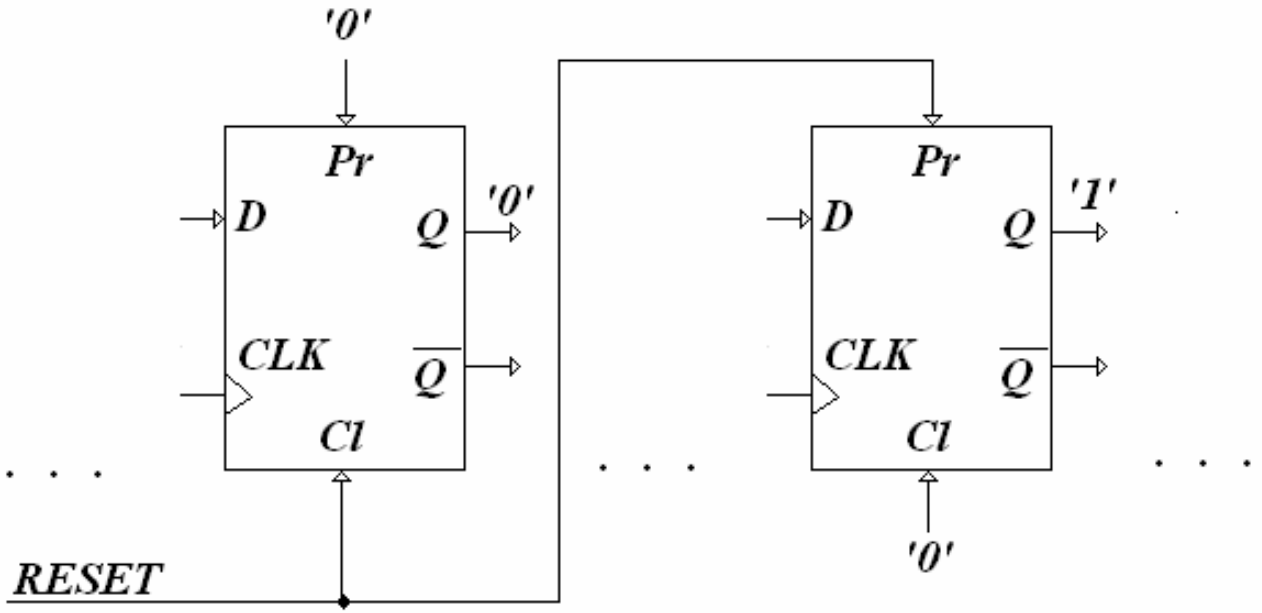
- *Szinkron sorrendi hálózatok kezdeti állapotának beállítása:*
  - *beállítás a ‚preset' (Pr) és a ‚clear' (Clr) segédbemenetek felhasználásával*
  - *beállítás az f_y hálózat kiegészítésével*
  - *(beállítás a szekunder változók aktuális állapotának módosításával)*

- *Aszinkron sorrendi hálózatok kezdeti állapotának beállítása:*
  - *közvetlenül visszacsatolt kombinációs hálózattal megvalósított aszinkron szekvenciális hálózat kezdeti állapotának beállítása*
  - *SR tárolókkal megvalósított aszinkron szekvenciális hálózat kezdeti állapotának beállítása*
  - *(beállítás a szekunder változók aktuális állapotának módosításával)*

Széchenyi István Egyetem

Szinkron sorrendi hálózatok kezdeti állapotának beállítása (I.)

- *beállítás a ‚preset' (Pr) és a ‚clear' (Clr) segédbemenetek felhasználásával*



Széchenyi István Egyetem

Szinkron sorrendi hálózatok kezdeti állapotának beállítása (II.)

- *beállítás az f_y hálózat kiegészítésével D-MS flip-flop esetében: kiegészítés RESET segédbemenettel*

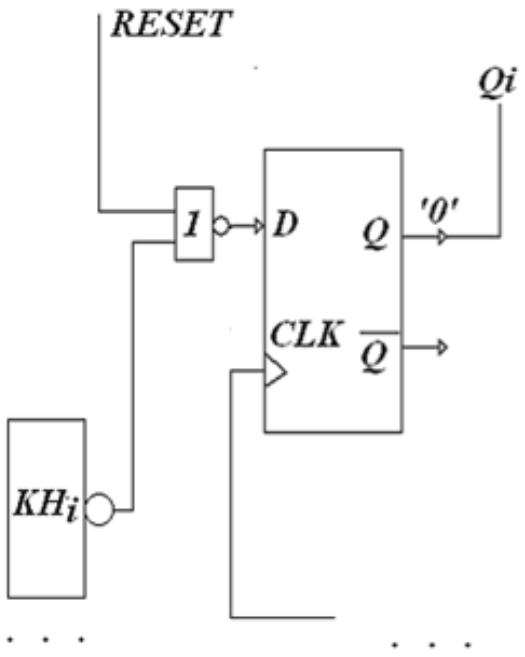
a) szükséges kezdeti állapot: ‚0' (Q=0)

$(D \rightarrow D_i)$

D_i	RESET	D_i'
0	0	0
0	1	0
1	0	1
1	1	0

$$D_i' = D_i \, \overline{RESET} = \overline{\overline{D_i} + RESET}$$

(Ez a módszer minden esetben biztosítja a kezdeti állapot beállását a szekunder változók és a bemenetek aktuális állapotától függetlenül!)



Széchenyi István Egyetem

Szinkron sorrendi hálózatok kezdeti állapotának beállítása (II.)

- *beállítás az f_y hálózat kiegészítésével D-MS flip-flop esetében: kiegészítés RESET segédbemenettel*

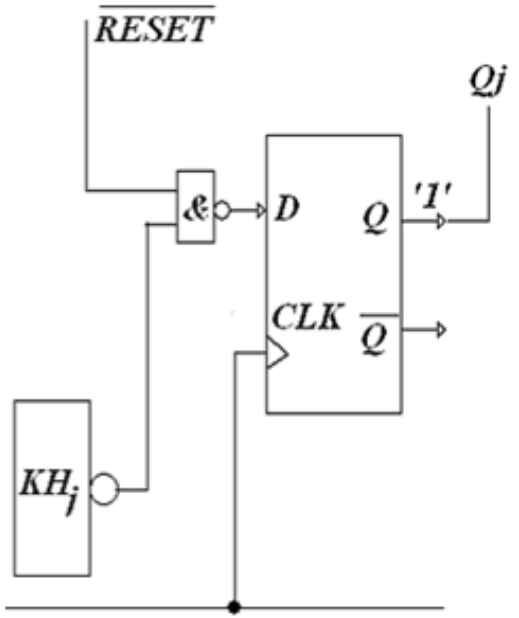
b) szükséges kezdeti állapot: ‚1' (Q=1)

$(D \rightarrow D_j)$

D_j	RESET	D_j'
0	0	0
0	1	1
1	0	1
1	1	1

$$D_j' = \overline{\overline{D_j}\ \overline{RESET}}$$

(Ez a módszer minden esetben biztosítja a kezdeti állapot beállását a szekunder változók és a bemenetek aktuális állapotától függetlenül!)



Széchenyi István Egyetem

Szinkron sorrendi hálózatok kezdeti állapotának beállítása (II.)

- *beállítás az f_y hálózat kiegészítésével JK-MS flip-flop esetében: kiegészítés RESET segédbemenettel*

a) szükséges kezdeti állapot: ‚0' (Q=0)

$(J \rightarrow J_i)$

J_i	RESET	J'_i
0	0	0
0	1	0
1	0	1
1	1	0

$(K \rightarrow K_i)$

K_i	RESET	K'_i
0	0	0
0	1	1
1	0	1
1	1	1

(Ez a módszer minden esetben biztosítja a kezdeti állapot beállását a szekunder változók és a bemenetek aktuális állapotától függetlenül!)

$$J'_i = \overline{\overline{J_i} + RESET}$$

$$K'_i = \overline{\overline{K_i}\ \overline{RESET}}$$

Széchenyi István Egyetem

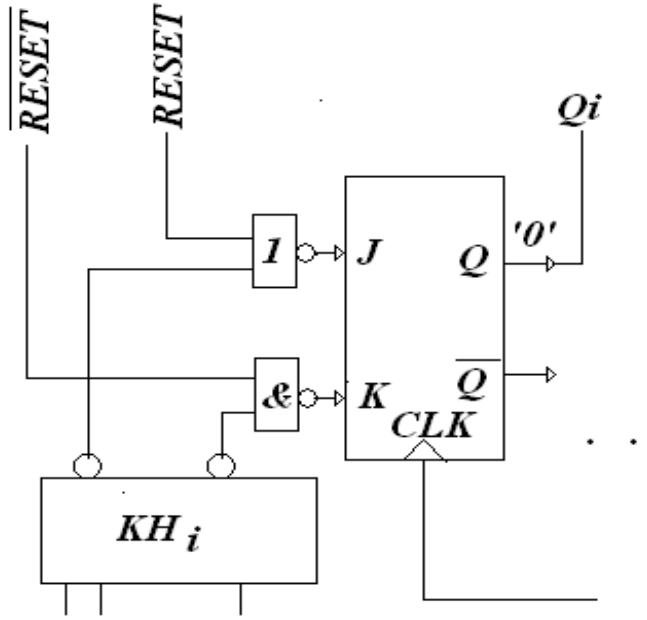
Szinkron sorrendi hálózatok kezdeti állapotának beállítása (II.)

- *beállítás az f_y hálózat kiegészítésével JK-MS flip-flop esetében: kiegészítés RESET segédbemenettel*

a) szükséges kezdeti állapot: ‚0' (Q=0)

$$J_i' = \overline{\overline{J_i} + RESET}$$

$$K_i' = \overline{\overline{K_i}\,\overline{RESET}}$$



Széchenyi István Egyetem

Szinkron sorrendi hálózatok kezdeti állapotának beállítása (II.)

- *beállítás az f_y hálózat kiegészítésével JK-MS flip-flop esetében: kiegészítés RESET segédbemenettel*

a) szükséges kezdeti állapot: ‚1' (Q=1)

$(J \rightarrow J_j)$

J_j	RESET	J'_j
0	0	0
0	1	1
1	0	1
1	1	1

$(K \rightarrow K_j)$

K_j	RESET	K'_j
0	0	0
0	1	0
1	0	1
1	1	0

$$J'_j = \overline{\overline{J_j}\ \overline{RESET}}$$

$$K'_j = \overline{\overline{K_j} + RESET}$$

(Ez a módszer minden esetben biztosítja a kezdeti állapot beállását a szekunder változók és a bemenetek aktuális állapotától függetlenül!)

Széchenyi István Egyetem

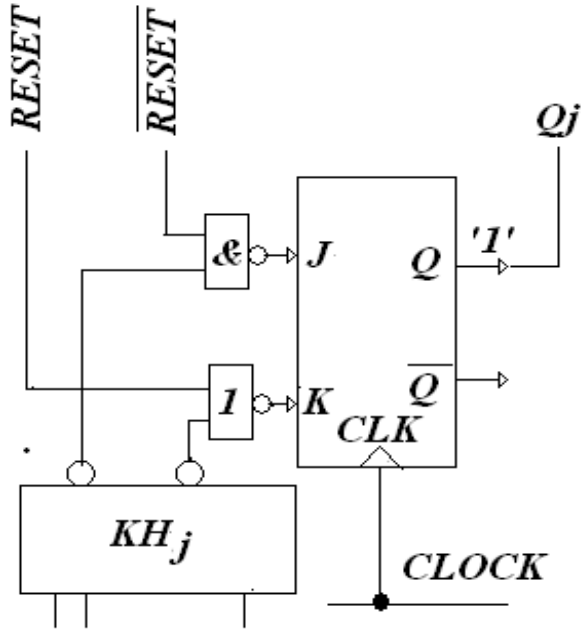
Szinkron sorrendi hálózatok kezdeti állapotának beállítása (II.)

- *beállítás az f_y hálózat kiegészítésével JK-MS flip-flop esetében: kiegészítés RESET segédbemenettel*

a) szükséges kezdeti állapot: ‚1’ (Q=1)

$$J_j' = \overline{\overline{J_j}\;\overline{RESET}}$$

$$K_j' = \overline{\overline{K_j} + RESET}$$



Széchenyi István Egyetem

Szinkron sorrendi hálózatok kezdeti állapotának beállítása (III.)

- *beállítás a szekunder változók aktuális állapotának módosításával (elv)*

a) legyen: Q→0 !

Q_i	RESET	Q_i'
0	0	0
0	1	0
1	0	1
1	1	0

b) legyen: Q→1 !

Q_j	RESET	Q_j'
0	0	0
0	1	1
1	0	1
1	1	1

(Ez a módszer egyszerűbb, de nem biztosítja minden esetben a bemenetektől függetlenül a kezdeti állapot beállítását.
pl.:

$J1 = Q_2^n \, X1 \, \overline{X2}$

$K1 = 1$

$Z = Q_1^n$

$J2 = \overline{X1} \; X2$

$K2 = \overline{Q_1^n \, Q_2^n \, X_1 \, X_2}$

Q_1^n és Q_2^n alacsony szintje nem garantálja mindkét J alacsony, és mindkét K magas szintjét!

Széchenyi István Egyetem

Aszinkron sorrendi hálózatok kezdeti állapotának beállítása (I.)

- *közvetlenül visszacsatolt kombinációs hálózattal megvalósított aszinkron szekvenciális hálózat kezdeti állapotának beállítása a szekunder változók módosításával*

a) szükséges kezdeti állapot: ‚0' (Y=0)

$(Y \rightarrow Y_i)$

Y_i	RESET	Y_i'
0	0	0
0	1	0
1	0	1
1	1	0

(Ez a módszer csak a bemenetekre megadott kezdeti bemeneti kombinációval együtt biztosítja a stabil kezdeti állapot beállítását!)

b) szükséges kezdeti állapot: ‚1' (Y=1)

$(Y \rightarrow Y_j)$

Y_j	RESET	Y_j'
0	0	0
0	1	1
1	0	1
1	1	1

Széchenyi István Egyetem

Aszinkron sorrendi hálózatok kezdeti állapotának beállítása (II.)

- *SR tárolókkal visszacsatolt aszinkron szekvenciális hálózatok kezdeti állapotának beállítása az fy hálózat S és R kimeneteinek kiegészítésével*

a) szükséges kezdeti állapot: ‚0' (Q=0)

$(S \rightarrow S_i)$

S_i	RESET	S_i'
0	0	0
0	1	0
1	0	1
1	1	0

$S_i' = \overline{\overline{S_i} + RESET}$

$(R \rightarrow R_i)$

R_i	RESET	R_i'
0	0	0
0	1	1
1	0	1
1	1	1

$R_i' = \overline{\overline{R_i}\ \overline{RESET}}$

(Ez nem biztosítja minden esetben a bemenetektől függetlenül a stabil kezdeti állapot beállítását!

pl.:

$S_1 = Y_2^{\vee}\overline{X}$ $\quad R_1 = \overline{Y_2^{\vee}}\ \overline{X}$

$S_2 = \overline{Y_1^{\vee}}\ X$ $\quad R_2 = Y_1^{\vee}\ X$

$Z = R_2$

Széchenyi István Egyetem

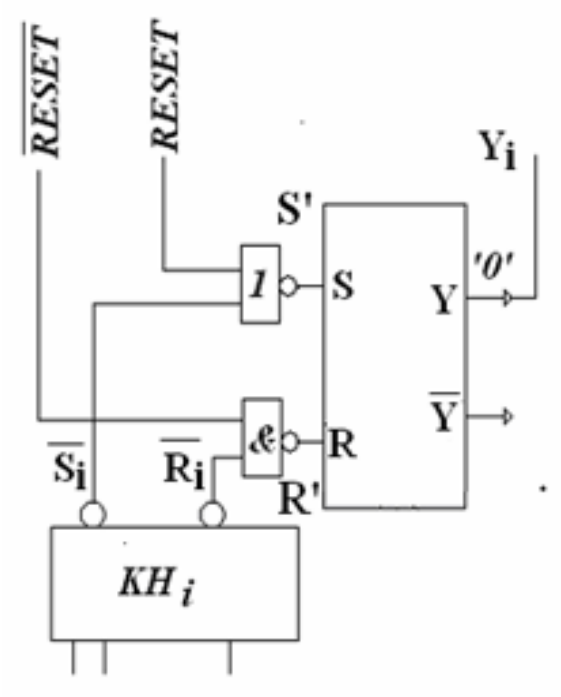
Aszinkron sorrendi hálózatok kezdeti állapotának beállítása (II.)

- *SR tárolókkal visszacsatolt aszinkron szekvenciális hálózatok kezdeti állapotának beállítása az f_Y hálózat S és R kimeneteinek kiegészítésével*

a) szükséges kezdeti állapot: ‚0' (Q=0)

$$S_i' = \overline{\overline{S_i} + RESET}$$

$$R_i' = \overline{\overline{R_i}\,\overline{RESET}}$$



Széchenyi István Egyetem

Aszinkron sorrendi hálózatok kezdeti állapotának beállítása (II.)

- *SR tárolókkal visszacsatolt aszinkron szekvenciális hálózatok kezdeti állapotának beállítása az f_Y hálózat S és R kimeneteinek kiegészítésével*

a) szükséges kezdeti állapot: ‚1' (Q=1)

$(S \rightarrow S_j)$

S_j	RESET	S'_j
0	0	0
0	1	1
1	0	1
1	1	1

$S'_j = \overline{\overline{S_j}\ \overline{RESET}}$

$(R \rightarrow R_j)$

R_j	RESET	R'_j
0	0	0
0	1	0
1	0	1
1	1	0

$R'_j = \overline{\overline{R_j} + RESET}$

(Ez nem biztosítja minden esetben a bemenetektől függetlenül a stabil kezdeti állapot beállítását!

pl.:

$S_1 = Y_2^v \overline{X}$ $\quad R_1 = \overline{Y_2^v}\ \overline{X}$

$S_2 = \overline{Y_1^v}\ X$ $\quad R_2 = Y_1^v\ X$

$Z = R_2$

Széchenyi István Egyetem

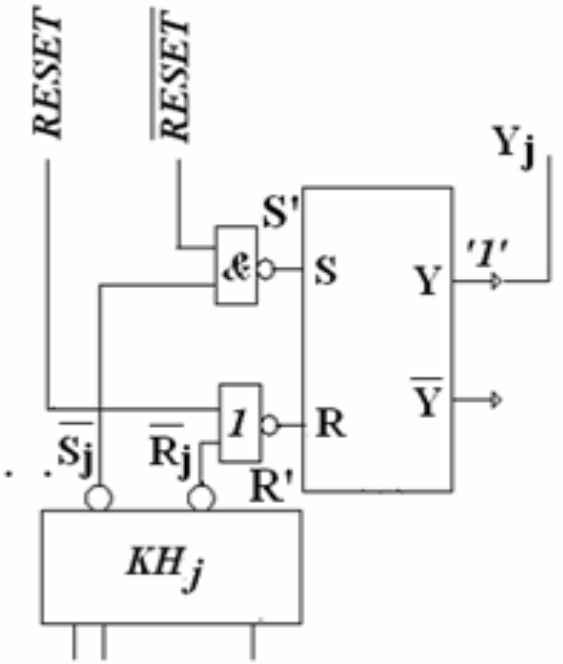
Aszinkron sorrendi hálózatok kezdeti állapotának beállítása (II.)

- *SR tárolókkal visszacsatolt aszinkron szekvenciális hálózatok kezdeti állapotának beállítása az f_Y hálózat S és R kimeneteinek kiegészítésével*

a) szükséges kezdeti állapot: ‚1' (Q=1)

$$S'_j = \overline{\overline{S_j}\;\overline{RESET}}$$

$$R'_j = \overline{\overline{R_j} + RESET}$$



Széchenyi István Egyetem

Aszinkron sorrendi hálózatok kezdeti állapotának beállítása (III.)

- *SR tárolókkal visszacsatolt aszinkron szekvenciális hálózat kezdeti állapotának beállítása a szekunder változók módosításával (elv)*

a) szükséges kezdeti állapot: ‚0’ (Y=0)

$(Y \rightarrow Y_i)$

Y_i	RESET	Y_i'
0	0	0
0	1	0
1	0	1
1	1	0

(Ez a módszer a bemenetekre megadott kezdeti bemeneti kombinációval együtt sem biztosítja mindig a kezdeti állapot beállítását! /lásd előző példa/.)

b) szükséges kezdeti állapot: ‚1’ (Y=1)

$(Y \rightarrow Y_j)$

Y_j	RESET	Y_j'
0	0	0
0	1	1
1	0	1
1	1	1

Állapot összevonási módszerek

- *Állapot összevonási módszerek teljesen specifikált szekvenciális hálózatok esetében*
  - *állapot összevonás az ekvivalencia-típusú reláció alapján*
  - *az összevonás lépései, a lépcsős tábla alkalmazása az ekvivalencia-típusú reláció alapján*

- *Állapot összevonási módszerek nem teljesen specifikált szekvenciális hálózatok esetében*
  - *állapot összevonás a kompatibilitás-típusú reláció alapján*
  - *az összevonás lépései, a lépcsős tábla alkalmazása a kompatibilitás-típusú reláció alapján*

Állapot összevonás teljesen specifikált előzetes szimbolikus állapottáblán

Emlékeztető

(…korábban…)

Az állapot összevonás feltételei:

az előzetes szimbolikus állapottábla két állapotát nem kell megkülönböztetni, ezért azok összevonhatók,

- *ha bemeneti kombinációnként egyeznek a hozzájuk rendelt kimeneti kombinációk,*
- *és bemenő kombinációnként ugyanarra a következő állapotra vezetnek.*

Állapot összevonás teljesen specifikált előzetes szimbolikus állapottáblán

Az összevonhatóság (szükséges és elégséges) feltételei:

Két állapot a teljesen specifikált hálózat állapottábláján nem megkülönböztethető (NMK), ha a két állapotból elindulva bármely bemeneti sorozatra ugyanazt a kimeneti sorozatot látjuk.

Ebből a magától értetődő definícióból kiindulva bizonyítható, hogy két állapot összevonható, ha a két állapotból bármely bemeneti kombinációra adott kimeneti kombinációk megegyeznek, és nem megkülönböztethető állapotokra vezetnek.

A „nem megkülönböztethetőség”(NMK), mint reláció

Tulajdonságok:

1. *reflexív*
2. *szimmetrikus*
3. *tranzitív*

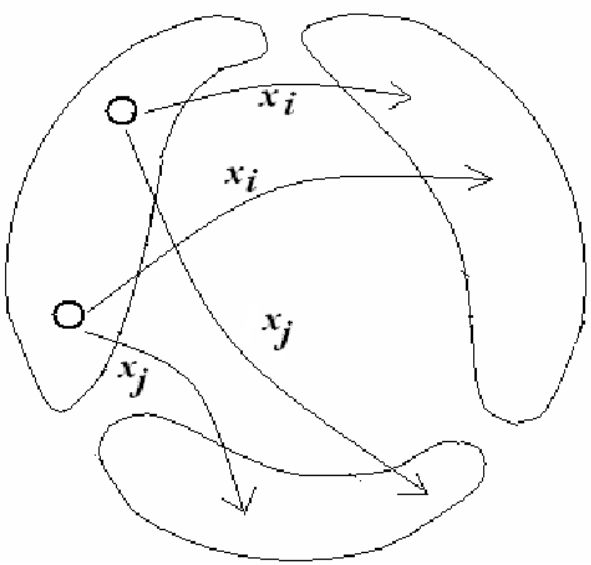
1. a *reflexivitás* az a trivialitás, hogy egy szimbolikus állapot saját magától nem különböztethető meg, azaz ***a*** ≡ ***a***.

2. *szimmetrikusak* azok a bináris relációk, amelyekre igaz, hogy amennyiben ***a*** ≡ ***b***, akkor bizonyosan fennáll a ***b*** ≡ ***a*** reláció is.

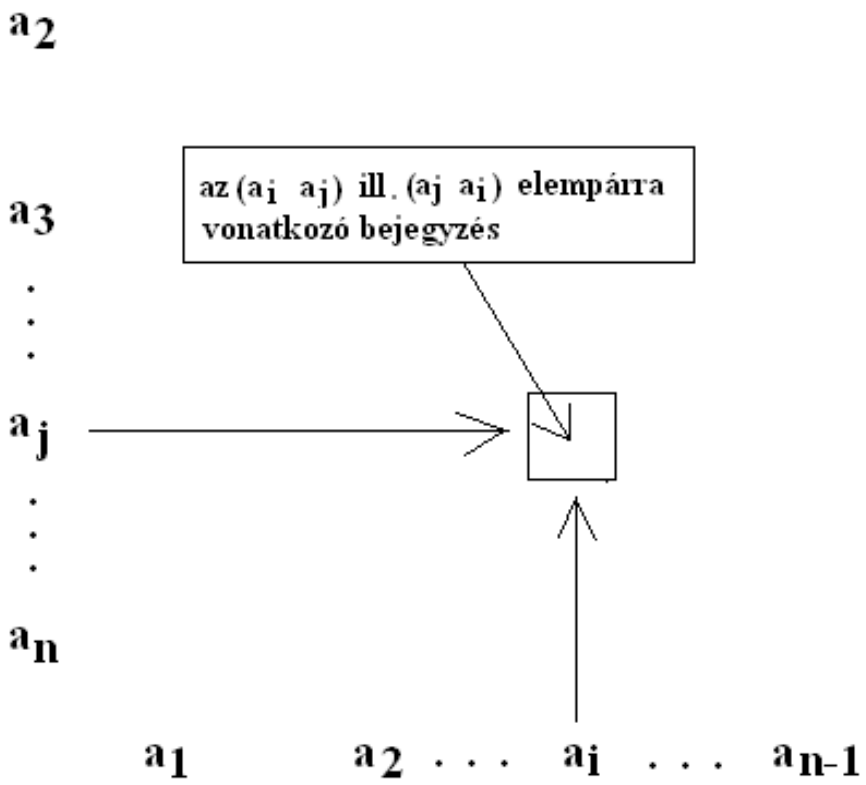
3. *tranzitív* relációk esetén igaz, hogy amennyiben ***a*** ≡ ***b*** és ***b*** ≡ ***c***, akkor teljesül az ***a*** ≡ ***c*** is.

Az ilyen relációkat ***ekvivalencia-típusú reláció***knak nevezzük.

Összevonható állapotok szemléltetése és a lépcsős tábla



Diszjunkt részhalmazokra bontás



Jelölések a lépcsős táblán

- ***a*** ≡ ***b*** : az ‚***a***' és a ‚***b***' állapotok ekvivalensek
- ***a*** <> ***b*** *:* az ‚***a***' és a ‚***b***' állapotok antivalensek
- *feltételes ekvivalencia:* a feltételes ekvivalenciát magával a feltétellel jelöljük. Például, ha a jelölés a következő felsorolás: *(**ab**, **cd** . . .) akkor az a két állapot, amelyekre ez vonatkozik, feltételesen ekvivalensek, azaz csak akkor ekvivalensek, ha **a** ≡ **b** és **c** ≡ **d**.*

Széchenyi István Egyetem

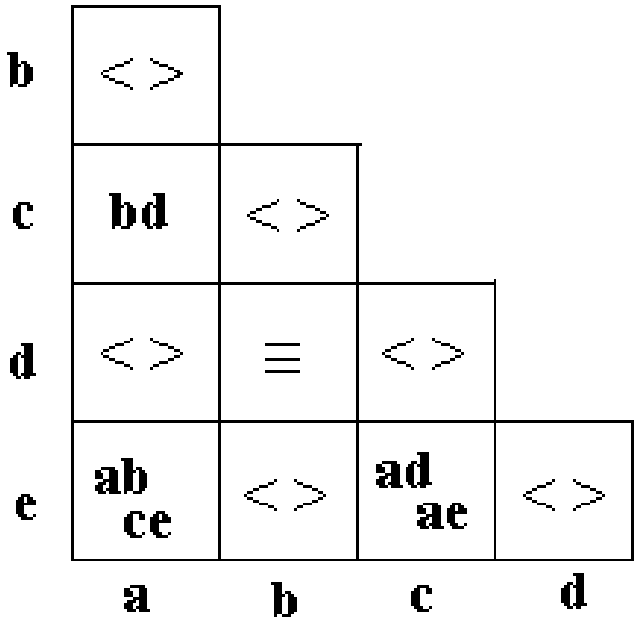
Mintapélda megoldása lépcsős táblán (1)

Példa:

előzetes szimbolikus állapottábla

aktuális állapot	következő állapot/kimenet	
	X=0	*X=1*
a	**c/0**	**b/1**
b	**d/1**	**e/0**
c	**a/0**	**d/1**
d	**b/1**	**e/0**
e	**e/0**	**a/1**

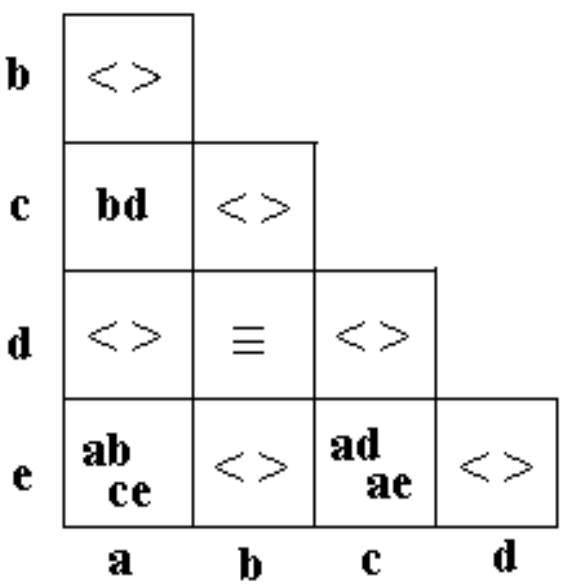
⇒



1.generációs lépcsős tábla

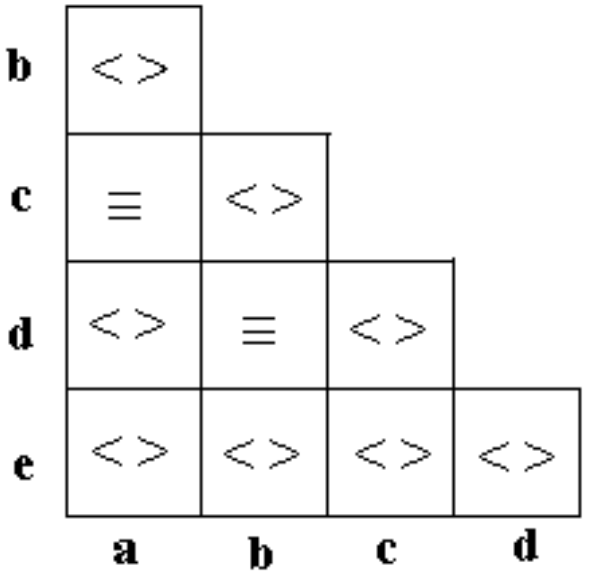
Széchenyi István Egyetem

Mintapélda megoldása lépcsős táblán (2)



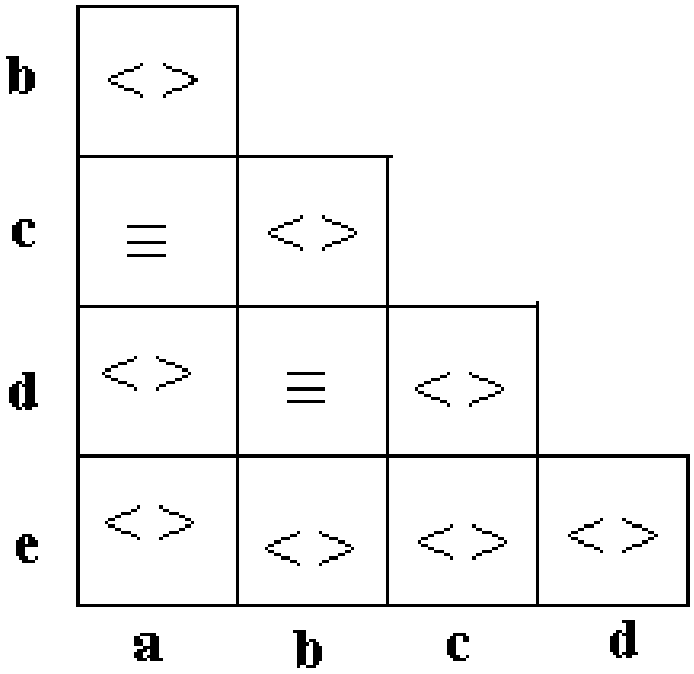
1.generációs lépcsős tábla

⇒



2.generációs lépcsős tábla

Mintapélda megoldása lépcsős táblán (3)



a ≡ c

b ≡ d

e

(a c) (b d) (e)

Mintapélda megoldása lépcsős táblán (4)

előzetes szimbolikus állapottábla

aktuális állapot	következő állapot/kimenet	
	X=0	*X=1*
a	**c/0**	**b/1**
b	**d/1**	**e/0**
c	**a/0**	**d/1**
d	**b/1**	**e/0**
e	**e/0**	**a/1**

$\mathbf{(ac) \rightarrow s_1}$

$\mathbf{(bd) \rightarrow s_2}$

$\mathbf{e \rightarrow s_3}$

$\Rightarrow$

összevont szimbolikus állapottábla

aktuális állapot	következő állapot/kimenet	
	X=0	X=1
s_1	$s_1/0$	$s_2/1$
s_2	$s_2/1$	$s_3/0$
s_3	$s_3/0$	$s_1/1$

Állapot összevonás ekvivalencia-típusú reláció alkalmazásával: összefoglaló

Lépések:

1. *az ekvivalens és antivalens állapotpárok megkeresése lépcsős tábla segítségével,*
2. *a maximális ekvivalencia osztályok meghatározása,*
3. *a maximális ekvivalencia osztályoknak megfelelő állapotokkal az összevont állapottábla elkészítése.*

Állapot összevonás nem teljesen specifikált előzetes szimbolikus állapottáblán

A nem megkülönböztethetőség, mint feltétel értelmezése a nem teljesen specifikált, előzetes szimbolikus állapottáblán:

a nem teljesen specifikált előzetes, szimbolikus állapottáblán két állapot nem megkülönböztethető, ha bemeneti kombinációnként megegyeznek a kimeneti kombinációk (amennyiben mindkettőre specifikálva vannak), és a következő állapotok is nem megkülönböztethetők (ha mindkettőre specifikálva vannak).

Széchenyi István Egyetem

Állapot összevonás nem teljesen specifikált előzetes szimbolikus állapottáblán: példa1.

Példa1.: állapot összevonás egy korábbi, nem teljesen specifikált előzetes szimbolikus állapottáblán

Egy lehetséges megoldás:
tegyük teljesen specifikálttá, és csináljunk összevonást ekvivalencia relációkkal!

X1 X2 bem. / akt. áll	szimb.köv.áll. / kim.			
	0 0	0 1	1 0	1 1
a	(a)/0	b/0	c/0	-/-
b	a/0	(b)/0	-/-	d/0
c	a/0	-/-	(c)/0	e/1
d	-/-	b/0	c/0	(d)/0
e	-/-	b/0	c/0	(e)/1

Sorrendi hálózatok

X1 X2 bem. / akt. áll	szimb.köv.áll. / kim.			
	0 0	0 1	1 0	1 1
a	(a)/0	b/0	c/0	-/-
b	a/0	(b)/0	-/-	d/0
c	a/0	-/-	(c)/0	e/1
d	-/-	b/0	c/0	(d)/0
e	-/-	b/0	c/0	(e)/1

$\longrightarrow$ *(abd), (ce)*

$\downarrow$

X1 X2 bem. / akt. áll	szimb.köv.áll. / kim.			
	0 0	0 1	1 0	1 1
a	(a)/0	b/0	c/0	d/0
b	a/0	(b)/0	c/0	d/0
c	a/0	b/0	(c)/0	e/1
d	a/0	b/0	c/0	(d)/0
e	a/0	b/0	c/0	(e)/1

Az állapot-kompatibilitás

Egy nem teljesen specifikált szimbolikus előzetes állapottáblával megadott hálózat adott állapotához tartozó specifikációs bemeneti sorozat az, amelyre a hálózat minden állapotátmenete és kimenete specifikálva van.

Két szimbolikus állapot a nem teljesen specifikált hálózat állapottábláján csak akkor megkülönböztethető, ha létezik *legalább egy* olyan specifikált bemeneti sorozat, amely mindkét állapotra *érvényes*, és amelynek *legalább egy elemére más kimeneti kombináció adódik*. Ha ilyen specifikációs bemeneti sorozat nem létezik, a két állapot nem megkülönböztethető.

Továbbá ha a kiválasztott két állapotra létezik olyan bemeneti kombináció, amelyre *vagy a kimenetek*, *vagy a következő állapotok*, *vagy mindkettő* specifikálva vannak, a két állapot akkor nem megkülönböztethető, ha a specifikált kimeneti kombinációk bemeneti kombinációnként megegyeznek, a specifikált következő állapotok pedig nem megkülönböztethetők.

A „nem megkülönböztethetőség”(NMK), mint reláció

Tulajdonságok:

1. *reflexív*
2. *szimmetrikus*
3. *nem tranzitív*

Az ilyen relációkat ***kompatibilitás-típusú relációknak*** nevezzük A nem teljesen specifikált hálózat állapotpárjaira fennálló ’nem megkülönböztethetőség’ tulajdonságot röviden *kompatibilitásnak*, illetve a pár tagjait *kompatibilis állapotoknak* fogjuk nevezni.

Jelölések a lépcsős táblán

- ***a ~ b*** : ‚***a***' és ‚***b***' állapotok kompatibilisek
- ***a !~ b*** : ‚***a'*** és ‚***b'*** állapotok nem kompatibilisek
- *feltételes kompatibilitás:* ***ab***, ***cd*** … az a két állapot, melyekre ez a bejegyzés vonatkozik, feltételesen kompatibilis, azaz csak akkor kompatibilis, ha ***a~b*** és ***c~d*** …

A kompatibilitás elégséges feltételei

A kompatibilitás elégséges feltételei a következők:

- *ha nincs olyan bemeneti kombináció, amelyre mindkét állapotból specifikált következő állapot és specifikált kimenet lenne az állapottáblán, akkor a két állapot kompatibilis,*

- *ha pedig létezik mindkét állapotra specifikált kimeneti kombinációt és következő állapotot definiáló bemeneti kombináció, és erre a két állapothoz tartozó kimeneti kombinációk megegyeznek, valamint a két állapothoz tartozó következő állapotok kompatibilisek, akkor a két állapot kompatibilis.*

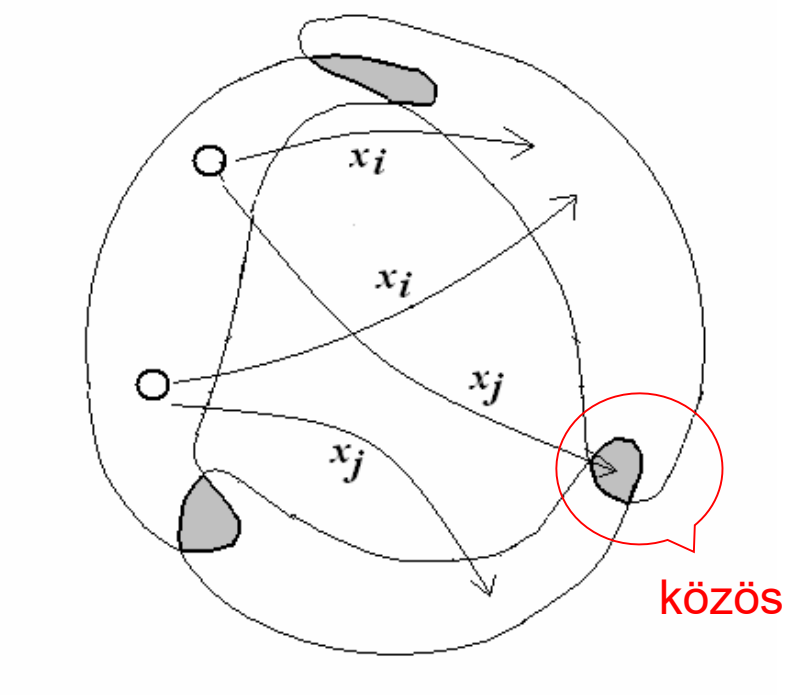
A kompatibilitási osztályok zárt halmaza

A kompatibilitási reláció az állapothalmazt nem diszjunkt osztályokra bontja, azaz lehetnek az osztályoknak közös elemeik is. Egy ilyen osztály valamennyi lehetséges állapotpárjára fennáll a kompatibilitás. Ezek az osztályok akkor maximálisak, ha további elemek egyetlen osztályba sem vonhatók be.

A kompatibilitási osztályok egy adott halmaza zárt, ha a halmazban szereplő bármelyik osztály tetszőleges két állapotából kiindulva minden olyan bemeneti kombinációra, amely mindkét állapotból specifikált következő állapotot ír elő, a következő állapotok is együtt szerepelnek a halmaznak *legalább egy osztályában*.

A kompatibilitási reláció jellegzetessége:
a nem diszjunkt részhalmazok zártsága

Példa:



közös osztályrész!

Kevesebb, vagy kisebb állapot-számú osztályból álló zárt kompatibilitási osztályhalmaz keresése

A maximális kompatibilitási osztályok megtalálása után a nem teljes specifikáció lehetőséget teremt arra, hogy az állapotok teljes lefedése és a zártság megőrzése mellett egyszerűbb kompatibilitási halmazrendszert válasszunk.

Ez azt jelenti, hogy úgy döntünk a közömbös bejegyzésekről, hogy döntésünk

- vagy kevesebb kompatibilitási osztályból álló,
- vagy az egyes osztályokban kevesebb állapotból álló osztályhalmazt eredményezzen.

Kevesebb, vagy kisebb állapot-számú osztályból álló zárt kompatibilitási osztályhalmaz keresése

Követelmények:

- a maximális kompatibilitási osztályoknak továbbra is zárt halmazrendszert kell alkotni
- minden állapotnak szerepelnie kell legalább egy osztályban

Kevesebb, vagy kisebb állapot-számú osztályból álló zárt kompatibilitási osztályhalmaz keresése

Vizsgálat: van-e olyan kompatibilitási osztály, amelynek valamennyi állapota szerepel valamely más osztályban is?

- Ha így van, megkísérelhetjük elhagyni ezt az osztályt. Ez akkor lehetséges, ha az osztály elhagyása után is zárt marad a kompatibilitási osztályok halmaza.

- Ha a zártság nem tartható fenn, akkor visszatesszük az elhagyni kívánt osztályt, és a többszörösen szereplő állapotok egyes osztályokból való elhagyásával próbálkozunk.

Ha találunk a teljes lefedettség és a zártság fenntartásával elhagyható állapotokat, akkor egyszerűbb összevont állapottáblát kapunk.

A fentiekből következik, hogy több megoldás is kínálkozhat, ezek közül kell választanunk a megvalósítandó összevont állapottáblát.

Állapot összevonás nem teljesen specifikált előzetes szimbolikus állapottáblán: példa2.

Példa2.: alkalmazzuk a kompatibilitás-típusú relációt az előző példában (példa1) bemutatott állapottáblán:

aktuális állapot	következő állapot/kimenet			
	X_1X_2			
	00	*01*	*10*	*11*
a	*a/0*	*b/0*	*c/0*	*-/-*
b	*a/0*	*b/0*	*-/-*	*d/0*
c	*a/0*	*-/-*	*c/0*	*e/1*
d	*-/-*	*b/0*	*c/0*	*d/0*
e	*-/-*	*b/0*	*c/0*	*e/1*

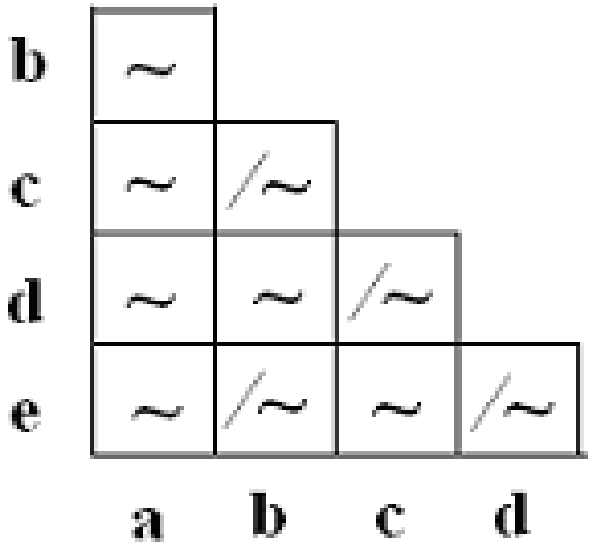
Széchenyi István Egyetem

Állapot összevonás nem teljesen specifikált előzetes szimbolikus állapottáblán: példa2.

előzetes szimbolikus állapottábla

aktuális állapot	következő állapot/kimenet X_1X_2			
	00	*01*	*10*	*11*
a	*a/0*	*b/0*	*c/0*	*-/-*
b	*a/0*	*b/0*	*-/-*	*d/0*
c	*a/0*	*-/-*	*c/0*	*e/1*
d	*-/-*	*b/0*	*c/0*	*d/0*
e	*-/-*	*b/0*	*c/0*	*e/1*

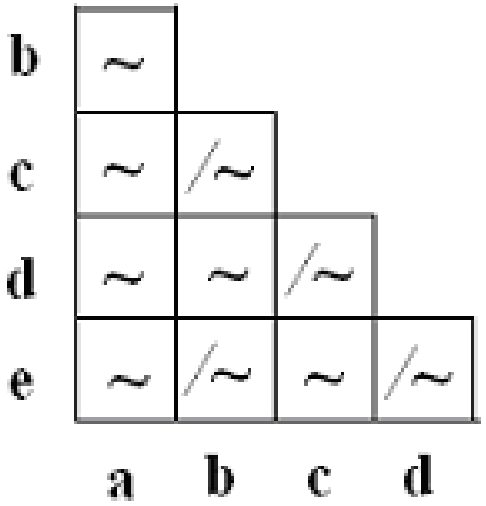
⇒



lépcsős tábla

Széchenyi István Egyetem

Állapot összevonás nem teljesen specifikált előzetes szimbolikus állapottáblán: példa2.



kompatibilis párok:

(a b), (a c), (a d), (a e), (b d), (c,e)

maximális kompatibilitási osztályok:

(a b d), (a c e)

Állapot összevonás nem teljesen specifikált előzetes szimbolikus állapottáblán: példa2.

Konklúzió

két redukált, zárt osztályhalmaz:

1) (**a b d**) (**c e**),

2) (**a c e**) (**b d**)

Az első osztályhalmaz zártságáról az állapottábla alapján meggyőződhetünk, és beláthatjuk, hogy az (**a b d**) minden eleme bemeneti kombinációnként ugyanabba az osztályba képződik le, illetve ez a (**c e**) osztály elemeire is igaz. Hasonlóan bizonyítható a második osztályhalmaz zártsága is. Ebből az következik, hogy a példának kétféle állapot összevonása is jó megoldáshoz vezet.

Széchenyi István Egyetem

Állapot összevonás nem teljesen specifikált előzetes szimbolikus állapottáblán: példa2.

Megoldás(1):

- **(a b d) (c e)**,

$\Downarrow$

(a b d) $\rightarrow S_1$
(c e) $\rightarrow S_2$

aktuális állapot	következő állapot/kimenet			
	X_1X_2			
	00	*01*	*10*	*11*
S_1	$S_1/0$	$S_1/0$	$S_2/0$	$S_1/0$
S_2	$S_1/0$	$S_1/0$	$S_2/0$	$S_2/1$

Megoldás(2):

- **(a c e) (b d)**

$\Downarrow$

(a c e) $\rightarrow S_1$
(b d) $\rightarrow S_2$

aktuális állapot	következő állapot/kimenet			
	X_1X_2			
	00	*01*	*10*	*11*
$S_1/0$	$S_1/0$	$S_2/0$	$S_1/0$	$S_1/1$
$S_2/0$	$S_1/0$	$S_2/0$	$S_1/0$	$S_2/0$

Állapotkódolási módszerek

- *Szinkron sorrendi hálózatok:*
  - nincs versenyhelyzet veszély, így az állapotkódolás arra irányul, hogy a legegyszerűbb struktúrát alakítsuk ki.

- *Aszinkron sorrendi hálózatok:*
  - *legfontosabb cél a kritikus versenyhelyzetek elkerülése*

Állapotkódolási módszerek

- *Szinkron sorrendi hálózatok állapotkódolási módszerei*
  - *az ‚1'-es súlyú állapotkódolás elve (ism.)*
  - *a szomszédos kódolás elve*
  - *a helyettesítési tulajdonságú partíció elve alapján végzett kódolás*

- *Aszinkron sorrendi hálózatok állapotkódolási módszerei*
  - *instabil állapotok beillesztésének elve*
  - *a Tracey-Unger módszer elve*

Szinkron sorrendi hálózatok állapotkódolási módszerei *(1)*

- *az ‚1'-es súlyú állapotkódolás („bit per state") elve (ism.)*

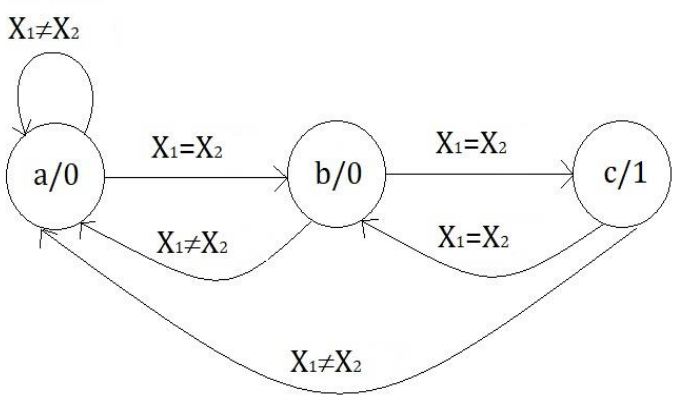
(…korábban…)

Az olyan kódolást, melyben a kódszóban csak egyetlen bit helyén szerepel '1'-es érték (a többi '0'), ‚1'-es súlyú kódolásnak nevezzük

Széchenyi István Egyetem

Szinkron sorrendi hálózatok állapotkódolási módszerei *(1)*

Egy korábbi példa ‚1'-es súlyozású kódolás alkalmazására:



	Q_a	Q_b	Q_c
a :	1	0	0
b :	0	1	0
c :	0	0	1

$$\Downarrow$$

$$D_a = R + \overline{R}\,(Q_a\,\overline{E} + Q_b\,\overline{E} + Q_c\,\overline{E}) = R + \overline{E}(Q_a + Q_b + Q_c)$$
$$= R + \overline{E}$$

$$D_b = \overline{R}\,E\,(Q_a + Q_c)$$

$$D_c = \overline{R}\,Q_b\,E$$

Szinkron sorrendi hálózatok állapotkódolási módszerei *(2)*

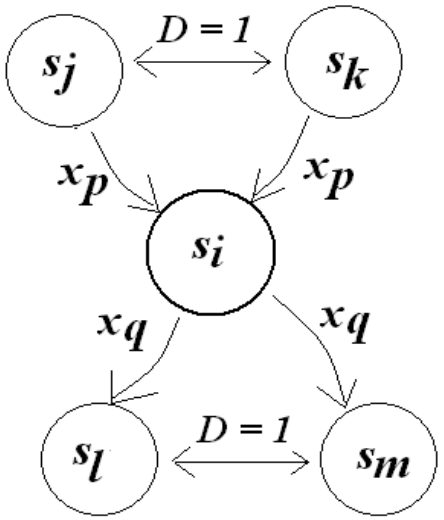
- *a szomszédos kódolás elve*

Egyszerűbb lesz a hálózat, ha egy adott állapot következő állapotainak kódjai bemenő kombinációnként szomszédosak, illetve csak egy szekunder változóban különböznek egymástól, azaz a kódok közötti távolság egységnyi (D=1). Ugyancsak előnyös, ha azok az állapotok, amelyek valamely adott állapotnak az elődjei, bemenő kombinációnként szomszédos kódúak.

Széchenyi István Egyetem

Szinkron sorrendi hálózatok állapotkódolási módszerei *(2)*

Példa:



- szomszédos kódolást kifejező állapotgráf a leírt előnyös kódolási helyzetekkel

aktuális állapot	következő állapot	
	X=0	*X=1*
a	*c*	*d*
b	*a*	*e*
c	*a*	*b*
d	*c*	*e*
e	*a*	*b*

- előzetes szimbolikus állapottábla

Széchenyi István Egyetem

Szinkron sorrendi hálózatok állapotkódolási módszerei *(2)*

állapotok	állapotok: „utódok”	
	X=0	*X=1*
a	*c*	*d*
b	*a*	*e*
c	*a*	*b*
d	*c*	*e*
e	*a*	*b*

állapotok	állapotok: „elődök”	
	X=0	*X=1*
a	*b,c,e*	
b		*c,e*
c	*a,d*	
d	*a*	
e		*b,d*

- szomszédos kódolás elve alapján átdolgozott „utódok” és „elődök” táblázatok

Széchenyi István Egyetem

Szinkron sorrendi hálózatok állapotkódolási módszerei *(2)*

- *az állapotok elhelyezése Karnaugh táblán, az előnyös szomszédosságok legnagyobb arányú biztosítására*

Közös utódja van X = 0ra:
b,c
b,e
c,e
a,d

Közös utódja van X = 1-re:
b,d
c,e

Közös előd van X = 0-ra:
c,d

Közös előd van X = 1-re
-

Szomszédos kódok:

mindkettő	*közös utód*	*közös előd*
—	a,d	c,d
	b,c	
	b,d	
	b,e	
	c,e	

Q1 Q2 / Q3	0 0	0 1	1 1	1 0
0	e	b		
1	c	d	a	

Széchenyi István Egyetem

Szinkron sorrendi hálózatok állapotkódolási módszerei (2)

- egy lehetséges kódkiosztás

	Q_1	Q_2	Q_3
a:	1	1	1
b:	0	1	0
c:	0	0	1
d:	0	1	1
e:	0	0	0
-	1	0	0
-	1	1	0
-	1	0	1

(…)

⇒

kódolt aktuális állapot,			kódolt következő állapot					
			X = 0			X = 1		
Q_1^n	Q_2^n	Q_3^n	D_1	D_2	D_3	D_1	D_2	D_3
1	1	1	0	0	1	0	1	1
0	1	0	1	1	1	0	0	0
0	0	1	1	1	1	0	1	0
0	1	1	0	0	1	0	0	0
0	0	0	0	1	0	1	1	1
1	0	0	-	-	-	-	-	-
1	1	0	-	-	-	-	-	-
1	0	1	-	-	-	-	-	-

- kódolt állapottábla

Szinkron sorrendi hálózatok állapotkódolási módszerei *(3)*

- *A helyettesítési tulajdonságú partíció elve alapján végzett kódolás*

Alap: az állapothalmazon értelmezett partíciópárok elmélete

Ha egy szinkron sorrendi hálózat állapotváltozóit *csoportokra* bontjuk, akkor a csoportokhoz *komponenseket* rendelhetünk. Minden egyes komponenshez *két partíciót* tartozik:

- az egyik partíció azon állapotokat sorolja egy osztályba, amelyeket a komponens egyformán kódol,
- a másik partíció azokat az állapotokat sorolja egy osztályba, amelyeket a komponensre kapcsolódó környezet egyformán kódol.

Szinkron sorrendi hálózatok állapotkódolási módszerei *(3)*

A környezeti partíció és a komponens partíció úgynevezett *partíciópárt* alkot.

Mivel a komponens a következő állapotának kialakításakor egy adott bemeneti kombináció esetén nem tesz különbséget két, a környezete által azonosan kódolt állapot között, a környezeti partíció ugyanazon osztályába tartozó állapotok következő állapotai bemeneti kombinációnként a komponens partíciónak is ugyanazon blokkjában vannak.

Ezért nevezzük ezt a partíció kettőst *partíciópár*nak. A komponens partíciók metszete a legfinomabb partíciót eredményezi.

Ha egy komponens bemeneteire nem kapcsolódnak más komponensek állapotváltozói, akkor a komponens partíció önmagával alkot partíciópárt.

Az ilyen partíciót helyettesítési tulajdonságú (HT) partíciónak nevezik.

(…)

Aszinkron sorrendi hálózatok állapotkódolási módszerei *(1)*

- *instabil állapotok beillesztésének elve*

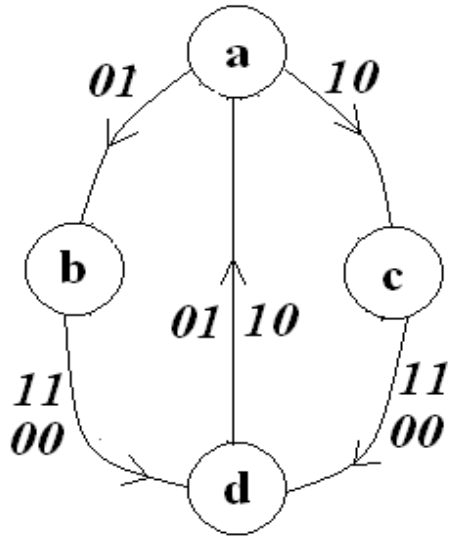
A módszer lényege: instabil állapotokkal bővítjük az összevont állapottáblát.
Az instabil állapotokkal való bővítést úgy kell megoldani, hogy a stabil állapotok egymás utáni sorrendje ne változzék, ugyanakkor az új instabil állapotokat úgy kell kódolni, hogy az egymás után következő tranziens kódok között csak egy szekunder változó értékében legyen különbség.

Széchenyi István Egyetem

Aszinkron sorrendi hálózatok állapotkódolási módszerei *(1)*

Példa:

	00	01	10	11
a	(a)	b	c	-
b	d	(b)	-	d
c	d	-	(c)	d
d	(d)	a	a	(d)



áll.kódok	0	1
0	a	b
1	c	(d)

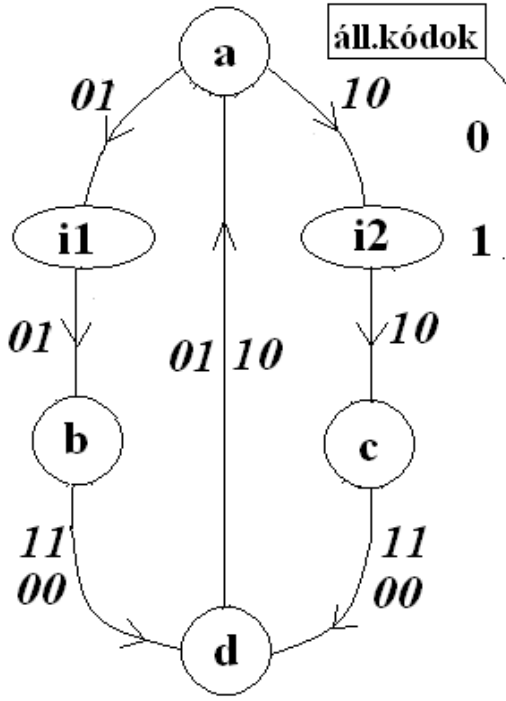
az a-val nem szomszédos

Széchenyi István Egyetem

Aszinkron sorrendi hálózatok állapotkódolási módszerei *(1)*

Megoldás:

- **i1** és **i2** beillesztése



áll.kódok	0 0	0 1	1 1	0 0
0	a	d	c	i2
1	i1	b		

	00	01	10	11
a	(a)	i1	i2	-
b	d	(b)	-	d
c	d	-	(c)	d
d	(d)	a	a	(d)
i1	-	b	-	-
i2	-	-	c	-

Aszinkron sorrendi hálózatok állapotkódolási módszerei *(2)*

- *a Tracey-Unger módszer elve*

Aszinkron sorrendi hálózatokban a szekunder változók között kritikus versenyhelyzet akkor áll elő, ha egy stabil állapotból kiindulva megváltoztatjuk a bemeneti kombinációt, és ennek hatására olyan átmeneti állapotkód áll elő, amelynek sorában és az adott bemeneti kombináció oszlopában ez az állapotkód szerepel. A nem kívánt átmeneti állapotkódot *hazárd kódnak* nevezzük.

Aszinkron sorrendi hálózatok állapotkódolási módszerei *(2)*

A Tracey-Unger módszer lényege, hogy a normális (tervezett) állapotátmenethez tartozó kiinduló és cél állapotok kódjai legalább egy adott szekunder változóban megegyezzenek, és ebben a változóban mindketten különbözzenek a hazárd kódtól.

(Ilyenkor ugyanis ez a szekunder változó az átmenet során állandó marad, és így soha sem áll elő a hazárd állapot kódja.)

Széchenyi István Egyetem

Aszinkron sorrendi hálózatok állapotkódolási módszerei *(2)*

Példa:

köv.áll / z

akt.áll \ D C	0 0	0 1	1 0	1 1
s1	(s1) / 0	(s1) / 0	s2 / 0	(s1) / 0
s2	s1 / 0	- / -	(s2) / 0	s3 / 1
s3	s4 / 1	(s3) / 1	(s3) / 1	(s3) / 1
s4	(s4) / 1	s1 / 0	s3 / 1	- / -

A megváltozott bemeneti kombináció oszlopában találjuk a tervezett új stabil állapot szimbólumát, valamint a stabilizálódott szimbólumot is. Az oszlopban szereplő minden más stabil állapot „leselkedő” *potenciális hazárd.*

Például : ha (00, s1) állapotból az (10, s2) állapotba mennénk, az s3 leselkedő, azaz el kéne kerülni, hogy a kódja tranziensként megjelenjen.

Széchenyi István Egyetem

Aszinkron sorrendi hálózatok állapotkódolási módszerei *(2)*

Megoldás:

köv.áll / z

akt.áll \ D C	0 0	0 1	1 0	1 1
s1	(s1) / 0	(s1) / 0	s2 / 0	(s1) / 0
s2	s1 / 0	- / -	(s2) / 0	s3 / 1
s3	s4 / 1	(s3) / 1	(s3) / 1	(s3) / 1
s4	(s4) / 1	s1 / 0	s3 / 1	- / -

A lehetséges bemeneti változások szerint listába vesszük a tervezett stabil-stabil átmeneteket, a leselkedő feltüntetésével.

00->01	00->10	01->00	01->11	10->00	10->11	11->01	11->10
s4->s1 s3 *1*	s1->s2 s3 *2*	s3->s4 s1 *4*	-	s2->s1 s4 *5*	s2->s3 s1 *6*	-	~~s1->s2~~ ~~s3~~
	s4->s3 s2 *3*			~~s3->s4~~ ~~s1~~			

Széchenyi István Egyetem

Aszinkron sorrendi hálózatok állapotkódolási módszerei *(2)*

00->01	00->10	01->00	01->11	10->00	10->11	11->01	11->10
s4->s1 s3 *1*	s1->s2 s3 *2*	s3->s4 s1 *4*	-	s2->s1 s4 *5*	s2->s3 s1 *6*	-	~~s1->s2~~ s3
	s4->s3 s2 *3*			~~s3->s4~~ s1			

- ahány különböző szabály, annyi szekunder változót írunk fel, és annak az állapotait a szabály alapján határozzuk meg

←

	Y1	*Y2*	*Y3*	*Y4*	*Y5*	*Y6*
	1 2	*1 2*	*1 2*	*1 2*	*1 2*	*1 2*
s1	*0 1*	*0 1*	*- -*	*1 0*	*0 1*	*1 0*
s2	*- -*	*0 1*	*1 0*	*- -*	*0 1*	*0 1*
s3	*1 0*	*1 0*	*0 1*	*0 1*	*- -*	*0 1*
s4	*0 1*	*- -*	*0 1*	*0 1*	*1 0*	*- -*

- minden szekunder változóra mindkét lehetséges választást felírjuk

←

Aszinkron sorrendi hálózatok állapotkódolási módszerei *(2)*

	Y1		Y2		Y3		Y4		Y5		Y6	
	1	2	1	2	1	2	1	2	1	2	1	2
s1	0	1	0	1	-	-	1	0	0	1	1	0
s2	-	-	0	1	1	0	-	-	0	1	0	1
s3	1	0	1	0	0	1	0	1	-	-	0	1
s4	0	1	-	-	0	1	0	1	1	0	-	-

- összevonható változók

- két szekunder változó csoport

	Y_p	Y_z
s1	0	0
s2	0	1
s3	1	1
s4	1	0

Széchenyi István Egyetem

Aszinkron sorrendi hálózatok állapotkódolási módszerei *(2)*

	Y_p	Y_z
s1	*0*	*0*
s2	*0*	*1*
s3	*1*	*1*
s4	*1*	*0*

köv.áll / z

akt.áll \ D C	0 0	0 1	1 0	1 1
s1	(s1) / 0	(s1) / 0	s2 / 0	(s1) / 0
s2	s1 / 0	- / -	(s2) / 0	s3 / 1
s3	s4 / 1	(s3) / 1	(s3) / 1	(s3) / 1
s4	(s4) / 1	s1 / 0	s3 / 1	- / -

↓

⇒

	$Y_p Y_z / Z$			
Y_p^v Y_z^v	*0 0*	*0 1*	*1 0*	*1 1*
0 0	*(0 0) / 0*	*(0 0) / 0*	*0 1 / 0*	*(0 0) / 0*
0 1	*0 0 / 0*	*- - / -*	*(0 1) / 0*	*1 1 / 1*
1 1	*1 0 / 1*	*(1 1) / 1*	*(1 1) / 1*	*(1 1) / 1*
1 0	*(1 0) / 1*	*0 0 / 0*	*1 1 / 1*	*- - / -*

- előzetes szimbolikus állapottábla
- a kódolt állapottábla

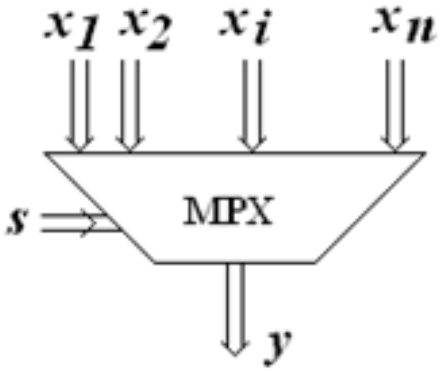
(…)

Az összetett digitális egységek csoportjai

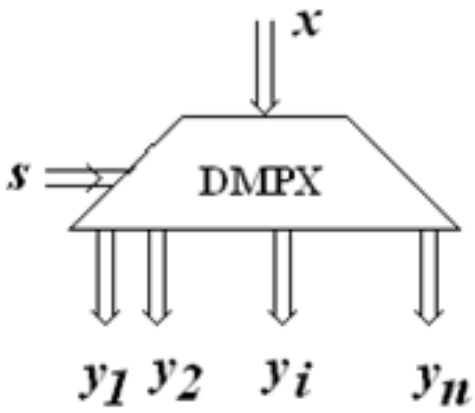
- *Multiplexerek, demultiplexerek:*
  - *feladatuk adatút szakaszok kijelölése*
- *Regiszterek:*
  - *adatokat tárolnak, és ezek elérését biztosítják*
- *Funkciós egységek:*
  - *adatok közötti műveleteket végeznek*

Multiplexerek, demultiplexerek

- *feladatuk adatút szakaszok kijelölése*



Multiplexer általános sémája



Demultiplexer általános sémája

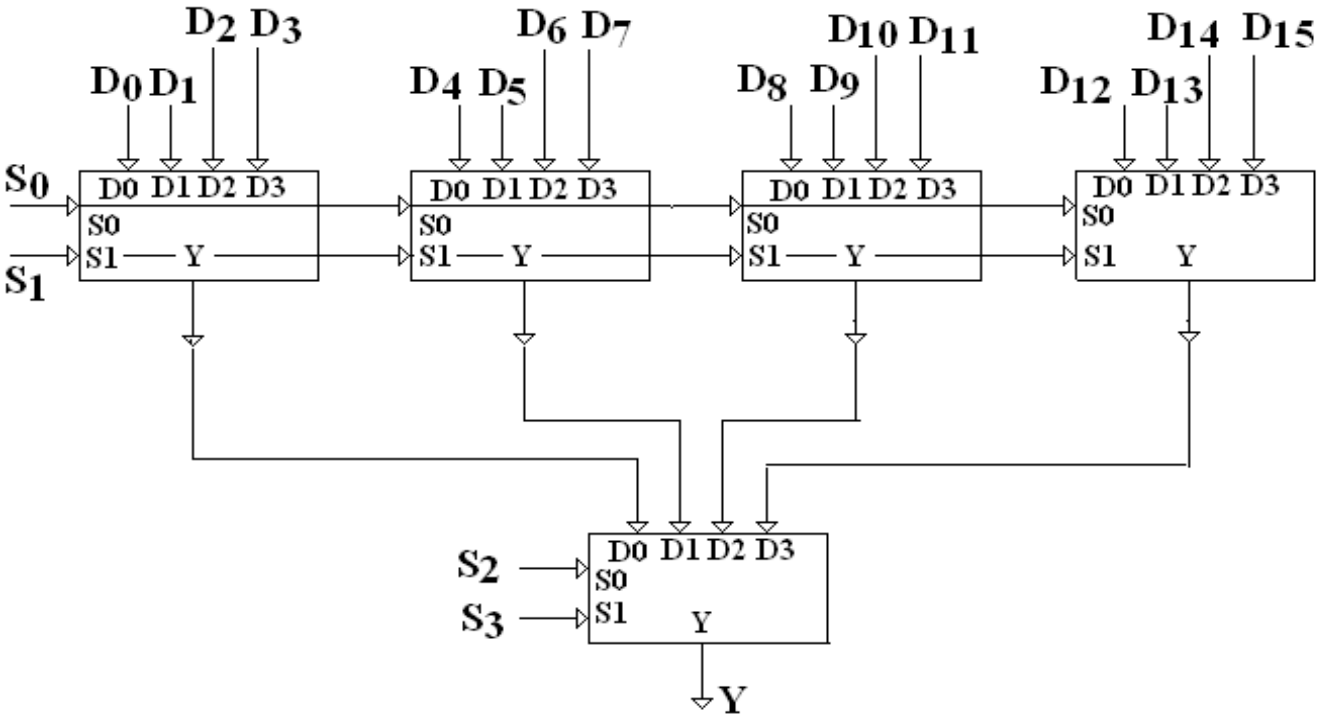
Multiplexerek

- *A multiplexerek, mint programozható logikai hálózatok (ism.):*
  - a mintermes kanonikus alakban megadott függvény mintermjeit a címző (vezérlő) bemenetekre adott címek képviselik, és a megcímzett adatbemenetre rá kell kapcsolnunk az adott mintermhez tartozó logikai értéket. Ezeknek a logikai konstansoknak a bemenetekre való kapcsolását tekinthetjük a multiplexerek programozásának.

(…)

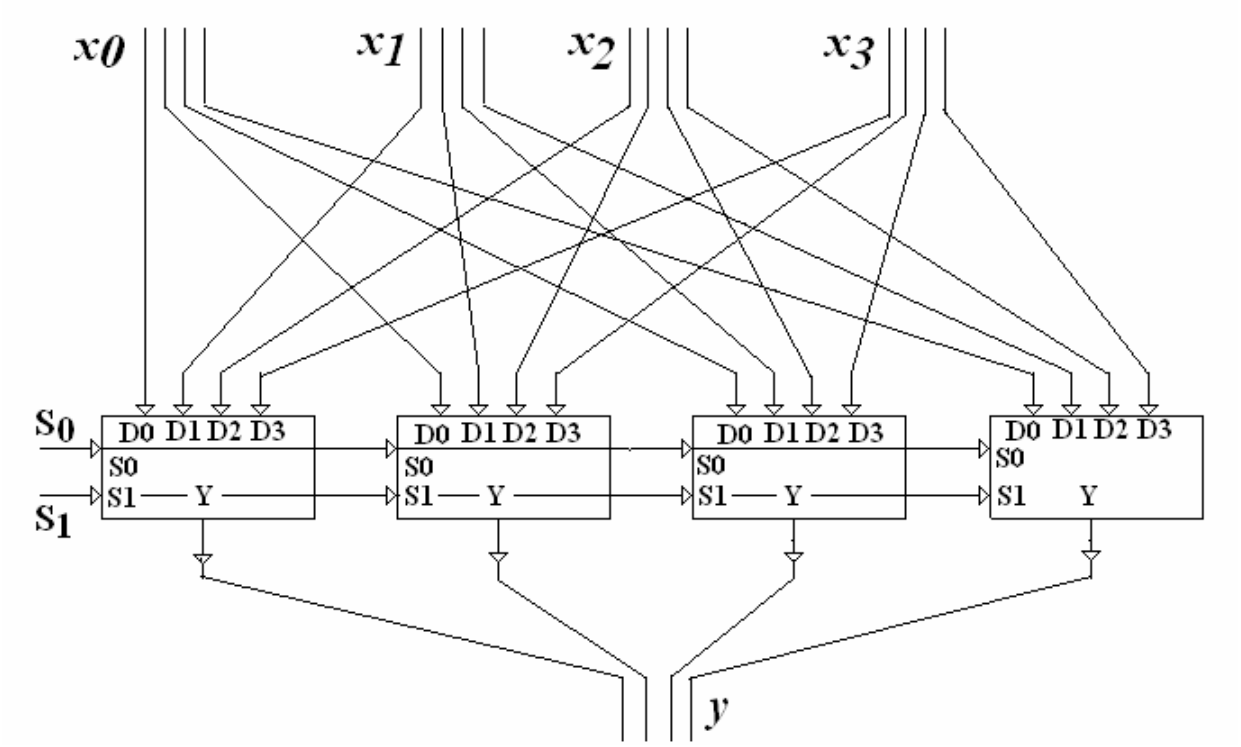
Multiplexerek

- *Alkalmazási példa1.: bővítés a bemenetek számának növelésére (MPX4-1 multiplexerek felhasználásával)*



Multiplexerek

- *Alkalmazási példa2.: bővítés sínek közötti választás céljából(MPX4-1 alapegységekből)*

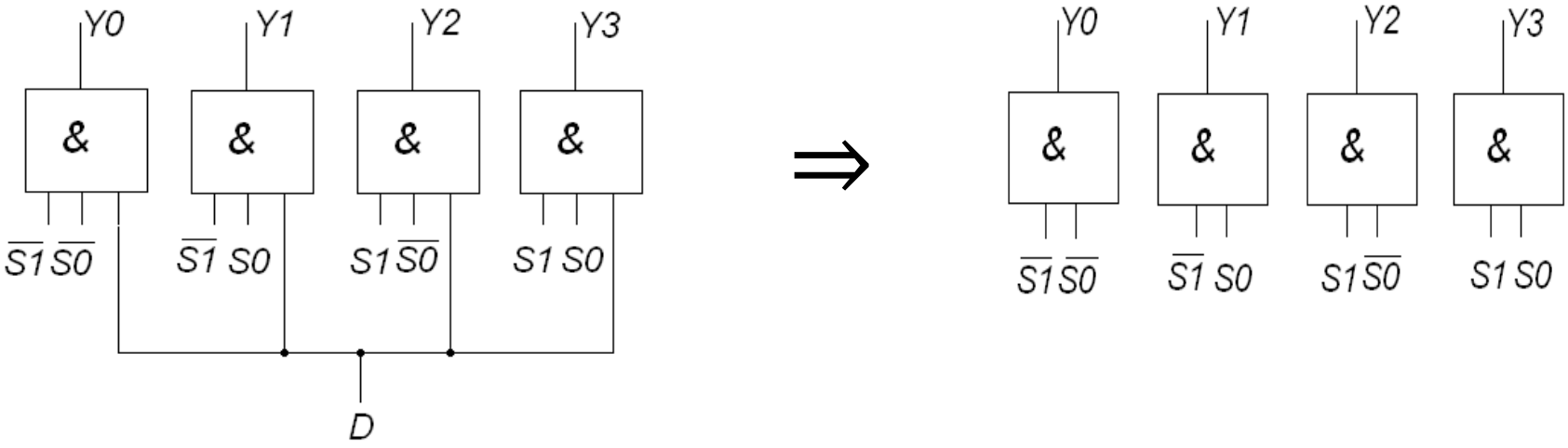


Demultiplexerek

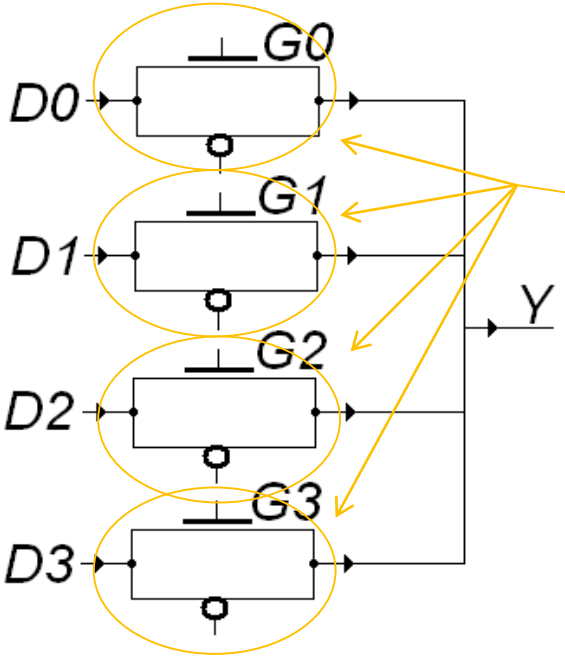
- *A demultiplexerek, mint dekóderek alkalmazása:*
  - ha egy demultiplexer adatbemenetét állandó, logikai ’1’ szintre kapcsoljuk, akkor ez egyenértékű azzal, hogy a hálózatában szereplő ’ÉS’ kapuk bemenetei közül elhagyjuk az adatbemenetet. Az ilyen áramkör sajátossága, hogy a kiválasztó bemenetekre kapcsolt kombinációk csak egyetlen, a kiválasztó kódnak megfelelő kimeneten eredményeznek magas szintet. Az ilyen áramköröket nevezzük dekódereknek.

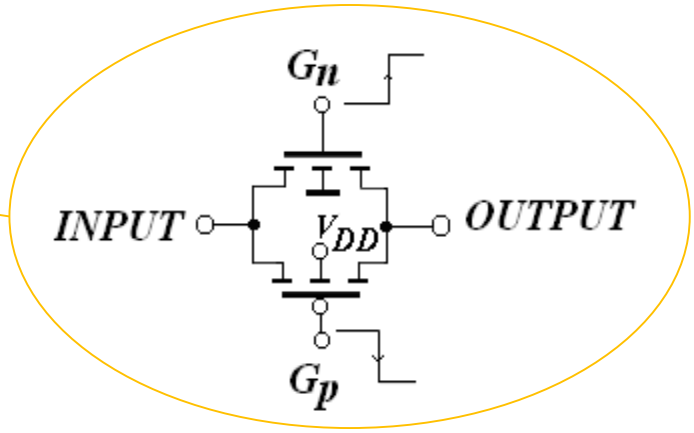
Demultiplexerek

- *Példa DMPX1-4 egység dekóderként történő felhasználására:*



Az átvivőkapu (transmission gate) alkalmazása a multiplexerekben és a demultiplexerekben





CMOS kapcsoló(transmission-gate): egy *n*- és egy *p*-csatornás MOS tranzisztorpár párhuzamosan összekapcsolva

- ***A harmadik logikai állapot, a „lebegő” kimenet lehetősége!***

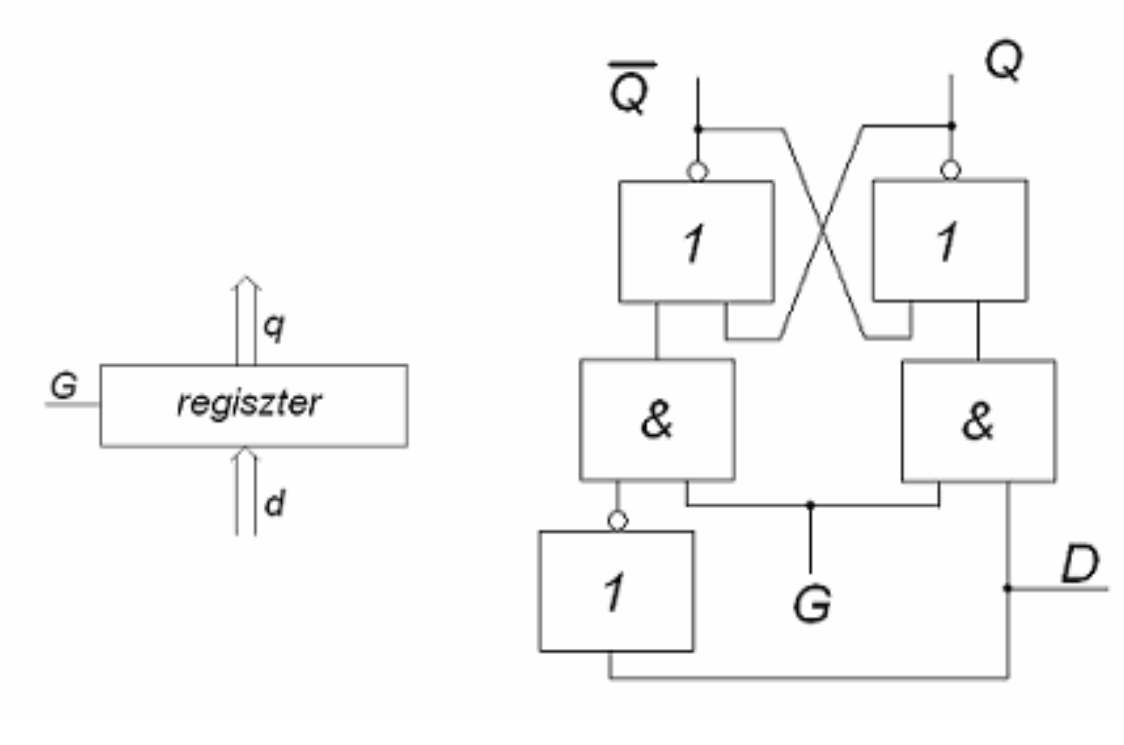
Regiszterek

- *adatokat tárolnak, és ezek elérését biztosítják*

Alapelemek:

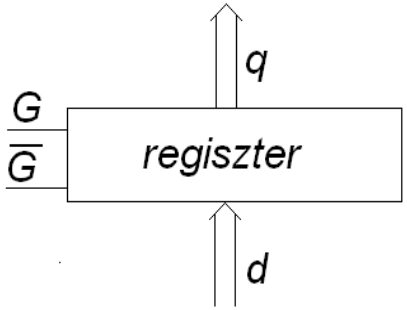
- *szintvezérelt statikus regiszter*
- *szintvezérelt statikus regiszter ponált és negált beírójelekkel*
- *kvázistatikus regiszter*
- *élvezérelt regiszter*

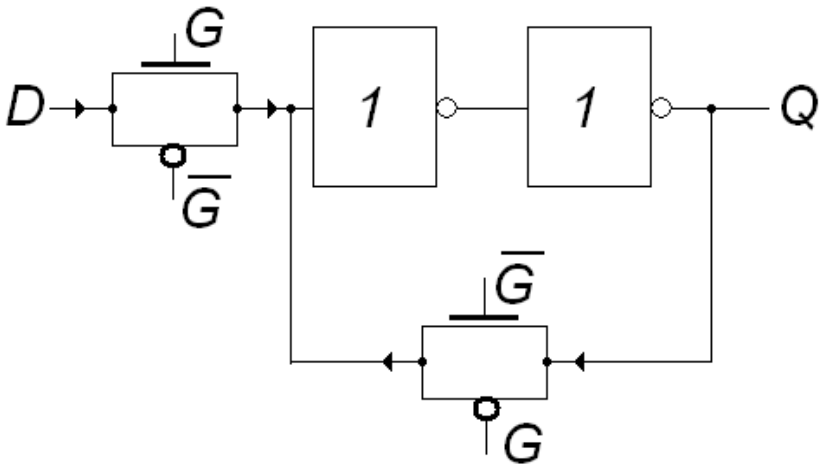
Szintvezérelt statikus regiszter



- A regiszter a ***G***=1 szint fennállásának idején „átlátszó”, azaz ***d*** változásai késleltetve ugyan, de kijutnak a kimenetre.

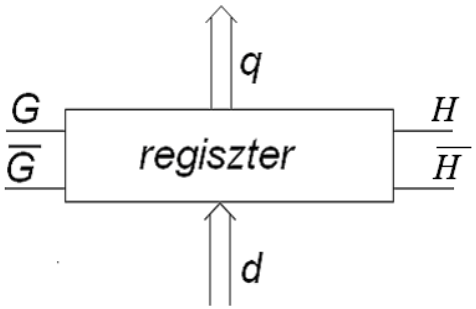
Szintvezérelt statikus regiszter ponált és negált beírójelekkel

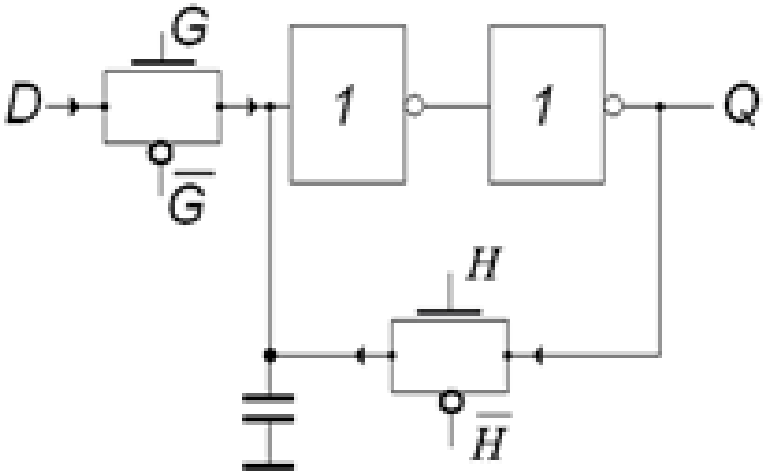




- CMOS átvivőkapu alkalmazása statikus regiszterben

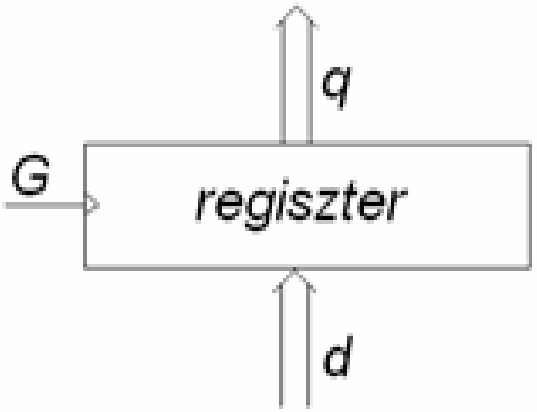
Kvázistatikus regiszter





- A kapacitás a ***G*** lefutása és ***H*** felfutása között tárolja a beírt szintet, az inverterek „frissítenek”.

Élvezérelt regiszter



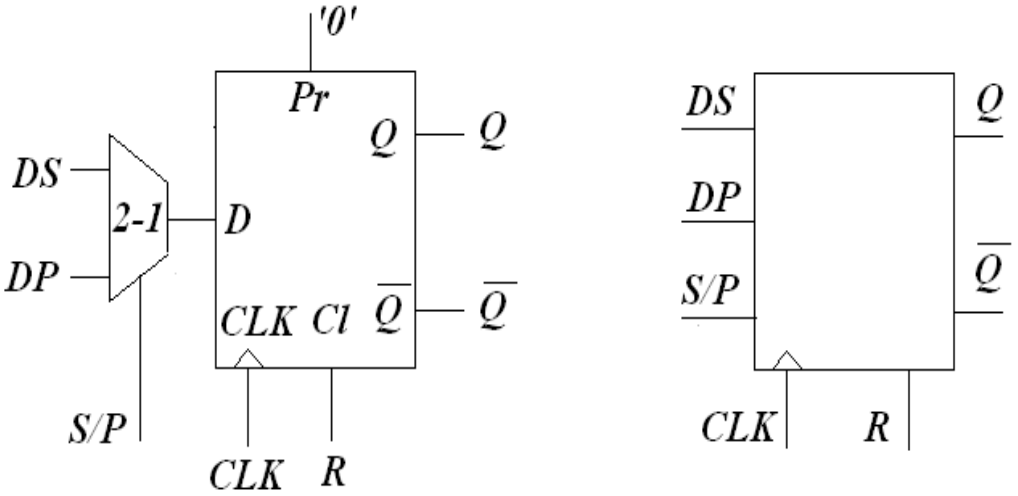
- Az átlátszóság a ***G*** jel felfutásának idejére szűkül! Igen sok előny származik ebből.

Tárolók

- soros elérésű memóriák
- párhuzamos hozzáférésű memóriák

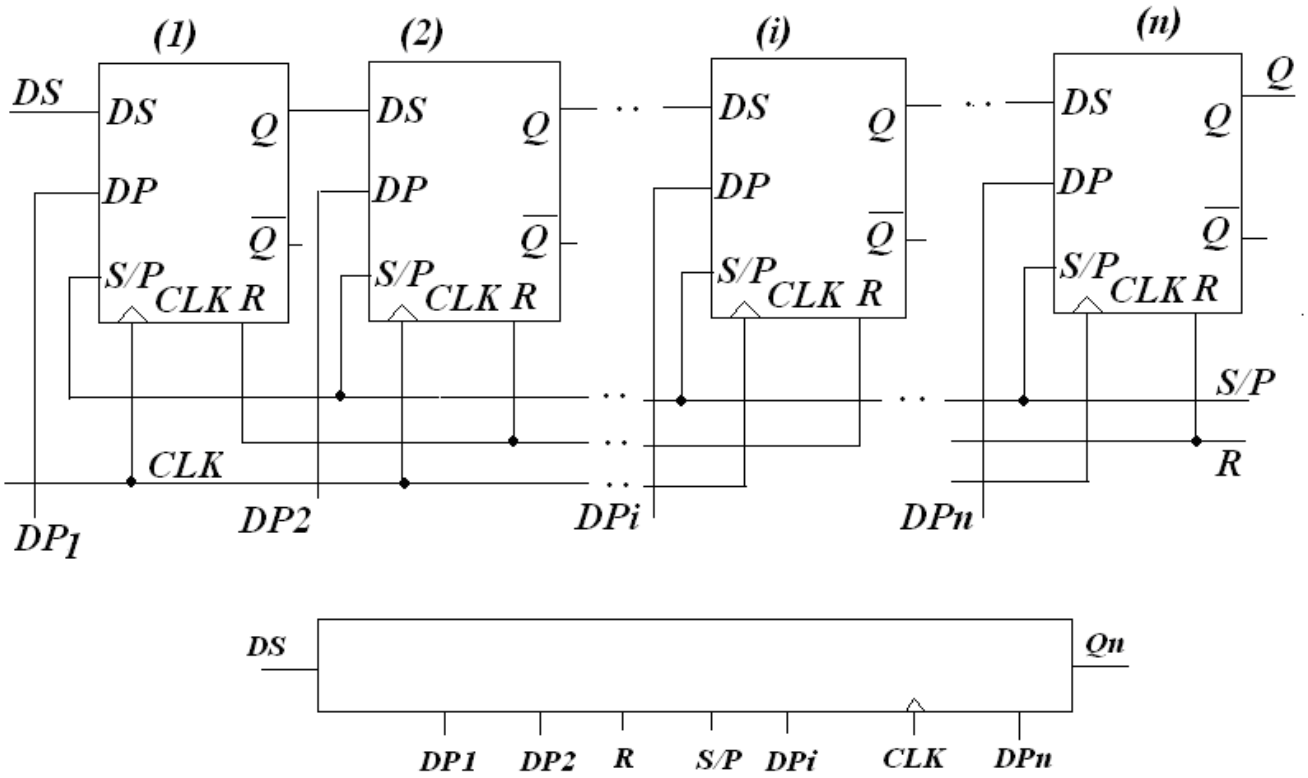
Soros elérésű memóriák

A soros memóriák alapeleme:

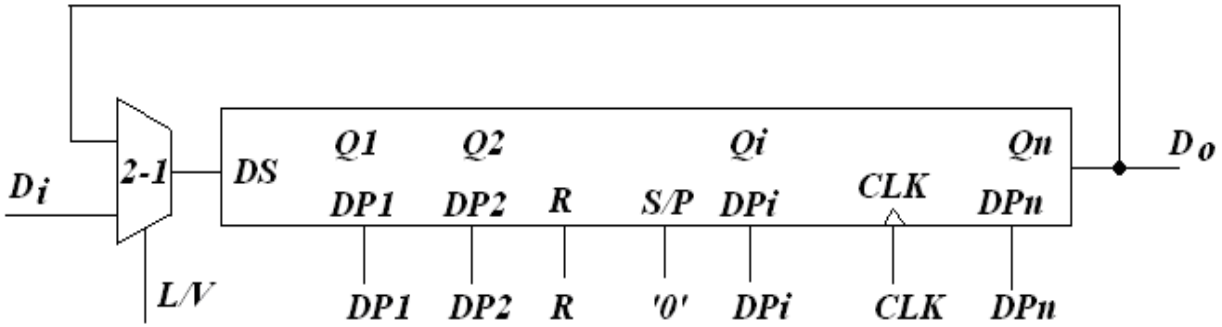


- két bemenetről (*DS; DP*) beírható élvezérelt D-MS flip-flop, a *D* bemeneten MPX2-1 multiplexerrel.

Nyitott, párhuzamosan is betölthető soros elérésű memória-sor (SHIFT-regiszter)

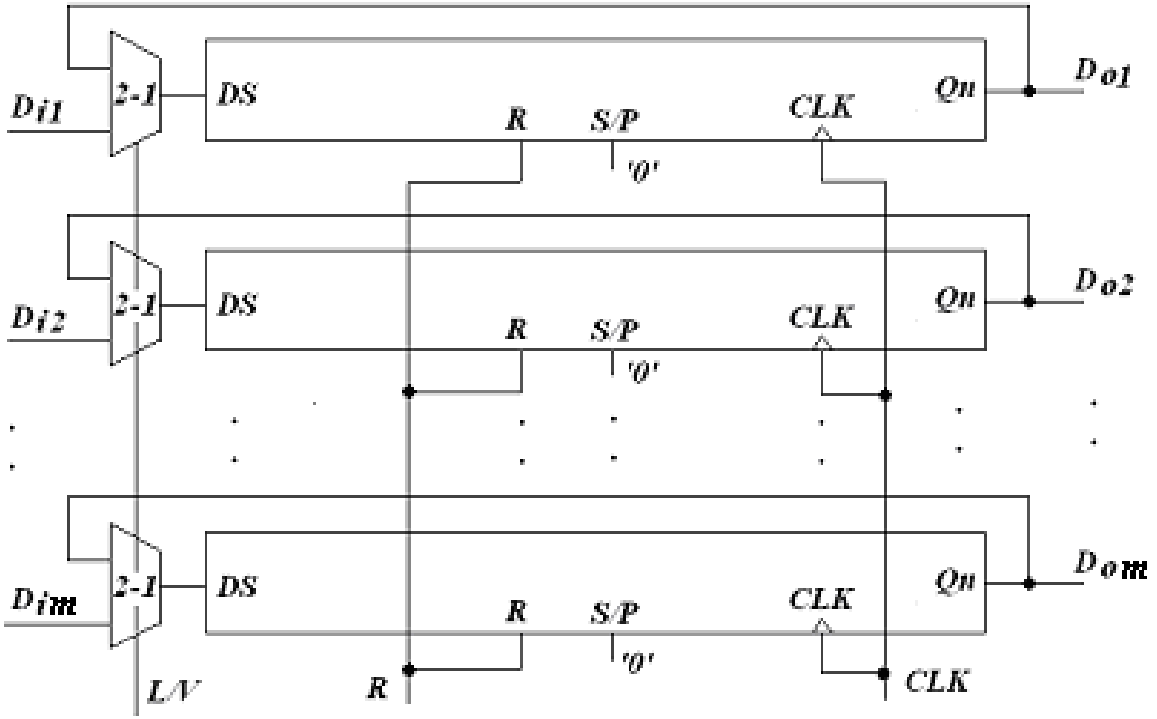


Bit-szervezésű, sorosan rátölthető, párhuzamosan is betölthető soros elérésű memória



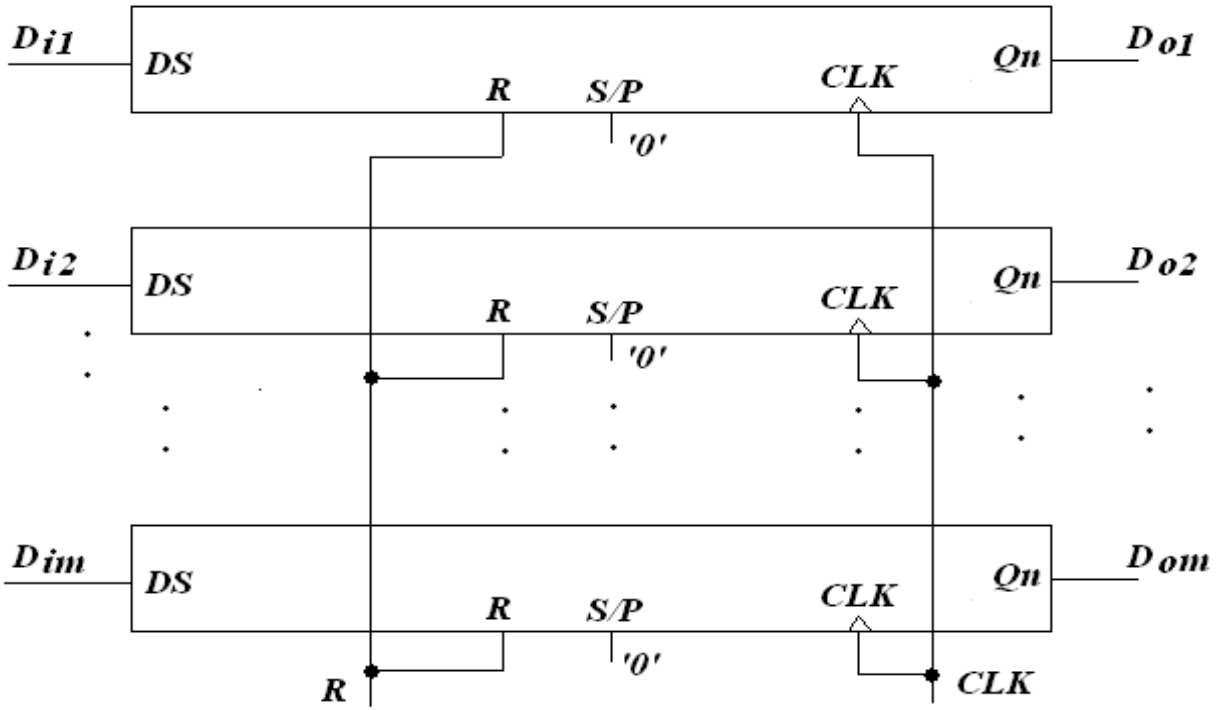
Széchenyi István Egyetem

Szószervezésű soros memóriák (1)**:** sorosan rátölthető soros elérésű memória



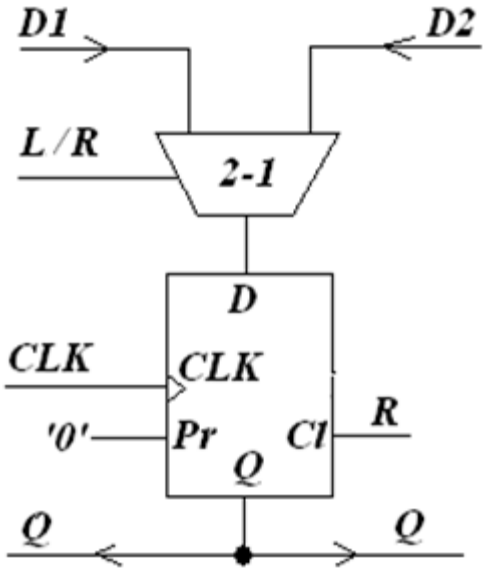
- *m*-bit szószélességű soros memóriaegység

Szószervezésű soros memóriák (2): F-I-F-O memóriák (First In First Out)

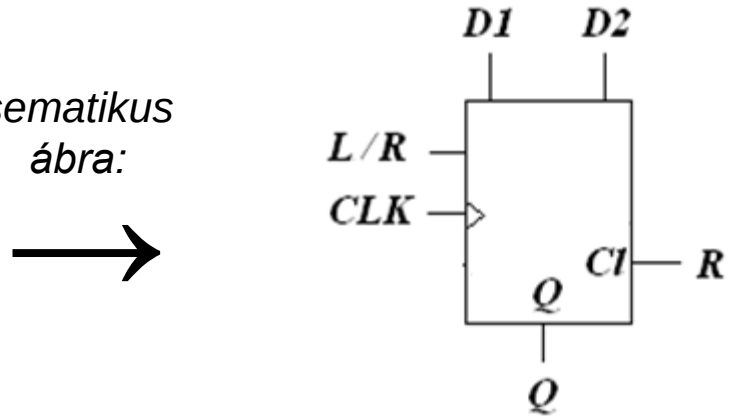


- *m*-bit szószélességű FIFO memória

Szószervezésű soros memóriák (3): L-I-F-O memóriák (Last In First Out)

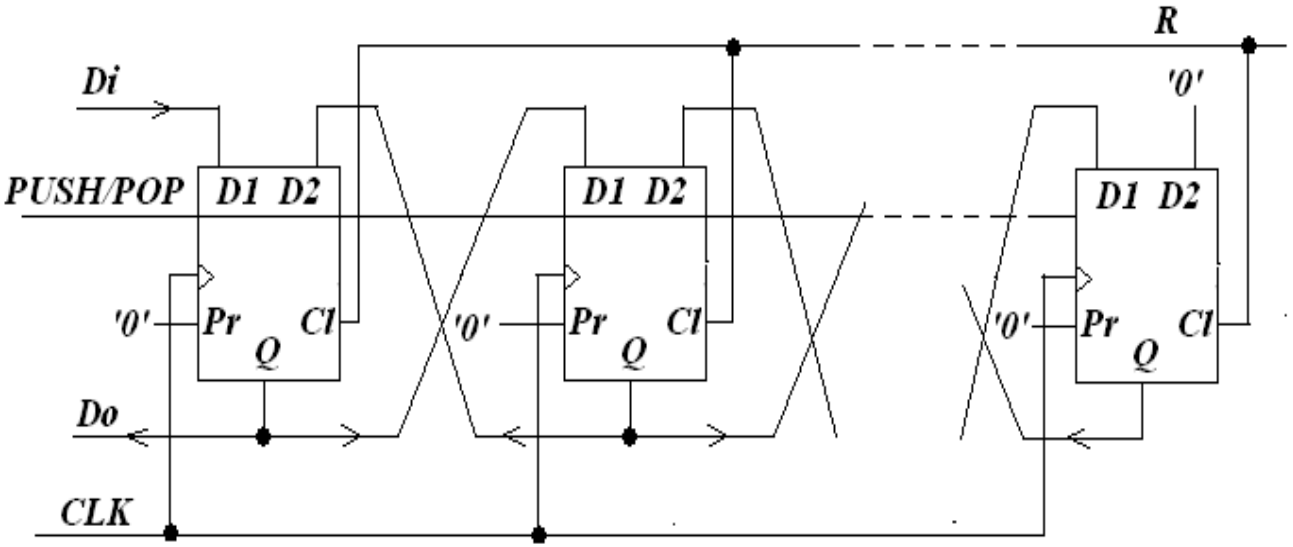


sematikus ábra:



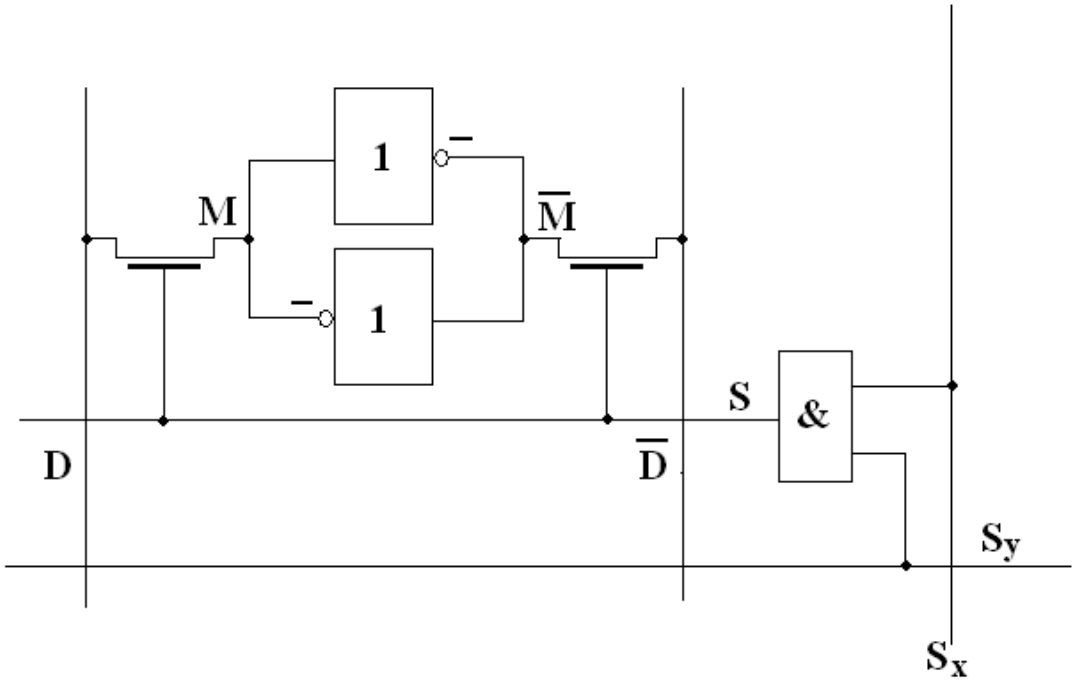
- LIFO memóriaelem

Szószervezésű soros memóriák (3): L-I-F-O memóriák (Last In First Out)



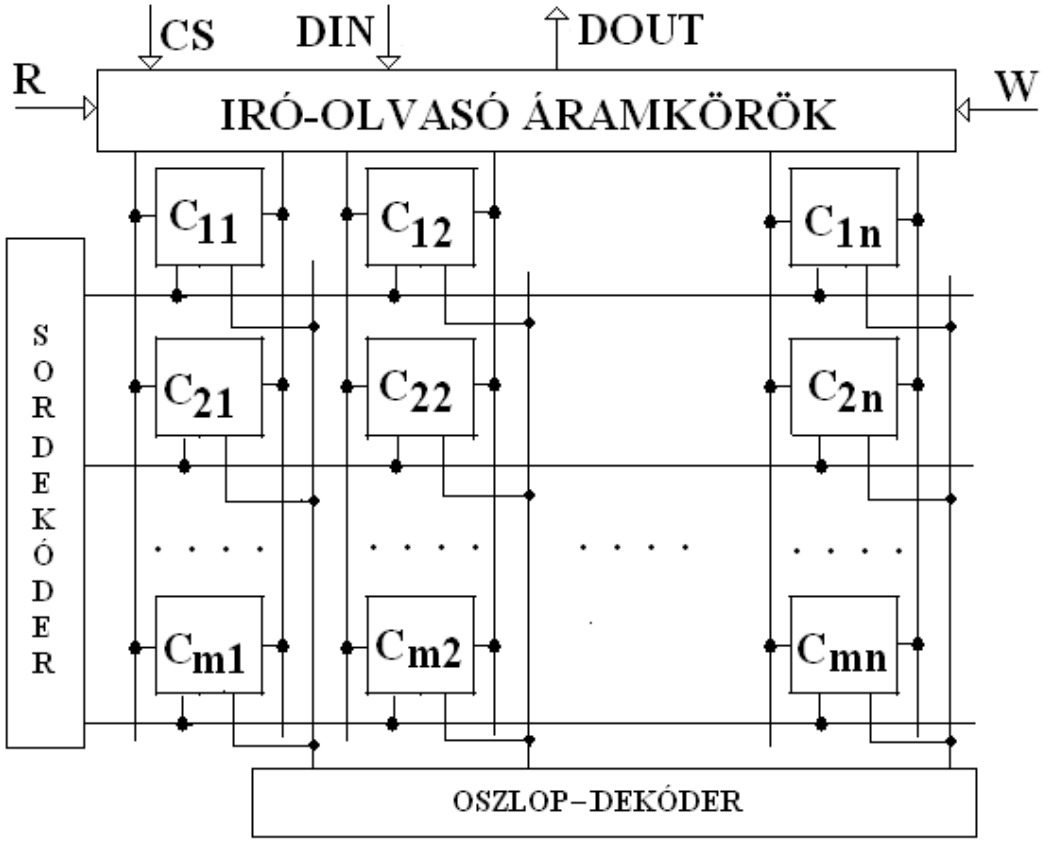
- LIFO memóriasor

Párhuzamos hozzáférésű memóriák (RAM-ok) (1)



- RAM alapcella felépítése

Párhuzamos hozzáférésű memóriák (RAM-ok) (2)

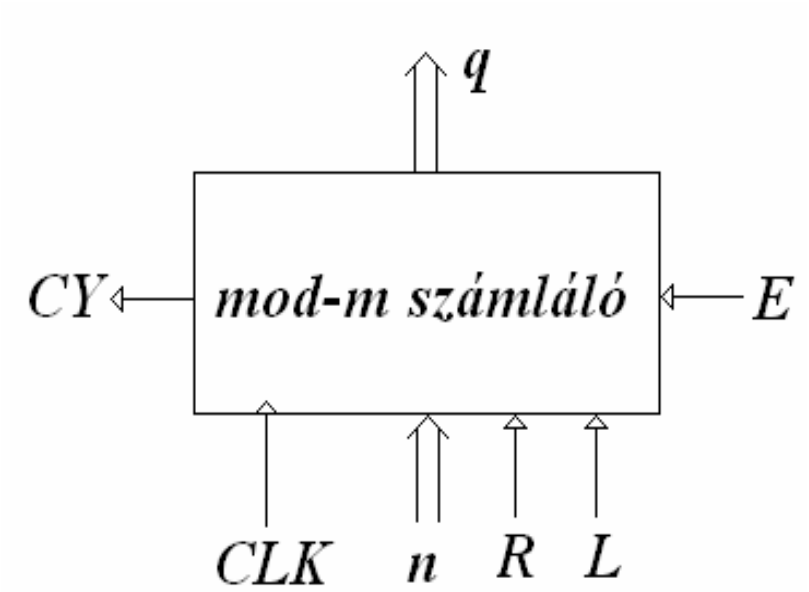


- Bit szervezésű RAM blokk

Állapotregiszterek, számlálók

- szinkron számlálók
- aszinkron számlálók

Szinkron számlálók (ism.)

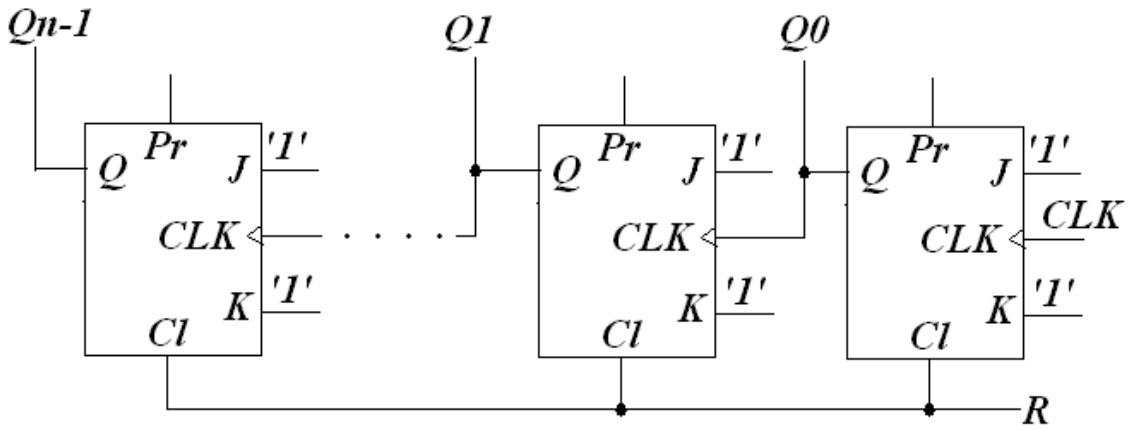


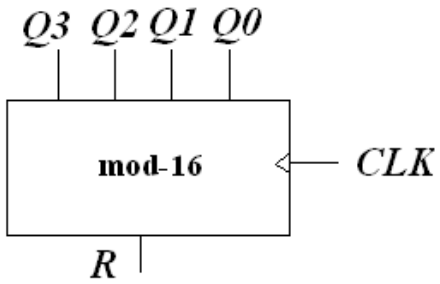
- mod-m szinkron számláló általános sémája

(…)

Széchenyi István Egyetem

Aszinkron számlálók





- Kettes osztók kaszkádja

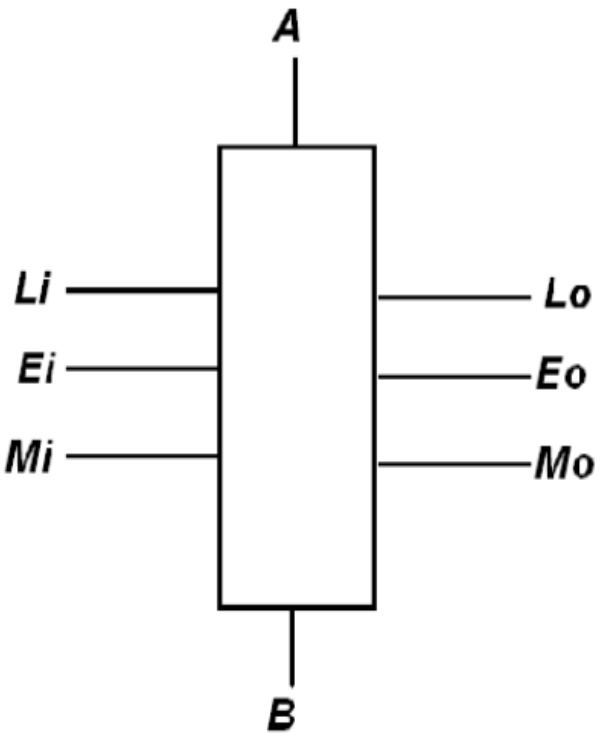
Funkciós egységek

- komparátorok
- összeadók, kivonók
- szorzók, osztók
- vezérlők

…

Komparátorok

Az 1-bites komparátor



$$Lo = Li + Ei\,(\overline{A}\,B)$$

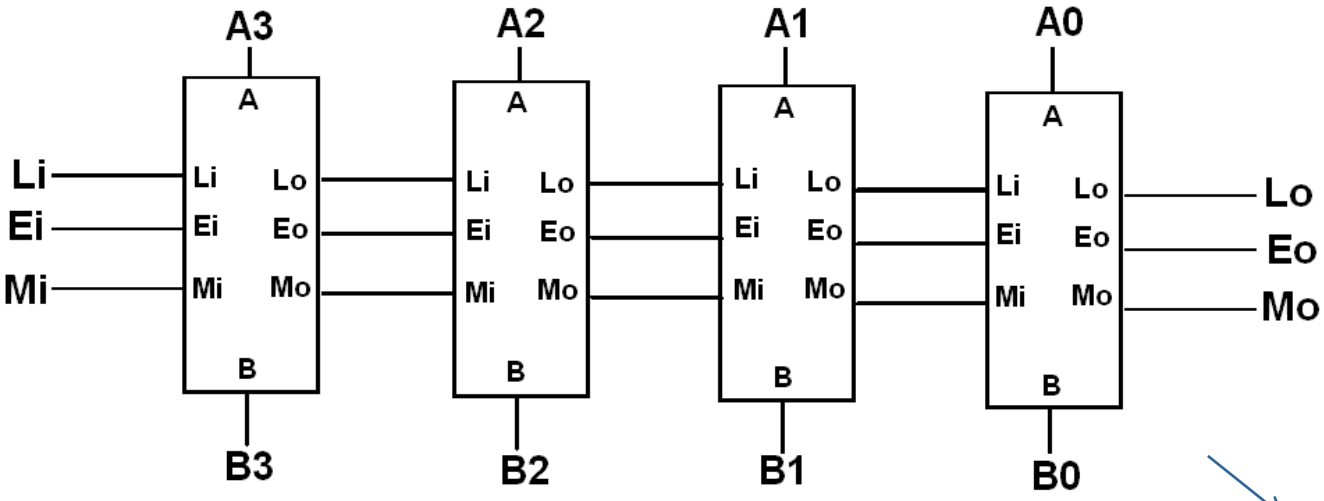
$$Eo = Ei\,(\overline{A}\,\overline{B} + A\,B)$$

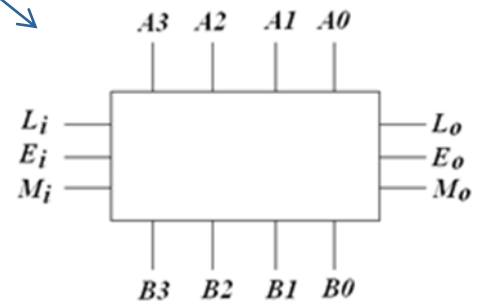
$$Mo = Mi + Ei\,(A\,\overline{B})$$

Széchenyi István Egyetem

Komparátorok

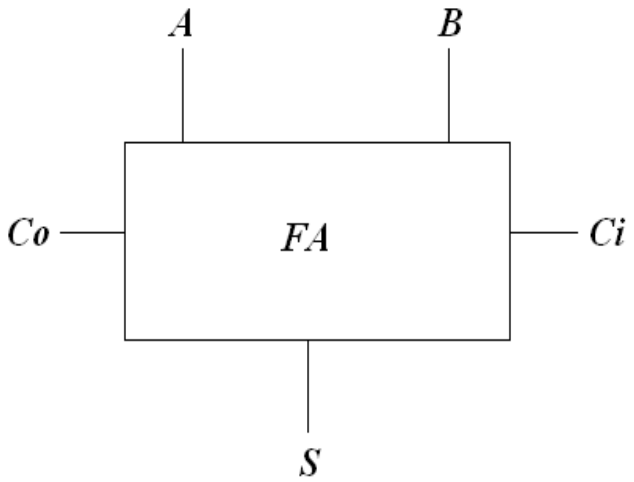
Példa: 4-bites komparátor kialakítása 1-bites alapegységekből





Összeadók

Az 1-bites (teljes) összeadó



- A teljes összeadó sémája

- igazságtábla

A	B	Ci	S	Co
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

$\Downarrow$

$$S = (A \oplus B) \oplus C$$

$$C_o = A\,B + B\,C_i + A\,C_i$$

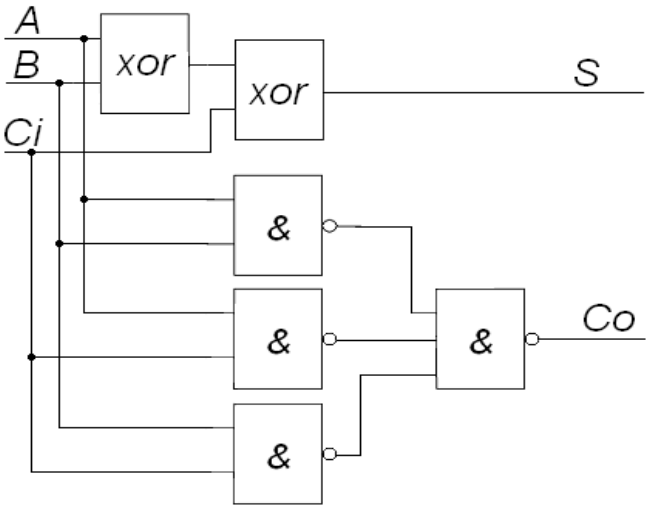
Összeadók

Az 1-bites (teljes) összeadó

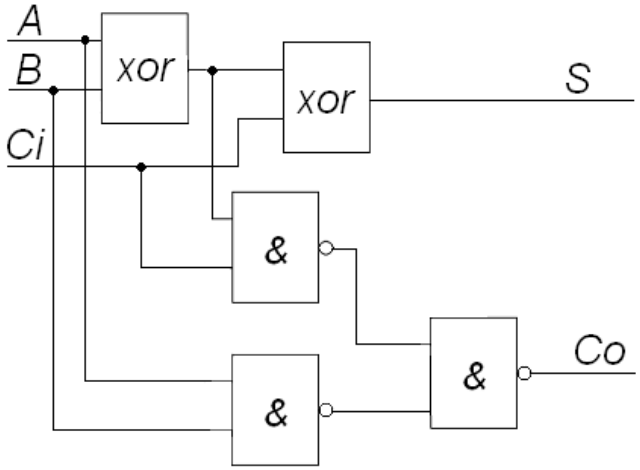
$S = (A \oplus B) \oplus C$

$C_o = A\,B + B\,C_i + A\,C_i = \overline{\overline{(AB)}\ \overline{(B\,C_i)}\ \overline{(A\,C_i)}}$ ***:(1)***

$C_o = (A \oplus B)\ C_i + A\,B = \overline{\overline{(A \oplus B)\,C_i}\ \overline{(AB)}}$ ***:(2)***



:(1)



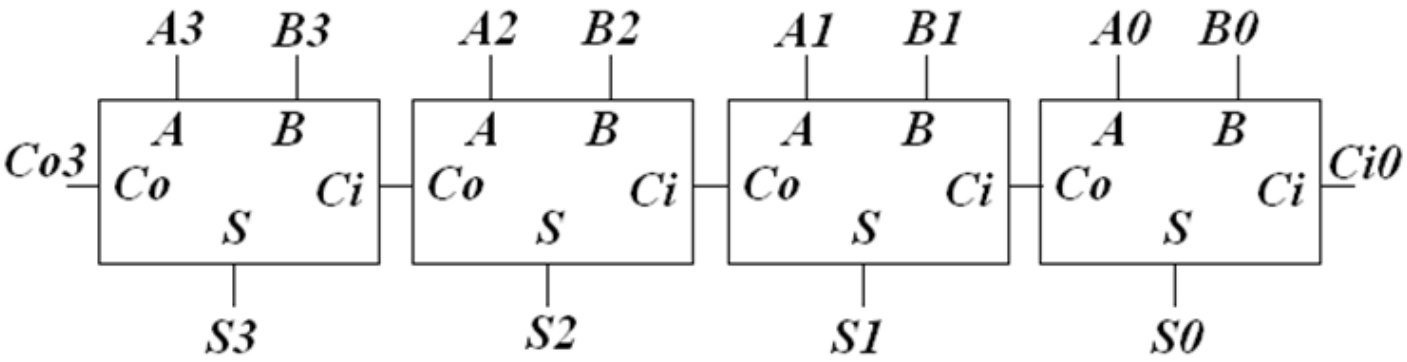
:(2)

Széchenyi István Egyetem

Összeadók

Soros átvitelképzésű összeadó

Példa: soros átvitelképzésű 4-bites összeadó egység, 1-bites alapegységek kaszkádosításával



Összeadók

Párhuzamos átvitelképzésű bit-vektor összeadó

- eredményképzés algoritmusa:

$$P_k = A_k \oplus B_k$$

$$G_k = A_k \, B_k$$

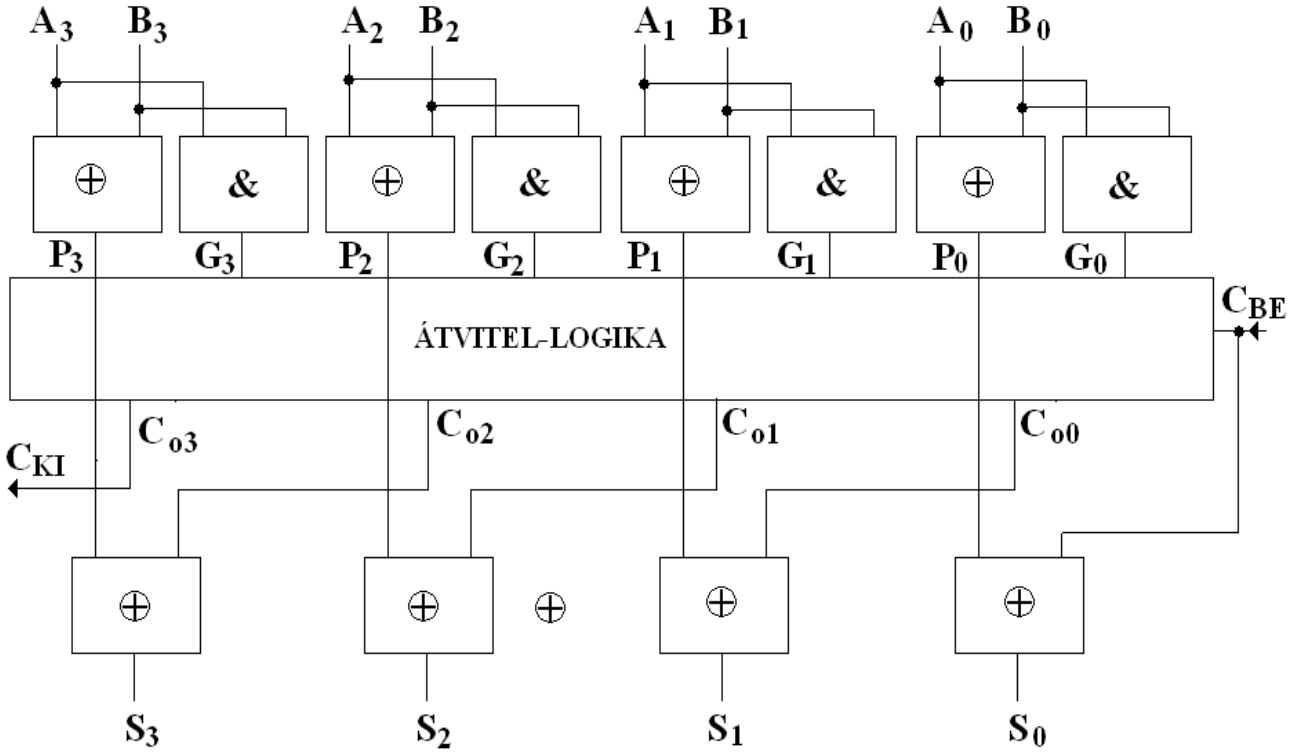
$$S_k = (A_k \oplus B_k) \oplus C_{ik} = P_k \oplus C_{o(k-1)}$$

$$C_{ok} = (A_k \oplus B_k) C_{ik} + A_k \, B_k = P_k C_{o(k-1)} + G_k$$

Összeadók

Párhuzamos átvitelképzésű bit-vektor összeadó

Példa: 4-bites, párhuzamos átvitelképzésű összeadó



Kivonók

A komplemens kód alkalmazása a kivonás műveletében

- elv: a kivonás műveletét a bináris számok kódolásának megfelelő megválasztásával összeadásra lehet visszavezetni.

- megoldás: vegyük a kivonandó kettes komplemensét, és a kissebbítendőhöz adjuk hozzá!

A kettes-komplemens kódú számábrázolás

A: szám, súlyozott bináris kóddal

KK(A) : a szám kettes komplemense, adott szabály szerint előállítva.

Egy kettes komplemens kódú szám (-1)szerese a szám kettes komplemense:

$$A + KK(A) = 0!$$

A kettes komplemens kód:

MSB : előjel

(***MSB****-1****) – LSB*** : számérték

- *ha a szám pozitív, előjele 0, a számérték pedig a szám binárisan súlyozott abszolút értéke*
- *ha a szám negatív, előjele 1, és az abszolút érték a kettes komplemens előállításával határozható meg*

A kettes komplemens előállítása:

1. *lépés : bitenkénti negálás (egyes komplemens)*
2. *lépés : 000….1 hozzáadása az egyes komplemenshez*

Példa: 0 1 0 1 pozitív szám, abszolút értéke 5,
ez tehát a (+5) kettes komplemens kóddal.

Ennek 2-es komplemense -5 kell hogy legyen:

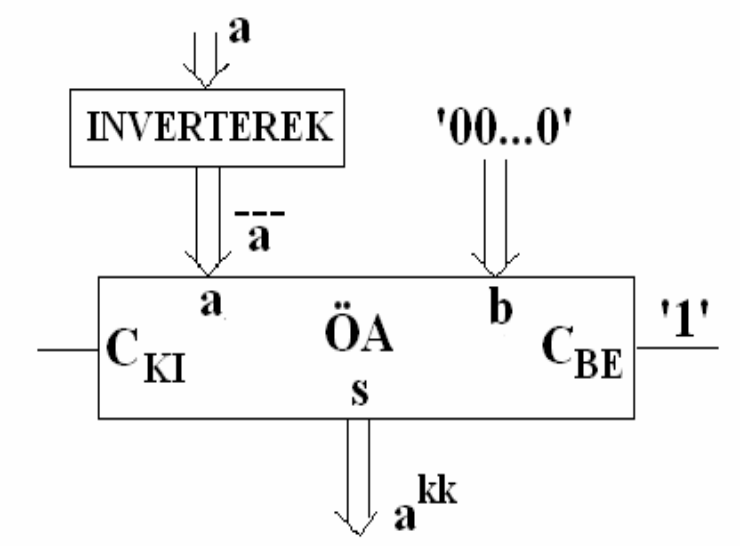
1-es komplemens : 1 0 1 0

2-es komplemens : 1 0 1 1

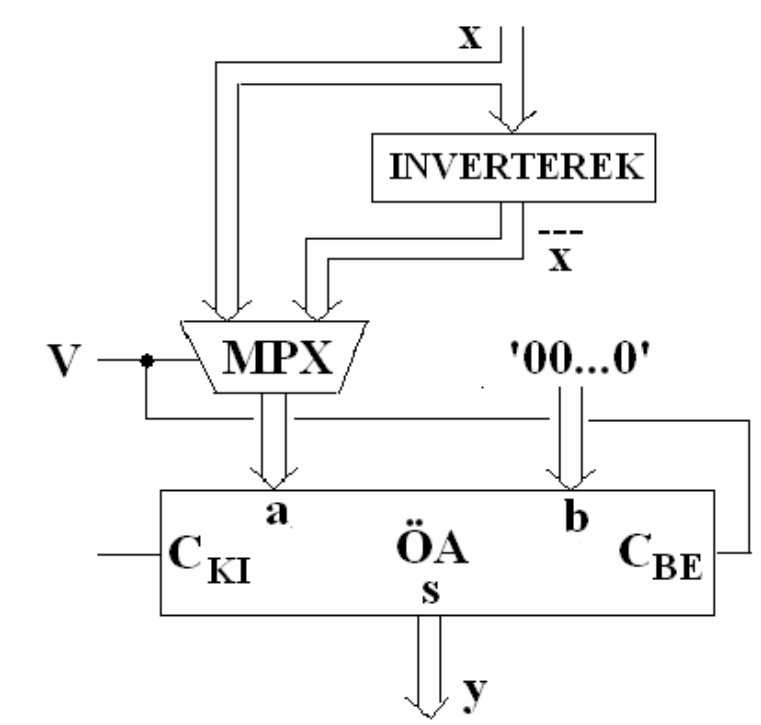
Próba : 0 1 0 1
+ 1 0 1 1
---
(1)0 0 0 0

Kivonók megvalósítása (1)

kettes-komplemens képző egységek

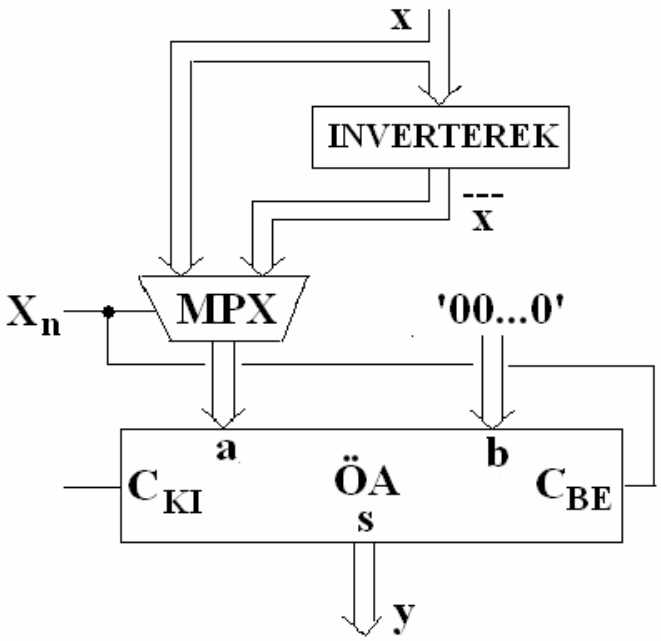


- *kettes-komplemens képző egység*

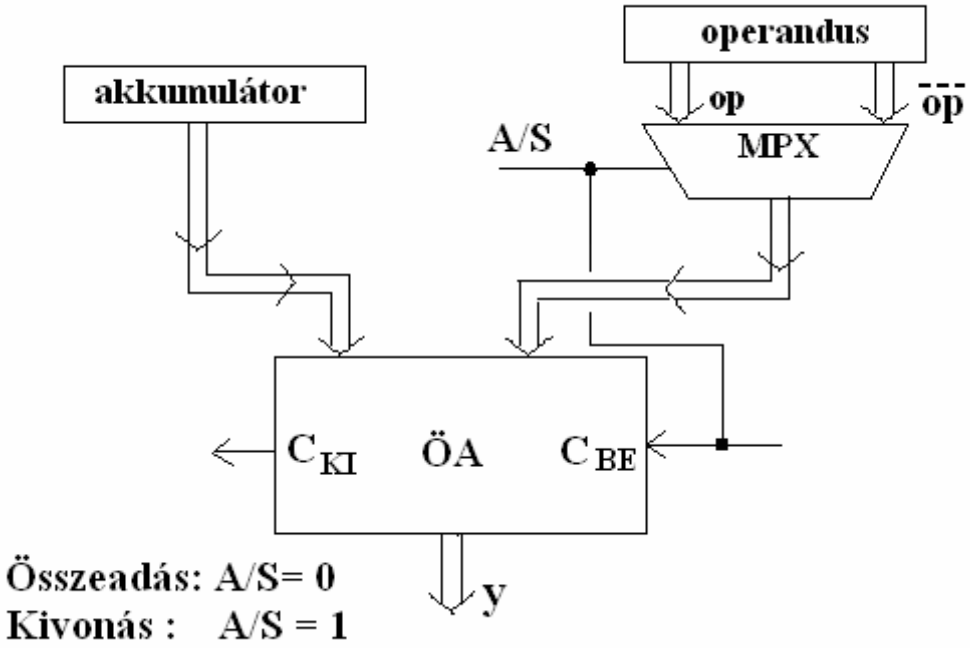


- *vezérelhető kettes-komplemens képző egység*

Kivonók megvalósítása (2)



- *abszolútérték képző egység*



- *kivonás mikroprocesszorok aritmetikai egységében*

Szorzók,osztók

➢ *különböző elvek → sokféle megvalósítás*

néhány példa:

- szorzók:
  - előjeles vagy előjel nélküli szekvenciális szorzók
  - tömbszorzók (array-szorzók)
  - fa-szorzók (Booth-Wallace szorzók)
  - …

- osztók:
  - előjel nélküli visszaállító osztó
  - előjeles nem visszaállító osztó
  - Radix 4 SRT osztó
  - funkcionális iterációs osztók (Newton-Raphson algoritmus)
  - …

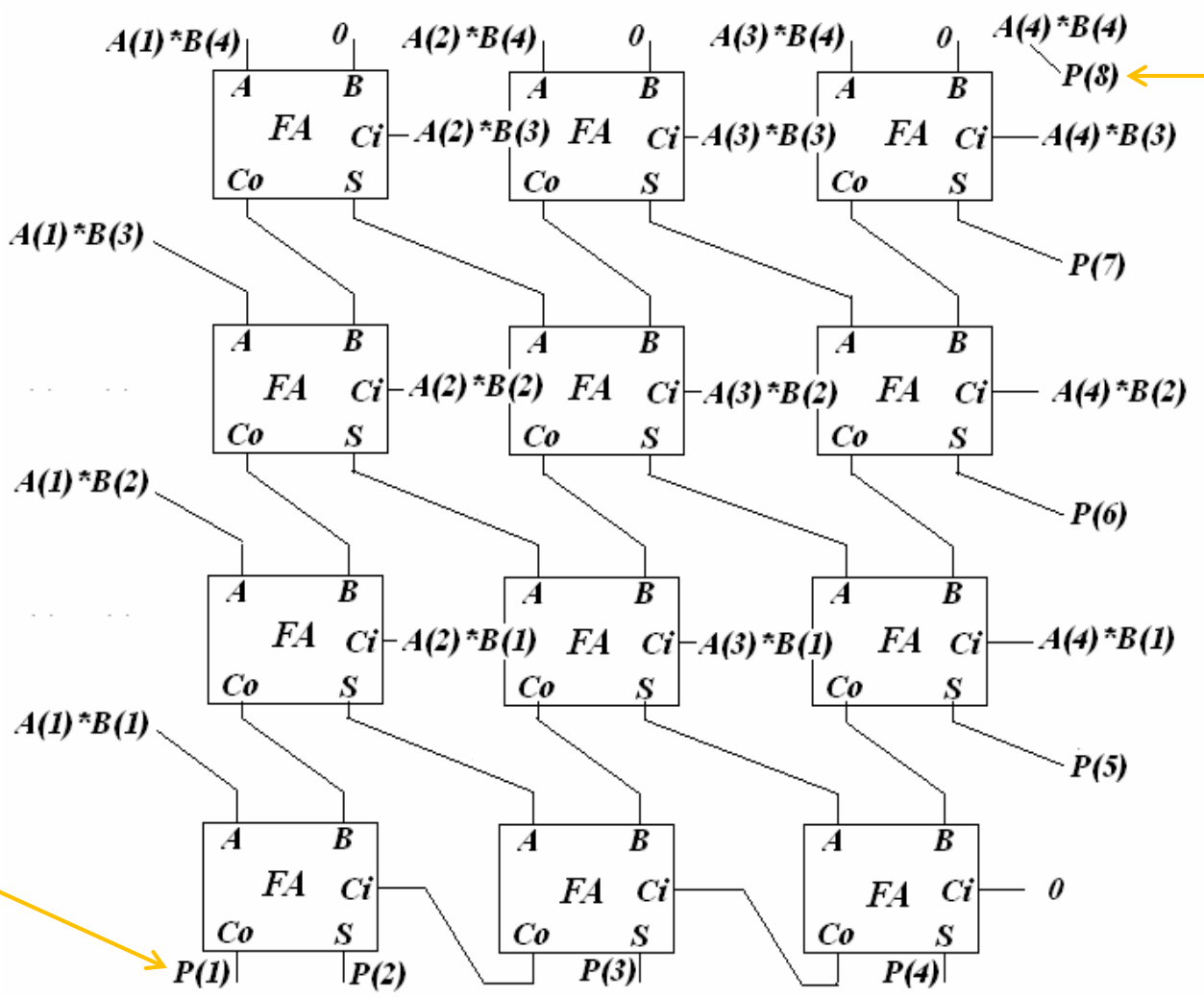
Összetett digitális egységek

Széchenyi István Egyetem

Példa1:

elv:

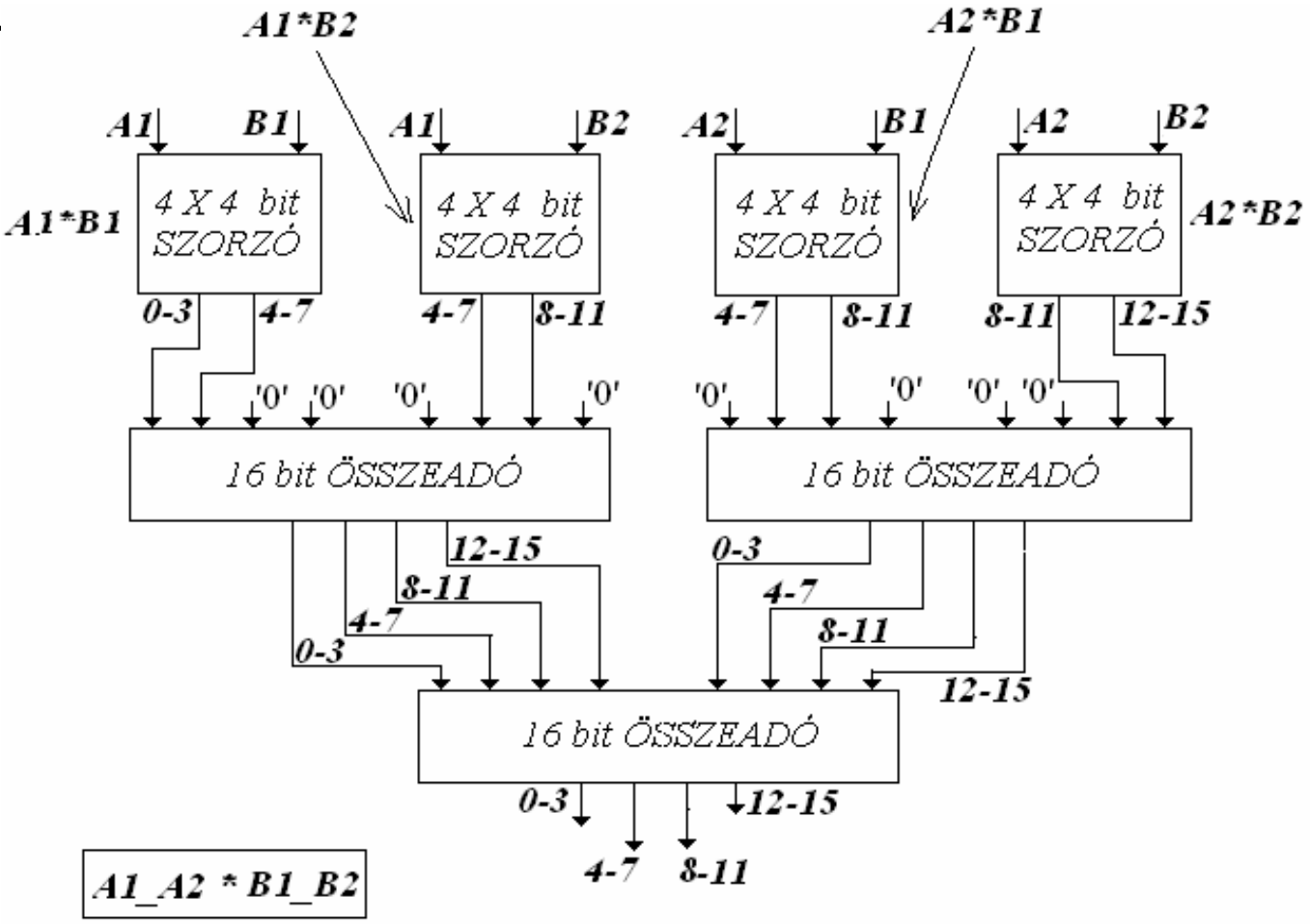
bitenkénti
szorzás
↓
léptetés
↓
összeadás
műveletsor



- ***4 x 4–es tömbszorzó kialakítása teljes összeadókból valamint 'ÉS' kapukból***

Összetett digitális egységek

Példa2:

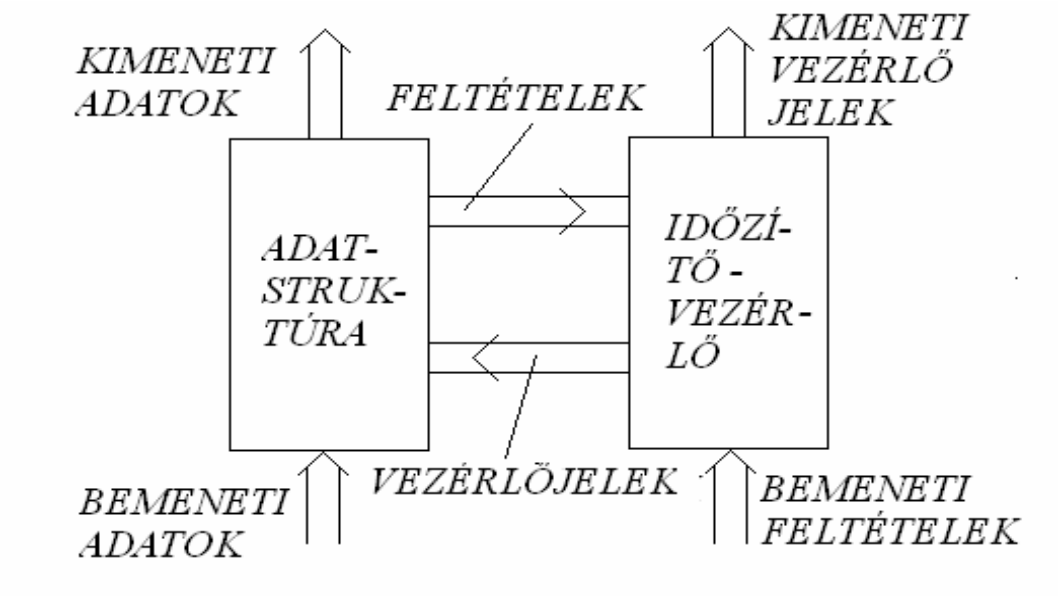


- *8-bites szorzó 4-bites egységekből*

Vezérlők

- *Elv: dekompozíció (ism.)*

  - elkülönítjük az adatutakat az azok kijelölését és időbeli mozgatását meghatározó időzítő vezérlő egységtől:



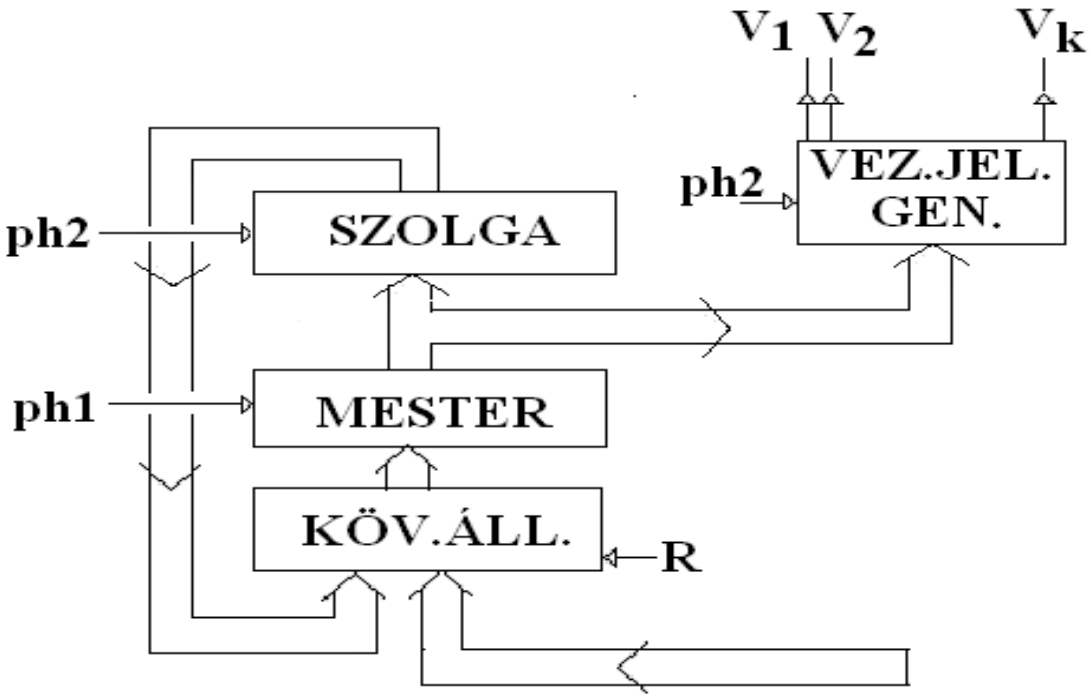
Vezérlők *(1)*

- *számláló bázisú időzítő-vezérlők (ism.)*
- *FSM típusú vezérlők*
  - **F**inite **S**tate **M**achine: az időzítő véges állapotszámú, MS tárolókból az adott célra felépített szinkron sorrendi hálózat.

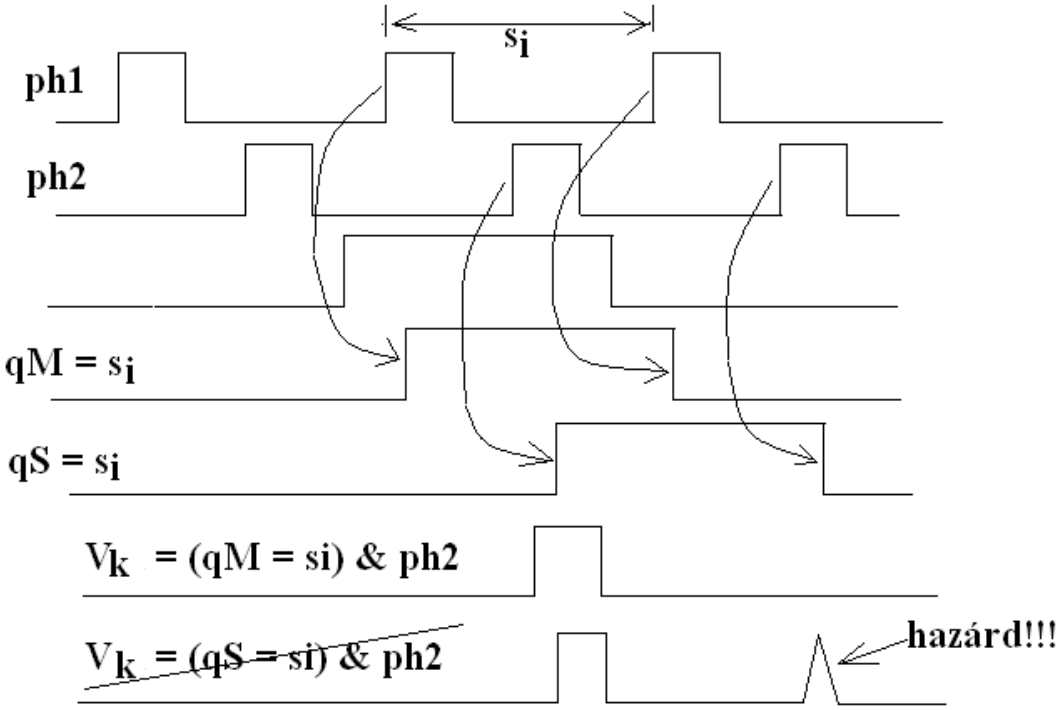
Széchenyi István Egyetem

Vezérlők *(2)*

Kétfázisú (ph_1, ph_2) órajellel működő FSM vezérlő struktúrája:



Vezérlők *(3)*



- *kétfázisú órajellel működtetett FSM típusú időzítő idő-diagramja*