

**Pannon Egyetem**

**Képfeldolgozás és Neuroszámítógépek Tanszék**



# Digitális Rendszerek (BSc)

5. előadás: Szekvenciális hálózatok I.  
Szinkron és aszinkron tárolók, regiszterek

Előadó: Vörösházi Zsolt

[voroshazi@vision.vein.hu](mailto:voroshazi@vision.vein.hu)

# Jegyzetek, segédanyagok:

- Könyvfejezetek:

- <http://www.knt.vein.hu>

- ⇒ Oktatás ⇒ Tantárgyak ⇒ Digitális Rendszerek (BSc).

- (04\_chapter.pdf + további részek amik a könyvben nem szerepelnek!)

- Fóliák, óravázlatok .ppt (.pdf)

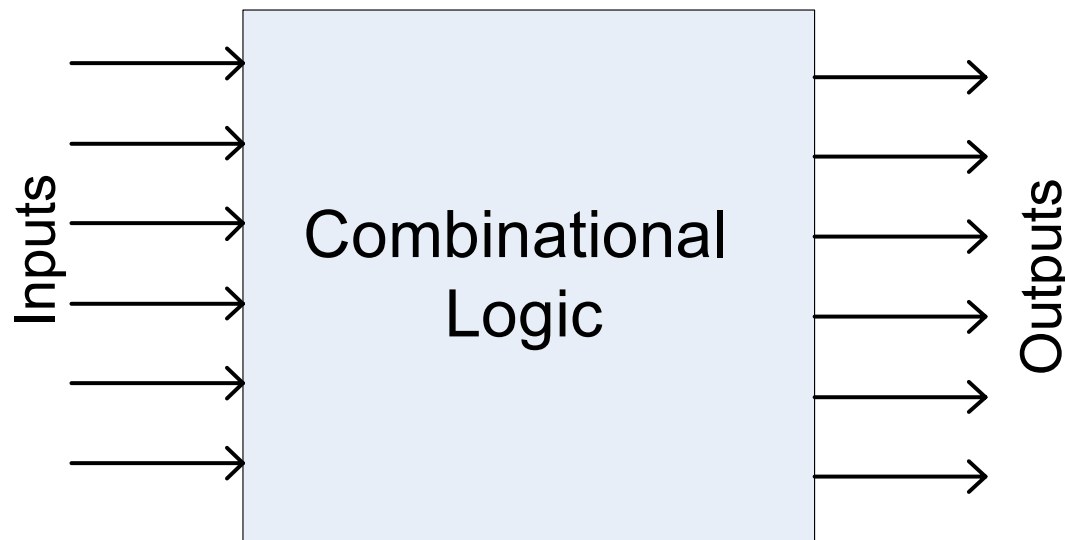
- Feltöltésük folyamatosan

# Digitális-logikai hálózatok csoportosítása:

- 1.) Kombinációs Hálózatok (K.H.)
- **2.) Sorrendi Hálózatok (S.H.)**  
[ könyv 4. és 12. fejezete ]

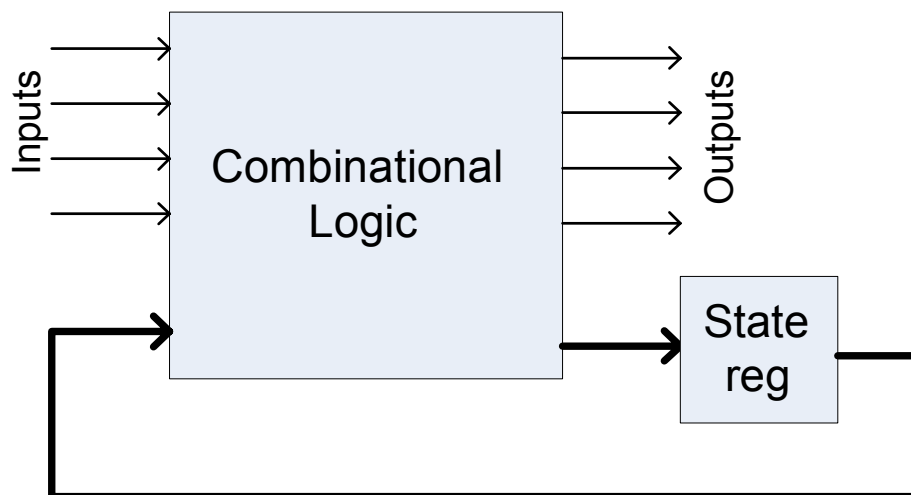
# Ism: Kombinációs hálózatok


- **(K.H.) Kombinációs logikai hálózatról** beszélünk: ha a mindenkori kimeneti kombinációk értéke csupán a bemeneti kombinációk pillanatnyi értékétől függ (tároló „kapacitás”, vagy memória nélküli hálózatok).



# Ism: Sorrendi hálózatok:

- **(S.H.) Sorrendi (szekvenciális) logikai hálózatról** beszélünk: ha a mindenkori kimeneti kombinációt, nemcsak a pillanatnyi bemeneti kombinációk, hanem a korábban fennállt bemeneti kombinációk és azok sorrendje is befolyásolja. (A *szekunder /másodlagos kombinációk* segítségével az ilyen hálózatok képessé válnak arra, hogy az ugyanolyan bemeneti kombinációkhoz más-más kimeneti kombinációt szolgáltatassanak, attól függően, hogy a bemeneti kombináció fellépésekor, milyen értékű a szekunder kombináció, pl. a State Register tartalma)





# Szekvenciális (sorrendi) hálózatok (S.H.)

# Szekvenciális hálózatok

- Hazárdjelenségek
- Visszacsatolás szerepe
- Építő elemek:
  - Szinkron tárolók és flip-flopok (szint- vagy él-vezérelt)/
  - Aszinkron tárolók (latch)
  - Regiszterek (Flip-flopok összekapcsolásából)
  - Számlálók (counters)
  - Memóriák: RAM, ROM – nagy memóriatömbök
  - Időzítő-vezérlő egységek

# Eddig:

- a kommunikáció (két kapu közötti információátvitel) sebességét végtelenül gyorsnak tekintettük (K.H).
  - (S.H.) Valóságban azonban a kapuknak véges kapukésleltetéssel (propagálási idő) rendelkeznek, amelyet figyelembe kell venni!
- A kimeneti értékek generálása csak az aktuális bemeneti állapottól függött. (K.H)
  - (S.H.) Azonban a korábbi állapotok értékét is figyelembe kell vennünk!



# Hazárd jelenségek

- Def: **Hazárdok:** Késleltetés okozta nem-kívánt kimenetek, állapotok.
- *Hazárd alakulhat* ki, ha egy kapu kimenete a bemenetek változásához képest csak véges időn belül változik (szilícium lapkán lévő elektron- és lyuk- vezetés következtében).  $T_{\text{propagation delay}}$  (Nem feltétlenül alakul ki, de lehetséges!)
- *Hazárdoknak* több fajtája lehetséges:
  - Funkcionális / Statikus
  - Dinamikus

# Hazárdok kialakulása I.

## ■ a.) Jelterjedési (propagation delay) késleltetés:

*a logikai kapu bemeneteinek és a kimeneteinek változása közötti időkülönbség miatt.*

## ■ Függ:

- ☐ Jelalak a bemeneten (waveform)
- ☐ Hőmérséklet
- ☐ Kimenet terhelése (output loading – Fan-out)
- ☐ Disszipált teljesítmény (operating power)
- ☐ Logikai eszköz típusa (type / device family)

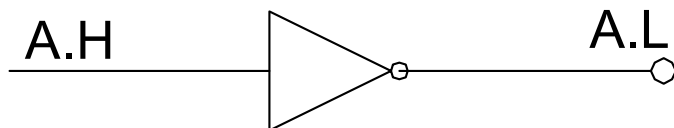
Példa: egy TTL 74LS eszközöknél, 1-gates kapu esetén a propagációs késleltetés kb. 5ns lehet.

# Hazárdok kialakulása II.

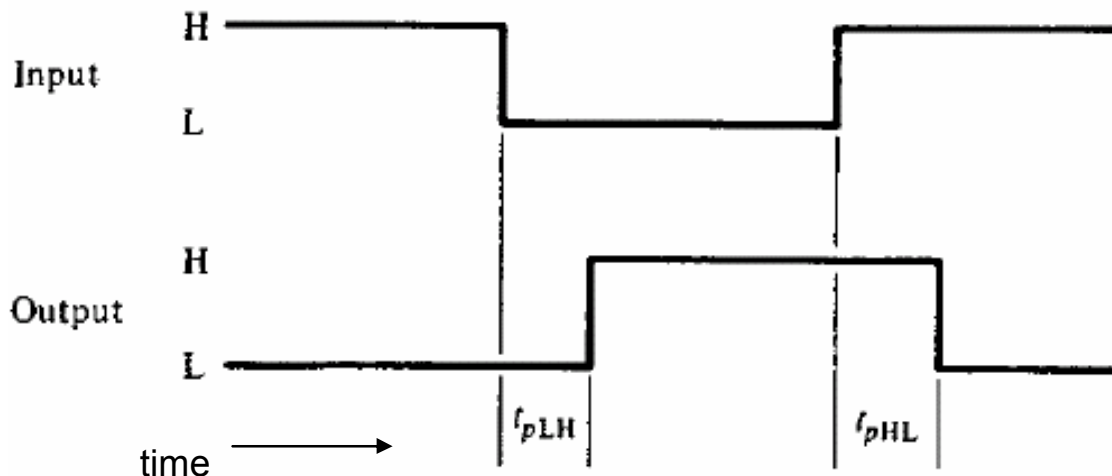
- **b.) Összeköttetési (interconnection delay) késleltetés:**  
*a logikai kapukat összekötő vezetéken lévő véges jelterjedés miatt.*
  - PI:  $\sim 20$  cm/ns sebességű jelátvitel az elektromos vezetéken
  - bizonyos vezetékhosszúság felett léphet fel

# Példa: hazard jelenségre

- Input:  $A.H \rightarrow A.L$  (fesz. polaritását változtatjuk)



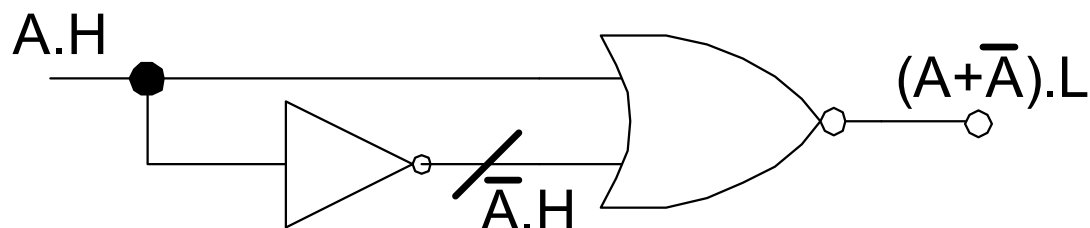
- **Idődiagram analízis:** a bemenet változását a kimenet csak véges idő alatt követi ( $t_{pLH}$  ill.  $t_{pHL}$ )



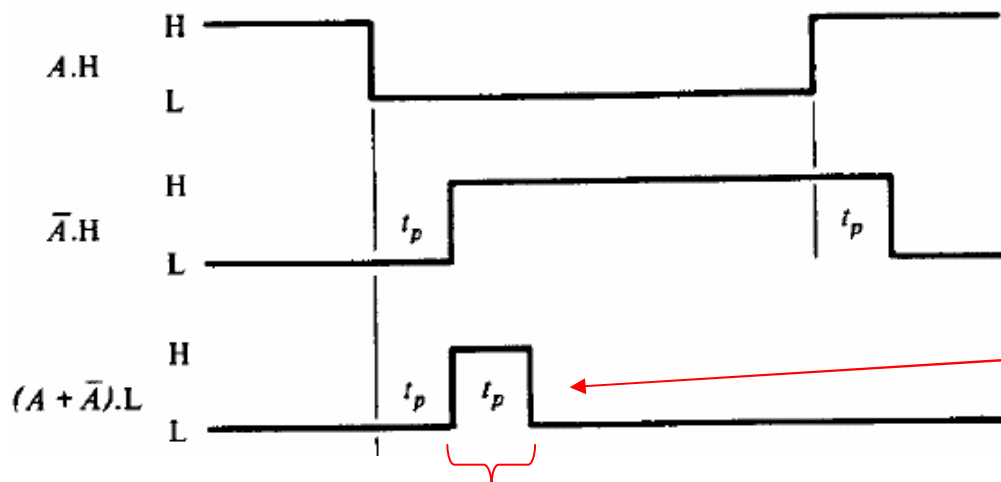
Propagációs  
(jelterjedési)  
késleltetések!  
Waveform-ok

# Példa: Késleltetés szerepe áramkörnél

- Tekintsük a  $A + \bar{A}$  -t realizáló áramkört:



- Legyen  $t_p$  a jelterjedési késleltetés.
- Ha „A” változik T→F, akkor egy nem-kívánt („spurious”) kimenet lesz, ami egységnyi kapu-késleltetésig tart („H”)



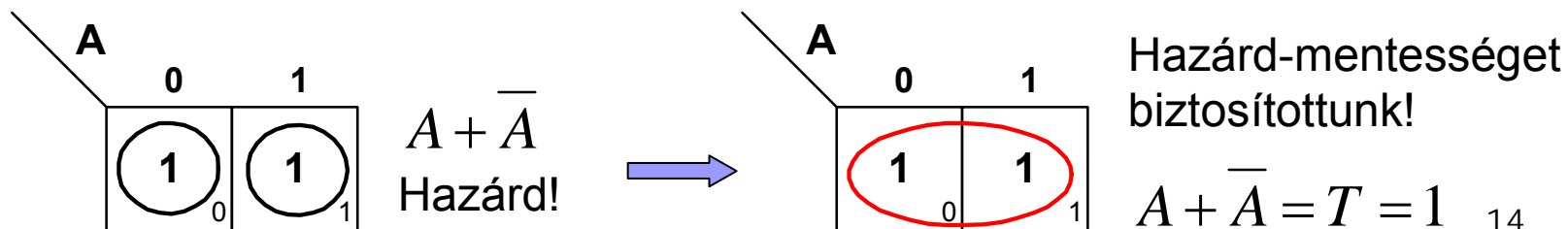
Tudjuk, hogy „A” bármely logikai értékére, a kimenet „T”. és azt is hogy „L” lesz a kimeneti fesz. értéke.

Hazárd (glitch = impulzus hiba)

## a.) Statikus / Funkcionális hazard

- Általában elegendő időt várakozva a kimenet a megfelelő (becsült) logikai- és feszültség- értékre áll be (lásd előző példa).
- De vannak olyan hazard-jelenségek is, melyek idővel nem szűnnek meg, ekkor a tervezőnek kell beavatkozni (**funkcionális hazard**).

- Pl. ha szomszédos 1-esek vannak Karnaugh táblában, amelyek nincsenek egy tömbbe összevonva, akkor hazard kialakulása lehetséges:



# Példa: Statikus hazárd

- Vegyünk egy komplexebb, 3-változós esetet:

		C			
		B			
A	BC	00	01	11	10
0	1	1	0	0	2
1	0	1	1	0	6

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

hazárdmentesítés

- Ha a szomszédos, itt kételemű tömbök között a „szaggatottal” jelölt összevonást is képezzük, akkor biztosíthatjuk a hazárdmentességet (de extra hardver szükséglet: 1 AND ill. 1 OR kapu – ezért költségesebb is.)

# Példa: hazárdmentesítésre

- Legyen  $F = \sum_{i=0}^{15} (0,1,2,5,7,10,11,13,15)$  //DNF!!
- Ekkor a következő K-tábla írható fel:

CD		C			
		00	01	11	10
AB	00	1	1	0	1
	01	0	1	1	0
	11	0	1	1	0
	10	0	0	1	1

The Karnaugh map is a 4x4 grid with rows labeled AB (00, 01, 11, 10) and columns labeled CD (00, 01, 11, 10). The cells contain 0 or 1. A blue arrow points from the map to the DNF expression. Red dashed lines and black solid lines group the 1s. Red dashed lines group (0,0), (0,1), (1,0), (1,1) and (0,1), (1,1), (1,0), (1,1). Black solid lines group (0,1), (1,1), (1,0), (1,1) and (1,1), (1,0), (1,1), (1,0).

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

$$+ A \cdot C \cdot D + \overline{B} \cdot C \cdot \overline{D} +$$

$$+ \overline{A} \cdot \overline{C} \cdot D + A \cdot \overline{B} \cdot C + \overline{A} \cdot \overline{B} \cdot \overline{D}$$

Hazárdmentesítés miatt  
kellenek (extra kapuk)

Ebből már felrajzolható a  
hazárdmentesített áramkör!



## b.) Dinamikus hazard

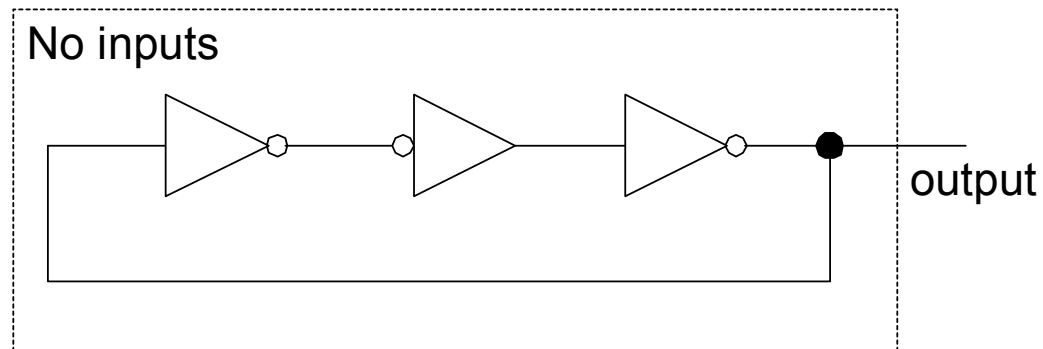
- Olyan többszintű hálózatokban jöhet létre, ahol a statikus hazard az alacsonyabb hierarchia szinteken nem lett kiküszöbölve.
- Megszüntethető: *szinkronizálással* (órajel fel-, vagy lefutó élére működtetjük a hálózatot)

# Oscilláció - visszacsatolt áramkörökben

- K.H. + **visszacsatolás/"feedback"** (iteratív, szekvenciális működés): nem csak a külső bemenetek, hanem a kimenet eredeti (korábbi) értékét is figyelembe veszi az aktuális kimenet meghatározásánál. „Szokatlan” működést biztosít.

- **Példa: Ring oszcillátor:**

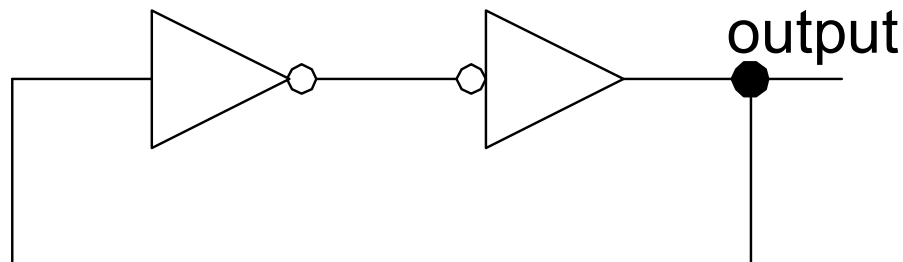
- Mindig páratlan számú invertert tartalmaz!
- Kimeneti feszültség/logikai-érték vissza van csatolva a bemenetre, amelynek értéke a kimeneten, mindig invertált formában jelenik meg! (**oszcillál** a hálózat!)



Mixed-logikai kapcsolás

# Stabilitás - visszacsatolt hálózatoknál

- Eltávolítva a ring oszcillátorból egy invertert a következőt kapjuk (páros számú elemmel):

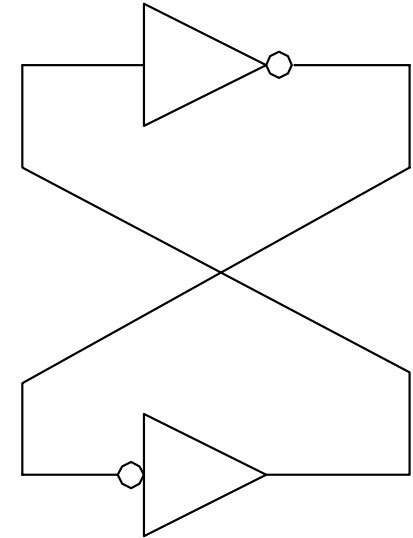


- Amíg egyik inverter alacsony, addig a másik magas feszültségszintre áll be (instabilitási periódus alatt: „*settle time*”). De miután beálltak, **stabil** értéket kapunk a kimeneten (függően attól, hogy mi volt a v.cs. output értéke!)

# Stabilitás:

## ■ Előző stabilitást biztosító áramkör átrajzolásából kapjuk:

- Mivel az áramkör nem rendelkezik külső bemenettel, ezért a viselkedésének leírására pusztán a K.H.-nál megismert Boole-algebra nem elegendő.
- „*Emlékező*” áramkör = tároló / memória (de nyilván külső bemenetek nélkül használata értelmetlen lenne)
- Tárolók / flip-flopok esetén alkalmazzuk ezt a jelölést (más logikai kapukkal is helyettesítve az invertert).



# Szekvenciális hálózatok

- Van visszacsatolás
- Korábbi állapot, és az aktuális külső bemenetek függvényében  $\Rightarrow$  határozzuk meg a kimeneteket.
- Digitális rendszert – kontrollálható memóriával szekvenciális rendszernek nevezzük.
- Építőelemei:
  - ☐ Latch (retesz)
  - ☐ Flip-flop
  - ☐ Regiszter, számláló
  - ☐ Memória

# Szekvenciális hálózatok csoportosítása – memória elemekre:

- 1.) órajel nélküli, **aszinkron** hálózatok:
  - ☐ a.) Latch (retesz)
  - ☐ b.) Aszinkron RS-tároló
- 2.) órajellel vezérelt / időzített **szinkron** hálózatok:
  - ☐ a.) Szinkron RS tároló – szintvezérelt
  - ☐ b.) MS Flip-Flop (tároló)
  - ☐ c.) Tiszta (pure) él-vezérelt FF
  - ☐ d.) JK Flip-Flop (tároló)
  - ☐ e.) D Flip-Flop (tároló)
  - ☐ f.) T Flip-Flop (tároló)

Él-vezérelt

# Fontos megjegyzések:

- **Aszinkron** esetben a „**tároló**” kifejezést használjuk (órajel nélküli)
- **Szinkron** esetben (órajellel vezérelt).
  - **Szint-vezérelt (level-triggered)** eszközök: adott logikai / feszültség szintet jelent – „**tároló**” kifejezést használjuk
  - **Él-vezérelt (edge-triggered)** eszközök: le / felfutó élre – „**flip-flop** kifejezést használjuk. (ekkor fontos, hogy csak egy meghatározott időpontban vegyünk mintát!)

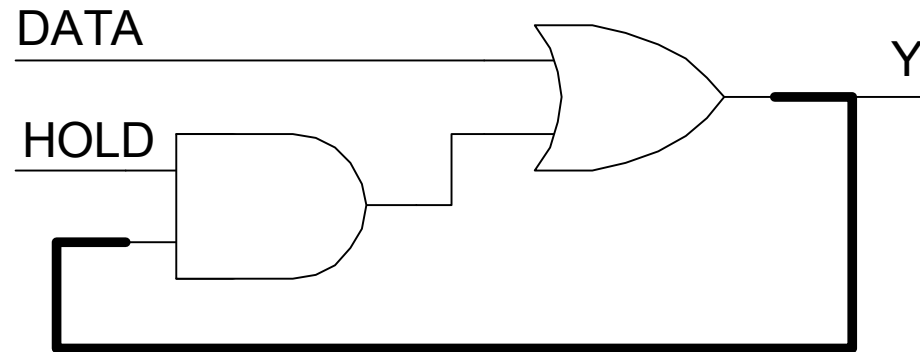


# 1. ) Órajel nélküli, aszinkron sorrendi hálózatok

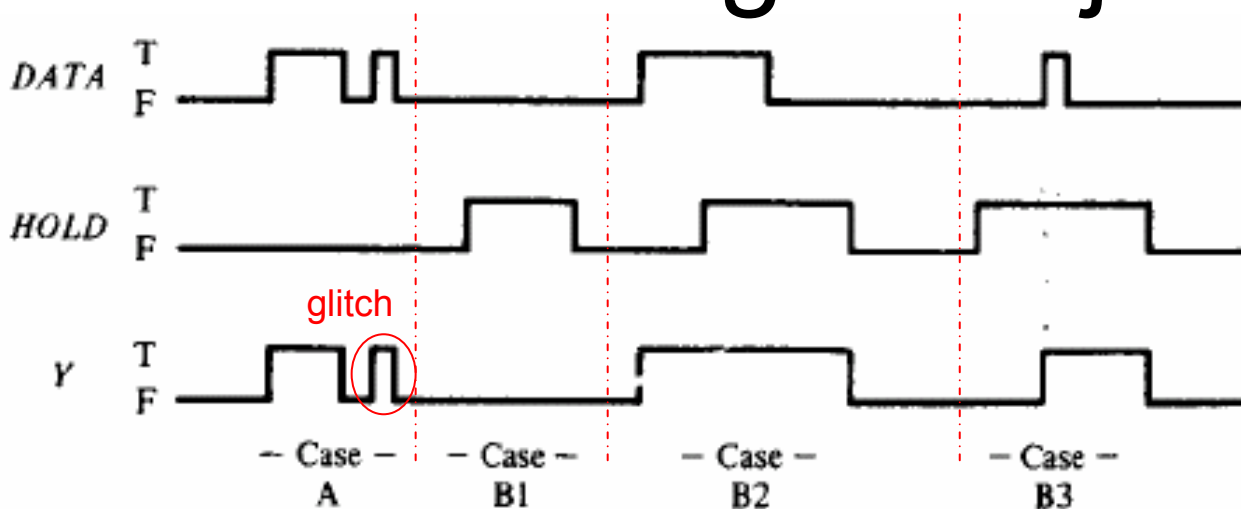


# 1/a.) Latch

- **Latch (retesz):** legegyszerűbb tároló elem. Az idő mint fontos paraméter játszik szerepet működésében.
- Tulajdonsága: a bemenetén lévő log. igaz ('T') adat azonnali kimenetre helyezése.
- Hibája, hogy amíg  $HOLD=T$ , ha egy impulzus zaj érkezik, akkor egy  $Y=T$  jelenik meg (glitch v. impulzus hiba).
- További tul. hogy a pillanatnyi 'T' érték a kimeneten  $Y=T$  megjelenik mindaddig, amíg  $HOLD=F$  nem lesz. (Ezt hívják **1's catching**). Néha hasznos, de veszélyes is lehet pl. leragadásos hiba!



# Latch időzítési diagrammja:

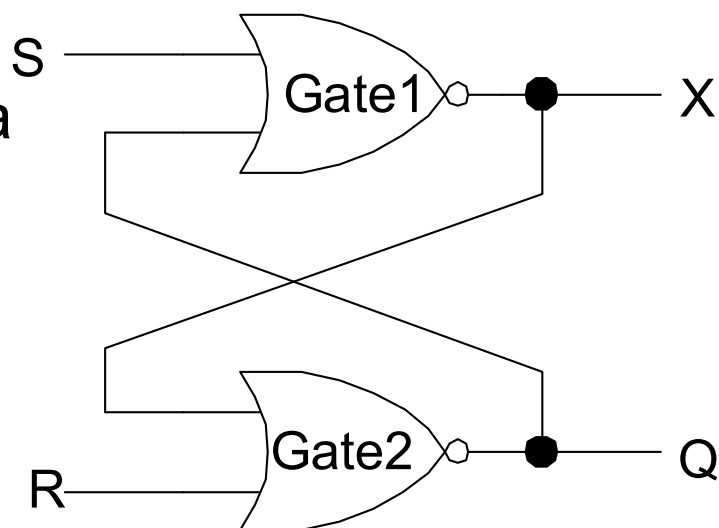


## ■ Több eset – a eset lehetséges:

- **Case A:** Ha HOLD=F, akkor Y=DATA
- **Case B:** Ha HOLD=T, akkor bármely előfordulásakor a DATA=T esetén az adatot tárolja („hold”=tartja), mindaddig amíg HOLD=F nem lesz. Ekkor a kimenetére helyezi. Három a eset lehetséges:
  - **Case B1:** DATA=F, amikor HOLD=T. Ekkor Y=F
  - **Case B2:** DATA=T és HOLD=F. Amikor HOLD=T lesz, tárolja az adatot, addig ameddig HOLD=F nem lesz újra. Ekkor a kimenetre helyezi
  - **Case B3:** DATA=F, amikor HOLD=F. Majd az adat DATA=T lesz, és ezáltal Y=T (DATA) lesz. Mindaddig kitartja Y-t, amíg HOLD=F.

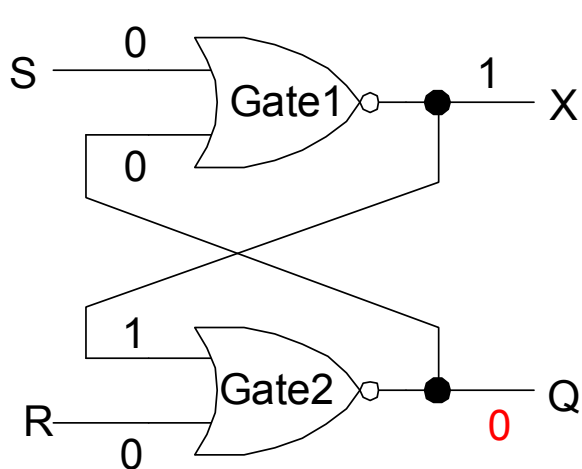
# 1/b.) Aszinkron RS-tároló (latch)

- Visszacsatolt hálózatok a memóriák egy sajátos típusát képezik: „visszaemlékezik” a feszültség, v. logikai szintek állapotára (1-bites tároló)
- Két stabil állapota lehetséges (**bistabil** eszköz), amelyeket eddig azonban külsőleg nem tudtunk befolyásolni.
- Ezért kell más logikai kapukat alkalmazni az inverterek helyett a visszacsatolásnál!  
így kapunk **aszinkron RS-tárolót**.
- Például: itt **NOR** kapukat használva
  - S: Set (beállítás)
  - R: Reset (újrabeállítás / törlés)
  - Kimenetek: Q (állapot), X (Q negáltja).
- (Megj: X-et jelölik  $\bar{Q}$ -al is!)

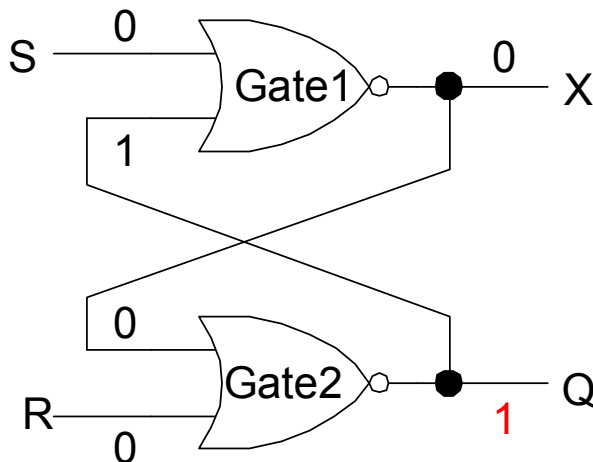


# Aszinkron RS-tároló - két stabil (**bi-stabil**) állapota:

## ■ i.) Logikai **NOR** kapcsolással



I.) RS tároló Q='0'-ás állapotban



II.) RS tároló Q='1'-es állapotban

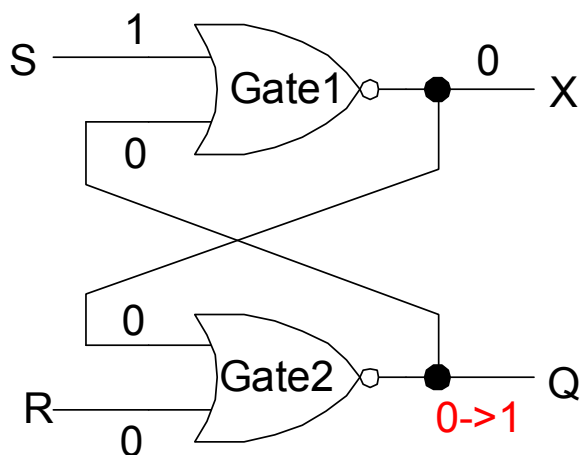
A	B	NOR
0	0	1
0	1	0
1	0	0
1	1	0

NOR igazságtáblázat

**S=R='0'='F' logikai szinten rögzítve tárolás során!**

# Aszinkron RS-tároló – „Set/Reset”:

## ■ i.) Logikai **NOR** kapcsolással (folyt)

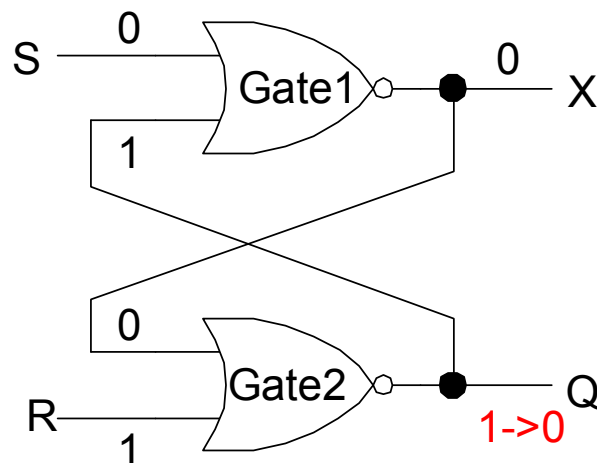


III.) Tfh: **S=1**, és Q=0.

Ekkor X=0

Ezután Q=1 lesz.

Tehát Q **0->1** (**Set**) történt!



IV.) Tfh: **R=1**, és Q=1.

Ekkor X=1

Ezután Q=0 lesz.

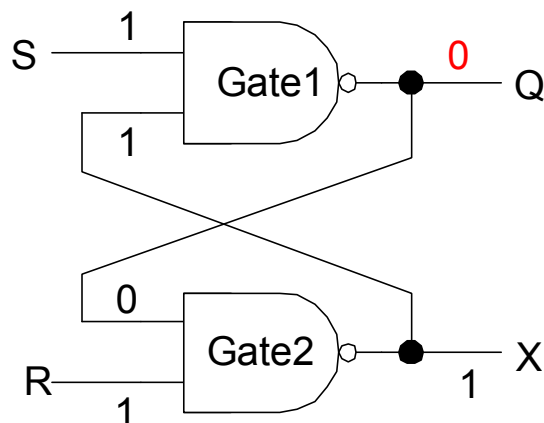
Tehát Q **1->0** (**Reset**) történt!

A	B	NOR
0	0	1
0	1	0
1	0	0
1	1	0

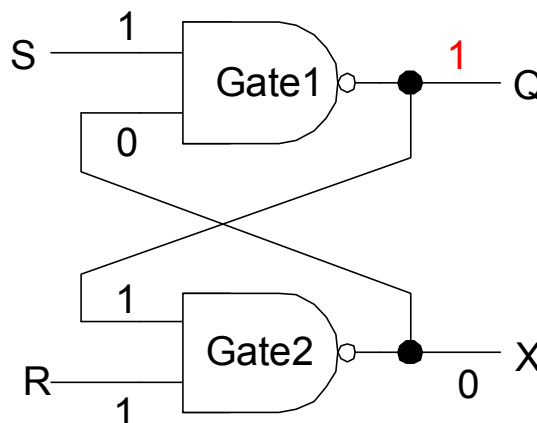
NOR  
igazságtáblázat

# Aszinkron RS-tároló - két stabil (bi-stabil) állapota:

## ■ ii.) Logikai **NAND** kapcsolással



I.) RS tároló Q='0'-ás állapotban



II.) RS tároló Q='1'-es állapotban

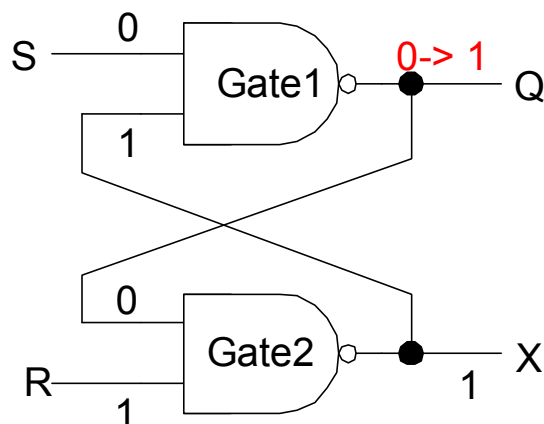
A	B	NAND
0	0	1
0	1	1
1	0	1
1	1	0

NAND igazságtáblázat

**S=R='1'='T' logikai szinten rögzítve tárolás során!**  
(Duálisa a NOR-nak!)

# Aszinkron RS-tároló – „Set/Reset”:

## ■ ii.) Logikai **NAND** kapcsolással (folyt.)

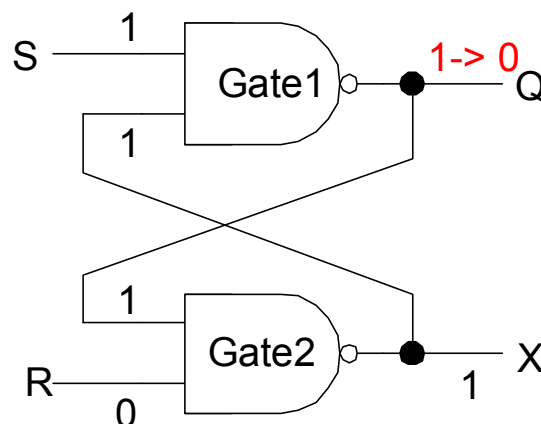


III.) Tfh: **S=0**, és Q=0.

Ekkor X=0

Ezután Q=1 lesz.

Tehát Q **0->1** (**Set**) történt!



IV.) Tfh: **R=0**, és Q=1.

Ekkor X=1

Ezután Q=0 lesz.

Tehát Q **1-> 0** (**Reset**) történt!

A	B	NAND
0	0	1
0	1	1
1	0	1
1	1	0

NAND  
igazságtáblázat

# RS tárolók Karnaugh táblái:

## ■ NOR esetben:

Ha  $S=1$ ,  
írás akkor  $q$ :  
 $0 \rightarrow 1$  lesz

		R			
		S			
q	SR	00	01	11	10
	0	0 <sub>0</sub>	0 <sub>1</sub>	- <sub>3</sub>	1 <sub>2</sub>
q	1	1 <sub>4</sub>	0 <sub>5</sub>	- <sub>7</sub>	1 <sub>6</sub>

Stabil  
állapotok  
(tárol)

Ha  $R=1$ ,  
törlés akkor  
 $q$ :  $1 \rightarrow 0$  lesz

Nem  
megengedett  
állapot  
( $R=S=1$ )

## ■ NAND esetben:

		R			
		S			
q	SR	00	01	11	10
	0	- <sub>0</sub>	1 <sub>1</sub>	0 <sub>3</sub>	0 <sub>2</sub>
q	1	- <sub>4</sub>	1 <sub>5</sub>	1 <sub>7</sub>	0 <sub>6</sub>

Ha  $S=0$ ,  
írás akkor  $q$ :  
 $0 \rightarrow 1$  lesz

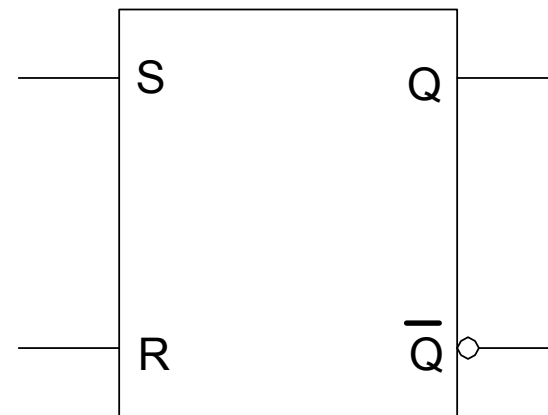
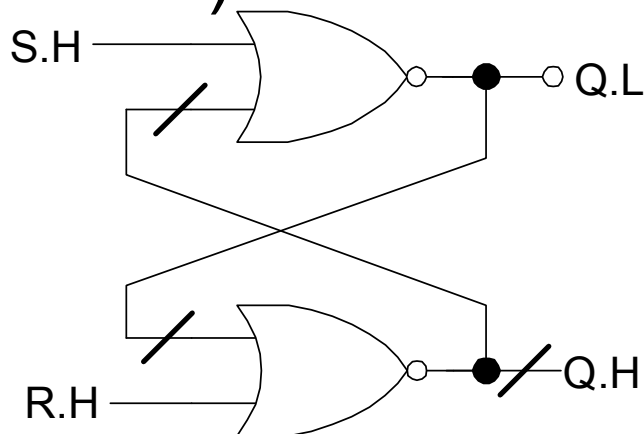
Stabil  
állapotok  
(tárol)

Ha  $R=0$ ,  
törlés akkor  
 $q$ :  $1 \rightarrow 0$  lesz



# Aszinkron RS-tároló feszültség- logikai viselkedése (**NOR** kapuval)

- Mixed-logikai kapcsolása ( 'T'='H' *pozitív* logika a bemeneteknél):



Amíg **S.H** és **R.H** „L” („F”) feszültség szinten van, a tároló aktuális állapotát tárolja, Q-tól függően (a bistabil állapot közül az egyikbe kerülünk).

Ha S-t, vagy R-t változtatjuk:

- S: control bemenet – Set State-be helyezi az áramkört ( $Q=H$ ), **S=H** ('T') esetén.
- R: control bemenet – Reset State-be helyezi az á.k.-t ( $Q=L$ ), amikor **R=H** ('T').

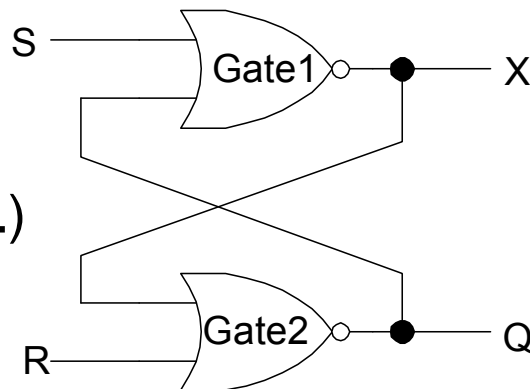
# Aszinkron RS-tároló feszültség- logikai viselkedése (**NOR** kapuval)

- Tekintsük a hálózatot pozitív logikával, ahol  $T=H$  ( $F=L$ )
- Tfh:  $R.L, S.L$  és  $Q=L$ . Ekkor  $\Rightarrow X.H$  lesz. Ezt visszacsatolva  $Q.L$  lesz megint. Tehát addig nincs változás a stabil viselkedésben, ameddig az  $R$  és  $S$  állapotokon nem változtatunk! ('L' / '0' állapotban van)
- Bi-stabilitása miatt ez igaz lesz  $Q.H \Rightarrow X.L$  -re is ('H' / '1' állapot).

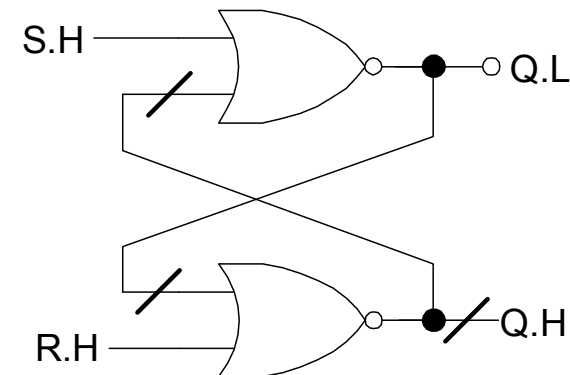
- **Set/ Reset:**

- Set state: ( $Q=L \rightarrow H$ )
- Reset state: ( $Q=H \rightarrow L$ )

**Terminológia:  $\bar{Q}$ -al jelöli!**



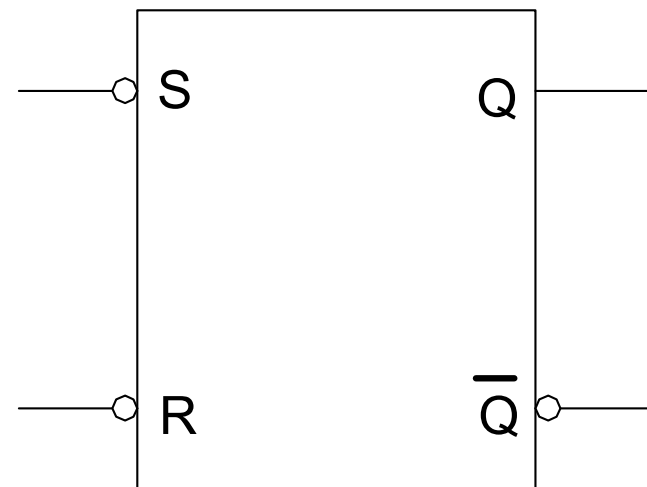
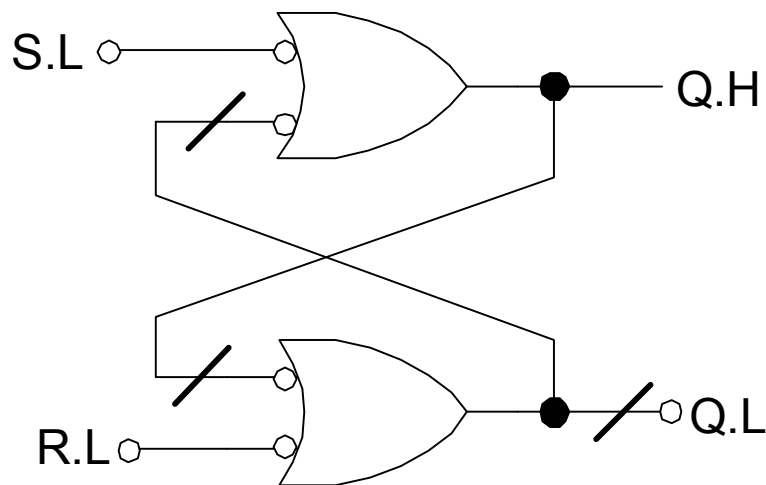
Logikai NOR kapcsolási rajza



Mixed-logikai NOR  
kapcsolási rajza

# Aszinkron RS-tároló feszültség- logikai viselkedése (**NAND** kapuval)

- Mixed-logikai kapcsolása (NOR duálisa) és szimbóluma ('T'='L' *negatív* logika a bemeneteknél):



Amíg **S.L** és **R.L** „H” feszültség szinten van, a tároló aktuális állapotát tárolja, Q-tól függően. A bistabil állapot közül az egyikbe kerülünk.

Ha S-t, vagy R-t változtatjuk:

- S: control bemenet – Set State-be helyezi az áramkört (Q=H), **S='L'** esetén.
- R: control bemenet – Reset State-be helyezi az á.k.-t (Q=L), amikor **R='L'**.

# Aszinkron RS-tároló tulajdonságai

- **Aszinkronitás:** nincs közös rendszer órajel, amely működtetné. Csupán az S és R control jelek hatására válaszol a kimenet („azonnal” – véges időn belül).
- Asynchronous (unlocked) latch
- !Hibája, hogy érzékeny impulzus zajokra: *glitch*-ek lehetnek az S / R bemeneteken, amikor a másik Reset- / Set- state állapotban vagyunk.
  - Általános tervezési eszközként ezért nem ajánlatos használni.

# Gerjesztési tábla:

„feszültség értékek” ábrázolására

- **Gerjesztési (Excitation) tábla:** sorrendi hálózatoknál a logikai- és feszültség- értékek felírására használatos, az *időbeliség* figyelembe vételével! (t: idő múlva,  $\delta$ : beállási (settle) idő)
- A **NOR** kapukból felépülő RS tároló működésének leírása gerjesztési táblázat segítségével (feszültség értékekre):

$S$	$R$	$Q_{(t)}$	$X_{(t)}$	$Q_{(t+\delta)}$	$X_{(t+\delta)}$	
L	L	$q$	$x$	$q$	$x$	Hold
L	H	$q$	$x$	L	H	Reset
H	L	$q$	$x$	H	L	Set
H	H	$q$	$x$			Disallowed

Kétértelműség! (NOR miatt)

# Gerjesztési tábla:

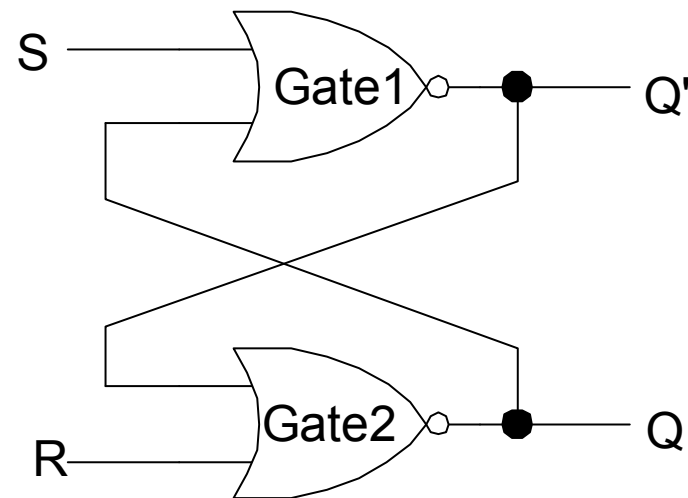
## „logikai értékek” ábrázolására

- A **NOR** kapukból felépülő RS tároló működésének leírása gerjesztési táblázat segítségével (logikai értékekre):

S	R	Q	Q'
0	0	q	$\sim q$
0	1	0	1
1	0	1	0
1	1	-	-

Hold  
Reset  
Set  
Disallow

Kétértelműség! (NOR miatt, ha  $R=S=1$ )



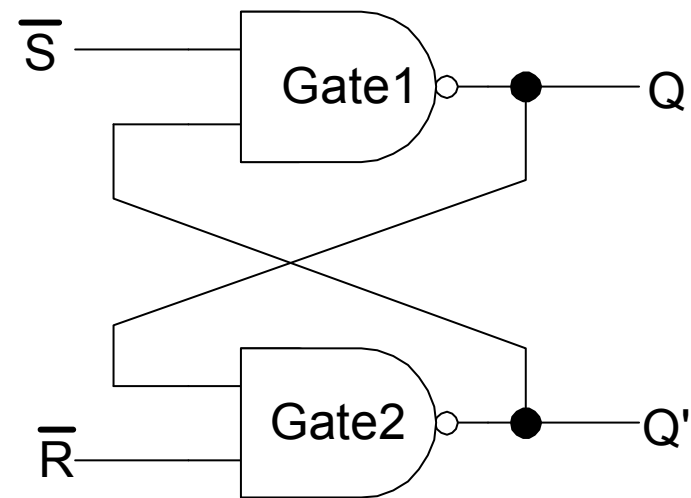
# Gerjesztési tábla:

## „logikai értékek” ábrázolására

- A **NAND** kapukból felépülő RS tároló működésének leírása gerjesztési táblázat segítségével (logikai értékekre):

$\bar{S}$	$\bar{R}$	Q	Q'	
0	0	-	-	Disallowed
0	1	1	0	Set
1	0	0	1	Reset
1	1	q	$\sim q$	Hold

Kétértelműség! (NAND miatt, ha  $R=S=0$  vagyis ha  $R=S=1$ )



NOR duálisa: kapuk, bemenetek és állapotok felcserélésével kapjuk.

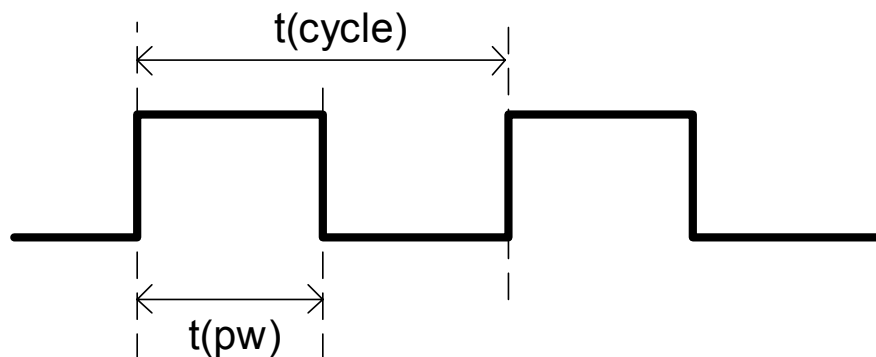


## 2. ) Órajellel vezérelt, szinkron sorrendi hálózatok



# Óra (Clock)

- A digitális áramkörökben az események történésének sorrendje kritikus (megfelelő időzítés kell  $\Rightarrow$  órajelel vezérléssel)
- **Óra**: impulzusok sorozatát bocsátja ki, pontosan meghatározott szélességgel  $[t(pw)]$ , és időintervallummal.
- **Ciklus-idő** (clock-cycle): két egymást követő pulzus élei közötti időintervallum  $[t(cycle)]$ .
  - Példa: Órajelel Frekvencia:  $f=100\,000\,000$  [Hz] ( $[1/s]$ )
  - Ekkor  $T=1/f=1 / 100\,000\,000 = 10$  [ns]
- Kristály-oszcillátor szolgáltatja ált. az órajelet.



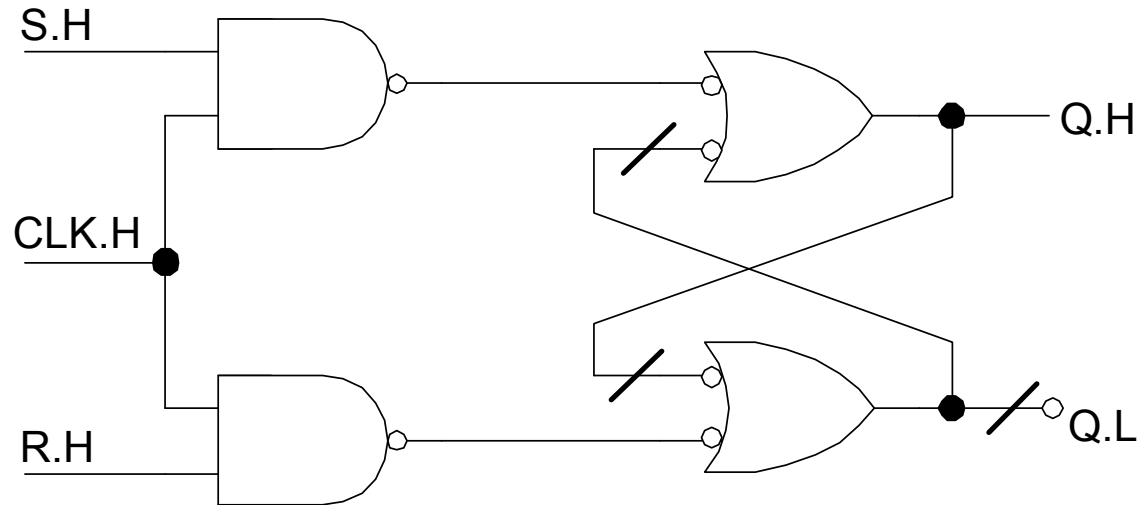
Négyszögjel.

# Órajellel vezérelt szinkron sorrendi hálózatok:

- Eml: aszinkron tárolók ('1' / 'T' catching)
- Szinkron tárolók (flip-flop-ok): általánosabb építő elemek
- **Szinkronizálás:** A kimenet mindaddig nem fog változni, amíg egy rendszer órajelre (CLK) nem engedélyeződik.
- CLK: órajel impulzus ált. négyszögjel formájában adott (timing waveform)

## 2/ a.) Szinkron RS-tároló

- Felépítése hasonló az aszinkron RS tárolóhoz, csupán az R / S állapotok aktivitását órajellel szabályozzuk.



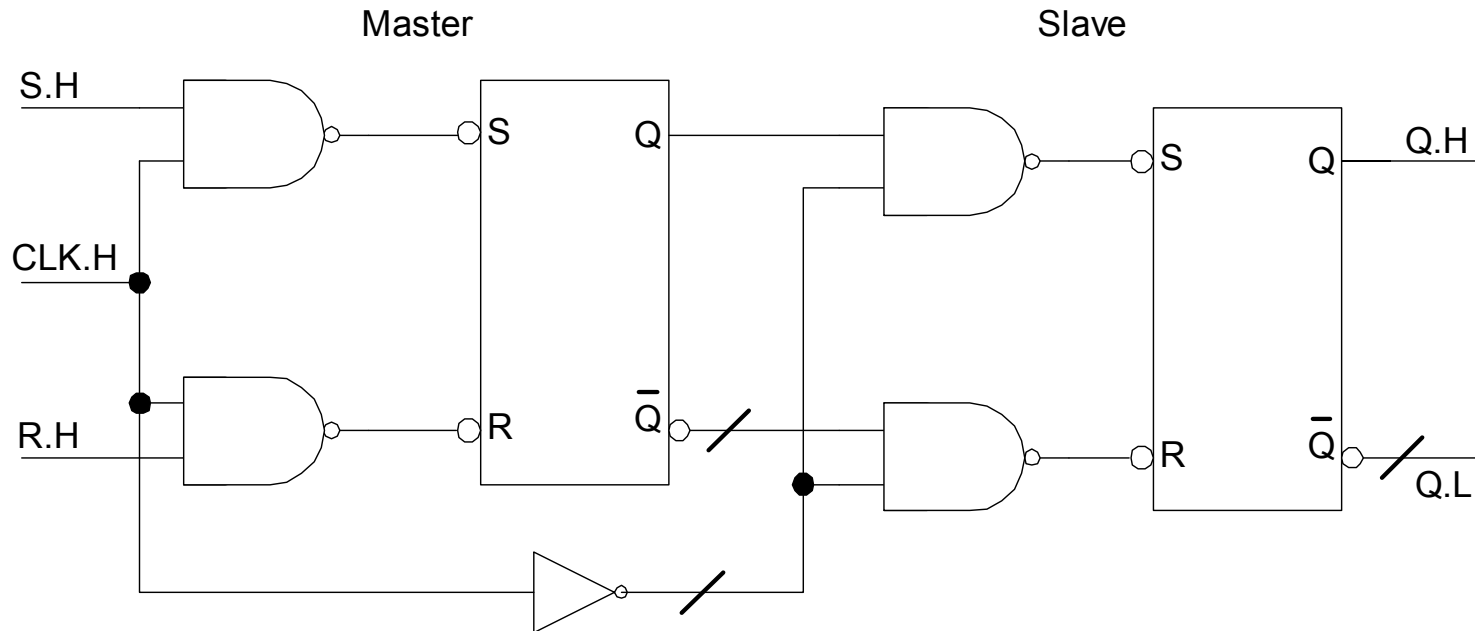
- A kimenet értéke csak az órajel 'igaz' ('T') állapota alatt változhat meg („**Level-driven**”: szint vezérelt eszköz)
- De az órajel (itt magas aktív) állapotát elelegendően szűkre kell beállítani, különben hazardok lehetségesek („shock”).

# Órajel meghatározása

- *Órajel beállítása:* a célunk, hogy elegendően kis intervallumot biztosítsunk, amelynek hatására a kimenet megváltozhat, a bemenettől függően (a szint-vezérelt működés helyett, él-vezérelt működést kell biztosítani!)
- **Él-vezérelt működés:** (edge-driven / edge-triggered): órajel feszültség átmenete
  - Pozitív (felfutó) él:  $\text{CLK.L} \rightarrow \text{H}$      $\uparrow$
  - Negatív (lefutó) él:  $\text{CLK.H} \rightarrow \text{L}$      $\downarrow$
- *Megj:* Az órajellel vezérelt szekvenciális rendszereket a továbbiakban *él-vezérelt* működésű eszközöknek tekintjük (más megnevezés: időzített flip-flopok)

## 2/ b.) Szinkron MS flip-flop

- Régóta széles körben használt eszköz, egyszerű felépítésű



- Ha a CLK magas aktív ('H'), akkor a kimenet megváltozik a bemenettől függően.
- H -> L átmenetnél (lefutó él) leválasztja az **Master** tárolót a S / R bemenetekről (ekkor tárol – változatlan tartalom)
- Feszültség invertálás miatt a **Slave** tároló ellenütemben működik: negatív (lefutó) él esetén lesz aktív (S / R bemeneteit a Master állapottól kapja)

## 2/ c.) Tiszta (pure) él-vezérelt FF

- Célunk: elkerüljük a hazárdokat (glitch, noise-zaj stb.)
- MS-FF helyett alkalmazzuk: *1's catching* tulajdonságot próbáljuk elkerülni használatával (clk pozitív részén)
- Az órajelciklus negatív részén viszont az R / S control bemeneteket kell stabil állapotba hozni
- Tiszta él-vezérlés: állapot-átmeneteket is használni kell
  - Aktív él:  $F \rightarrow T$  amelyre az átmenet megtörténik (lehet  $H \rightarrow L$  vagy  $L \rightarrow H$ )
  - Aktív élre a bemeneteket (R / S) kell figyelni (sensing)
  - Aktív él eredményeként (adott log / fesz. szinten) szabad csak megváltoznia az állapotnak (Q)

# Gerjesztési tábla: logikai és feszültség értékek ábrázolására

- Él-vezérelt Flip-flopok esetén használatos táblázat
- Jelölések:
  - Minden aktív élre vagy *Set*, vagy *Reset* állapotba jut (bistabil) a bemeneti értékeknek és az aktuális tárolt állapotnak megfelelően.
  - **Q(n)**: n. órajel **trigger** („**aktív él-váltás**”) állapota
  - Ha ekkor a tároló *Set* állapotban van, akkor  $Q(n) = 'T'$
  - Ha ekkor a tároló *Reset* állapotban van, akkor  $Q(n) = 'F'$
  - **Q(n+1)**: n. utáni (következő) aktív élre (táblázat készítése összes lehetséges kombinációjára)
  - **Setup time**: megbizonyosodni az R / S control inputok stabil voltáról, az aktív él előtt röviddel.
  - **Hold time**: R / S control inputok stabilitása az aktív él után röviddel.

## 2/ d.) Szinkron JK flip-flop

- Ism: Az RS tárolónál *kétértelműség* volt az R / S bemenetek azonossága esetén (ott nem-megengedett volt!)
- Azonban **JK** esetén az összes R / S bemeneti kombinációra egyértelmű **kimeneti eredményt kapunk.**
- **Gerjesztési** tábla a JK tároló *logikai* vizsgálatához:
  - Megfeleltetés: **J:=Set / K:=Reset** - mint control bemenetek

Clock	J	K	$Q_{(n)}$	$Q_{(n+1)}$	
F	X	X	$q$	$q$	} Szint-vezérelt viselkedés: Stabil 'T' v. 'F' esetén a JK kimenete érzéketlen
T	X	X	$q$	$q$	
↑	F	F	$q$	$q$	Hold
↑	F	T	$q$	F	Reset
↑	T	F	$q$	T	Set
↑	T	T	$q$	$\bar{q}$	Toggle (complement)
					} negált

↑ : CLK felfutó élére (pozitív) vezérelt (F → T)



# JK flip-flop Karnaugh táblája

- Nagyon hasonló az RS-tárolóhoz, de itt az  $J=K=1$  állapot is megengedett (toggle)!

The Karnaugh map for the JK flip-flop is shown below. The vertical axis represents the current state  $q$  (0, 1) and the horizontal axis represents the inputs  $J$  and  $K$  (00, 01, 11, 10). The cells are numbered 0 through 7. Annotations explain the behavior of different input combinations:



- Stabil állapotok (tárol)**: Cells 0, 1, 4, and 6 are circled in blue, indicating stable states where the output  $q$  remains the same as the input  $q$ .
- Ha  $J=1$ , írás akkor  $q: 0 \rightarrow 1$  lesz (Set)**: Cell 2 (J=1, K=0) is highlighted in red, indicating a set operation where  $q$  transitions from 0 to 1.
- Ha  $K=1$ , törlés akkor  $q: 1 \rightarrow 0$  lesz (Reset)**: Cell 5 (J=0, K=1) is highlighted in red, indicating a reset operation where  $q$  transitions from 1 to 0.
- Megengedett állapot „toggle” mód ( $J=K=1$ )**: Cell 7 (J=1, K=1) is highlighted in red, indicating the toggle operation where  $q$  transitions from 0 to 1 or 1 to 0.

JK		K			
		J			
q		00	01	11	10
0	0	0	0	1	1
1	1	1	0	0	1

# JK tárolók típusai:

- Kereskedelmi forgalomban több típussal, jelöléssel is találkozhatunk:

- Aktív-éle az órajelnek lehet:

- Pozitív (felfutó) él-vezérelt:   $(F \rightarrow T)$   $\uparrow$
- Negatív (lefutó) él-vezérelt:   $(T \rightarrow F)$   $\downarrow$

- J / K control jelek aktív feszültség-szintjei:

- J / K is magas aktív (T=H): pozitív logika
- J aktív magas ('H') / K aktív alacsony ('L')

- Aszinkron R / S módú viselkedés (availability):

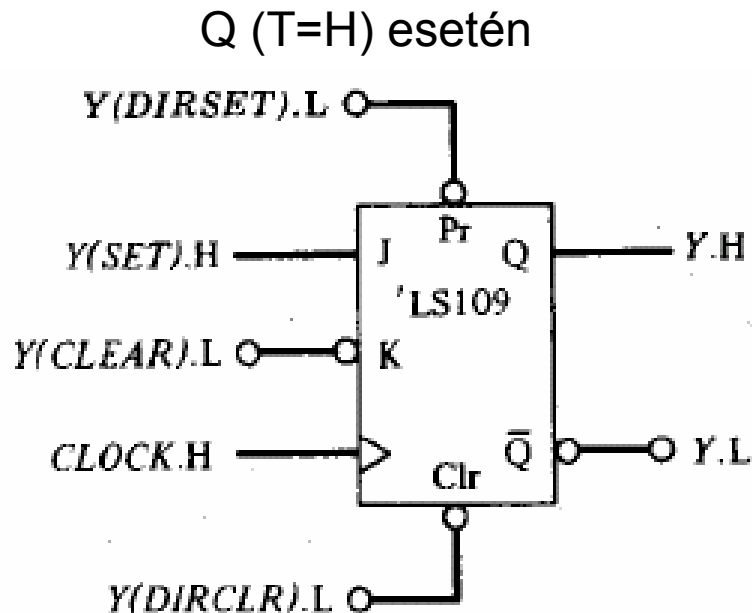
- Direct-clear (Pre-clear): aszinkron Reset (szimbólum alján)
- Direct-set (Pre-set): aszinkron Set (szimbólum tetején)

- Szinkron R / S módú viselkedés (alap)

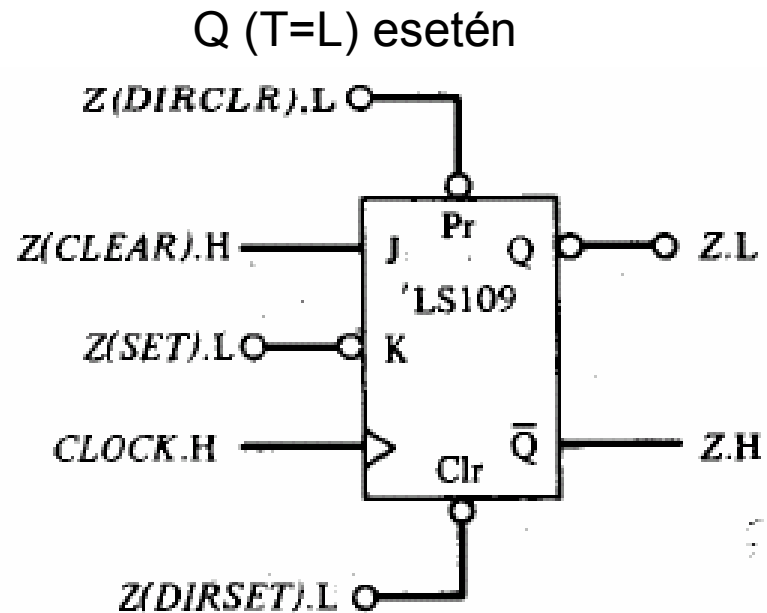
- Clear: lehet J, vagy K – control (logikai clear:  $Q=F$ )
- Set: lehet J, vagy K – control (logikai set:  $Q=T$ )

# Példa: 74LS109 Dual JK Flip-flop

- Pozitív (felfutó) él-vezérelt SSI tároló elem
- Control jelei: J magas aktív (H) / K alacsony aktív (L)
- Set: Q-t 'T' be állítja (logikai set)
- Clear: Q-t 'F' be állítja, törli (logikai clear)



(a) Setting with *J*,  
clearing with *K*



(b) Setting with *K*,  
clearing with *J*

# Példa: 74LS109 Gerjesztési tábla – feszültség értékek esetén

Q: T=H eset!

Preset	Preclear	Clock	J	K	$Q_{(n+1)}$	Action if Q is active-high
L	H	X	X	X	H	Direct set
H	L	X	X	X	L	Direct clear
L	L	X	X	X	—	Disallowed configuration
H	H	↑	L	L	L	Clear (Reset)
H	H	↑	L	H	$Q_{(n)}$	Hold
H	H	↑	H	L	$\sim Q_{(n)}$	Toggle
H	H	↑	H	H	H	Set

- $\sim Q_{(n)} = Q_{(n)}$  negáltja (komplementese: „toggel mód”-ban)
- X: don't care állapot
- —: nem megengedett állapot

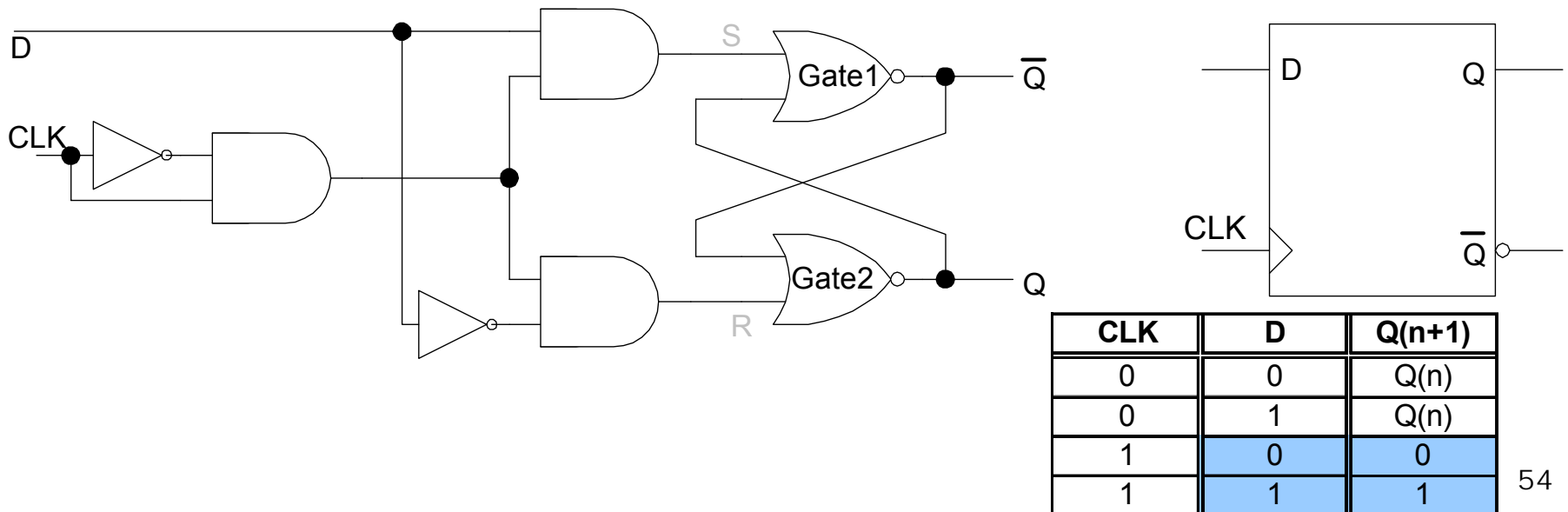
Aszinkron  
viselkedés

# JK Flip-flopok általános tulajdonságai

- Általánosan használt, jól controllálható működésű tároló (flexibilitás)
- Adattárolásra (hold): gerjesztési tábla alapján
  - $Q(n+1) = Q(n) = q$  lesz bármikor, ha  $J=K=F$  (felfutó élre) – ez egy egyszerű adattárolási mód
- Adatbevitelre: J / K nem adat, hanem vezérlő vonalak! Háromféleképpen használható:
  - a.) Clear -> majd Set
  - b.) Set -> majd Clear
  - c.) Tárolás: egy órajel ciklus ideig (részletesen a könyvben)

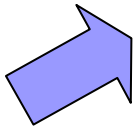
## 2/ e.) Szinkron D-flip-flop

- Rendkívül egyszerű működésű, általános tároló elem.
- **D** (Delay) tároló: késleltetési-alapú működés (egy órajel ciklus ideig tartja az értéket, amelyet a következő ciklusban a kimenetére helyez)
- **CLK=0→1** értékére működik (1→0-nál tárolt érték  $q(n)$ )
- Kapcsolása, szimbóluma, és logikai gerjesztési táblázata:



# D-flip-flop Karnogh táblája

- JK tárolóból származtatható, ahol csak a **különböző** bemeneti értékű JK kombinációk a megengedettek  $\rightarrow$  D



		JK			
		K		J	
q		00	01	11	10
	0	0 <sub>0</sub>	0 <sub>1</sub>	1 <sub>3</sub>	1 <sub>2</sub>
q	1	1 <sub>4</sub>	0 <sub>5</sub>	0 <sub>7</sub>	1 <sub>6</sub>

		D	
		0	1
q	0	0 <sub>0</sub>	1 <sub>1</sub>
	1	0 <sub>2</sub>	1 <sub>3</sub>

Ha D=0,  
törlés lesz  
q: 1- $\rightarrow$ 0 lesz  
(tárolás)

Stabil  
állapotok  
(tárol)

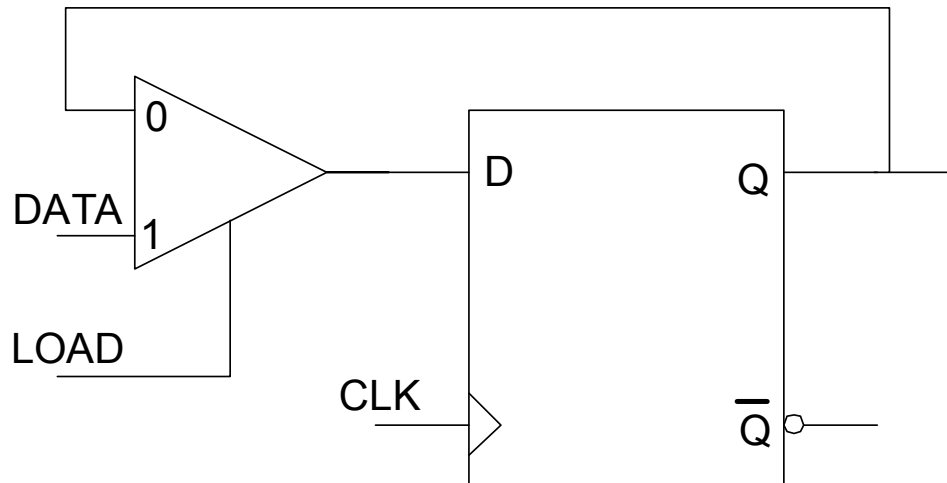
# D-tároló jelei

- Kereskedelmi forgalomban a következő variációkban, jelölésekkel kapható
  - i.) CLK órajel aktív éle:
    - Pozitív (felfutó):  $L \rightarrow H$
    - Negatív (lefutó):  $H \rightarrow L$  (kis kör jelöli)
  - ii.) Direct (aszinkron) Set / Clear (ha jelölik akkor általában alacsony aktív állapotúak – 1's catcher)
  - iii.) Legtöbb esetben csak a Q kimenete van feltüntetve, vagy a negált Q-val biztosítja mindkét polaritást



# D-tároló alkalmazása, mint:

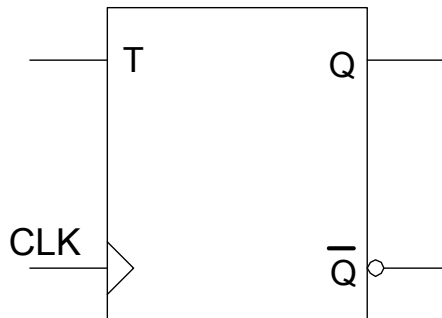
- **Késleltető elem:** egy órajel ciklusig késlelteti a bemenetre adott értéket, mire az a kimenetre kerül
- **Szinkronizáló elem:** különböző jelek rendszer órajelhez való időbeli ütemezése (szinkronizálatlan jel a bemenetén).
- **Adat tároló elem:** adatbevétel és tárolás
- **Engedélyező elem:** LOAD engedélyező bemenet (MUX-on keresztül választódik ki a bemenet, vagy v.cs. értéke



- LOAD = 0, visszacsatolt Q állapot kering
- LOAD = 1, külső bemenet (DATA)

## 2/ f.) Szinkron T-flip-flop

- Rendkívül egyszerű működésű
- **T** (toggle) tároló: késleltetési-alapú működés (egy órajel ciklus ideig tartja az értéket, a következő ciklusban viszont az érték **negáltját** teszi a kimenetére)
- CLK=0->1 értékére működik (tárol)
- szimbóluma, és logikai gerjesztési táblázata:



T	Q(n+1)
0	1
1	0

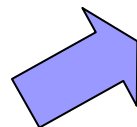
# T-flip-flop Karnogh táblája

- JK tárolóból származtatható, ahol csak az **azonos** bemeneti értékű JK kombinációk a megengedettek  $\rightarrow$  T
- A tárolót a JK bemenetek összekötéséből kapjuk!

JK K

q	JK			
	00	01	11	10
0	0 <sub>0</sub>	0 <sub>1</sub>	1 <sub>3</sub>	1 <sub>2</sub>
1	1 <sub>4</sub>	0 <sub>5</sub>	0 <sub>7</sub>	1 <sub>6</sub>

Red lines indicate the mapping from JK to T: (00, 01) to (0, 0), (11, 10) to (1, 1), and (01, 10) to (0, 1).



T

q	T	
	0	1
0	0 <sub>0</sub>	1 <sub>1</sub>
1	1 <sub>2</sub>	0 <sub>3</sub>

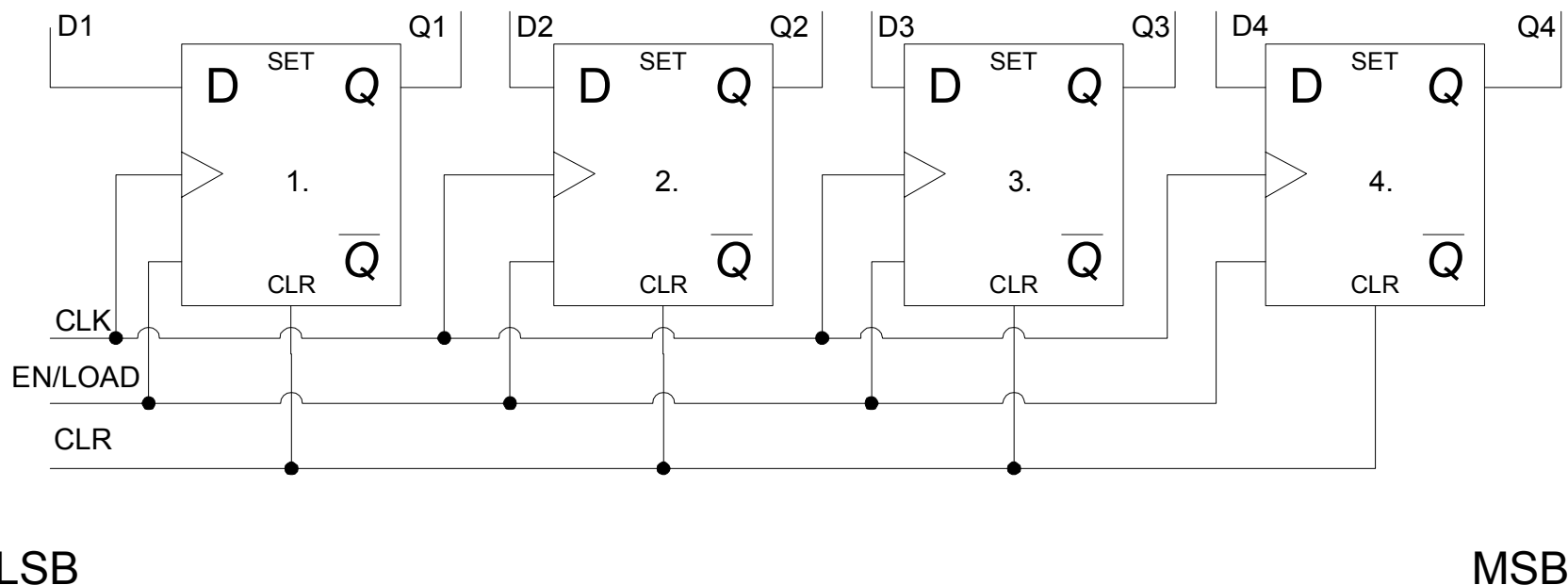
Stabil  
állapotok  
(tárol)

Ha  $T=1$ ,  
negálás  
lesz ( $\sim q$ )  
(toggle)

# Regiszterek

- **Regiszter:** n db tároló (Flip-flop) elemekből felépülő rendezett tömb
- Ideiglenes adattárolásra használjuk (néhány bites érték tárolása aritmetikai műveletekhez)
- Byte, vagy word (szóhosszúság) szervezésű
- MSI-szintű építőelem
- Példák:
  - Engedélyező bemenetű D-regiszter (EN)
  - Tiszta (pure) D-regiszter (EN=1 nek feltételezzük)
    - 4, 6, 8, 16, 32 ...számú D Flip-flop-ból épülhetnek fel.
    - Közös órajel (esetleg törlő és engedélyező jel)
  - Shift-regiszter (kimenetek sorba kötésével – léptetés)

# 4-bites Parallel In/ Parallel Out regiszter (D-tárolókból felépítve)



LOAD: a D1...D4 bemeneteknek párhuzamos beírására is lehetőség van. (LOAD=1)

# Számlálók (counters)

## ■ Bináris számlálók

- Moduló-N számláló: M moduló N ( $M / N$  utáni maradék) értékét tárolja
- 3-bites számláló:  $10^3$  különböző értékre 000-999 –ig, majd 999 után újból 000-tól indul, inicializálódik. (Ez egy „M moduló 1000” számláló.)

## ■ Számláló JK tárolókból: „toggle mode”-ban használjuk a tárolót

- Modulo-2 számláló: (q negált).

■ CLK impulzusa: 0 1 2 3 4 5 6 7 8 ...

■ FF Q kimenete: 0 1 0 1 0 1 0 1 0 ... (  $q / \overline{q}$  ) //alternáló jelleg

- Szinkron Modulo-4 számláló: (közös CLK)

■ CLK impulzusa: 0 1 2 3 4 5 6 7 8 ...

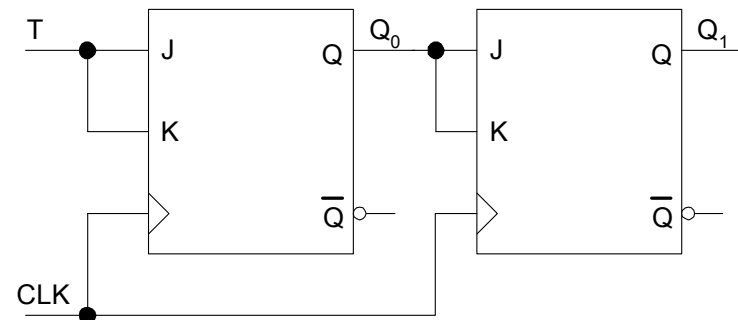
■ FF  $Q_1, Q_0$  kimenete: 00 01 10 11 00 01 10 11 00 ...

Logikai  
értékeket  
reprezentál

LSB bit:  $Q_0$  balra (alternáló jelleg)

MSB bit:  $Q_1$  jobbra (akkor változtatja az értékét alternáló jelleggel, amikor  $Q_0 = 'T' / '1'$ )

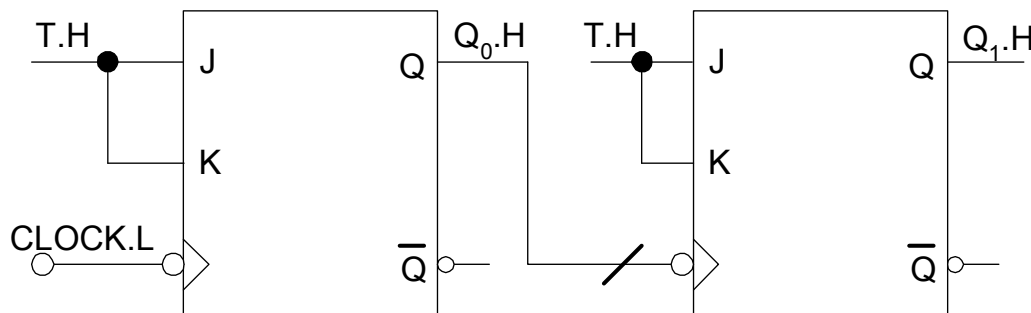
Ezekből nagyobb méretű modulo-N számláló is felépíthető sorbakötésükkel!



# Aszinkron bináris moduló-4 számláló JK tárolóból

## ■ Hasonlóan működik az előzőhöz:

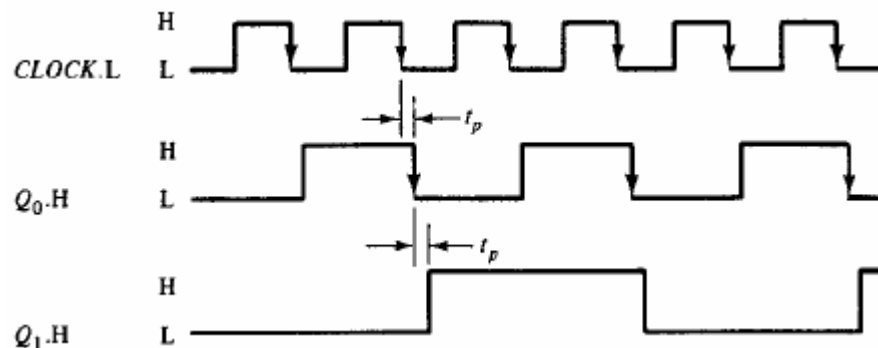
- $Q_0$  alternál („toggle mode”) minden CLK-ra (mivel  $J=K='T'$ ) //bal FF
- $Q_0$  generál egy  $T \rightarrow F$  ( $H \rightarrow L$  ebben az esetben) átmenetet
- $Q_1$  is alternál, de  $Q_0$  tól függően ( $J=K='T'$  re) //jobboldali FF
- $Q_1$  –es FF órajelét a  $Q_0$  biztosítja (aszinkron működés)



LSB bit:  $Q_0$  balra (alternáló jelleg)

MSB bit:  $Q_1$  jobbra (akkor változtatja az értékét alternáló jelleggel, amikor  $Q_0 = 'T' / '1'$ )

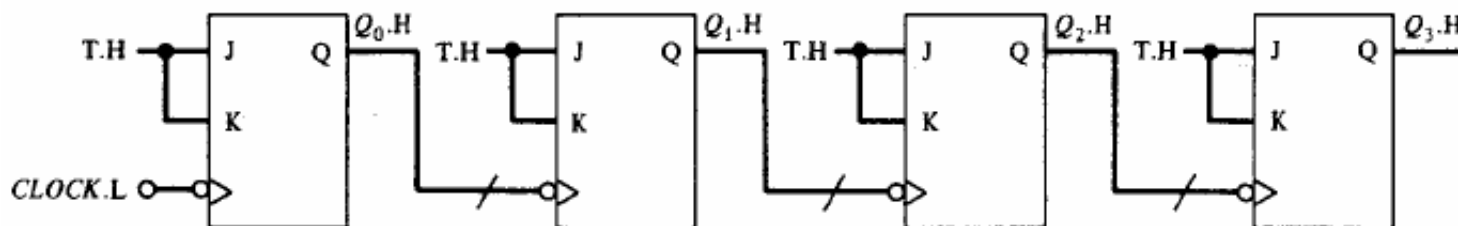
## □ Idődiagram:



$t_p$ : propagációs késleltetés [ns]

# 4-bites moduló-16 számláló (counter)

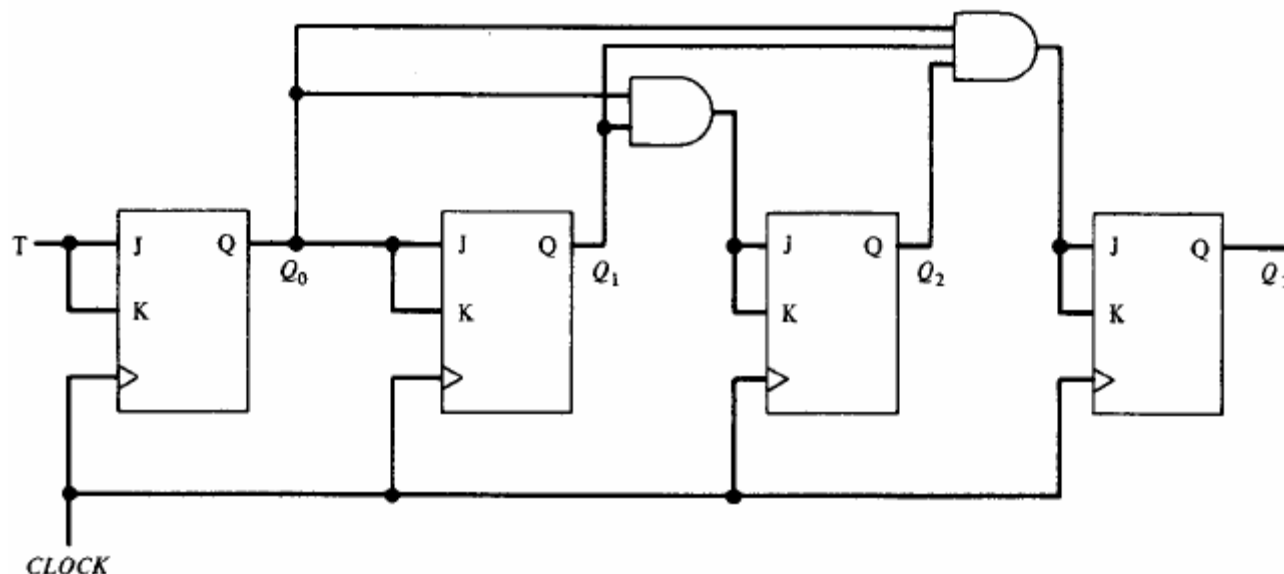
## ■ Aszinkron (ripple) counter:



Pl. átmenet 3 -> 4 között

Time	$Q_2$	$Q_1$	$Q_0$
0	0	1	1
$t_p$	0	1	0
$2t_p$	0	0	0
$3t_p$	1	0	0

## ■ Szinkron counter:



Extra AND kapuk:

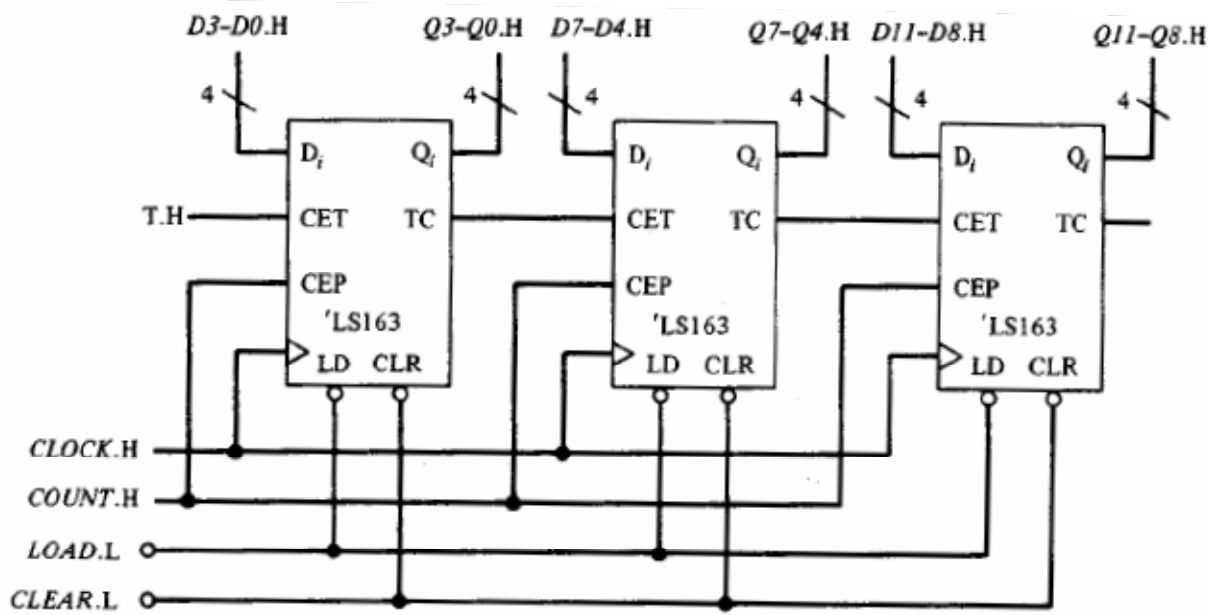
„toggle-mode”  
szinkronizációja  
az egyes FF-nál

Közös CLK!



# Szinkron MSI számlálók:

- Kereskedelmi forgalomban is kaphatóak
  - Moduló-10 (dekád) számláló
  - Moduló-16 (4-bites bináris) számláló
  - **Példa: 12-bites bináris számláló** 3 db 'LS163 – 4-bites szinkron bináris számláló összekapcsolásából.



Közös CLK: szinkron működés

CLOCK.H: rendszer órajel

Szinkron Clear (0) / Load (Set)

TC: terminal count

CET: Count Enable Trickle

CEP: Count Enable Parallel (master)

} Eszköz vezérlők

# Példa: számláló tervezése D-FF-ből

- Tervezzünk bináris számlálót szinkron D-tárolóból, amely 0...5-ig tárolja az értékeket. (6 érték → összesen 3 db D-tárolóra lesz szükségünk).
- Igazságtáblázata:

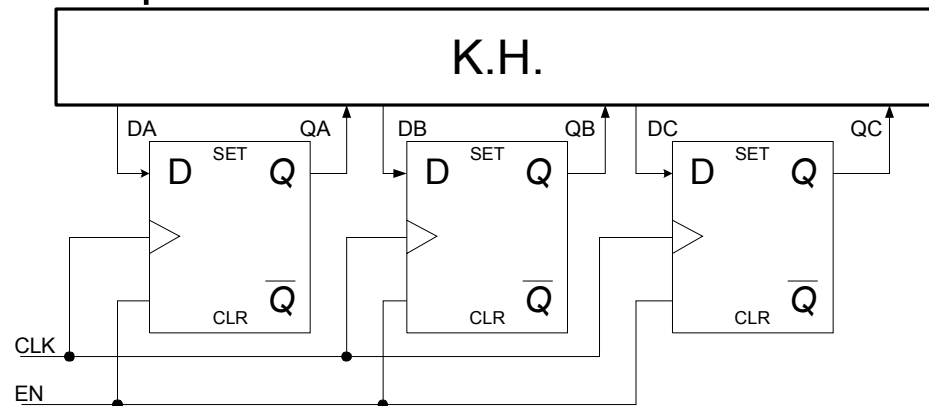
count	QA(i)	QB(i)	QC(i)	DA(i-1)	DB(i-1)	DC(i-1)	i. clk
0	0	0	0	0	0	1	1
1	0	0	1	0	1	0	2
2	0	1	0	0	1	1	3
3	0	1	1	1	0	0	4
4	1	0	0	1	0	1	5
5	1	0	1	0	0	0	6
0	-	-	-	-	-	-	-
1	-	-	-	-	-	-	-

egy órajellel később jelennek meg Dx értékei a Qx kimeneteken!

Realizáljuk pl. a DC-t (A,B,C kimenetű K.H-ból) → DC:

(Hasonlóan lehet képezni a Karnough táblákat a DA, DB-kre is!)

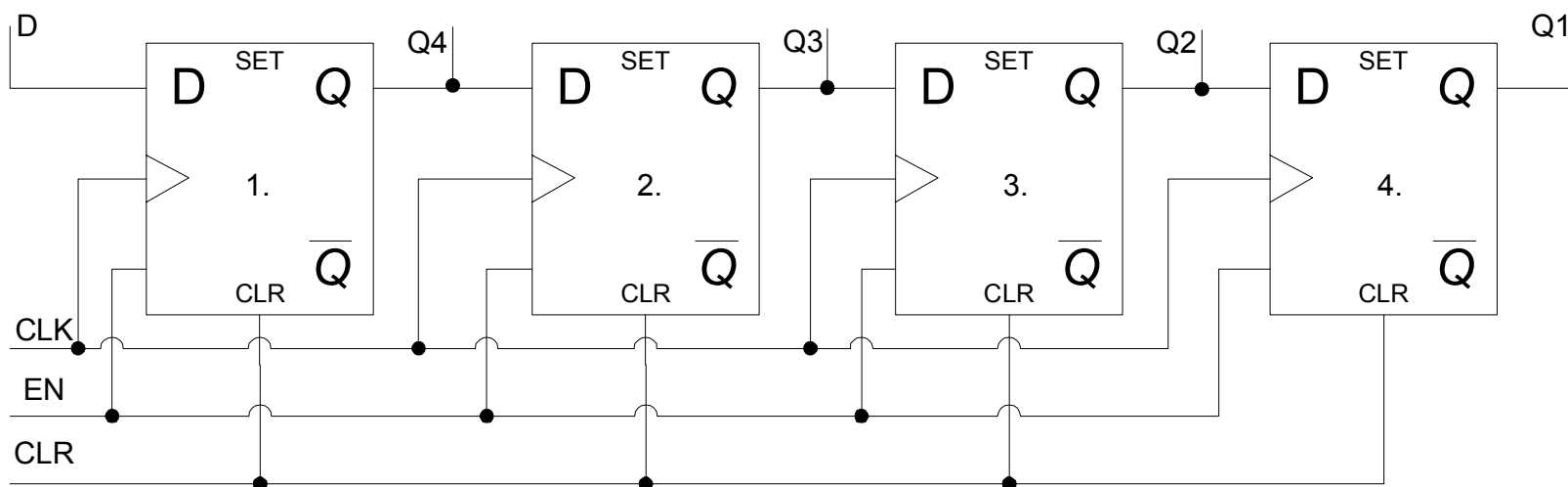
Kapcsolása:



		C			
		B			
A	BC	00	01	11	10
	0	1 <sub>0</sub>	0 <sub>1</sub>	0 <sub>3</sub>	1 <sub>2</sub>
1	1	1 <sub>4</sub>	0 <sub>5</sub>	-/0 <sub>7</sub>	-/1 <sub>6</sub>

$$DC = \overline{C}$$

# 4-bites Shift (léptető) regiszter (Serial in/Parallel Out – D-tárolós)



**Shift regiszter.** Oldalirányú (laterális) léptetés, az egyik bitpozíciótól a szomszédosig.

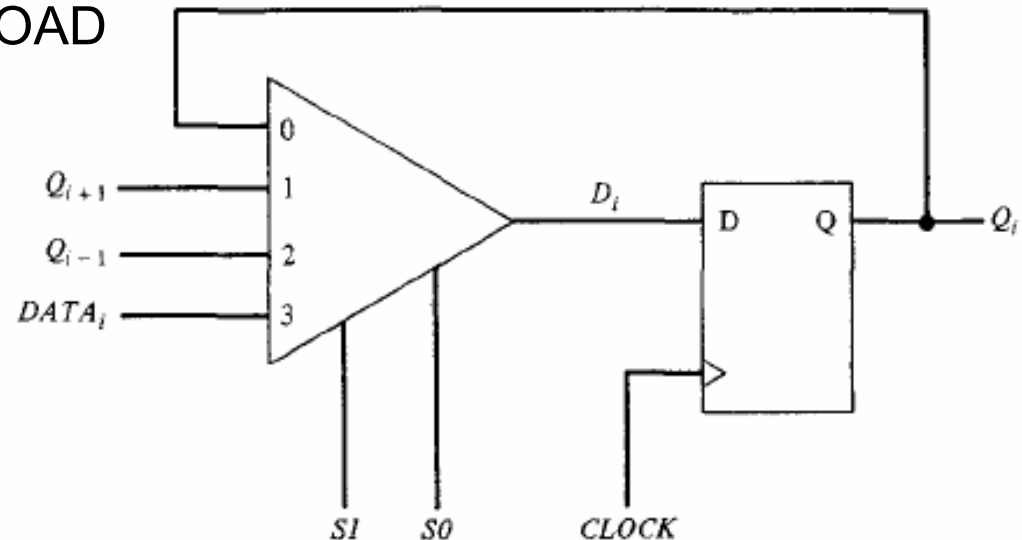
**D:** Data bemeneten lévő adatot lépteti sorosan balról-jobbra minden egyes órajel ciklusban. 1.clk-ban a 1. tárolóba a D1-et, majd 2 clk-ban 2. tárolóba a D1-et és az 1. be a D2-t.

**Q:** a kimeneteken párhuzamosan kapjuk az adatot

CLK	Q4	Q3	Q2	Q1
1	D1	-	-	-
2	D2	D1	-	-
3	D3	D2	D1	-
4	D4	D3	D2	D1
5	D5	D4	D3	D2
6	D6	D5	D4	D3
7	D7	D6	D5	D4

# 4-bites Shift-regiszter működése

- Közös szinkron CLK
- Két szelektor jel:  $S_0$ ,  $S_1$  (az állapotok kiválasztásához! – 4:1 MUX)
- 4-állapot: HOLD/ SHL /SHR /LOAD
- Kapcsolási rajz:



- Táblázat:

Clock	$S_1$	$S_0$	Result desired	Selected mux position	Required mux input
↑	0	0	Hold present data	0	$Q_i$
↑	0	1	Shift right	1	$Q_{i+1}$
↑	1	0	Shift left	2	$Q_{i-1}$
↑	1	1	Load new data	3	$DATA_i$

- 
- Ajánlott: a fejezetek végén lévő feladatok (Exercises) részek áttekintése.