Kapcsolómezők

Segédlet a Távközlő hálózatok és szolgáltatások c. előadáshoz

Németh Krisztián, tanársegéd, BME TMIT
Gabrovits Pál, hallgató, BME VIK
2009. november 15.

E segédlet a telefonközpontokban használt tér-, illetve időosztásos kapcsolás elvét igyekszik röviden bemutatni, az előadáson bemutatott fóliákat szöveggel kiegészítve.

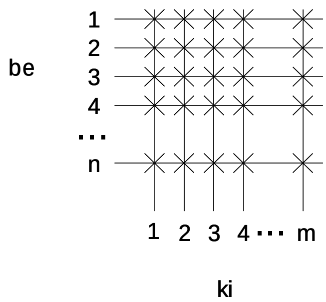
A kapcsolómezők típusai:

Elvben 4-féle kapcsolómező képzelhető el, de ebből a gyakorlatban csak kettőt használnak:

- **térkapcsolás**: az egyes kapcsolatok térben különülnek el egymástól
- **időkapcsolás**: az egyes kapcsolatok időben különülnek el egymástól
- **frekvenciaosztásos**: frekvenciában különíti el az egyidejű összeköttetéseket (a gyakorlatban nem használják)
- **kódosztásos:** az alkalmazott kóddal különíti el az egyidejű összeköttetéseket (a gyakorlatban nem használják)

<u>Térkapcsolás (Space Division Switching, „S”)</u>

Egyfokozatú kapcsolások:



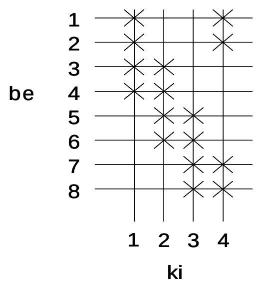
1. ábra. Egyfokozatú kapcsolás

Ennek az egyszerű kapcsolómezőnek *n* bemenete és *m* kimenete van. A megvalósítás az ábrán látható, melyhez $n \cdot m$ db kis kapcsoló, ún. kapcsolópont kell. E kapcsolópontok korábban mechanikusan hoztak létre galvanikus összeköttetést, manapság pedig elektronikusan működnek. Ezek közül a kapcsolópontok közül minden sorban és minden oszlopban egyszerre legfeljebb egy lehet aktív. Ha például a 4. sorban és 2. oszlopban lévő kapcsolópont aktív, az azt jelenti, hogy a 4. bemenet éppen össze van kapcsolva a 2. kimenettel.

Ennek a rendszernek az előnye, hogy egyszerű, és – ellentétben a később bemutatott rendszerekkel – sosem fordulhat benne elő blokkolás, azaz ha egy bemenet és egy kimenet szabad, azokat mindig össze is tudja kapcsolni. A rendszer azonban drága, mert $n \cdot m$ darab kapcsolópont kell hozzá, ami nagy sok, ha *n* és *m* nagy.

A továbbiakban a szükséges kapcsolópontok számát próbáljuk meg csökkenteni.

Egy lehetséges „spórolósabb” elrendezés például ez:



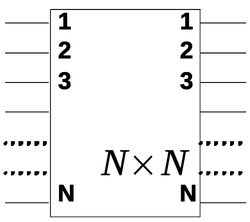
2. ábra. Egyfokozatú kapcsolás kevesebb kapcsolóponttal

Ilyen módon megspóroljuk a kapcsolópontok felét. Ugyanakkor látszik, hogy ekkor csak a kimenetek felével lehet egy bemenetet összekapcsolni. Az ötlet mégsem teljesen elvetendő: akkor használható, ha nem gond, hogy nem minden kimenettel lehet minden bemenetet összekapcsolni. Pl. ha egy banki tanácsadóval akarok beszélni, akkor mindegy, hogy pontosan melyikkel, csak egyet kapcsoljanak közülük.

Ez még mindig elég drága rendszer, ráadásul nem is túl praktikus, így nem használják a gyakorlatban.

Többfokozatú kapcsolások:

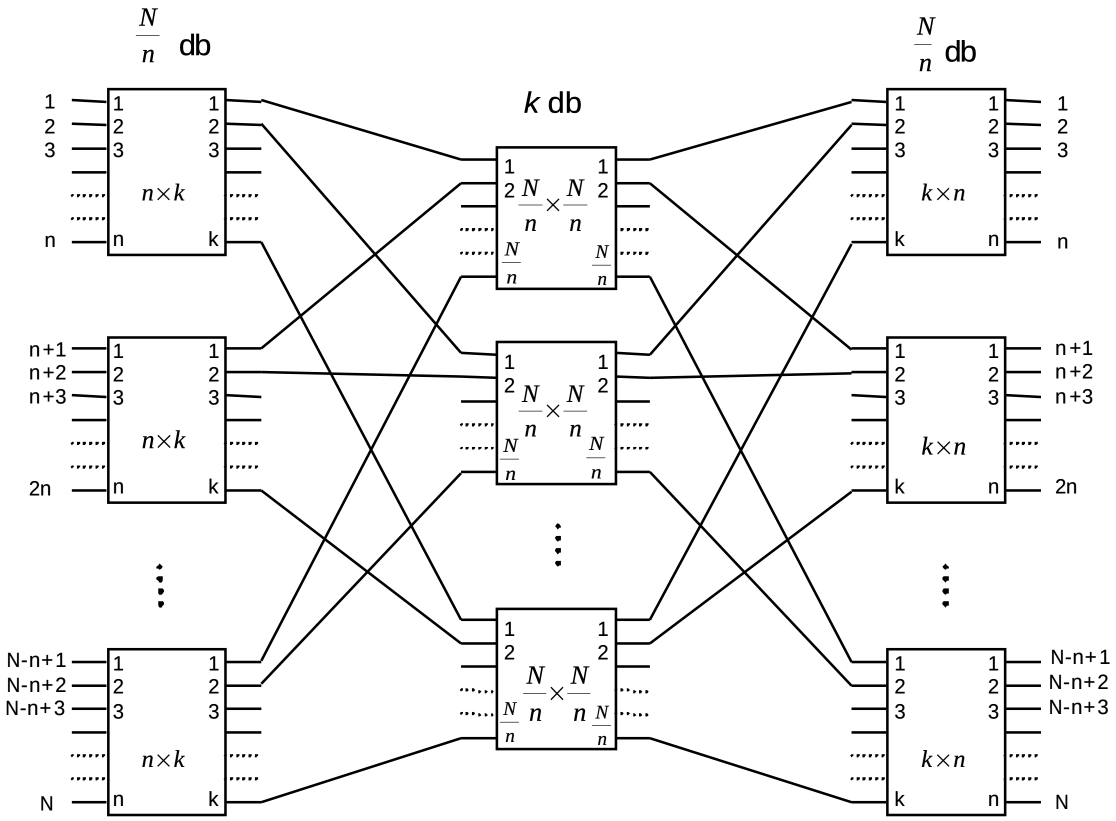
Nézzünk most egy hagyományos egyfokozatú térkapcsolót, amely N bemenettel és N kimenettel rendelkezik! Jelöljük így:



3. ábra

Tudjuk róla, hogy működése egyszerű, de drága, mert nagyon sok kapcsolópont kell hozzá (N^2). Ezek kihasználtsága ugyanakkor kicsi, hiszen egy sorban és egy oszlopban egyszerre legfeljebb csak egy kapcsolópont aktív. Keresünk erre a problémára egy jobb megoldást, ami funkciójában ezzel teljesen megegyezik, de kevesebb kapcsolóponttal realizálható.

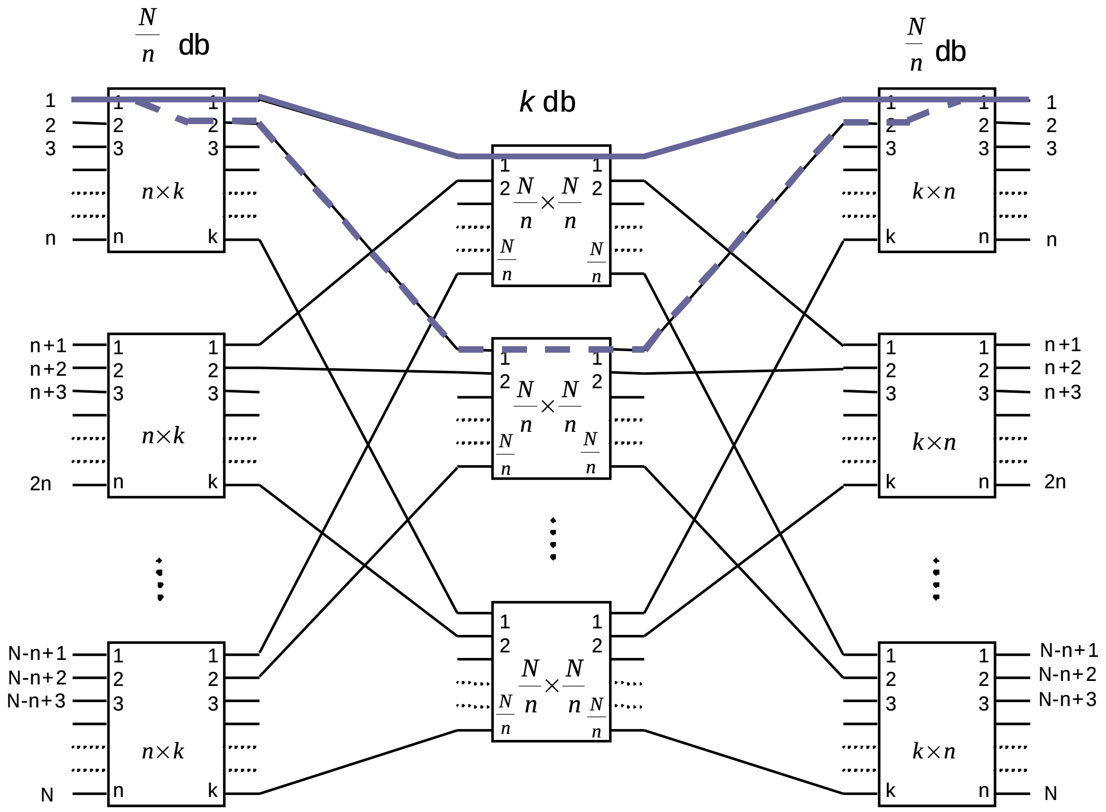
Elsőre meglepő lehet, de létezik ilyen megoldás. Ez az ún. a többfokozatú térkapcsoló. A 4. ábra a háromfokozatú térkapcsolót szemlélteti.



4. ábra. Háromfokozatú térkapcsoló (SSS kapcsoló)

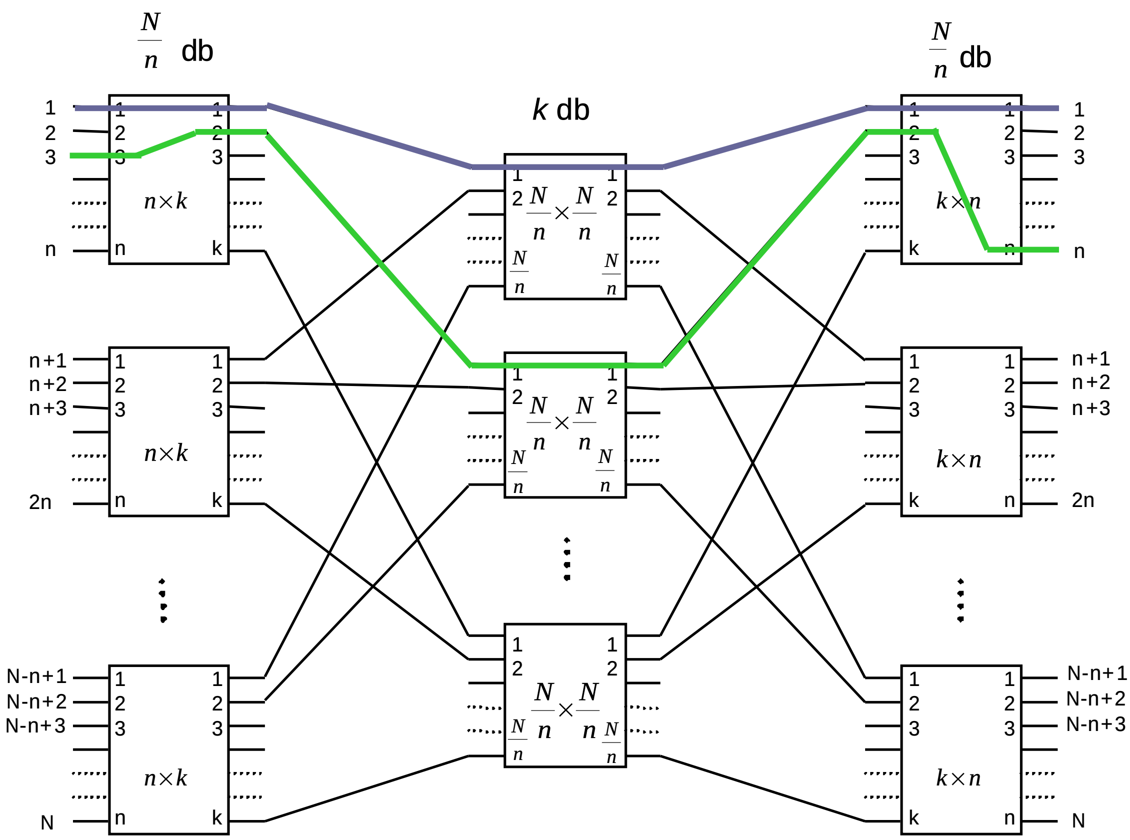
A felépítéséből adódóan ezt a kapcsolót SSS (tér-tér-tér, space-space-space) kapcsolónak is szokták nevezni. Láthatjuk, hogy az egész rendszer be-, illetve kimenetit tekintve megegyezik egy NxN-es egyfokozatú kapcsolóval.

A többfokozatú térkapcsolók egyfokozatú kapcsolókból épülnek fel: az ábrán minden téglalap egy egyfokozatú térkapcsoló. Nézzük meg, hogy kapcsol egy ilyen rendszer! Tegyük fel, hogy az első bemenetét szeretnénk az első kimenetével összekötni. Erre több lehetőség is van (ld. az 5. ábrát). Az első, teli vonallal jelölt esetben az első fokozat első kapcsolóblokkja az első bemenetét köti össze az első kimenetével, a második fokozat első blokkja szintén, és a harmadik fokozat első blokkja szintén. A szaggatott vonallal jelölt esetben az első fokozat első blokkja az első bemenetet a második kimenettel, a második fokozat második blokkja az elsőt az elsővel és a harmadik fokozat első blokkja a másodikat az elsővel. Mindkét megoldás teljesen egyenértékű, kívülről nem látszik a különbség.



5. ábra. Hívásfelépítés egy háromfokozatú térkapcsolóban

Természetesen nem mindig lehetséges bármelyik középső fokozatot használni. Nézzük meg most a következő ábrát:



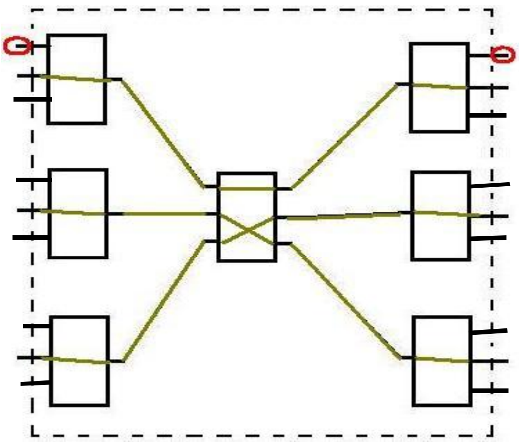
6. ábra. Több hívás egy háromfokozatú térkapcsolóban

Tegyük fel, hogy a lilával jelölt kapcsolat az 1. bemenet és 1. kimenet között már fel van építve, és ekkor érkezik egy igény a kapcsolómátrixhoz, hogy hozzon létre egy kapcsolatot az 3. bemenete és n. kimenete között. Látható, hogy ekkor a középső fokozat első dobozát már nem használhatjuk, hiszen az ahhoz az első fokozat első blokkjából, illetve az abból a harmadik fokozat első blokkja felé vezető kapcsolat is foglalt már. A középső fokozat második blokkja viszont használható, ahogy azt a zöld vonal mutatja.

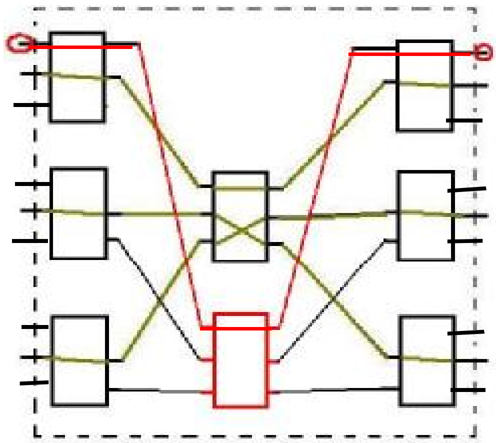
Tehát amellett, hogy egy külső szemlélőnek a többfokozatú kapcsolás úgy működik, mint egy egyfokozatú, tulajdonképpen merőben más belső működés áll a háttérben. Az egyfokozatú kapcsolásban egy utat egyértelműen választottunk ki: egy sorban és egy oszlopban egyszerre egy kapcsolópont volt aktív. Többfokozatú kapcsolás esetén már nem egyértelmű egy be és kimenet közötti út, ezért kell egy, a megfelelő utat kiválasztó algoritmus, ezáltal bonyolultabb a vezérlése.

Felmerül a kérdés, hogy hány darab középső fokozatbeli blokk kell egy ilyen rendszerbe, azaz mennyi legyen *k* értéke? Érdekes kérdés, hiszen elvileg a rendszer tetszőleges *k*-ra működik. Ugyanakkor ha *k* túl kicsi, megjelenhet a blokkolás. A blokkolás azt jelenti, hogy szabad egy bemenet, szabad egy kimenet, de mégsem tudunk közöttük kapcsolatot létesíteni, mert valamilyen belső erőforrás (pl. olyan középső fokozat, aminek szabad a megfelelő be és kimenete) nem áll rendelkezésre. A 6. ábra példájában: ha *k* értéke 1 lenne, akkor a zöld kapcsolatot már nem tudnánk létrehozni, azaz ha a lila kapcsolat fenn áll, akkor a zöld már blokkolódik.

Ugyanez egy másik példán:



7. ábra. Az adott hívás itt blokkolódik



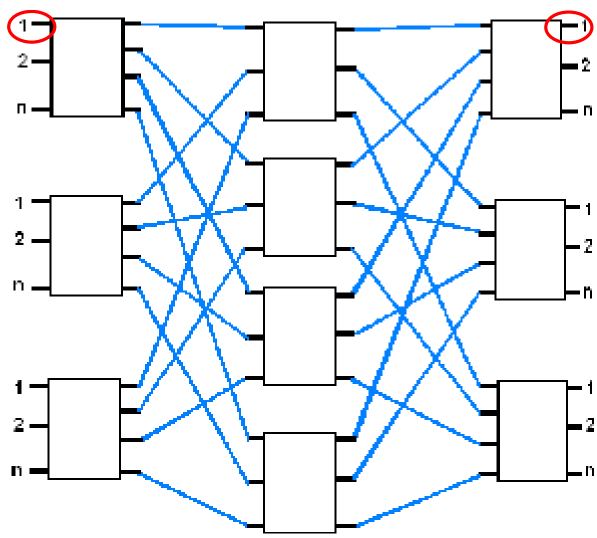
8. ábra. Az adott hívás itt nem blokkolódik

Láthatjuk, hogy a 7. ábrán már nem tudunk a két pirossal jelölt pont között kapcsolatot létesíteni, ha a zölddel jelölt utak foglaltak. Erre megoldásként a 8. ábrán bevezethetünk egy újabb középső fokozatot, így már nyilván felépíthető a kívánt kapcsolat.

Felmerül a kérdés, hogy vajon van-e olyan véges *k*, amelyre már akárhogy próbálkozok is, biztosan sosem lesz blokkolás. A válasz természetesen igen. Például nyilván, ha minden bemenet-kimenet párhoz van saját középső fokozatbeli blokk (N^2 db) akkor nem léphet fel blokkolás. Ez természetesen borzasztóan pazarló megoldás (és a célunk egyébként is az volt, hogy ezt az egyfokozatú kapcsolókra jellemző N^2 kapcsolópontszámot csökkentsük), de arra jó, hogy lássuk, van felső korlát.

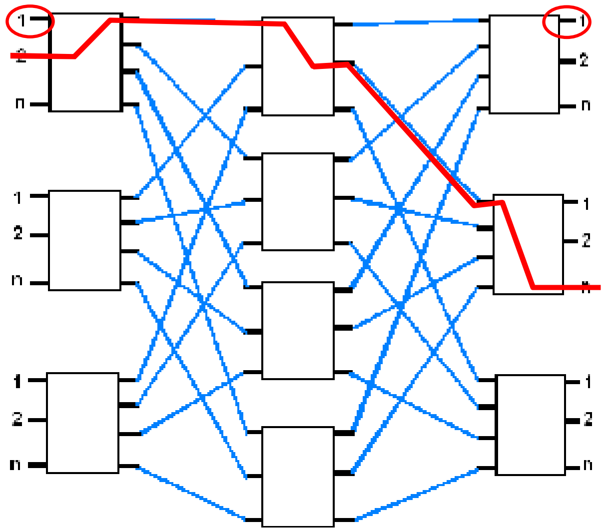
Adódik a kérdés, hogy mennyi a minimális *k*, azaz melyik az a *k*, amelyre még blokkolásmentes a kapcsoló, de ahol *k-1* középső fokozatbeli blokk esetén már blokkolás léphet fel.

A válaszhoz vezető gondolatmenetet egy példán nézzük végig! Tegyük fel, hogy a 9. ábrában szeretnénk az első bemenetet és az első kimenetet összekapcsolni. (Az egész rajz szimmetrikus, szóval akármelyik be- és kimenetről lehetne szó, ezzel nem csorbítunk az általánosságon.)



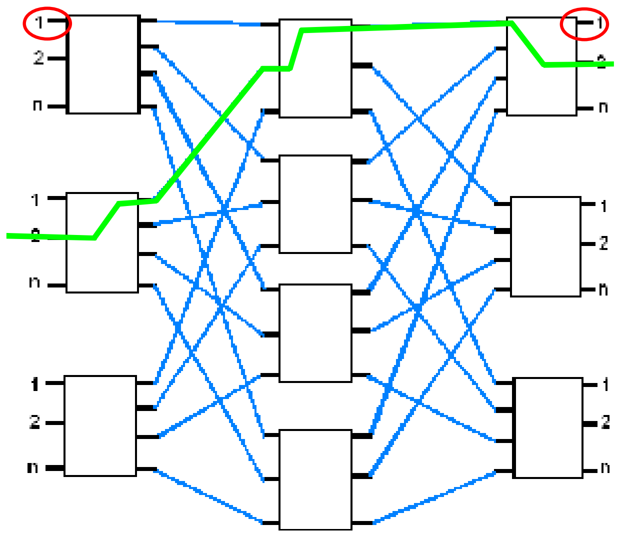
9. ábra. Egy példa kapcsoló

Játsszuk azt, hogy már fel van építve néhány kapcsolat, és ezek miatt nem tudjuk az általunk kívánt összeköttetést létrehozni, és számoljuk meg, hány középső fokozat foglalt ilyenkor. A legrosszabb esetre készüljünk fel (worst case design), és számoljuk meg ekkor hány középső fokozatbeli blokk foglalt. Nézzük meg az alábbi ábrát:



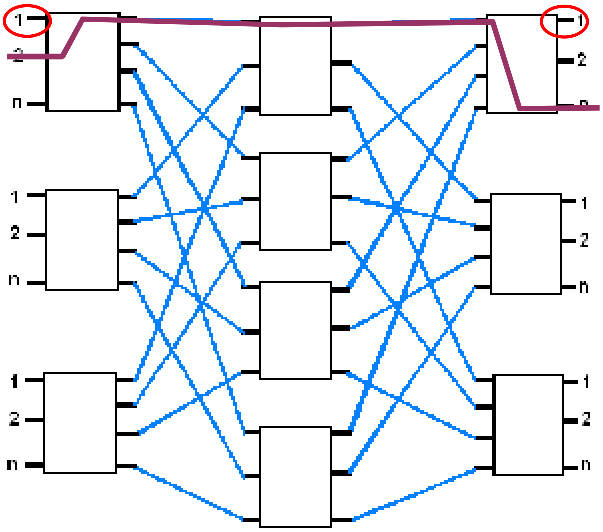
10. ábra. Bemenetén foglalt középső fokozatbeli blokk

Ez estben az első középső blokkot nem használhatjuk a kapcsolatunk felépítésére, mert annak a számunkra érdekes bemenete foglalt. Nézzük most ezt az esetet:



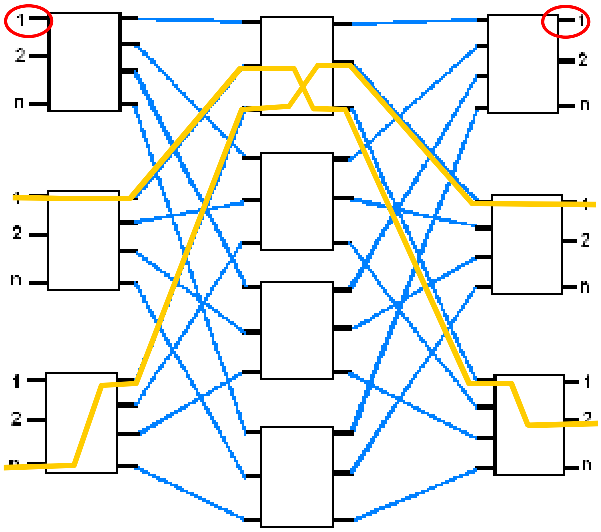
11. ábra. Kimenetén foglalt középső fokozatbeli blokk

Ekkor ugye az első középső blokk kimenete foglalt. Persze lehet mindkettő is foglalt:



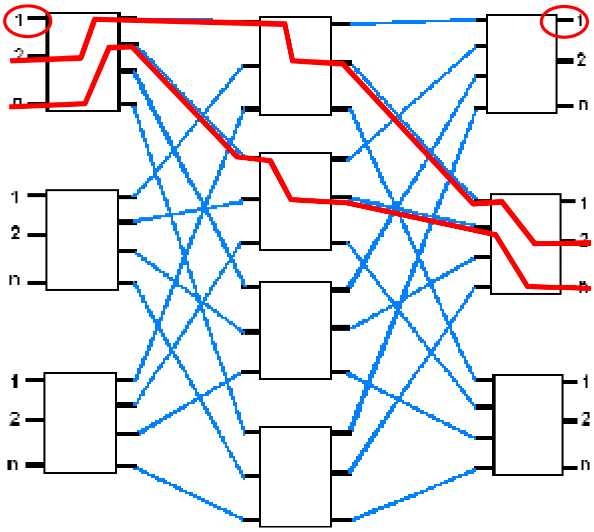
12. ábra. Be- és kimenetén foglalt középső fokozatbeli blokk

Illetve lehet mindkettő szabad, úgy, hogy azért a blokkon megy keresztül kapcsolat, akár több is:



13. ábra. A középső fokozatbeli blokk be- és kimenete is használható

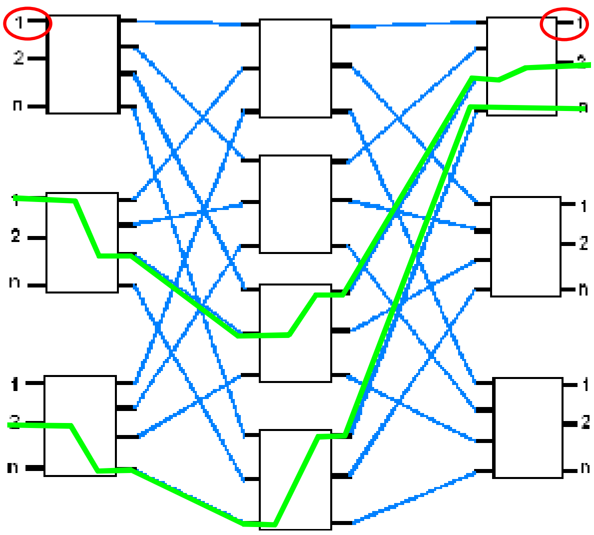
Számoljuk meg, hány középső fokozatot tudunk úgy „elrontani”, hogy a megfelelő bemenetük lesz foglalt:



14. ábra. Foglaltság a bemeneten

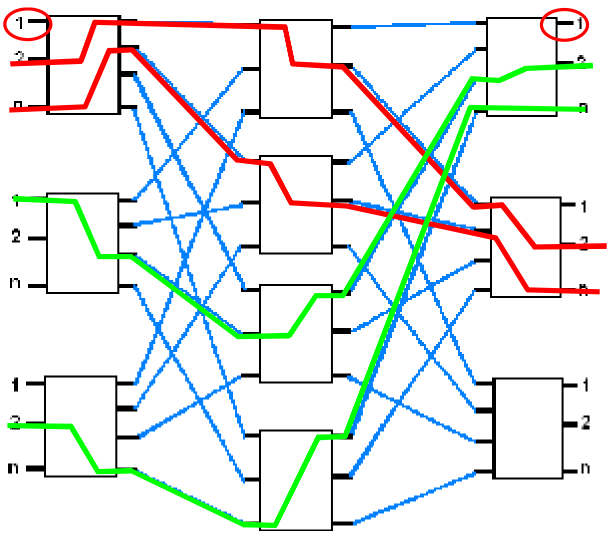
Ahogy az ábrán is látszik, legfeljebb *n-1*-et, hiszen minden ilyen kapcsolatnak az első fokozat első blokkjából, annak is valamelyik bemenetéről kell indulnia. Ott *n* bemenet van, de abból a feltevésünk szerint egy szabad, marad *n-1*.

Hasonló a helyzet a foglalt kimenetű középső blokkokkal:



15. ábra. Foglaltság a kimeneten

Ezekből is max. *n-1* lehet. Térjünk vissza most arra a kérdésre, hogy összesen legfeljebb hány középső fokozat lehet „rossz” a mi felépítendő kapcsolatunk szempontjából! Nyilván az a legrosszabb eset, ha az első fokozat első blokkjának mind az *n-1* bemenete foglalt, továbbá a harmadik fokozat első blokkjának mind az *n-1* kimenete foglalt, ráadásul mindez úgy, hogy nincs a 12. ábrán látható egybeesés. Ekkor *2n-2* középső fokozatbeli blokk lesz használhatatlan a mi kapcsolatunk felépítésére:



16. ábra. Foglaltság a be-, illetve kimeneteken

Ennél eggyel több, tehát *2n-1* középső fokozatbeli blokk tehát minden esetben elég. A válasz tehát *k=2n-1*.

Nézzük most újra a 4. ábrát! Hány kapcsolópontunk van a többfokozatú kapcsolóban? (Tudjuk, hogy egy egyfokozatú kapcsolóban található kapcsolópont száma *i* bemenet és *j* kimenet esetén $i \cdot j$)

- <u>első oszlop</u>: van $\frac{N}{n}$ darab $n \cdot k$ méretű egyfokozatú térkapcsolónk. Mivel egy kapcsolóblokkban $n \cdot k$ darab kapcsolópont van, ez összesen $\frac{N}{n} \cdot (n \cdot k)$.
- <u>második oszlop</u>: az első oszlopban összesen k darab $\frac{N}{n}$ bemenetű és $\frac{N}{n}$ kimenetű kapcsolóblokkunk van, ami $k \cdot \frac{N}{n} \cdot \frac{N}{n}$ kapcsolópont összesen.
- <u>Harmadik oszlop</u>: hasonlóan az első oszlophoz $\frac{N}{n} \cdot (n \cdot k)$ kapcsolópont.

Tehát összesen: első+második+harmadik oszlop= $\frac{N}{n} \cdot (n \cdot k) + k \cdot \frac{N}{n} \cdot \frac{N}{n} + \frac{N}{n} \cdot (n \cdot k)$

kapcsolópontunk van ebben a kapcsolásban, ez összesen: $2Nk + \frac{N^2 k}{n^2}$

Felmerül a kérdés, hogy ez valóban kevesebb-e, mint az egyfokozatú kapcsolás N^2 db. kapcsolópontja.

A válasz n illetve k értékeitől függ. Már megvan a legkisebb megfelelő *k* értéke (*2n-1*), de mekkora *n* értéket érdemes választanunk?

Ehhez keressük meg a $2Nk + \frac{N^2 k}{n^2}$ kifejezés minimumát, ha $k = 2n - 1$.

Ekkor a képlet: $2N(2n-1) + \frac{N^2(2n-1)}{n^2} = \frac{4Nn^3 - 2Nn^2 + 2N^2 n - N^2}{n^2}$. Deriváljuk és keressük meg a derivált nulla helyeit. Itt lehet a függvény maximuma ill. minimuma. Azért, hogy eldöntsük melyik, deriváljuk még egyszer és tudjuk, hogy ha a 2. derivált értéke nagyobb mint 0, akkor ott lokális minimuma lesz a függvénynek. A konkrét számításokat most mellőzzük, de higgyük el, hogy nagy N-re $n \approx \sqrt{\frac{N}{2}}$ esetén lesz a függvénynek minimuma.

Vajon tehát tényleg kevesebb kapcsolópont kell?

- Az egyfokozatú kapcsolóban N^2 kapcsolópont volt (mindkét oldalon n előfizető esetén).
- A többfokozatúban $2Nk + \frac{N^2k}{n^2}$ kapcsolópontról beszéltünk. Behelyettesítve blokkolásmentes esetre a *k*-t és a minimum helyre kapott *n* értéket:

$$\frac{4Nn^3 - 2Nn^2 + 2N^2n - N^2}{n^2} \approx \frac{4N\left(\sqrt{N/2}\right)^3 - 2N\left(\sqrt{N/2}\right)^2 + 2N^2\left(\sqrt{N/2}\right) - N^2}{\left(\sqrt{N/2}\right)^2} = 4N\left(\sqrt{(2N)} - 1\right)$$

kifejezéshez jutunk. Mivel nagy N-re $4N\left(\sqrt{2N} - 1\right) < N^2$, ezért tényleg kevesebb kapcsolópont kell. Nézzünk pár számpéldát, ahol ez jobban látszik:

N	kapcsolópontok 1 fokozat esetén	kapcsolópontok 3 fokozat esetén	n	N/n	k=2n-1
128	16 384	7680	8	16	15
8192	67 millió	4,2 millió	64	128	127
131072	17 milliárd	268 millió	256	512	511

1. táblázat. Blokkolásmentes többfokozatú kapcsolás

Ez jelentős megtakarítás, de még mindig elég sok a kapcsolópont. Mit tehetünk még?

Az egyik lehetőség, hogy további fokozatokat vezetünk be, például úgy, hogy a 4. ábra egyik vagy több kapcsolóblokkját egy komplett háromfokozatú kapcsolóval helyettesítjük. Ettől egy darabig még csökken a kapcsolópontok száma, bár egyre kevésbé, viszont egyre komplexebb lesz a rendszerünk.

A másik lehetőség, hogy megengedjük a blokkolást egy kis eséllyel. Ez esetben a kapcsolópontok száma attól fog függeni, hogy mekkora egy bemenet kihasználtsága, hiszen nyilvánvalóan több középső fokozatbeli blokk kell akkor, ha egy bemenetet gyakrabban használnak. (Blokkolásmentes esetben ez nem számított, hiszen a legrosszabb esetre terveztünk.) A számítások ekkor túl bonyolultak ahhoz, hogy e jegyzet kereteiben ismertessük. Elégedjünk meg egy példával, melyben a megengedett blokkolási valószínűség (azaz annak az esélye, hogy egy felépítendő hívás blokkolódik) 0,002, azaz 0,2%. E példát mutatja a 2. táblázat:

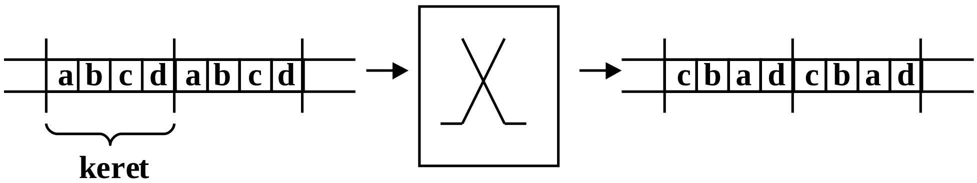
N	1 fokozat (blokkolás-mentes)	3 fokozat, blokkolás-mentes	3 fokozat, bemenet kihasználtság: 0,7		3 fokozat, bemenet kihasználtság: 0,1	
	kapcsoló-pontok	kapcsoló-pontok	kapcsoló-pontok	k/n	kapcsoló-pontok	k/n
128	16 384	7680	7168	1,75	2560	0,625
8192	67 millió	4,2 millió	2,1 millió	1,0	491520	0,234
131072	17 milliárd	268 millió	113 millió	0,84	21,5 millió	0,160

2. táblázat. Többfokozatú kapcsolás blokkolással

Figyeljük meg, mennyivel kevesebb kapcsolópont kell, illetve hogy a k/n arány a blokkolásmentesbeli kb. 2-ről akár 0,2 alá is csökkenhet!

Időosztásos kapcsolás (Time Division Switching, „T”)

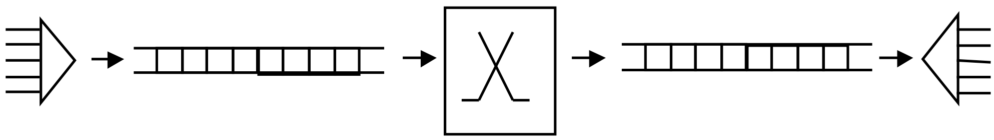
Az időosztásos kapcsolásnak, vagy rövidebben időkapcsolásnak TDM nyalábol jelek esetén van értelme. Ekkor az időkapcsoló a TDM kereteken belüli időréseket cseréli fel egymással (minden keretben ugyanúgy):



17. ábra. Időkapcsoló

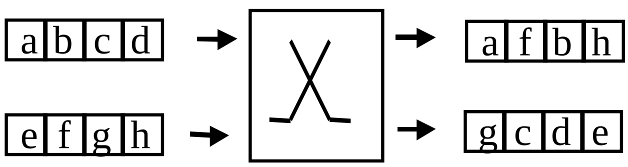
Viszonylag egyszerűen meg lehet ezt valósítani egy memóriával: soros beírással és a meghatározott sorrendbeli kiolvasással, vagy nem soros beírással és soros kiolvasással. Természetesen ez a kapcsolás késleltetést visz a rendszerbe, tipikusan egy keretnyit. Probléma, hogy a memória sebessége véges, csak néhány száz időrés lehet egy 125 μs hosszú keretben. Továbbá PDH esetén (ld. majd ott) az újrakeretezés miatt csak az első hierarchiaszinten valósítható meg.

Mire jó ez? Például egy ilyen időkapcsolóval megvalósíthatunk egy térkapcsolót a Digitális technika c. tárgyból tanult multiplexer/demultiplexer áramkörök segítségével:



18. ábra. Időkapcsolóból térkapcsoló

Még jobb alkalmazás az, hogy megvalósíthatunk vele tér- és időkapcsolót együtt:



19. ábra. Kombinált tér- és időkapcsoló

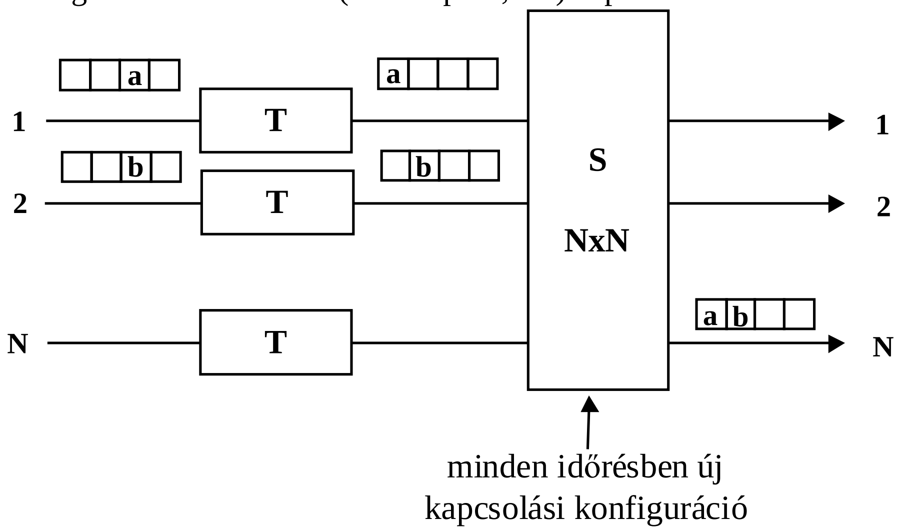
Egy ilyen kombinált tér- és időkapcsoló képes TDM nyalábolt jeleket fogadni és ugyanilyen jeleket kibocsátani, és ezek között úgy kapcsolni, hogy bármelyik bemenet bármelyik időrését bármelyik kimenet bármelyik időrésére kiteszi. Ez azért jó, mert egy központhoz a trönkökön a távoli központból érkező jelek amúgy is TDM nyaláboltak, és így nem szükséges teljesen lebontani őket. Ez azt jelenti, hogy egy központnak innentől lehet olyan bemenete, amin 30 beszédcsatorna van időosztásosan nyalábolva, és ezt nem kell lebontani pl. 30 érpárra csak azért, hogy a kimeneten újra összenyaláboljuk őket. Ráadásul, ahogy látni fogjuk, maga a kapcsolás is egyszerűbben végrehajtható így.

A továbbiakban azt fogjuk megnézni, hogy hogyan épülhet fel egy ilyen kombinált tér- és időkapcsoló.

Tér- és Időkapcsoló

TS

A legegyszerűbb megvalósítás az idő-tér (Time-Space, TS) kapcsoló:

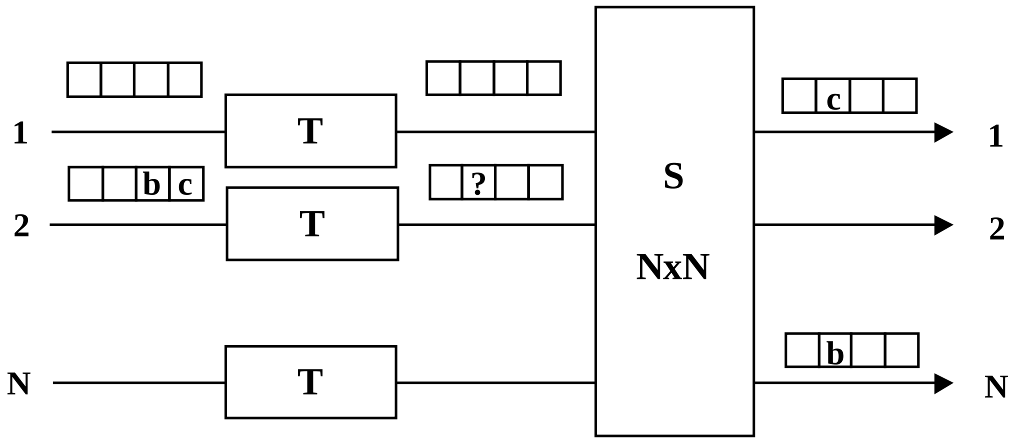


20. ábra. Egy TS kapcsoló

A T-k (time) az időkapcsolók, S (space) a térkapcsoló. Az időkapcsoló feladata, hogy berakja a jelet a megfelelő időrésbe, majd a térkapcsoló kirakja a megfelelő kimenetére.

Fontos, hogy bár a térkapcsoló működési elve változatlan (NxN db. kapcsolópont van benne, melyből egyszerre egy sorban és egy oszlopban max. egy aktív), most más üzemmódban használjuk, mint eddig. Ezelőtt a térkapcsolóban, ha felépült egy kapcsolat, akkor az adott kapcsolópont aktív volt, amíg a kapcsolat élt. Most ez nem így van: például a 20. ábrán az első időrésben a kapcsoló első bemenete van a N. kimenetével összekötve, majd a második időrésben a második bemenet lesz az N. kimenetre kapcsolva, és így tovább. A keret végén az egész kezdődik elölről. Más szavakkal, itt minden időrésben új kapcsolási konfigurációt használ a térkapcsoló, ezzel egyébként sokkal jobban kihasználja a benne lévő kapcsolópontokat.

Ez a TS kapcsoló azonban egy fontos hibát tartalmaz: akár már két darab szerencsétlenül érkező hívás is blokkolást okozhat. Figyeljük meg az alábbi példában, hogy már két kapcsolat esetén is blokkolás léphet fel, hiszen b és c is a második időrésbe kellene kerüljön egyszerre:

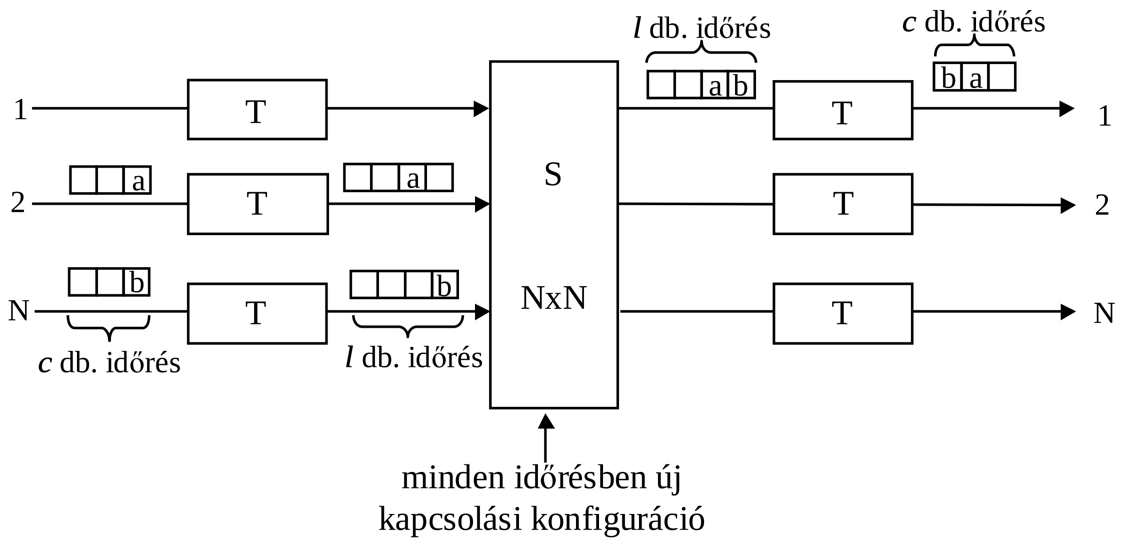


21. ábra. A TS kapcsoló nem jó

Ez pedig súlyos hiba. Azt ugyan bizonyos esetek között elfogadhatjuk, hogy egy nagyon leterhelt kapcsolóban blokkolás léphet fel, de az, hogy egy gyakorlatilag teljesen üres kapcsolóban lép fel blokkolás, elfogadhatatlan. Ennek a hibának a küszöbölésére találták ki a TST, illetve STS kapcsolókat.

TST

A TST kapcsoló felépítését mutatja 22. ábra:



22. ábra. TST kapcsoló

A kapcsoló érdekessége, hogy nem ugyananyi időrés van az első fokozat bemenetén (*c* db), illetve kimenetén (*l* db). Tipikusan egyébként $l > c$, ezért az első fokozat kimenetén lesznek szabad időrések. Mivel az S kapcsoló bemenetén és kimenetén ugyanannyi időrésnek kell lenni, hiszen az időkapcsolást nem végez, ezért a harmadik fokozat bemenetén szintén *l* db. időrés lesz. Továbbá az is biztos, hogy az egész kapcsoló bemenetén és kimenetén is ugyanannyi időrés kell legyen (hiszen a 19. ábra funkcionalitását szeretnénk elérni), ezért a harmadik fokozat kimenetén *c* időrés lesz.

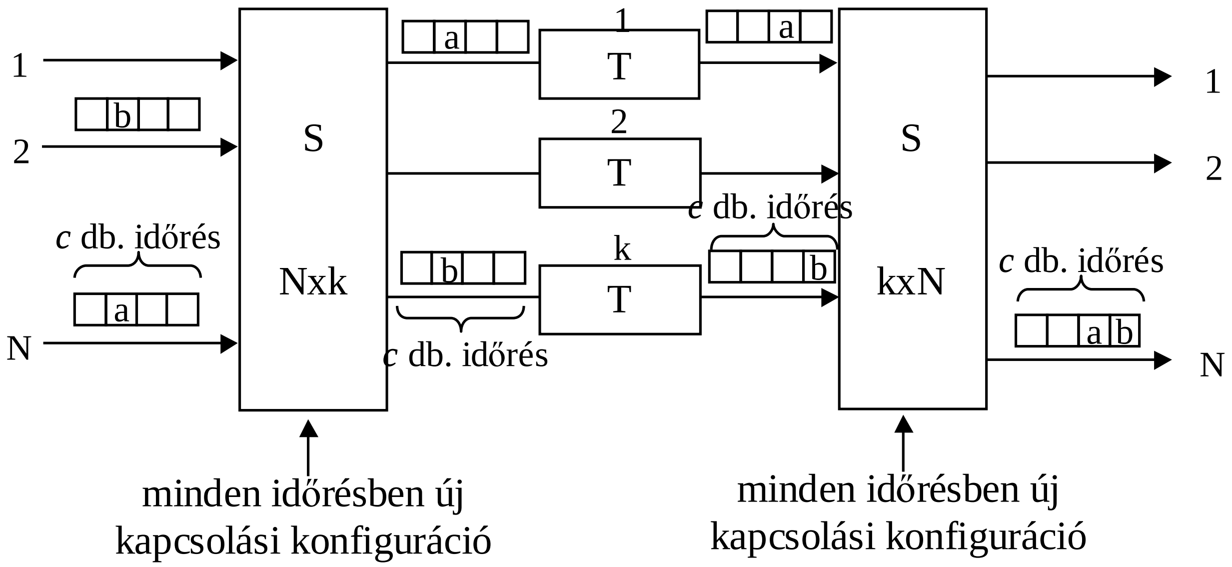
Könnyen végiggondolható, hogy ez a kapcsoló már fel tudja építeni a 21. ábrán bemutatott két kapcsolatot, ami a TS kapcsolóban még blokkolást okozott.

Ismét csak felmerül a kérdés, hogy mikor lesz blokkolásmentes ez a kapcsoló. Használjuk a háromfokozatú térkapcsolásnál (SSS) használt gondolatmenetet! Ott azt számoltuk meg, hogy hány olyan középső fokozatbeli blokk van, amelynek foglalt a megfelelő bemenete, és hánynak foglalt a megfelelő kimenete. Itt ennek analógiájára nem megfelelő középső fokozatbeli blokkot kell keresnünk, hanem megfelelő időrést az S kapcsoló be-, illetve kimenetén. A 22. ábra példáját tekintve adott, hogy a *b* kapcsolat az S kapcsoló *N*. bemenetére fog érkezni és az elsőn kell távozzon. Találnunk kell tehát az *l* darabból egy olyan időrést, amely szabad az S kapcsolónak mind a bemenetén, mind a kimenetén. Adott esetben a például a harmadik időrés (balról kezdve a számozást) szabad ugyan a bemeneten, de nem szabad a kimeneten.

Hasonlóan az SSS kapcsolóbeli esethez, könnyen belátható, hogy itt *c-1* darab időrés lehet foglalt a megfelelő bemenetén az S kapcsolónak, és szintén *c-1* darab lehet foglalt a megfelelő kimenetén. Ebből szerencsétlen esetben egy sem esik egybe (worst case design), azaz *2c-2* lesz foglalt. Ennél eggyel több, *l=2c-1* időrés mindig elég lesz a blokkolásmentes kapcsolatfelépítéshez.

STS

Az STS kapcsoló felépítését mutatja 23. ábra:



23. ábra. STS kapcsoló

Itt először térben, majd időben és végül megint térben kapcsoljuk a hívásokat. A fenti példában érkezik egy *a* hívás egy keretben, amit az S térkapcsoló az egyik kimenetére kapcsol. Ezután az adott időkapcsoló behelyezi *a*-t a megfelelő időrésbe, majd az utolsó fokozatban levő térkapcsoló a megfelelő kimenetére kapcsolja a már helyes időrésben levő *a*-t.

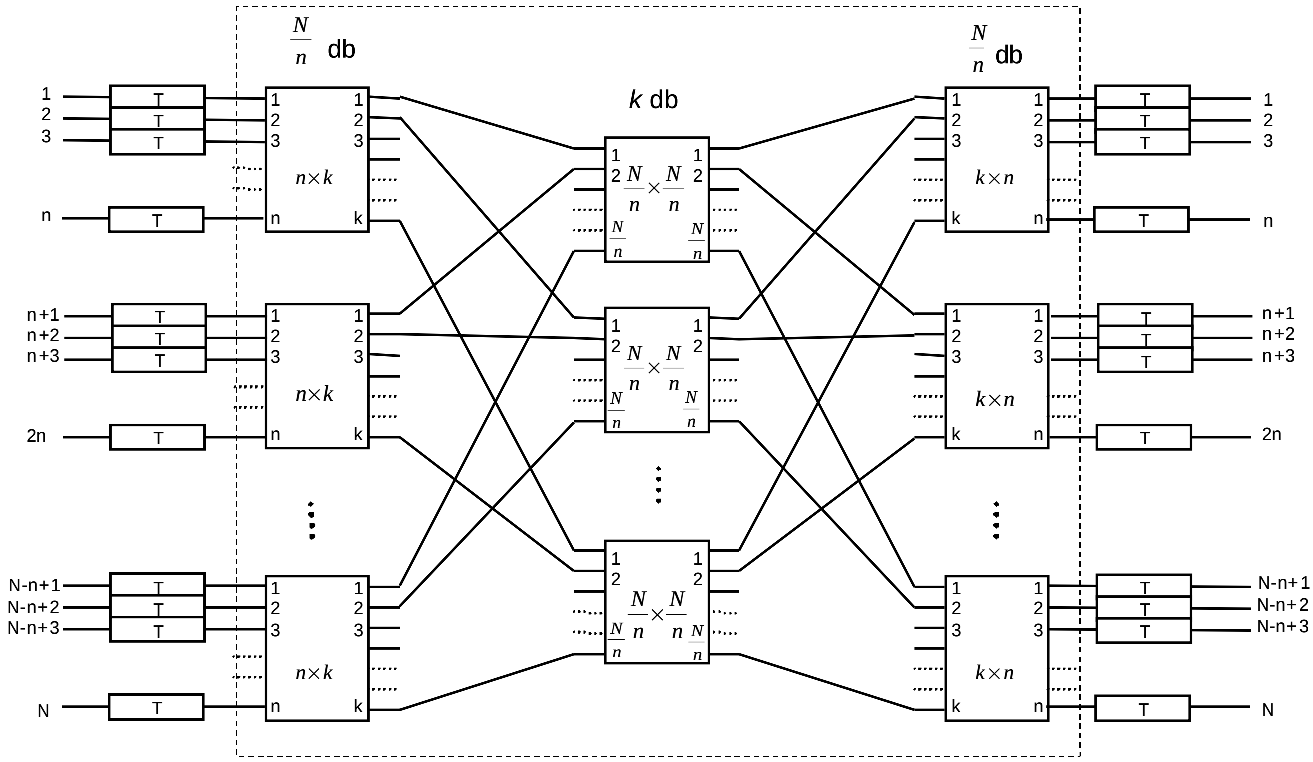
Vegyük észre, hogy itt az időrések száma végig *c*, fix. Ennek az oka egyszerű: az első kapcsoló bemenetén és kimenetén ugyanannyi időrésnek kell lenni, hiszen a térkapcsoló ezt nem tudja megváltoztatni. Az is biztos, hogy az egész rendszer kimenetén ugyanannyi időrés kell legyen, mint a bemenetén, ezért a harmadik fokozat kimenetén is *c* darab lesz. Végül pedig a harmadik fokozat is időkapcsoló, így annak a bemenetén is ugyanannyi időrés lesz, mint a kimenetén, tehát az is *c*.

A kérdés a már szokásos: mikor lesz ez a rendszer blokkolásmentes? Ez *k* értékétől, a középső fokozatban lévő időkapcsolók számától függ. A gondolatmenet természetesen hasonló lesz az SSS, illetve a TST kapcsolók esetében bemutatottakhoz. Tegyük fel, hogy az első bemenet *i*-edik időréséből szeretnénk kapcsolatot felépíteni az első kimenet *j*-edik időrésébe. Ehhez egy olyan középső fokozatot kell találnunk, amelynek szabad az *i*-edik bemeneti és a *j*-edik kimeneti időrése.

Maximum hány *i*-edik bemeneti időrés lehet szabad? Ugye *N-1*, hiszen *N* bemenetünk van, de abból egy (példánkban az első) egyelőre biztosan szabad. Ugyanez a helyzet a kimenetekkel, és szerencsétlen esetben nincs ezek között egybeesés. Ekkor *2N-2* középső fokozatbeli időkapcsoló lesz számunkra foglalt, ennél eggyel több, *k=2N-1* pedig mindig elég.

TSSST

A gyakorlatban ennél bonyolultabb felépítésű kapcsolómezőket is használnak, pl. TSSST kapcsolót. Ez tulajdonképpen egy TST kapcsoló, ahol az S egy háromfokozatú térkapcsolóval van megvalósítva, így valahogy:



24. ábra. TSSST kapcsoló