

Híradó
Híradó

A1

73.180/MK

KIVONAT

Eljárás és berendezés adatok átvitelére összesorolással és azt követő sebesség átalakításával lyuggatás és ismétlés útján

Eljárás és berendezés adatkeretek átvitelére, valamint eljárás és berendezés adatsebesség átalakítására.

Az átvitelre szánt elemeket egy összesorolóval rádiókeretekre bontjuk, majd lyuggatjuk, vagy ismételjük, ahol is a lyuggatást, vagy az ismétlést egy minta segítségével úgy végezzük el, hogy az megakadályozza az egymással szomszédos, vagy egymáshoz közel eső elemek lyuggatását, illetve ismétlését, amikor az elemek összesorolás előtti, eredeti sorrendjére alkalmazzuk (10. ábra).





S.B.G. & K.

Nemzetközi

Szabadalmi Iroda

H-1062 Budapest, Andrásy út 113.

Telefon: 34-24-950, Fax: 34-24-323

73.180/MK

KÖZZÉTÉTELI
PÉLDÁNY

41

ELJÁRÁS ÉS BERENDEZÉS ADATOK ÁTVITELÉRE ÖSSZESOROLÁSSAL ÉS AZT
KÖVETŐ SEBESSÉG ÁTALAKÍTÁSSAL LYUGGATÁS ÉS ISMÉTLÉS ÚTJÁN

A szóban forgó találmány eljárást és berendezést ad adatok átvitelére összesorolással és azt követő sebesség átalakítással lyuggatás és ismétlés útján.

A digitális távközlési rendszerek úgy vannak adatok átvitelére kialakítva, hogy az adatokat olyan alakra hozzák, ami megkönnyíti a távközlési eszközön keresztül történő átvitelüket. Például rádióhullámos átvitel esetén az adatokat rádiójel formájában állítják elő, ami a távközlési rendszerben az adó és a vevő között átvitelre kerül. Szélessávú távközlési rendszerek esetében az adatok fény formájában is előállhatnak, és például egy fényvezető száloptikai hálózatban a rendszer adói és vevői között átvihetők.

Az átvitel során az átvitt adat egyes bitjei vagy szimbólumai megmásíthatnak, aminek eredményeként a vevőben ezek a bitek vagy szimbólumok nem ismerhetők fel helyesen. Ebből az okból kifolyólag az adatközlési rendszerek gyakran tartalmazzanak az adatok átvitele során fellépő megmásítódásának kiküszöbölésére szolgáló eszközöket. Az egyik ilyen eszköz a rendszer adóinak kódolókkal történő felszerelése, melyek az adatokat átvitel előtt megfelelően egy hibavédelmi kóddal kódolják. A hibavédelmi kód úgy van kialakítva, hogy az adatokhoz szabályozott módon redundáns információt fűz. A vevőben, ahol a hibavédelmi kód dekódolásra kerül - ami által újratерemthetők az

eredeti adatok - az átvitel alatt fellépő hibák korrigálhatók. A dekódolás egy hibadekódoló algoritmus felhasználásával történik, ami megfelel a vevő által ismert hibavédelmi kódnak.

Az adatok kódolása után az adatok átvitelét megelőzően a sebesség átalakításhoz (rate matching) gyakran szükség van egy kódolt adatblokkból származó adatbitek vagy szimbólumok lyuggatására vagy megismétlésére. A lyuggatás fogalma itt egy kódolt adatblokkból bitek eltávolításának vagy törlésének folyamatát jelenti, aminek eredményeként a lyuggatott bitek nem kerülnek átvitelre az adott adatblokkal. Pontozásra például akkor lehet szükség, amikor az adatok adatátviteli közegen történő átvitelére szolgáló többszörös hozzáférésű eljárás az adatoknak egy meghatározott blokkméretet szab meg, ami nem felel meg a kódolt adatkeretek nagyságának.

Egy átviteli adatblokkban a kódolt adatkeret meghatározott mérethez való igazításához a kódolt adatkeretből adatbiteket lyuggatnak ki, amennyiben a kódolt adatkeret nagyobb, mint az átviteli blokk mérete, vagy megismétlik a kódolt adatkeret bitjeit abban az esetben, amikor a kódolt adatkeret kisebb, mint az átviteli blokk előre meghatározott mérete. A továbbiakban ezt példaként egy mobil távközlési rendszer kapcsán részletesebben ismertetjük:

A mobil távközlési rendszerek többszörös hozzáférésű rendszerekkel vannak ellátva, melyek a többszörös hozzáférést például időbeli multiplexálással (TDMA - Time Division Multiple Access, időmegosztásos többszörös hozzáférés) való-sítják meg, mint például az, melyet a globális mobil távközlési rendszerekben (GSM - Global System for Mobile Phones, globális mobil távközlési rendszer) használnak, ami az Európai

Távközlési Szabványhivatal által szabványosított mobil távközlési szabványok egyike. Egy másik lehetőségként a mobil távközlési rendszer egy olyan többszörös hozzáférésű rendszerrel rendelkezhet, mely a többszörös hozzáférést kódmultiplexálással (CDMA - Code Division Multiple Access, kódmegosztásos többszörös hozzáférés) valósítja meg, mint például az univerzális mobil távközlési rendszer harmadik generációjaként javasolt UMTS (Universal Mobile Telecommunication System, univerzális mobil távközlési rendszer) rendszer. Mindenképpen egyértelmű, hogy a találmány egy kiviteli példája tetszés szerinti adatközlési rendszer esetében alkalmazható, mint például egy helyi adathálózat, vagy egy szélessávú távközlési hálózat, melyek megfelelően aszinkron átvitelt valósítanak meg. Ezeket a például szolgáló adatközlési rendszereket az jellemzi, hogy az adatok keretek, csomagok, vagy blokkok formájában kerülnek átvitelre. Egy mobil távközlési rendszer esetében az adatokat adathordozó rádiójel-keretekben továbbítják, melyek egy előre meghatározott adatméretet jelentenek. Egy ilyen mobil távközlési rendszer példája látható a 7. ábrán.

A 7. ábrán három BS bázisállomás látható, melyek MS előfizetői állomásokkal váltanak rádiójeleket egy rádiós lefedési területen, melyet a 2 szaggatott vonalak által meghatározott 1 cellákkal hoznak létre. A BS bázisállomások egy NET közvetítőhálózattal (Netzrelaysystem) vannak összekapcsolva. Az MS előfizetői állomások és a BS bázisállomások adatokat váltanak, ahol is 4 rádiójeleket visznek át 6 antennák között, melyek az MS előfizetői állomásokhoz, illetve a BS bázisállomásokhoz vannak kötve. Az MS előfizetői állomások és a BS bázisállomá-

sok között az adatokat egy adatközlő készülék segítségével viszik át, melyben az adatokat 4 rádiójelekké alakítják, melyeket a vevő 6 antennához továbbítanak, ami felismeri a rádiójeleket. A vevő segítségével a rádiójelekből újra előállítják az adatokat. Emellett a találmány a felfelé irányuló (MS -> BS, uplink) kapcsolatnál és a lefelé irányuló (BS -> MS, downlink) kapcsolatnál egyaránt alkalmazható.

A 8. ábra egy példaszerű adatközlő berendezést ábrázol, mely rádiós távközlési kapcsolatot létesít az MS előfizetői állomások egyike, illetve a BS bázisállomások egyike között, ahol a 7. ábrán is látható alkatrészek rendre ugyanazon számjelölésekkel vannak ellátva. A 8. ábrán egy 10 adatforrás 8 adatkereteket állít elő olyan sebességgel, melyet a forrás által előállított adat határoz meg. A 10 adatforrás által előállított 8 adatkeretek egy 12 sebesség-átalakítóba kerülnek, ahol végbemegy a 8 adatkeretek 14 átviteli adatblokkokká történő átalakítása. A 14 átviteli adatblokkok egy előre meghatározott mérettel és egy adatmennyiséggel jönnek létre, melyek adathordozó rádiókeretek formájában átvihetők, illetve segítségével az adatok továbbíthatók egy rádiós interfészen keresztül, melyet egy 18 adóból és egy 22 vevőből álló páros alkot.

A 14 átviteli adatblokk egy 16 rádiós hozzáférési processzorhoz kerül, mely a 14 átviteli adatblokkok rádiós hozzáférési interfészen keresztül történő átvitelének folyamatvezérlését végzi. Egy megfelelő időpontban a 14 átviteli adatblokk a 16 rádiós hozzáférési processzoron keresztül egy 18 adóhoz kerül, ahol végbemegy az átviteli adatblokkok átalakítása adathordozó rádiókeretekké, melyek azon időtartam alatt



kerülnek átvitelre, melyet az adó a rádiójelek továbbítására kap. A 22 vevőben a vevő 6'' antennája felismeri a rádiójeleket, és az adatkereteken egy visszafele történő átalakítást hajt végre, illetve újrateremti azokat, melyek ezután egy 24 rádiós hozzáférés folyamatvezérlési irányváltó berendezéshez kerülnek. A 24 rádiós hozzáférés folyamatvezérlési irányváltó berendezés a vett átviteli adatblokkokat egy 26 sebesség-átalakítási irányváltó berendezéshez továbbítja a 24 rádiós hozzáférés folyamatvezérlési irányváltó berendezés irányítása mellett, ami egy 28 vezérlőn keresztül történik. Ezek után a 26 sebesség-átalakítási irányváltó berendezés az újrateremtett 8 adatkereteket előállítja egy cél - illetve egy nyelő számára, amit a 30 blokk alkot.

A 12 sebesség-átalakító és a 26 sebesség-átalakítás irányváltó berendezések úgy vannak kialakítva, hogy amennyire csak lehet, optimálisan használják ki a 14 átviteli adatblokk által nyújtott adatátviteli kapacitást. A találmány egy mintaszerű kiviteli példájában erre a 12 sebesség-átalakító szolgál, mely az adatkeretek kódolását, és azt követően a kódolt adatkereteket kiválasztott adatbitek, vagy szimbólumok lyuggatását végzi, aminek eredményeként egy olyan átviteli adatkeret áll elő, mely beleillik a 14 adatblokkokba. A 12 sebesség-átalakító tartalmaz egy kódolót és egy lyuggatót. A kódolóhoz továbbított 8 adatkereteket egy kódolt adatkeret előállítása végett kódolják, melyet a lyuggatóhoz továbbítanak. Ezt követően a lyuggató lyuggatja a kódolt adatkereteket, ezáltal egy 14 átviteli adatblokkot állít elő. A keretek lyuggatása kiviteli példaként a felfelé, illetve a lefelé irányuló kapcsolatoknál egyaránt elvégezhető.

A GB 2296165 A -ból ismert egy olyan multiplex távközlési rendszer, mely tartalmaz egy lyuggatást és egy összesorolást.

Szakemberek számára köztudott, hogy egy kódolt adatkeret lyuggatásának hatására az eredeti adatok hiba nélküli újrateremtésének valószínűsége lecsökken. Emellett az ismert hibavédelmi kódok, illetve a hibavédelmi kódokhoz tartozó dekódolók akkor nyújtják a legjobb teljesítményt, amikor az adatátvitel alatt fellépő hibákat Gauss-eloszlású zaj okozza, ugyanis annak hatására a hibák az átviteli adatblokkban függetlenül oszlanak el. Amikor egy kódolt adatkeretet lyuggatni kell, a kódolt adatkeretben lehetőleg minél távolabb kell lenniük egymástól azoknak a helyeknek, ahol a bit lyuggatásra kerül. Ennek értelmében a lyuggatási helyeknek egyenletesen kell eloszlanuk az adatkeretben. Mivel a hibák az átvitel során gyakran halmozottan lépnek fel - különösen olyan rádiós távközlési rendszerek esetében, ahol nem használnak összesorolást -, és mivel a bitek ismétlésével az adatkeret minőségét annak nem csak egy meghatározott részén, hanem lehetőleg egyenletesen kell feljavítani, egy kódolt vagy kódolás nélküli adatkeretben úgy kell elhelyezni azokat a helyeket, ahol adatbitek ismétlésére van szükség, hogy azok egyenletesen, egymástól elválasztva helyezkedjenek el az adatkeret egészén belül.

Egy kódolt adatkeretben a lyuggatásra kerülő bitek, vagy szimbólumok helyének kiválasztására szolgáló eljárások közé tartozik a keretben található bitek, vagy szimbólumok számának elosztása a lyuggatandó bitek, vagy szimbólumok számával, és a helyeknek az osztásnak megfelelő egész számokkal történő kiválasztása. Abban az esetben, amikor a lyuggatandó bitek száma nem maradék nélküli osztója az adatkeretben található bitek

számának, a lyuggatott helyek távolságeloszlása nem lesz egyenletes, aminek az a hátránya, hogy az egyes lyuggatott helyek közti távolság kisebb lesz, mint az a bizonyos egész szám, és a lyuggatott helyek gyakran akár közvetlenül egymás mellé is esnek.

A továbbiakban ennek az összetett találmánynak egy szűkebb technikai területét és az ott fellépő problémákat írjuk le az 1-6, illetve 9. ábrák kapcsán, melyek legalább részben a harmadik mobil távközlési generáció szabványosításának a találmány előtti pillanatnyi állásából adódnak, mely a következő dokumentumban van megadva: S1.12 v0.0.1, 3GPP FDD, Multiplexing, channel coding and interleaving description (multiplexálás, csatornakódolás és összesorolás leírása).

Egy átvitel-multiplexálási eljárás kereteiben az összesorolás gyakran két lépésben kerül végrehajtásra. A lyuggatás/ismétlés elvégzésére szolgáló különböző megoldásoknak különböző következményei vannak, amikor a lyuggatás az első összesoroló (1st interleaver) után történik, mint ahogy az UMTS rendszer számára elő van irányozva. Az UMTS rendszerben ezen kívül egy második összesorolót is használnak, melyet a fizikai csatornafelosztás (Physical channel segmentation) után, illetve a fizikai csatornaleképzés (Physical channel mapping) előtt hajtanak végre (lásd: 1. ábra). Ez az összesoroló ugyan a lehető legegyenletesebbé teszi az elküldött bitek eloszlását, azonban nincs hatással a lyuggatott/ismételt bitek eloszlására, ezért a találmány kapcsán a továbbiakban nem tárgyaljuk.

Az 1. ábra egy FS-MIL összesoroló (FS-Multistage Interleaver) felfelé irányuló kapcsolat multiplexálási eljárásban összesorolóként történő alkalmazását mutatja be egy

közismerten az UMTS-hez előterjesztett sebesség-átalakító algoritmussal összefüggésben.

Vegyünk példaként egy olyan esetet, ahol a 2 réteg (Layer 2) egy 160 bites átviteli blokkot továbbít egy 80 ms átviteli ciklusú átviteli csatornán. Ez a bitsorozat adatkeretként, vagy adatkeretek sorozataként is leírható. Ez azt jelenti, hogy az adatok az első összesoroló után nyolc rádiókeretben (radio frame) (a későbbiekben gyakran keretként, vagy oszlopként említjük) továbbítódnak. Az összesorolás itt a bitek egy sor szerinti beolvasásából, illetve a bitek egy oszlop szerinti kiválasztásából áll az azt követő oszlopcserével (column randomizing).

Egy jó lyuggató algoritmus első célja az, hogy a lyuggatott biteket olyan egyenletesen ossza el a bitpozíciókban azok eredeti sorrendjében, amennyire csak lehet. Az UMTS esetében alkalmazott lyuggató algoritmus megalkotásánál is ez volt a meghatározó alapelv, ahogy például a fent nevezett S1.12 specifikációban is le van írva. Ez minden n . bit lyuggatásával érhető el a legjobban, részenként nem egész számú lyuggatási arány mellett minden $(n+1)$. bitnél.

Egy második cél az, hogy a különböző kereteket (a későbbiekben a kereteket gyakran oszlopként, vagy rádiókeretként említjük) ugyanolyan sűrűséggel lyuggassa, és ezzel a lyuggatott biteket az összes keret fölött is egyenletesen ossza el, és a különböző keretek között is egyenletes lyuggatást érjen el. Egy oszlop (keret) lyuggatása, illetve ismétlése alatt egy elemnek, pontosabban az oszlop (keret) egy bitjének lyuggatását, illetve ismétlését is értjük.

Vegyünk például azt, amikor minden keretben

(rádiókeretben) négy bitet kell lyuggatni az adott átviteli csatorna szolgáltatásának minőségi követelménye, illetve a többi csatorna közti egyensúly megteremtéséhez. A korábban az UMTS rendszer számára előterjesztett sebesség-átalakító algoritmus abból áll, hogy minden keretben (rádiókeretben) a 4, 9, 14 és 19. bitek (az index 0-tól kezdődik, számozás a bitek első összesorolás utáni sorrendje szerint) vannak lyuggatva. A 2. ábrán a lyuggatott bitek vastagon vannak szedve. Ennek következtében nyolc egymással szomszédos bit lyuggatásra kerül, ami - ahogy azt korábban kifejtettük - nem kívánatos. A fent megnevezett első célkitűzésnek így ez nem tesz eleget kielégítő mértékben.

Ennek a problémának az elkerülésére egy módszer lehet a lyuggatási minta eltolása minden keretben. Legyen N_i egy keretben a bitek száma a sebesség-átalakítás előtt, N_c a bitek száma a sebesség-átalakítás után, m_j a lyuggatandó bitek helyei egy összesorolt keretben, k a keretszám, valamint K az összesorolt keretek száma.

Tekintsük az $N_i < N_c$ esetet, illetve a lyuggatást. A fent említett példában $N_i=20$, $N_c=16$, $m_1=4$, $m_2=9$, $m_3=14$, $m_4=19$, $k=1, \dots, 7$ és $K=8$. A fent nevezett probléma elkerülésére a lyuggatott bitek helyeinek eltolása a következő képlettel írható le:

$$m_{j, \text{eltolt}} = (m_j + k * \lceil N_c / (N_i - N_c) / K \rceil) \bmod N_i$$
, ahol a $\lceil \rceil$ a felfele kerekítést jelenti.

Az ebből a képletből adódó lyuggatandó bitek helyei a fenti példára vonatkozóan a 3. ábrán láthatók.

Amint az a 3. ábrán látható, a szomszédos bitek lyuggatása egy bizonyos mértékig kiküszöbölődik, azonban adódik egy

peremhatás, amikor például a 43. és 44. bitek lyuggatásra kerülnek, ami - ahogy azt korábban kifejtettük - nem kívánatos. A fent megnevezett első célkitűzésnek így ez ismét nem tesz eleget kielégítő mértékben.

Amennyiben a lyuggatási arány nagyon kicsi, csökken a szomszédos bitek lyuggatásának valószínűsége. A 4. ábrán egy 10 százalékos lyuggatási arány látható. Amint az a 4. ábrából kitűnik, ennek ellenére még mindig szomszédos bitek (91. bit és 92. bit) vannak lyuggatva, ami teljesítményesést vonhat maga után. A fent megnevezett első célkitűzésnek ez megint nem tesz eleget kielégítő mértékben.

A leírt sebesség-átalakító algoritmust helyettesítő lehetőségként javasolták az első összesorolónak egy olyan irányú optimalizálását, hogy a leírt sebesség-átalakító algoritmus lyuggatásának használatára ne legyen szükség. Egy optimalizált első összesorolónak úgy kellene a biteket átrendeznie, hogy a szomszédos biteket elkülönítse egymástól. Ennek megfelelően a lyuggatás könnyen elvégezhető azáltal, hogy az összesorolás után az egymást követő biteket eltávolítják. Azonban két további lehetőség áll fenn, melyeket az 5. ábrán bemutatott helyzetek kapcsán közelebbről megvizsgálunk:

A TrCH A -n ("A" átviteli csatorna) a négy átviteli adatblokkot összesorolják, majd elvégzik a sebesség-átalakítást. A lyuggatásnál minden keretben eltávolítják az egymást követő biteket. Így nagyon kicsi a valószínűsége annak, hogy egy keretben a lyuggatott bitek az összesorolás előtt, azaz a kódolás után egymással szomszédosan helyezkedtek el. Ugyanakkor nincs semmilyen biztosíték arra, hogy a különböző keretekben a lyuggatott bitek nem voltak szomszédosak a kódolás után. Így

tehát ennek a módszernek a használata során teljesítményesés léphet fel.

A 4. ábra kapcsán ismertetett problémára megoldást adhat a következő eljárás is, melyet a 6. ábra kapcsán ismertetünk, melynek során az egyik kereten alkalmazott lyuggatási mintaeltolást a többi keretre is alkalmazzák, ahol is az eltolt mintát az összesorolás előtt alkalmazzák a keretekre. A 6. ábrán egy, a 3. ábrán bemutatott példaszerű bitsorozatra való lyuggatási minta látható. Az ábrán látható, hogy szomszédos bitek lyuggatására - legalábbis ebben az esetben - nem kerül sor. Így a lyuggatásból adódó teljesítményesés ebben az esetben elkerülhető.

A fenti sebesség-átalakításnak a véletlenszerű oszlopcsere előtt történő elvégzése valójában nem szükségszerű. A véletlenszerű oszlopcsere után elvégezhető egy azzal egyenértékű sebesség-átalakítás, melyben figyelembe veszik az oszlopcserélési szabályt, és ez könnyen elérhető a kezdeti, az oszlopra jellemző e_{offset} offszetérték cseréjével, melyet a lyuggatási minta használatának eltolása egy egyszerű képlettel határoz meg. Az offszetérték számításánál nem a véletlenszerű oszlopcsere utáni oszlopszámot, hanem az oszlopcsere előtti oszlopszámot használják fel, ami az oszlopcsere művelet leírásának a megfordításával kiszámítható. Ebből eredően az e_{offset} nem csak a lyuggatásnál, hanem az ismétlésnél is felhasználható. Ilyen módon az ismétlési bitek is egyenletesebben helyezhetők el.

Összefoglalás gyanánt a továbbiakban még egyszer megmutatjuk, hogy az eddig javasolt megoldások, azaz a javasolt lyuggatási/ismétlési minták továbbra sem optimálisak minden esetben.

A 2. ábra kapcsán egy példaszerű eset vizsgálatával - ahol a 2 réteg (Layer 2) egy 160 bites átviteli blokkot továbbít egy 80 ms átviteli ciklusú átviteli csatornán amellet a feltétel mellett, hogy minden keretben négy bitet kell lyuggatni - megmutattuk, hogy nyolc szomszédos bit kerül lyuggatásra, ami nyilvánvalóan nem kívánatos. A fent megnevezett első célkitűzésnek így ez nem tesz eleget kielégítő mértékben.

A 3. ábra, illetve 4. ábra szerinti javaslat a lyuggatási minta minden egyes keretben történő eltolását javasolja. Láthatóan ez is szomszédos bitek lyuggatását eredményezte (43. és 44. bitek, illetve 91. és 92. bitek). A fent megnevezett első célkitűzésnek így ez nem tesz eleget kielégítő mértékben.

A 6. ábra szerinti javaslat eltolt lyuggatási minták használatát terjeszti elő az összesorolás után, ahol az oszlopokra jellemző eltolásokat az oszlopcserék figyelembevételével állítják elő. Így az adott példa során nem adódnak egymással szomszédos lyuggatott bitek.

Ennek ellenére a 6. ábra szerinti eljárás mellett is vannak olyan esetek, melyeknél a lyuggatási aránytól függően egymással szomszédos bitek kerülnek lyuggatásra. A 9. ábra példaképpen azt az esetet mutatja be, ahol $N_i=16$, $N_c=14$, $m_1=4$, $m_2=14$, $k=1, \dots, 7$ és $K=8$. A 9 és 10. ábrákon az egyszerűség kedvéért csak az összesorolás előtti mező van feltüntetve, ahol ugyanakkor az összesorolás utáni lyuggatott bithelyek vastag szedéssel vannak ábrázolva. Látható, hogy a 31. és 32. bitek, illetve a 95. és 96. bitek lyuggatásra kerülnek, ami nyilvánvalóan nem kívánatos. A fent megnevezett első célkitűzésnek így ez ismét nem tesz eleget kielégítő mértékben.

Amennyiben egyszerűen minden n . bitet az eredeti sorrend-

nek megfelelően lyuggatnánk az összesorolás után, úgy a második célkitűzésnek nem tudnánk mindig teljes mértékben eleget tenni. Vegyünk például egy 80 ms-os összesorolást (mint a 9. ábrán) egy 1:6-os lyuggatási aránnyal. Minden 6. bit lyuggatásával csak a 0., 2., 4. és 6. keretek lesznek lyuggatva, az 1., 3., 5. és 7. keretek pedig nem, ami természetesen nem kívánatos, és a második célkitűzéssel össze nem egyeztethető. Az első célkitűzésnek ezzel szemben kielégítő mértékben eleget tennénk.

Ebből kiindulva tehát az a feladat szolgál a találmány alapjául, hogy a technika állásának ezeket a hátrányait mérsékeljük.

Ezt a feladatot az egymástól független igénypontokban összefoglaltakkal oldjuk meg. A találmány továbbfejlesztései az alpontokból adódnak.

A találmány kiviteli példái csupán példa gyanánt vannak leírva a csatolt rajzokra való hivatkozással. Ezek a következőket ábrázolják:

1. ÁBRA: Leegyszerűsített folyamatábra egy összesorolóval sebesség-átalakítás előtt (a technika mai állása);
2. ÁBRA: összesorolási és lyuggatási minta keretenként négy bites lyuggatáshoz (a technika mai állása);
3. ÁBRA: összesorolás és eltolt lyuggatási minta keretenként négy bites lyuggatáshoz (a technika mai állása);
4. ÁBRA: összesorolás és eltolt lyuggatási minta 10 százalékos arányú lyuggatáshoz (a technika mai állása);
5. ÁBRA: átviteli csatornák leegyszerűsített rajza (a tech-

- nika mai állása);
6. ÁBRA: összesorolás és eltolt lyuggatási minta keretenként négy bites lyuggatáshoz (a technika mai állása);
 7. ÁBRA: egy mobil távközlési rendszer blokkdiagramja (a technika mai állása);
 8. ÁBRA: egy adatközlési elrendezés blokkdiagramja, mely utat képez a 7. ábrán látható távközlési hálózat egy előfizetői állomása és egy bázisállomása között (a technika mai állása);
 9. ÁBRA: lyuggatási minta eltolt lyuggatási mintához keretként két bites lyuggatásnál (a technika mai állása);
 10. ÁBRA: a lyuggatás optimalizálásának két célja közül az egyiknek az alapelvének a leegyszerűsített ábrázolása;
 11. ÁBRA: referencia-táblázat;
 12. ÁBRA: lyuggatási minta 20 százalékos arányú lyuggatáshoz;
 13. ÁBRA: lyuggatási minta 1:8 arányú lyuggatáshoz;
 14. ÁBRA: lyuggatási minta keretként változó lyuggatandó bitszám melletti lyuggatáshoz.

Ha az összesorolást követően egyszerűen minden n. bitet lyuggatnánk az összefűzés előtti, eredeti sorrendjüknek megfelelően, a második célkitűzésnek - mint azt a fentiek során tárgyaltuk - nem tennénk mindig eleget kielégítő mértékben.

A találmány egy kiviteli példája - eltérve az összesorolás előtti, eredeti sorrendnek megfelelő egyenletes lyuggatás-

tól - a lyuggatási távolság legalább egyszeri, szükség szerint többszöri megváltoztatásával a fent megnevezett mindkét célkitűzésnek kielégítő mértékben megfelelné, mivel megelőzi, hogy egyes oszlopokat előnyben részesítsünk, miközben más oszlopokat pedig egyáltalán nem lyuggatunk. Ez a 10. ábrán látható. A vékonyan körvonalazott P6 vízszintes nyilak egy 6-os lyuggatási távolságot jelölnek, illetve a P5 függőleges nyíl egy 5-ös lyuggatási távolságot jelöl annak megakadályozására, hogy az első oszlopot kétszer lyuggassuk. Miután minden oszlopot egyszer lyuggattunk, a mintát eltoljuk 6 sorral lefelé a függőleges nyilak mentén a további lyuggatandó bitek meghatározására. Ez láthatóan megfelel minden oszlop minden 6. bitjének lyuggatásának, azaz egy szabványos sebesség-átalakító algoritmus használatának a különböző oszlopokban a különböző lyuggatási minták egymáshoz képest történő eltolásához.

A továbbiakban ezt az eljárást képletek segítségével ismertetjük:

Legyen N_i a bitek száma egy keretben a sebesség-átalakítás előtt, N_c a bitek száma a sebesség-átalakítás után, m_j a lyuggatott/ismételt bitek indexe, k az oszlop- illetve keretszám az összesorolás után, K pedig az összesorolt oszlopok, illetve keretek száma. Főként az $N_i < N_c$ esetet, illetve a lyuggatást kell tekintenünk, a képletek ugyanakkor ismétléshez is felhasználhatók.

A fenti példában $N_i=20$, $N_c=16$, $m_1=4$, $m_2=9$, $m_3=14$, $m_4=19$, $k=1, \dots, 7$, ahol k az oszlop- illetve keretszámot jelöli az összesorolás után, valamint $K=8$. A magyarázatokat az előttük álló "--" jelöli. A lyuggatási vagy ismétlési minta k keretekre való alkalmazásának $V(k) = S(k) + T(k) * Q$ eltolása a következő el-

járással állítható elő:

```
-- Az átlagos lyuggatási távolság számítása
q:=  $\lfloor N_c / (N_i - N_c) \rfloor \bmod K$  -- ahol a  $\lfloor \rfloor$  a lefele kerekítést, a  $\|$ 
pedig az abszolút értéket jelöli.
Q:=  $\lfloor N_c / (N_i - N_c) \rfloor \div K$ 
if q páros -- kivétel kezelés:
    then q = q - lcd(q, K)/K -- ahol az lcd(q, K) q és K leg-
nagyobb közös osztóját jelenti.
    -- Megjegyezzük, hogy az lcd könnyen kiszámítható bitmű-
veletekkel, mivel K kettő hatványa.
    -- Ugyanezen oknál fogva q-val könnyen végezhetünk fix-
pontos bináris aritmetikai számításokat (vagy egész számú
aritmetikai számításokat, illetve néhány eltolási műveletet).
endif

-- S és T kiszámítása; S a sorok eltolását állítja elő
mod K-val, T pedig az eltolási értéket div K-val; S tehát q-
hoz viszonyítva adja meg a sorok eltolását (azaz mod K), míg T
Q-hoz viszonyítva (azaz div K) adja meg az eltolási értéket;
for i = 0 to K-1
     $S(R_K(\lceil i * q \rceil \bmod K)) = (\lceil i * q \rceil \div K)$  -- ahol a  $\lceil \rceil$  a fel-
fele kerekítést jelenti.
     $T((R_K(\lceil i * q \rceil \bmod K)) = i$  --  $R_K(k)$  megfordítja az összeso-
rolót,
end for
```

Egy tényleges megvalósításnál ezek a képletek a 11. ábrán bemutatott módon referencia-táblázat formájában valósíthatók meg. A táblázat ezen kívül tartalmazza az oszlopcsere $R_K(k)$ -

val elért átképzésének korábban ismertetett eredményét. S-t egy további kiviteli lehetőségként nyilván T-ból is kiszámíthatjuk.

Ezt követően az e_{offset} a következő módon számítható ki:

$$e_{\text{offset}}(k) = ((2 \cdot S) + (2 \cdot T \cdot Q + 1) \cdot y + 1) \bmod 2N_c$$

Ezután az $e_{\text{offset}}(k)$ -val előre inicializáljuk e -t az UMTS sebesség-átalakító eljárásában. Az e_{offset} -nek ez a megválasztása az oszlopok lyuggatási mintáinak egymáshoz képest történő eltolását eredményezi az $S + T \cdot Q$ eredmény alapján.

A továbbiak során egy leegyszerűsített ábrázolást mutatunk be, mely egyszerűen abból adódik, hogy q , illetve Q számítását a K -val való osztás maradékánál, illetve a K többszörösénél nem külön-külön, hanem a két részhez összevonva végezzük. Hasonlóképpen S és T nem számolható külön-külön q -hoz és Q -hoz, hanem csak összevonva. A $q + K \cdot Q \rightarrow q$, illetve $S + Q \cdot T \rightarrow S$ behelyettesítés a következő, a korábban megadott eljárásnak megfelelő leírást eredményezi, ahol is a $V(k)$ eltolásra $V(k) = S(k)$ érvényes. Az megvalósítás részleteitől függően egyik-másik számítási módszer (vagy további azzal egyenértékű módszerek) olcsóbban is elvégezhető.

-- Az átlagos lyuggatási távolság számítása

$q := \lfloor N_c / \lfloor N_i - N_c \rfloor \rfloor$ -- ahol a $\lfloor \rfloor$ a lefele kerekítést, a $||$ pedig az abszolút értéket jelöli.

if q páros -- kivétel kezelés:

then $q = q - \text{lcd}(q, K) / K$ -- ahol az $\text{lcd}(q, K)$ q és K legnagyobb közös osztóját jelenti.

-- Megjegyezzük, hogy az lcd könnyen kiszámítható bitműveletekkel, mivel K kettő hatványa.



-- Ugyanezen oknál fogva q -val könnyen végezhetünk fix-pontos bináris aritmetikai számításokat (vagy egész számú aritmetikai számításokat, illetve néhány eltolási műveletet).

endif

-- $S(k)$, a sorok eltolásának kiszámítása;

for $i = 0$ to $K-1$

$S(R_k(\lceil i * q \rceil \bmod K)) = (\lceil i * q \rceil \div K)$ -- ahol a $\lceil \rceil$ a felfele kerekítést jelenti.

-- $R_k(k)$ megfordítja az összesorolót,

end for

Ezt követően az e_{offset} a következő módon számítható ki:

$$e_{\text{offset}}(k) = ((2 * S) + y + 1) \bmod 2Nc$$

Ezután az $e_{\text{offset}}(k)$ -val előre inicializáljuk e -t a sebességátalakító eljárásban.

Amennyiben a lyuggatási arány egy páratlan szám törtrésze, mint pl. 1:5 vagy 1:9, ez az eljárás a két célkitűzésből az egyik tekintetében optimális lyuggatási mintát hoz létre, melyet a sebességátalakító eljárás alatt a lyuggatásnál, közvetlenül az összesorolás előtt alkalmazunk. Egyéb esetekben szomszédos biteket soha nem lyuggatunk, azonban az egyik távolság két szomszédos bit között akár $\text{lcd}(q, K)+1$ -el nagyobb is lehet a többinél. Ez az eljárás bitismétlések esetében szintén használható. Noha a szomszédos bitek ismétlése nincs akkora hatással a hibajavító kódok teljesítményére, mint a szomszédos bitek lyuggatása, mégis előnyös, ha az ismételt biteket a lehető legegyszerűsebben osztjuk el.

Ennek az eljárásnak az alapvető célkitűzése az, hogy egyenletes távolságot hozzunk létre az eredeti sorrendben a

lyuggatott bitek között, azonban figyelembe vesszük azt a megszorítást, hogy a különböző keretekben ugyanolyan mennyiségű bitet kell lyuggatni. Ezt úgy érjük el, hogy a lyuggatási távolságot meghatározott esetekben 1-el csökkentjük. A bemutatott eljárás így optimális, ugyanis a távolságot sohasem csökkentjük 1-nél nagyobb számmal, és csak szükség szerinti gyakorisággal csökkentjük. Ez a fent említett megszorítások mellett a lehető legjobb lyuggatási mintát eredményezi.

A következő példa a 12. ábra kapcsán az 1:5-ös lyuggatási arány melletti lyuggatást ábrázolja. Az optimalizált algoritmus nem csak elkerüli a szomszédos bitek lyuggatását, hanem emellett az eredeti sorrendnek megfelelően egyenlő távolságra is osztja el a biteket. Valójában ugyanazokat a tulajdonságokat kapjuk, mintha a lyuggatást közvetlenül a kódolás után, az összesorolás előtt végeztük volna el. Az 1:5-ös lyuggatásnak ebben az esetben, és általában véve minden esetben, amikor a lyuggatási arány $1:q$ tört alakban felírható, ahol is q egész szám, valamint q -nak és K -nak (a keretek számának) nincs közös osztója, elmondhatjuk, hogy optimális lyuggatási minta jön létre annak ellenére, hogy a lyuggatást az első összesoroló után hajtjuk végre. Ez a lyuggatási minta minden q . bit lyuggatását eredményezi, akár csak egy olyan optimális lyuggatási minta, melyet közvetlenül a kódolás után, az összesorolás előtt alkalmaznánk.

A 13. ábra kapcsán egy 1:8-as lyuggatási arány melletti lyuggatást tekintünk meg. A szomszédos bitek lyuggatását ismét elkerüljük. Ebben az esetben nincs lehetőség egyenlő távolságú lyuggatás megvalósítására, ugyanis akkor az egyik keret minden bitjét lyuggatni kellene, ami a második célkitűzésre való te-

kintettel teljességgel elfogadhatatlan. Ebben az esetben a legtöbb szomszédos bit között a távolság 7 (csak 1-el kevesebb, mint egy optimális eloszlásnál). Emiatt viszont némelyik távolság nagyobb (minden nyolcadik).

Amikor a bemenő bitek N_i száma nem osztható K -val, a sebesség-átalakítás változhat az átvitel időtartama alatt. Az utolsó keretek olyankor egy bittel kevesebbet visznek át, mint az elsők, ugyanakkor valamivel kisebb lyuggatási aránnyal rendelkeznek. Erre az esetre a találmány egy kiviteli példája a lyuggatási minta változatlanul hagyását irányozza elő az utolsó sorokra. Ehelyett ugyanazt a lyuggató algoritmust használjuk, mint amit az első oszlophoz használtunk, azonban az utolsó lyuggatás elvégzése nélkül. A 14. ábra kapcsán azt a példát vesszük, amikor úgy kell lyuggatni 125 bemeneti bitet, hogy 104 bit maradjon hátra, melyeket nyolc keretbe sorolunk össze. Az utolsó két oszlop egy-egy bemeneti bittel kevesebbet használ, mint az elsők; a két utolsó oszlopban az utolsó lyuggatás kihagyásával mindegyik oszlop 13 bitet tartalmaz.

Az itt előterjesztett eljárással a fent megnevezett célkitűzések értelmében optimális lyuggatási minták adhatóak meg, amikor a sebesség-átalakítást az első összefűzés után véghezvük. Az eljárás egyszerű, kevés számítási kapacitást igényel, valamint keretenként csak egyszer kell elvégezni, nem pedig bitenként. Az eljárás nem korlátozódik rádiós átviteli rendszerekre.

73.180/MK

SZABADALMI IGÉNYPONTOK

1. Eljárás adatsebesség átalakítására, **azzal jellemezve**, hogy az átvitelre kerülő bitek által alkotott adatokat egy első összesorolóval K keretek halmazára bontjuk, valamint az összesorolást követően úgy hajtunk végre lyuggatási, vagy ismétlési eljárást a sebesség-átalakításhoz, hogy a keretenként megegyező számú bitek lyuggatásánál, vagy ismétlésénél megváltoztatjuk a lyuggatott, vagy ismételt bitek közötti távolságot a bitek első összesoroló előtti sorrendjéhez viszonyítva, ahol is a távolságra a következő összefüggés érvényes:

$q - 1 \leq \text{távolság} \leq q + \text{lcd}(q, K) + 1$, ahol

$q := \lfloor N_c / (N_i - N_c) \rfloor \bmod K$, ahol a $\lfloor \rfloor$ a lefele kerekítést, a $\|$ pedig az abszolút értéket jelöli, valamint $N_i :=$ bitek száma a sebesség-átalakítás után, illetve $N_c :=$ bitek száma a sebesség-átalakítás előtt;

$\text{lcd}(q, K) := q$ és K legnagyobb közös osztója.

2. Az 1. igénypont szerinti eljárás, **azzal jellemezve**, hogy az alábbi összefüggés akkor is érvényes, ha a lyuggatási arány, vagy az ismétlési arány $1/K$ -val egyenlő:

$q - 1 \leq \text{távolság} \leq q + \text{lcd}(q, K) + 1$, ahol

$q := \lfloor N_c / (N_i - N_c) \rfloor \bmod K$, ahol a $\lfloor \rfloor$ a lefele kerekítést, a $\|$ pedig az abszolút értéket jelöli, valamint $N_i :=$ bitek száma a sebesség-átalakítás után, illetve $N_c :=$ bitek száma a sebesség-átalakítás előtt;

$\text{lcd}(q, K) := q$ és K legnagyobb közös osztója.

3. Az 1. vagy 2. igénypontok bármelyike szerinti eljárás, **azzal jellemezve**, hogy a bitek első összesoroló előtti sorrendjére nézve az egymást követő lyuggatott, vagy ismételt biteket egy olyan eljárással állítjuk elő, mely a következő lépésekből áll:

a) Lyuggatás, vagy ismétlés a bitek első összesoroló előtti sorrendjéhez viszonyított távolsággal a szomszédos lyuggatott, vagy ismételt bitek között, melyet q határoz meg;

b) A szomszédos lyuggatott, vagy ismételt bitek közötti távolságot $q-1$ -re, vagy $q+1$ -re változtatjuk, amennyiben egy keretben a lyuggatott, vagy ismételt bitek száma 1-nél nagyobb számmal haladja meg egy másik keretben található lyuggatott, vagy ismételt bitek számát, továbbá amennyiben a lyuggatást, vagy ismétlést a bitek első összesoroló előtti sorrendjéhez viszonyított távolsággal végezzük, melyet q határoz meg;

c) amennyiben vannak még további lyuggatandó, vagy ismétlendő bitek, az a) lépéssel folytatjuk.

4. Az 1-3. igénypontok bármelyike szerinti eljárás, **azzal jellemezve**, hogy a lyuggatási, vagy ismétlési eljárást úgy hajtjuk végre, hogy az egy keretben használt lyuggatási, vagy ismétlési mintát eltolva a halmaz további kereteiben is felhasználjuk.

5. A 4. igénypont szerinti eljárás, **azzal jellemezve**, hogy a $V(k) = S(k) + T(k) * Q$ eltolásnál a lyuggatási, illetve ismétlési mintának a k keretre való alkalmazását a következő lépésekből kapjuk:

- Kiszámítjuk az átlagos q lyuggatási távolságot, ahol

- $q := \lfloor N_c / \lfloor N_i - N_c \rfloor \rfloor \bmod K$, ahol a $\lfloor \rfloor$ a lefele kerekítést, a $||$ pedig az abszolút értéket jelöli, valamint
- $N_i :=$ bitek száma a sebesség-átalakítás után,
- $N_c :=$ bitek száma a sebesség-átalakítás előtt;
- Q kiszámítása, ahol $Q := \lfloor N_c / \lfloor N_i - N_c \rfloor \rfloor \div K$
 - ha q páros, akkor elvégezzük a $q := q - \text{lcd}(q, K)/K$ értékátadást, ahol az $\text{lcd}(q, K)$ q és K legnagyobb közös osztóját jelenti;
 - Egy i változót nullázunk;
 - Az alábbi lépéseket ismételjük mindaddig, amíg $i \leq K-1$:
 - $S(R_k(\lceil i * q \rceil \bmod K)) = \lceil i * q \rceil \div K$, ahol a $\lceil \rceil$ a felfele kerekítést jelenti;
 - $T((R_k(\lceil i * q \rceil \bmod K)) = i$, ahol $R_k(k)$ megfordítja az összesorolót;
 - i -t növeljük 1-el.

6. A 4. igénypont szerinti eljárás, **azzal jellemezve**, hogy a $V(k) = S(k) + T(k) * Q$ eltolásnál a lyuggatási, illetve ismétlési mintának a k keretre való alkalmazása a következő lépésekből adódik:

- Kiszámítjuk az átlagos q lyuggatási távolságot, ahol
- $q := \lfloor N_c / \lfloor N_i - N_c \rfloor \rfloor \bmod K$, ahol a $\lfloor \rfloor$ a lefele kerekítést, a $||$ pedig az abszolút értéket jelöli, valamint
- $N_i :=$ bitek száma a sebesség-átalakítás után,
- $N_c :=$ bitek száma a sebesség-átalakítás előtt;
- ha q páros, akkor elvégezzük a $q := q - \text{lcd}(q, K)/K$ értékátadást, ahol az $\text{lcd}(q, K)$ q és K legnagyobb közös osztóját jelenti;
-
- Egy i változót nullázunk;
-

- Az alábbi lépéseket ismételjük mindaddig, amíg $i \leq K-1$:
 - $S(R_K(\lceil i * q \rceil \bmod K)) = (\lceil i * q \rceil \text{ div } K)$, ahol a $\lceil \rceil$ a felfele kerekítést jelenti;
 - $R_K(k)$, ahol $R_K(k)$ megfordítja az összesorolót;
 - i -t növeljük 1-el.

7. Az 1-6. igénypontok bármelyike szerinti eljárás, **azzal jellemezve**, hogy a lyuggatandó, vagy ismétlendő biteket egy olyan eljárással állítjuk elő, mely a következő lépésekből áll:

a) Meghatározzuk az átlagos lyuggatási távolság q egész részét:

$q := \lfloor N_c / (N_i - N_c) \rfloor \bmod K$, ahol a $\lfloor \rfloor$ a lefele kerekítést, a $||$ pedig az abszolút értéket jelöli, valamint

N_i := bitek száma a sebesség-átalakítás után,

N_c := bitek száma a sebesség-átalakítás előtt;

b) Egy lyuggatandó, vagy ismétlendő bit kiválasztása egy első keretben;

c) A következő keretben a következő lyuggatandó, vagy ismétlendő bit kiválasztása az előző keretbeli legutolsó lyuggatandó, vagy ismétlendő bitből kiindulva, azáltal, hogy ezzel az utolsó lyuggatandó, vagy ismétlendő bittel kezdve mindig az eredeti sorrendhez viszonyított q távolságra eső következő bitet kiválasztjuk, mindaddig, amíg ez nem eredményezi egy keret kétszeres lyuggatását, vagy ismétlését, máskülönben egy $q-1$, vagy $q+1$ távolságra eső bitet választunk ki lyuggatásra, vagy ismétlésre.

d) Addig ismételjük a c) lépést, amíg minden oszlopot egyszer lyuggatunk, illetve ismétlünk.

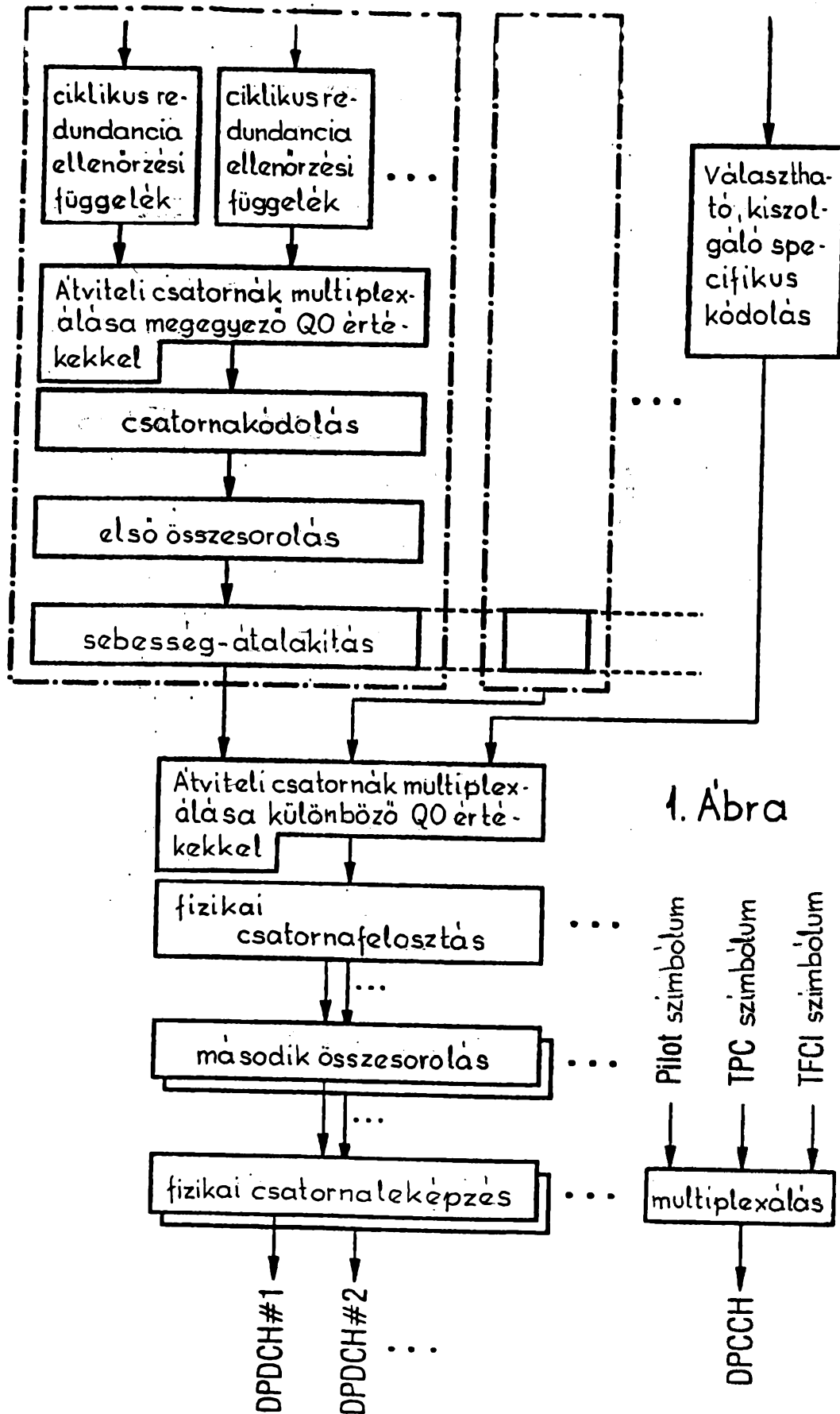
8. A 7. igénypont szerinti eljárás, **azzal jellemezve**, hogy egy első keret bitjeit egy előre meghatározott lyuggatási minta, vagy ismétlési minta alapján lyuggatjuk, vagy ismételjük, valamint a lyuggatási mintát, vagy az ismétlési mintát eltolva alkalmazzuk a további keretekre a további lyuggatandó, vagy ismétlendő bitek kiválasztására, ahol a lyuggatási minta, vagy az ismétlési minta egy további keretre való alkalmazása megfelel a következő keret 7. igénypont c) lépésében kiválasztott bitje eltolásának a 7. igénypont b) lépésében kiválasztott bithez képest.

9. Adatsebesség-átalakító berendezés, pontosítva feldolgozóegység, **azzal jellemezve**, hogy képes az 1-8. igénypontok bármelyike szerinti eljárás elvégzésére.

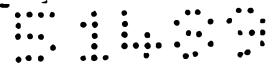
12. maj. Z. N. K. K.

A meghatalmazott:

J. K.



1. Ábra



KÖZZÉTÉTELI PÉLDÁNY

2/12

bitsorozat

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 ... 159

2. Ábra

soronkénti
feldolgozás
 $8[4[2 \times 2] \times 2]$

első
összesorolás

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87
88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103
104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119
120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135
136	137	138	139	140	141	142	143
144	145	146	147	148	149	150	151
152	153	154	155	156	157	158	159

0	4	2	6	1	5	3	7
8	12	10	14	9	13	11	15
16	20	18	22	17	21	19	23
24	28	26	30	25	29	27	31
32	36	34	38	33	37	35	39
40	44	42	46	41	45	43	47
48	52	50	54	49	53	51	55
56	60	58	62	57	61	59	63
64	68	66	70	65	69	67	71
72	76	74	78	73	77	75	79
80	84	82	86	81	85	83	87
88	92	90	94	89	93	91	95
96	100	98	102	97	101	99	103
104	108	106	110	105	109	107	111
112	116	114	118	113	117	115	119
120	124	122	126	121	125	123	127
128	132	130	134	129	133	131	135
136	140	138	142	137	141	139	143
144	148	146	150	145	149	147	151
152	156	154	158	153	157	155	159

radiókeret #1

radiókeret #2

...
radiókeret #8

INFORMATIKAI
FELADAT

3/12

bitsorozat

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 ... 159

3. Ábra

soronkénti
feldolgozás
 $8[4[2 \times 2] \times 2]$

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87
88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103
104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119
120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135
136	137	138	139	140	141	142	143
144	145	146	147	148	149	150	151
152	153	154	155	156	157	158	159

0	4	2	6	1	5	3	7
8	12	10	14	9	13	11	15
16	20	18	22	17	21	19	23
24	28	26	30	25	29	27	31
32	36	34	38	33	37	35	39
40	44	42	46	41	45	43	47
48	52	50	54	49	53	51	55
56	60	58	62	57	61	59	63
64	68	66	70	65	69	67	71
72	76	74	78	73	77	75	79
80	84	82	86	81	85	83	87
88	92	90	94	89	93	91	95
96	100	98	102	97	101	99	103
104	108	106	110	105	109	107	111
112	116	114	118	113	117	115	119
120	124	122	126	121	125	123	127
128	132	130	134	129	133	131	135
136	140	138	142	137	141	139	143
144	148	146	150	145	149	147	151
152	156	154	158	153	157	155	159

radiókeret #1

radiókeret #2

radiókeret #8

ÖSSZEKÖLT MŰKÖDÉS

4/12

bitsorozat

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 ... 159

4. Ábra

soronkénti
feldolgozás
 $8[4[2 \times 2] \times 2]$

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87
88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103
104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119
120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135
136	137	138	139	140	141	142	143
144	145	146	147	148	149	150	151
152	153	154	155	156	157	158	159

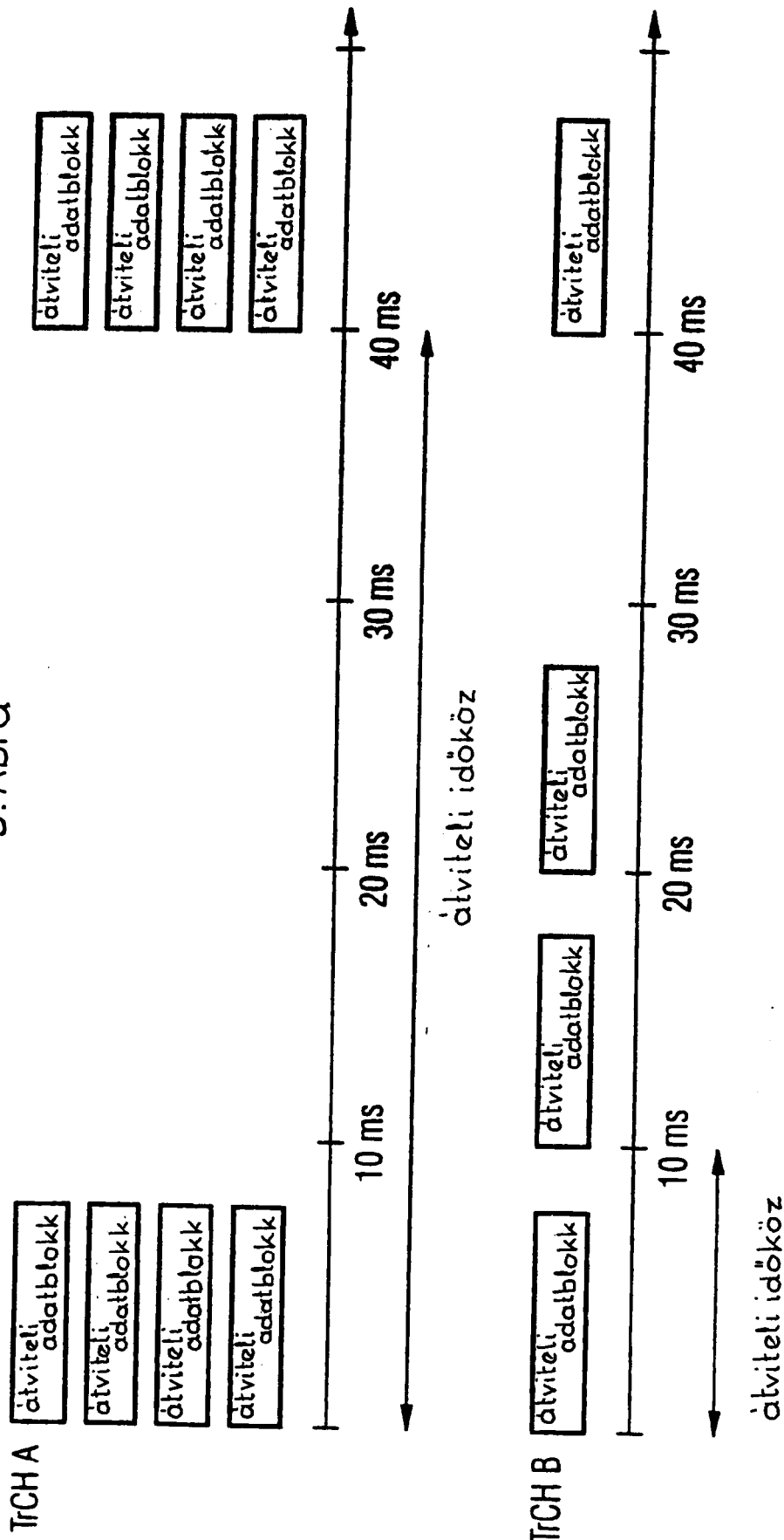
0	4	2	6	1	5	3	7
8	12	10	14	9	13	11	15
16	20	18	22	17	21	19	23
24	28	26	30	25	29	27	31
32	36	34	38	33	37	35	39
40	44	42	46	41	45	43	47
48	52	50	54	49	53	51	55
56	60	58	62	57	61	59	63
64	68	66	70	65	69	67	71
72	76	74	78	73	77	75	79
80	84	82	86	81	85	83	87
88	92	90	94	89	93	91	55
96	100	98	102	97	101	99	103
104	108	106	110	105	109	107	111
112	116	114	118	113	117	115	119
120	124	122	126	121	125	123	127
128	132	130	134	129	133	131	135
136	140	138	142	137	141	139	143
144	148	146	150	145	149	147	151
152	156	154	158	153	157	155	159

radiókeret #1

radiókeret #2

radiókeret #8

5. Ábra



REKONSTRUKCIÓ
MÉREK/ÁNY

6/12

Bemeneti bitsorozat
1. blokk összetar-
tozás

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 ...159

6. Ábra

lyuggatás
egyszerű el-
tolással

(véletlenszerű)
oszlopcsere,
oszlop rando-
mizálás

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87
88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103
104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119
120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135
136	137	138	139	140	141	142	143
144	145	146	147	148	149	150	151
152	153	154	155	156	157	158	159

0	4	2	6	1	5	3	7
8	12	10	14	9	13	11	15
16	20	18	22	17	21	19	23
24	28	26	30	25	29	27	31
32	36	34	38	33	37	35	39
40	44	42	46	41	45	43	47
48	52	50	54	49	53	51	55
56	60	58	62	57	61	59	63
64	68	66	70	65	69	67	71
72	76	74	78	73	77	75	79
80	84	82	86	81	85	83	87
88	92	90	94	89	93	91	95
96	100	98	102	97	101	99	103
104	108	106	110	105	109	107	111
112	116	114	118	113	117	115	119
120	124	122	126	121	125	123	127
128	132	130	134	129	133	131	135
136	140	138	142	137	141	139	143
144	148	146	150	145	149	147	151
152	156	154	158	153	157	155	159

keret #1 → keret #2 ... keret #8

KÖZTISZKELI
FELHÍVÁS

7/12

7. Ábra

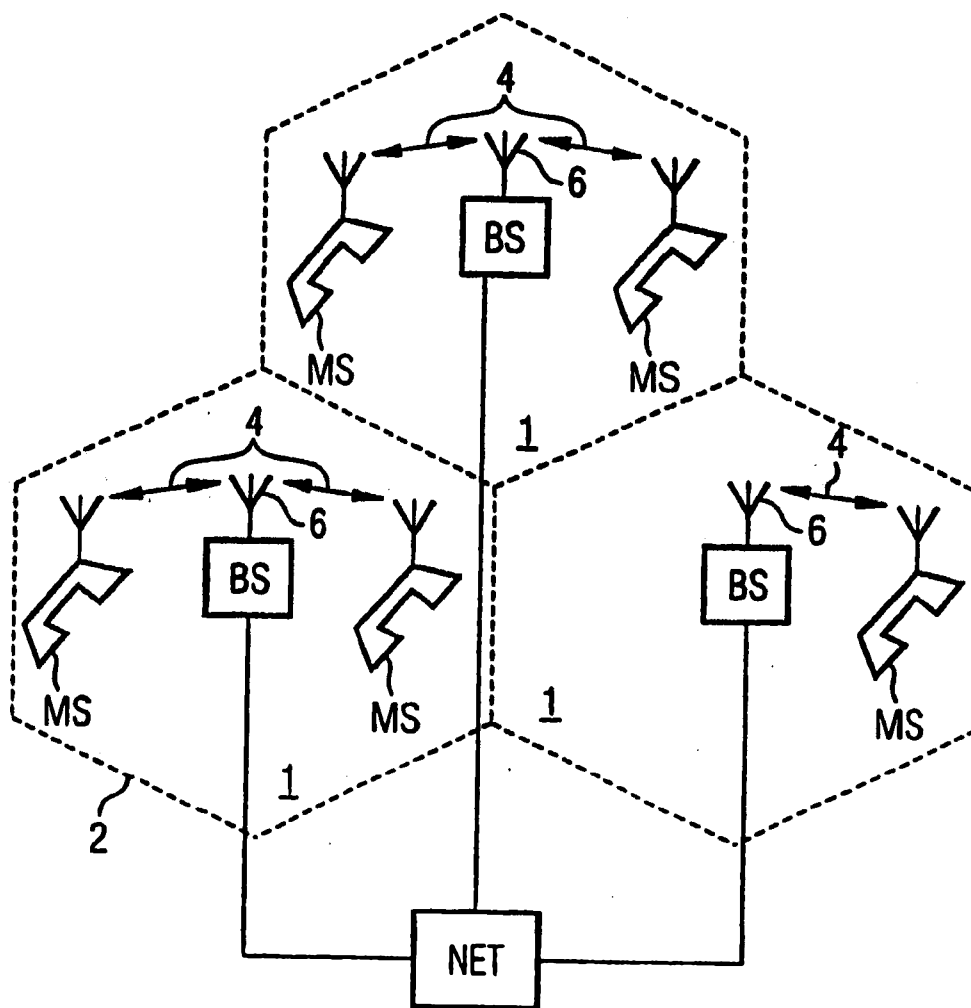
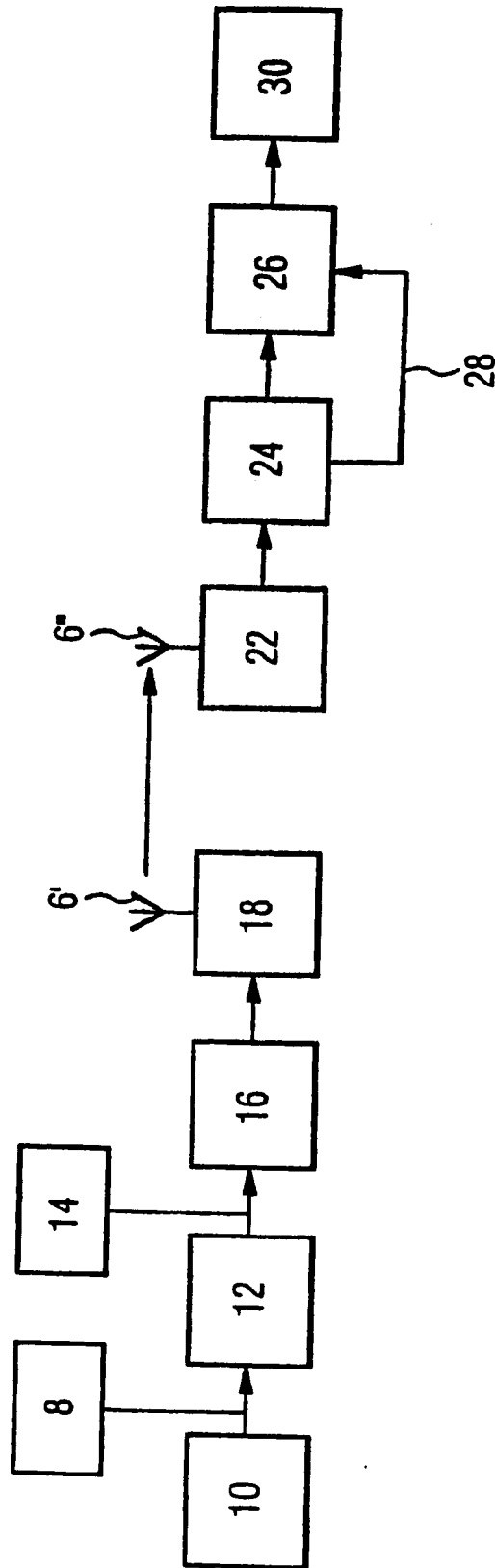


FIG. 1

8. Abra



10/12/2011
F.10.10.10

11. Abra

S:T	K	2		4				8							
		1	2	0	1	2	3	0	1	2	3	4	5	6	7
Q	k	0	0	1	0	1	2	3	0	0	0	4	0	2	7
	1	0:0	0:0	0:1	0:0	0:2	0:1	0:3	0:0	0:2	0:6	0:1	0:5	0:3	0:7
	2		0:0	1:1	0:0	0:1	1:3	0:2	0:0	0:2	0:3	1:5	1:7	1:6	0:4
	3				0:0	1:2	2:3	0:1	0:0	2:6	0:2	1:3	2:7	0:1	1:5
	4				0:0	1:2	2:3	0:1	0:0	2:5	1:4	3:7	2:6	1:3	0:2
	5								0:0	3:4	4:6	4:5	1:1	5:7	2:3
	6								0:0	1:2	2:3	0:1	5:7	4:6	2:4
	7								0:0	3:4	1:2	6:7	2:3	4:5	0:1
	8								0:0	3:4	1:2	6:7	2:3	4:5	0:1

12. Abra

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87
88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103
104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119
120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135
136	137	138	139	140	141	142	143
144	145	146	147	148	149	150	151
152	153	154	155	156	157	158	159

13. Abra

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87
88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103
104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119
120	121	122	123	124	125	126	127

0	8	16	24	32	40	48	56	64	72	80	88	96	104	112	120
1	9	17	25	33	41	49	57	65	73	81	89	97	105	113	121
2	10	18	26	34	42	50	58	66	74	82	90	98	106	114	122
3	11	19	27	35	43	51	59	67	75	83	91	99	107	115	123
4	12	20	28	36	44	52	60	68	76	84	92	100	108	116	124
5	13	21	29	37	45	53	61	69	77	85	93	101	109	117	125
6	14	22	30	38	46	54	62	70	78	86	94	102	110	118	
7	15	23	31	39	47	55	63	71	79	87	95	103	111	119	

																			A legjobb megoldás n bit lyuggatására
																			n+1 bit lyuggatása a fentimodan. egy bittel kiegészítve
																			n+1 bit lyuggatása optimalizált algoritmussal