Az előadó betegsége miatt az óra kissé szokatlan volt. A nebulók a **4.7**, **5.10** feladatok megoldásával folytatták a Kódok feladatgyűjtemény feldolgozását, majd egy esszét olvashattak a CD működésének matematikai alapjairól. Új házi feladat nincs, de korábbról megmaradt még a **hétszögminták**, **5.11**, az **Általános Hamming-kód** és az "**Ideális 1001-es**".

A 20. szakkör részletezett anyaga

4.7 Összehasonlítunk három kódot! Mindhárom kód kilenc szóból áll, mindegyik egy hárombetűs ABC-t használ.
$\boldsymbol{k}_1$) ***A legegyszerűbb***: A szavak kétbetűsek, a kilenc kódszó az összes kétbetűs szó.
$\boldsymbol{k}_2$) ***Mindent háromszor***: A szavak hatbetűsek és úgy készülnek, hogy a kétbetűs szót háromszor írjuk le egymás után.
$\boldsymbol{k}_3$) ***A sűrű***: A szavak négybetűsek, a 4.6 feladatban meghatározott 1-hiba javító tökéletes kódot alkotják.

Tegyük fel, hogy egy betű a kommunikáció során 10% eséllyel megváltozik és 90% eséllyel jól továbbítódik. $\boldsymbol{k}_2$ és $\boldsymbol{k}_3$ esetén a kiolvasott jelsorozatot mindig a tőle legkevesebb jelben eltérő kódszóra változtatjuk. Ha ez nem egyértelmű, akkor nem javítunk. Határozzuk meg a három esetben külön-külön, hogy mi az esélye, hogy egy szót úgy olvasunk ki, ahogy elküldték!

Megoldás
$\boldsymbol{k}_1$ esetén mindkét jelet jól kell kiolvasni, így $0{,}9^2 = 0{,}81$ alapján 81% az esély.
$\boldsymbol{k}_2$ esetén mindkét jel három példányából háromnak vagy kettőnek kell helyesen megérkeznie, tehát a számolás: $(0{,}9^3 + 3\cdot 0{,}1\cdot 0{,}9^2)^2 = 0{,}944784$, azaz 94,4784% az esély.
$\boldsymbol{k}_3$ esetén a négy jelből négynek vagy háromnak kell helyesnek lennie, így $0{,}9^4 + 4\cdot 0{,}1\cdot 0{,}9^3 = 0{,}9477$ alapján 94,77% az esély.

5.10 (***Dienes Péter javaslata***)
Hanyag Hugó az 1, 2, 3, ..., 16 számok egyikére gondolt. Egy-egy cetlire kell fölírni kérdéseinket, s mind odaadni neki, majd amikor ráér egyszerre mindegyikre válaszol fog. De lehet rá számítani, hogy az egyik választ elveszti mielőtt az eljutna hozzánk. Legalább hány kérdésre van így szükség ahhoz, hogy kitaláljuk a gondolt számot?

Megoldás
Öt kérdés nyilván szükséges, de ennyi elég is. Öt megfelelő kérdés megkapható az 5.9 feladatra adott bármelyik konstrukció mintájára (lásd a 18. szakkör anyagát).

Most 16 db olyan 5 hosszúságú 0 - 1 sorozatot keresünk, amelyek közül bármelyik kettő legalább két helyen eltér egymástól. Ha ugyanis az egyik kérdésre nem érkezik válasz, azaz a sorozatokban az egyik helyen álló elem eltűnik, akkor továbbra is 16 különböző sorozatot kell kapjunk. Ez felel meg annak, hogy a négy megmaradt kérdésre adott válaszból minden esetben kitalálható a gondolt szám. Az alábbi táblázat oszlopaiba 16 megfelelő sorozatot írtunk be.

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

Látható, hogy az utolsó sor az előzőekhez hozzátett paritásjelző-bit. Az öt kérdés közül az első négy megegyezik az 5.8 b) feladat megoldásaként konstruált négy kérdéssel, az ötödik
pedig azokra a számokra kérdez rá, amelyekre addig páratlan sokszor kérdeztünk rá. Így összességében minden számra páros sokszor kérdezünk rá, tehát ha mind az öt kérdésre kapnánk választ, akkor páros sok "Igen"-t hallanánk. Ezért, ha egy válasz hiányzik, akkor az már kitalálható a többi négyből.

Megjegyzések
1. Az 5.9 feladatra (hazudós barkochba) adott II. konstrukció szellemében fenti kérdéseink így is írhatók:
"$\boldsymbol{x}_8 = 0$?"; "$\boldsymbol{x}_4 = 0$?"; "$\boldsymbol{x}_2 = 0$?"; "$\boldsymbol{x}_1 = 0$?"; "$\boldsymbol{x}_8 + \boldsymbol{x}_4 + \boldsymbol{x}_2 + \boldsymbol{x}_1 = 0$?".

2. Táblázatunk oszlopai most egy 16 kódszóból álló ötbetűs 1-törlés javító kódot alkotnak.

A CD-ről szóló esszé előtt felelevenítjük a **házi feladatokat:**

"Ideális 1001-es" Oldjuk meg az "ideális 101-es" feladatot 101 helyett mindenütt 1001-gyel!

Hétszögminták Ebben a feladatban egy szabályos hétszög csúcsaira írunk 0-t vagy 1-et. Összesen $2^7=128$ ilyen kitöltés van. A kitöltések egy $\boldsymbol{I}_7$ részhalmaza "7-szögre ideális",
1. ha $\boldsymbol{h}\in\boldsymbol{I}_7 \Rightarrow \boldsymbol{h}$ bármely ($\boldsymbol{n}\cdot 360°/7$-kal való) elforgatottja is $\boldsymbol{I}_7$-ben van.
2. ha bármely két $\boldsymbol{I}_7$-beli kitöltés csúcsonként és mod 2 számított összege is $\boldsymbol{I}_7$-ben van.
A kitöltéseknek hány "7-szögre ideális" részhalmaza van?

5.11 (***Juhász Istvántól és Szegedy Balázstól is hallottam***)
Egy kém az ellenséges ország televíziójánál dolgozik. Esténként alkalma van az adásba kerülő 8×8-as fekete fehér tábla egyetlen mezőjének színét megváltoztatni. Nem feltétlenül szükséges változtatnia. Sajnos sohasem tudja előre, hogy milyen mintázatú lesz a 64 mező, amikor eléje kerül. Hányféle információt tud így küldeni a TV-n keresztül?

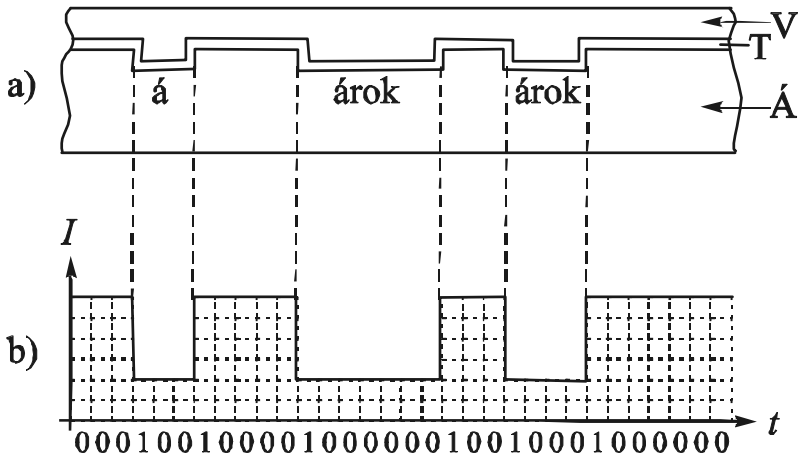
A CD matematikája

Ennek a fejezetnek az anyagát nagyrészt a J. H. van Lint, The Mathematics of the Compact Disc, ***DMV-Mitteilungen*** 4/98, 25--29. és a Compact disc digital audio, ***Philips technical review***, Vol. 40, 1982, No. 6. művek alapján állítottuk össze.

A pálya

A CD lejátszót a Philips és a Sony cég fejlesztette ki.
Maga a lemez egyetlen egy pályából áll, ami 5 km hosszú. Ezen binárisan van tárolva az információ. "Síkságok" és "árkok" követik egymást, de az árok alja is sík. Az 1-es bitet a változás, az árok széle, a 0-át az egységnyi hosszúságú vízszintes rész jelöli, amely lehet síkságon vagy az árok alján is.

A kódolt üzenetnek nagyon sok feltételt tudnia kell:



a) A CD lemez keresztmetszete a pálya irányában
V: védőréteg; T: tükrözőfelület; Á: átlátszó réteg.
b) ***I***: Az optikai olvasófej által érzékelt jel intenzitása az idő (***t***) függvényében.

1) A digitalizálás ellenére az emberi fül számára hűen adja vissza a felvett zenét;
2) Az elszórt véletlenszerű bithibákat képes legyen kijavítani;
3) A csomóban keletkező hibák --- ujjlenyomatok, karcolások, anyaghibák --- korrigálását is megoldja;
4) A hallgató számára információt tudjon nyújtani, hogy hol tart;
5) A lézeres leolvasószerkezet hatékonyan olvasni tudja az információt;
6) A szerkezet működése ne keltsen az emberi fül számára hallható zajt (a lejátszandó zenén kívül).

Audiobit

A zene felvételekor az érzékelő bizonyos időnként "mintát vesz", azaz egy-egy pillanatban megméri milyen a beérkező hullám nyomása. Lényegében ez az érték a tárolt információ.
A felvétel lejátszásakor keletkező hang a pillanatnyi minták visszaadásából áll össze, így természetesen nem pont ugyanaz, mint amit fel szándékoztak venni. A mintavételezés gyakoriságától függően, a zenei hang bizonyos hullámhosszú összetevői hűen hallhatók.
Egy fizikai elv, Nyquist tétele, kapcsolatot teremt a mintavételezés gyakorisága, és az abból hűen rekonstruálható hanghullámok frekvenciája között. Ennek a tételnek az alapján és abból kiindulva, hogy az emberi fül kb. 20 000 Hz-ig hall jól másodpercenként 44 100-nek adódik a szükséges mintavételezési gyakoriság.

Minden minta eredményét 16 bitté alakítja a rendszer. A felvétel általában sztereó, így máris 32 bit - úgynevezett ***audiobit*** - tartozik egyetlen pillanatnyi mintához. A 32 bit 4 byte-ba van rendezve (egy byte az nyolc egymást követő bit).

2-hiba javító kód

Ha nem volnának hibajavító kódok hozzátéve a felvett audiobitekhez, akkor a CD lemez lejátszhatatlan minőségű lenne. A leggyakoribb "károkozók" az ujjlenyomatok, az apró karcolások, a lemezben vagy rajta lévő idegen anyagok, a műanyagban óhatatlanul meglévő légbuborékok, az eredeti felületi egyenetlenségek és a felvételkor vétett pontatlanságok.

Tegyük fel, hogy nem alkalmazunk hibajavító kódot. Ha a byte-hiba valószínűsége csupán 0,01% lenne, akkor az egy mintavételnek megfelelő információban, azaz 4 byte-ban megbúvó hiba esélye $1-0{,}9999^4 \approx 0{,}9996$ alapján kb. 0,04%-os lenne. Ez másodpercenként több mint 17 hibás lejátszott hangot okozna.

Tegyük fel először, hogy csak 1-hiba javítást alkalmazunk. A CD-n használt Reed-Solomon-féle kódolási eljárás egy-egy byte-ot egyben, egyetlen "számként" kezel. Ennek részleteibe most nem megyünk bele, inkább egyszerűsítésként képzeljük azt, mintha egy byte-on csak 31-féle információ lenne szállítható. Feleltessünk meg minden lehetőségnek egy 0 és 30 közötti számot és számoljunk velük mod 31.[1]

Az 1-hiba javítás megoldható úgy, hogy minden mintavételnek megfelelő byte-négyeshez még 2 byte-ot veszünk hozzá. Az R-S kódolás ezt az 1.1 feladathoz kicsit hasonlóan oldja meg. Az $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, $\boldsymbol{x}_3$, $\boldsymbol{x}_4$ eredeti byte-ok mellé az $\boldsymbol{x}_5$ és $\boldsymbol{x}_6$ ellenőrző biteket úgy kell választani, hogy teljesüljön az alábbi két összefüggés:

$$\boldsymbol{x}_1 + \boldsymbol{x}_2 + \boldsymbol{x}_3 + \boldsymbol{x}_4 + \boldsymbol{x}_5 + \boldsymbol{x}_6 \equiv 0 \pmod{31},$$
$$\boldsymbol{x}_1 + 2\boldsymbol{x}_2 + 3\boldsymbol{x}_3 + 4\boldsymbol{x}_4 + 5\boldsymbol{x}_5 + 6\boldsymbol{x}_6 \equiv 0 \pmod{31}.$$

Ha a dekódolásnál kiderül, hogy nem teljesül ez a két összefüggés, mondjuk a felső bal oldali kifejezés értéke 2, az alsóé pedig 41, azaz 10 (mod 31), akkor - abban bízva, hogy csak 1 hiba történt - kiszámolhatjuk a kódolt információt. Tudjuk, hogy a hiba értéke 2, tehát azt kell megkeresnünk, hogy melyik 1 és 26 közötti szám kétszerese ad 10 maradékot 31-gyel osztva. Ez a szám az 5, tehát az $\boldsymbol{x}_5$ byte hibás, 2-vel nagyobb a kelleténél (mod 31).

Nézzük, most mennyi hibára számíthatunk! Sajnos azzal, hogy megnöveltük a bitsorozat hosszát vagy kevesebb zenét kell rögzítenünk a lemezen, vagy kisebb jeleket sűrűbben kell írnunk a CD-re, ami növeli a hiba valószínűségét. Az első lehetőséget ne fogadjuk el. Tegyük fel, hogy a byte-hiba valószínűsége 0,02%-ra nő. Annak esélye, hogy egy mintavételnek megfelelő, most már 6 byte-ból álló adatsor hibás lesz, tehát 2 hiba kerül bele:

$$1-0{,}9998^6 - 6\cdot 0{,}0002\cdot 0{,}9998^5 \approx 0{,}0000006.$$

Ez másodpercenként csak átlagosan kb. 0,0264 hibás lejátszott hangot okozna, ami lényegesen jobb, mintha nem alkalmaztunk volna kódolást.

Egy, az előzőnél jobb, két hibát javító kóddal még sokkal jobb eredményt érhetünk el. Most byte-négyesek helyett byte-ok nyolcas csoportjaira osztjuk a kódolandó üzenetet, és ezekhez négy további byte-ot illesztünk. Nevezetesen, az eredeti byte-oknak megfelelő 8 szám legyen
$\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_8$, a négy ellenőrző szám pedig $\boldsymbol{x}_9, \boldsymbol{x}_{10}, \boldsymbol{x}_{11}, \boldsymbol{x}_{12}$. Szabályaink legyenek

$$\boldsymbol{x}_1 + \boldsymbol{x}_2 + \boldsymbol{x}_3 + \boldsymbol{x}_4 + ... + \boldsymbol{x}_{11} + \boldsymbol{x}_{12} \equiv 0 \pmod{31},$$
$$\boldsymbol{x}_1 + 2\boldsymbol{x}_2 + 3\boldsymbol{x}_3 + 4\boldsymbol{x}_4 + ... + 11\boldsymbol{x}_{11} + 12\boldsymbol{x}_{12} \equiv 0 \pmod{31},$$
$$\boldsymbol{x}_1 + 4\boldsymbol{x}_2 + 9\boldsymbol{x}_3 + 16\boldsymbol{x}_4 + ... + 121\boldsymbol{x}_{11} + 144\boldsymbol{x}_{12} \equiv 0 \pmod{31},$$
$$\boldsymbol{x}_1 + 8\boldsymbol{x}_2 + 27\boldsymbol{x}_3 + 64\boldsymbol{x}_4 + ... + 1331\boldsymbol{x}_{11} + 1728\boldsymbol{x}_{12} \equiv 0 \pmod{31}.$$

Annak részletes vizsgálatát, hogy így 2-hibajavító kódot kapunk, feladatnak hagyjuk. Jegyezzük meg, hogy ebben az esetben is csak másfélszer annyi üzenetet kell átküldenünk, mint kódolás nélkül. Ha tehát itt is a 0,02%-os hibájú berendezést használjuk, akkor a legalább 3 hibát tartalmazó csoportok lesznek azok, amelyeket nem tudunk javítani. Ennek valószínűsége

$$1-0{,}9998^{12} - 12\cdot 0{,}0002\cdot 0{,}9998^{11} - 66\cdot 0{,}0002^2\cdot 0{,}9998^{10} \approx 0{,}000000002.$$

Ha úgy számolunk, hogy ilyenkor a 8 byte-hoz tartozó mindkét hang rossz, akkor is másodpercenként átlagosan csak kb. 0,0000774 hibás lejátszott hangot kapnánk, az egy órányi zene alatt összesen 0,28-at. Ez már elfogadható eredmény.

[1] Az egyszerűsítésre azért van szükség, mert a természetesen adódó mod 256 számolás $256=2^8$ esetén az osztással baj van, pl.: a 2 fele az 1 és a 129 is, a 3-nak viszont nincsen fele. Lehet a 256 elemű halmazon úgy értelmezni az alapműveleteket, hogy ne forduljon elő hasonló probléma, ezt láttuk az első félévben.

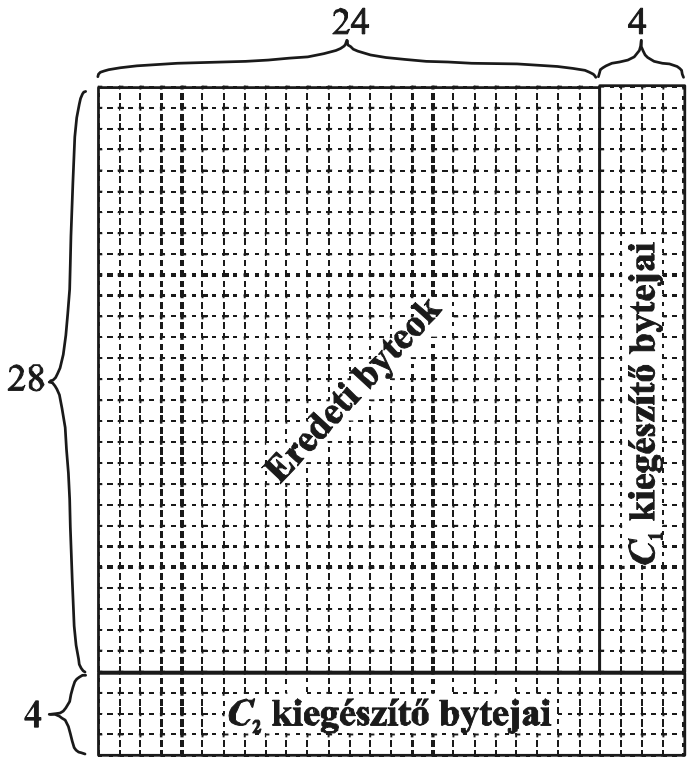
Lényeges, hogy a CD-n olyan kódot használjunk, amely a lehető legkisebb arányban növeli meg az adatsor hosszát, és nagyon gyorsan dekódolható.

"Keresztbe font" kódok

Eddig csak a véletlenszerű, elszórt hibák kiszűréséről volt szó. Sajnos, a hibák jelentős része csomóban jelentkezik. Ezek ellenében a CD-n két "keresztbe font" (Cross-Interleaved) Reed-Solomon kód biztosítja az információ megmaradását.

Ez egyrészt azt jelenti, hogy az egymáshoz kapcsolódó információk a lemez különböző helyein vannak elhelyezve, az egymás mellé kerülő jelek pedig a lejátszandó zenében valójában egymástól nagyobb időkülönbséggel következnek. Így a hibafolt ugyan több információba "belepiszkít", de mindegyikbe csak egy picit.

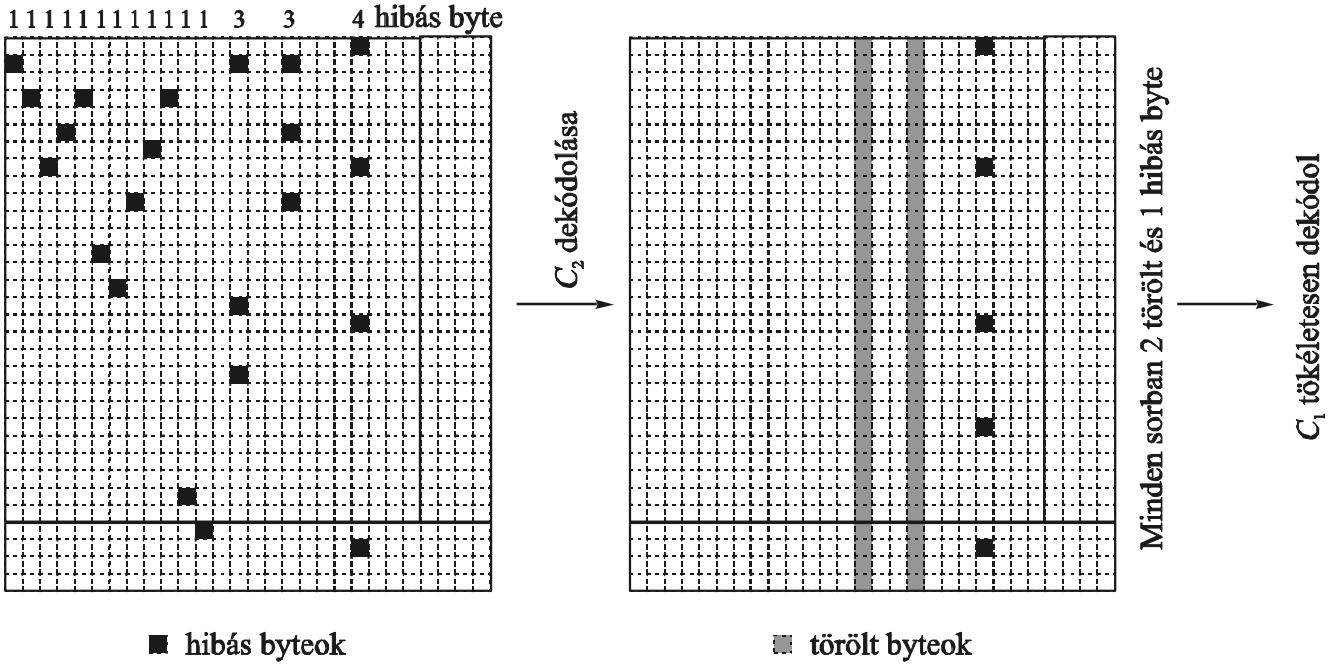
A másik trükk az, hogy a két kódolás az alábbi táblázatos formához hasonló módon működik együtt. Az első, C1 kód 24 byte-ot kezel együtt, ezekhez 4 byte-ot tesz hozzá 2-hiba javító módon, tehát a létrejövő 28 byte-os kódszavak minimális távolsága 5 lesz. Rendezzünk el 28 ilyen, már C1-gyel kódolt 28 byte-os adatsort egy táblázat soraiban! A második, C2 kód 28 byte-ot kezel együtt, a táblázat minden oszlopát további 4 byte-tal



Keresztbefont kódok

egészíti ki 2-hiba javító módon, tehát a létrejövő 32 byte-os oszlopkódok minimális távolsága 5 lesz.

A két kód alkalmazásával lényegében egy 24×28-as táblázat adatait kódoltuk át egy 28×32-es táblázattá. Bebizonyítható, hogy az így létrejövő táblázatok minimális távolsága 25, tehát a kettős kód alkalmas a 28×32 adatban keletkező 12 hiba kijavítására.



Keresztbefont kódok dekódolása

Képzeljünk el egy ennél sokkal rosszabb esetet, amikor 22 hiba van a táblázatban: tizenkét oszlopban 1-1 hiba, két oszlopban 3, egyben pedig 4 hiba is előfordul. Nem várható el, hogy ki tudjuk javítani a hibákat, hiszen az oszlopok 2-hiba javítása után még elképzelhető, hogy marad sor 3 hibával. Úgy döntünk, hogy a C2 kódot a dekódolásnál csak 1-hiba javításra használjuk.
Fel tudjuk ismerni a 3 hibát tartalmazó oszlopokat, mert azok legalább 2 távolságra vannak a legközelebbi kódszótól. Könnyen elképzelhető viszont, hogy a 4 hibát tartalmazó oszlop csak 1 távolságra került egy kódszótól, így azt "kijavítjuk" 5 hibásra. A trükk az, hogy kijavítjuk (legalábbis így képzeljük) az 1 hibát tartalmazó oszlopokat, azokat viszont, amelyekről látjuk, hogy ennél több hibát tartalmaznak csak megjelöljük, hogy "törlendő", ne vegyék figyelembe. Korábbi feladatainknak nyelvén a "hazugságot" "hanyagsággá" változtatjuk. Ez azt jelenti, hogy a 32 oszlopból C2 dekódolása után 29 jó, 2 törlendő, 1-ben pedig 5 hiba is van. Nézzük most a sorokat, amelyeket C1-gyel kódoltunk! Mindegyikben 2 jel törölve van, öt sorban pedig van egy-egy hibás is. Mivel C1 minimális távolsága 5, így mindegyiket ki tudja javítani.

Előfordulhat mégis, hogy a dekódoló észreveszi a hibát, de nem tudja kijavítani. Ez a helyzet pl. akkor, amikor túl sok legalább 2-hibás oszlop van. Ekkor a C2 dekóder végül is a "törlendő" jelet mellékeli jónéhány byte-hoz. A dekóderből kimenő byte-ok még mindig nem a lejátszás sorrendjében következnek egymás után, hanem összefonódva. A visszarendezéskor az "épségben" lévő társai közé érkező "törlendő" byte helyettesíthető szomszédai átlagával. Ilyen interpolációval, ami szukcesszíven is alkalmazható elérhető, hogy ahelyett, hogy "mute"-ra kapcsolna a zenegép, egy-egy pillanatban amolyan "kitöltő" hangot ad. Ez, ha tényleg csak pillanatokról van szó valószínűleg észre sem vehető.

A csomóhiba ma***x***imális hossza, amit a dekodolással még ki tud javítani a rendszer 4000 adatbit. Tehát ha ennyi egymást követő bit elromlik, akkor azt nem fogjuk észrevenni. Ez a disc-en 2,5 mm-nyi torzulást jelent a pálya irányában.
Az interpoláció segítségével pótolható egybefüggő bitsorozat ma***x***imális hossza 12 000.

A barázda-jelek

Láttuk, hogy a mintavételből keletkező byte-ok 24-es csoportját az egyik kód 28, majd a másik 32 byte hosszúságúra növeli. (A "keresztbe fonás" miatt ez ennél valamivel bonyolultabb, csak a számarányok tekintetében pontos a megállapítás.) A 32 byte-hoz hozzátesznek egy 33.-at, egy kontrol byte-ot, amely lehetővé teszi, hogy tudjuk, hogy hol tartunk.

Az így kapott adatsor nem alkalmas arra, hogy a "síkságok" és "árkok" módszerével az anyagba vive, azt a lézer olvasófej biztonságosan ki tudja olvasni. Az árok mélysége és a síkság magassága úgy van beállítva, hogy az árokból sokkal kisebb intenzitású fény verődik vissza a leolvasófejbe, mint a síkságról. Az egymáshoz túl közeli 1-esek, tehát közeli falak, olyan interferenciát okoznának, amely megzavarná a leolvasást. A túl távoli falak esetén nagy a bizonytalanság, hogy pontosan mennyi ideje tart a lapos rész, hány 0-t jelent ez. Ilyen megfontolások alapján a lemezen olyan jelsorozatokat érdemes tárolni, amelyben két 1-es között legalább két, de legfeljebb tíz 0 van. Az ilyen elrendezésű jelsorozatok a ***barázda-jelek*** (channel-bits).
A byte-ok 256 lehetséges értékét hány barázda-biten lehet tárolni? Ennek meghatározásához a mérnököknek az alábbi feladatot kellett megoldaniuk:

Ha F_n az n hosszúságú barázda-jelek száma, akkor melyik az a legkisebb n érték, amelyre $F_n \geq 256$?

A helyes eredmény 14. Tehát a byte-okat 14 bitre bővítik. Ez az eljárás az EFM átalakítás (Eight-to-Fourteen Modulation).
Problémát okoz, hogy két szomszédos barázda-jel együtt már nem feltétlenül teljesíti a barázda-jelekre kirótt követelményeket. Segítséget jelent viszont, hogy F_{14} értéke 267, tehát 11 darab 14 hosszúságú barázda-jel kidobható. 10-et azért hagytak ki, mert "nem jöttek volna jól ki a szomszédaikkal", 1-et pedig véletlenszerűen választottak. Ezután az EFM átalakítást úgy tervezték meg, hogy a szükséges logikai kapuk száma minimális legyen.

Az egymás mellé kerülő barázda-jelek problémáját nem teljesen oldotta meg a 10 kellemetlen sorozat kidobása. További 3-3 bitet kellett berakni a byte-oknak megfelelő 14 hosszúságú bitsorozatok közé.
A 3 bit egy kicsit túlbiztosított. Beállításukkor arra is törekednek, hogy a pálya kezdetétől a síkságok és az árkok hosszának különbsége minél kisebb legyen. Az ezt a különbséget leíró hullámszerű függvényt ugyanis a berendezés "érzi": annak megfelelő zaj indukálódik a berendezésben. Ebből ki kell küszöbölni az emberi fül számára hallható nagyobb amplitudójú komponenseket, tehát minél kisebb kitéréseket kell elérni.

Végül a 33 byte-ból adódó 33×17 bit után 27 szinkronizáló bit jön. Ez a 27 bitsorozat olyan, hogy nem keverhető össze a kódolt zenével, csak arra szolgál, hogy az egyik 33-as egység végét, a következő kezdetét jelölje.
Mindebből kiszámolható, hogy egyetlen másodpercnyi zenét 4 321 800 biten tárol a CD.