|  |
| --- |
| Universitatea politehnica bucurești |
| Detecția și corecția erorii |
| Proiect RC |
|  |
| **Gheață Ioana Cristina 442A** |
| **1/30/2015** |

|  |
| --- |
|  |

Cuprins

1. **Introducere**
2. **Coduri detectoare de erori**

 2.1 Codul de control al parității

 2.2 Codul suma de control

* 1. CRC (Control Redundant Ciclic)
1. **Protocoale ARQ**
2. **Coduri Corectoare de Erori.**

4.1 Codul Hamming

4.2 Codul convultional

4.3 Codul Reed-Solomon

4.4 Codurile de control al parităţii de mica densitate(Low denisity parity check)

1. **Concluzii**
2. **Introducere**

Detecția și corecția erorii reprezintă o tehnică introdusă pentru a mări probabilitate de trimitere a datelor digitale neeronate prin canalele de comunicații nesigure. Multe canale de comunicații conțin zgomot si de aceea pot fii introduse erori in timpul transmisiei de la sursă la receptor. Tehnicile de detecție a erorii permit detectarea unor astfel de erori, în timp ce tehnicile de corecție a erorii reconstituie datele originale.

Există mai multe cauze precum zgomotul, cross-talk-ul etc care sporesc probabilitatea datelor de a fii corupte in timpul transmisiei. Nivelele superioare au o vedere de ansamblu asupra arhitecturii rețelei și nu sunt conștiente de partea hardware a procesării datelor. Aplicațile voce sau video nu vor fii afectate foarte mult de aceste erori dar majoritatea aplicaților nu vor funcționa corespunzător dacă vor primi date eronate. In cazul comunicatiei wireless erorile sunt foarte mari.

Proiectanții de rețele au elaborat două strategii de bază pentru tratarea erorilor. Prima strategie este de a include in datele transmise sufieciente date redundante astfel încât receptorul să fie capabil să deducă singur care date ar trebui sa fie corecte. Iar cea de-a doua strategie este de a transmite doar suficiente date redundante astfel încât receptorul să poată deduce doar faptul ca există erori de la transmisie dar nu si despre ce erori este vorba dar să poată cere o retransmitere a datelor. Prima strategie reprezintă o strategie de corecție a erorilor iar cea de a doua de detecție a erorilor.

Nivelul legăturii de date **corectează** erorile de transmitere apărute la nivelul fizic, realizând o comunicare corectă intre două noduri adiacente ale rețelei. Mecanismul utilizat în acest scop este împărțirea biților în cadre (frame), cărora le sunt adăugate informații de control. Cadrele sunt transmise individual, putând fii verificate si confirmate de către receptor.

Există mai multe considerente ce trebuiesc luate în considerare în momentul alegerii tehnicii de tratare a erorilor. Tehnica de retransmitere a datelor se folosește doar pentru canle cu zgomot deoarece retransmisia datelor nu asigură complet faptul ca datele retransmise nu vor fii eronate din nou. Dar costul de implementare este mai scăzut. Se pot folosi de exemplu pe canalele care sunt foarte sigure, precum cele din fibră. Pe de altă parte în cazul canalelor cu transmisie wireless, care au un nivel de eroare ridicat, este de preferat să se folosească cea de a doua tehnică, a corecției erorilor, astfel încât receptorul să poată determina care bloc a fost transmis inițial.

Un alt considerent ce trebuie luat în considerare este reprezentat de tipul erorii . In momentul in care transmitem niste date pe un canal presupunem ca există posibilitatea sa avem anumite tipuri de erori la transmisie. Desigur, nici una din cele două tehnici nu poate corecta/detecta orice tip de eroare. Majoritatea erorilor ar trebui sa fie corectate de un cod sufiecient de bun. Un tip de eroare. (Referință bibleografică 4)

Un tip de eroare este reprezentat pe eroarea pe un singur bit:

|  |  |  |
| --- | --- | --- |
| Trimis: |  | Primit: |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |  | 1 | 0 | 1 | 1 | 0 | **1** | 1 | 1 |

La receptie un singur bit este gresit.

Eroarea pe mai multi biti:

|  |  |  |
| --- | --- | --- |
| Trimis: |  | Primit: |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |  | 1 | 0 | 1 | **0** | 0 | **1** | 1 | 1 |

La receptie mai mulți biți sunt eronați.

Eroarea de tip explozie (burst error):

|  |  |  |
| --- | --- | --- |
| Trimis: |  | Primit: |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |  | 1 | **1** | **0** | **0** | 0 | **1** | 1 | 1 |

Mai mulți biți consecutivi sunt eronați.

Ideea generala pentru obținerea detecției și corecției erorii este, prin urmare adăugarea la transmiterea mesajului mai multă redundanță astfel încât la recepție să se poată verifica consistența mesajului primit și datele corupte să poată fii recuperate.

Schemele de corecție a erorii și de detecție a erorii pot fii de două feluri: sistematice sau nesistematice.

 În schema sistematică, transmițătorul trimite datele originale și atașează un număr fix de biți de control (parity data) care derivă din datele de biti prin intermediul algoritmului determinist. Dacă se cere strict doar detecția erorii, un receptor poate să aplice același algoritm biților primiți și să compare rezultatele cu biții de control. Dacă valorile nu corespund rezultă ca o eroare a avut loc in timpul transmisei.

În schema nesistematică , mesajul inițial este transformatat într-un mesaj codat care are cel puțin la fel de mulți biți ca si mesajul original.

O performanță bună a controlului de erori necesită ca schema să fie aleasă pe baza caracteristicilor canalului de comunicații. Modelele de canale obișnuite includ modele fără memorie unde erorile nu apar într-o anumită ordine dar cu o probabilitate cunoscută, si modele dinamice unde erorile deobicei sunt de tipul burst.

Dacă capacitatea canalului nu poate fii determinată, ori este foarte variabilă, schema detecției de erori poate fii combinată sistem de retransmitere a datelor eronate. Acest procedeu se numeste Cerere automatica repetata (ARQ Automatic Repeat request) și este foarte folosită in internet. (Referință bibleografică 4)

**2. Coduri detectoare de erori**

 Codurile corectoare de erori sunt folosite de obicei pentru canalele fără fir unde probabilitatea de eroare este mai ridicată spre deobire de canalele care folosesc fibră optică sau fibre de cupru. Este de preferat în aceste cazuri o detectare a erorii si retransmitere a datelor decât corecția erorii care ar fii mult mai costisitoare.

**3.1 Codul de control al parității**

 Una dintre cele mai simple și utilizate metode de detectare a erorilor se bazează pe utilizare unor biți „de paritate”. Acestea pot detecta erorile simple (modificarea unui singur bit din cuvânt), dar nu și erorile multiple, și nici nu pot corecta erorile detectate. Paritatea este o formă simplă de detecție a erorilor care adaugă câte un bit la fiecare cuvânt (8 biți) din modulul de memorie. Acest bit suplimentar reține dacă în cei 8 biți ai fiecărui cuvânt există un număr par sau impar de cifre 1 și este memorat sau transmis împreună cu cuvântul. La utilizarea (citirea sau recepția) cuvântului paritatea se recalculează și rezultatul se compară cu bitul de paritate citit sau recepționat. Dacă nu se potrivește, atunci înseamnă că există o eroare în memorie sau la transmisie.

 Este codul autoverificator cel mai simplu. El se compune din m + 1 biţi: cei m biţi de informaţie la care se adaugă al (m + 1) - lea bit numit bit de paritate. Valoarea sa este aleasă astfel ca numărul total de biţi egali cu 1, calculat pe n + 1 biţi să fie par (în cazul unei parităţi pare), sau impar (paritate impară).

**Exemplu:** Transmiterea de caractere codificate în ASCII pe 7 biţi plus un bit de paritate între un calculator şi un terminal.

A → 1 1000001

 B → 1 1000010

 E → 0 1000101 ↑ bit de paritate impară

 Dacă un bit este schimbat din eroare în timpul transferului, paritatea nu mai este verificată. Dacă eroarea este detectată trebuie retransmisă informaţia deoarece eroarea nu poate fi localizată pentru a putea fi corectată. În general, controlul de paritate nu permite decât detectarea unui număr impar de erori, în cazul unui număr par efectele se pot anula. Controlul de paritate nu poate fi utilizat decât pentru transmisii în care posibilitatea apariţiei erorilor este scăzută (de exemplu, în interiorul unui calculator sau între calculator şi perifericele sale)

 Multe erori multiple însă nu afectează bitul de paritate și deci nu pot fi detectate cu această metodă simplă.

**3.2 Codul suma de control**

 Suma de control la nivel de bloc este un alt mecanism de detecție a erorilor de transmisie. Mai intâi, datele sunt împărțite în blocuri, apoi se însumeaza și se obține o sumă care va fi trunchiată, inversată și adaugată la sfârșit. La recepție, blocurile primite, care includ și suma de la sfârșit, se adună pe măsură ce sosesc. În cazul în care suma obținută nu este 0, înseamnă că datele sunt eronate și atunci secvența trebuie retransmisă. Corecția erorii nu este posibilă. În cazul metodei CRC se calculează suma de control prin împărțire aritmetică. Secvența de biți este împărțită cu un număr special ales. Împărțirea se face în modulo 2, adică folosind operatorul XOR. Restul împărțirii este de fapt semnatura care va fi adaugată la sfârțit, dupa biții utili. Folosind algoritmul de la codurile Hamming se obține divizorul. La recepție, se recalculează restul împărțirii. Dacă acesta nu coincide cu restul primit, atunci secvența este eronată.

 Acest cod se află în strânsă legătură cu codul parității. Un exemplu de control al putea fii chiar un grup de biți de pariate. Cu toate acesteam există alte coduri de sine stătoare de control al sumei, bazate pe o sumă care se realizează in mod continuu a biților de date dintr-un mesaj. Controlul sumei deobicei se realizează la sfârșitul mesajului ca o completare a funcției de insumare. In acest fel, erorile pot fii determinate însumând toate cuvintele de date recepționate impreună cu suma de control. Dacă în final această sumă este negativă înseamnă că nu am avut erori în canal.

 Un exemplu foarte cunoscut al acestui algoritm este reprezentat chiar de controlul de 16 biti al Internetului folosit în tot internetul ca o parte a protocolului IP. Acesta este un protocol al sumei formate din mesajele impărțite in cuvinte de 16 biti. Întrucât aceasta metodă se folosește la nivel de cuvinte de cod și nu la nivel de bit erorile ce au fost omise prin metoda parității pot afecta suma și prin urmare pot fii detectate prin această metodă.

De exemplu, daca bitul de ordinul cel mai mic din două cuvinte este schimbat de la 0 la 1, un control al parității nu va determina această eroare. Dar dacă doi de unu vor fii adunate la controlul sumei pe 16 biti va rezulta o sump diferită si eroarea va fii detectata.

In particular controlul sumei la Internet este o metodă eficientă si ușoară dar nu asigură o protecție suficient de bună în anumite cazuri, deoarece în final rămâne tot o simplă sumă. Nu determină adăugare sau ștergerea unui bit cu valoarea 0 sau interschimbare unor părți din același mesaj si oferă protecție slabă in ceea ce privește ruperea mesajului în două în locurile în care două pachete se întâlnesc. Aceste erorii nu sunt foarte probabile să apară prin procesee aleatoare.

 Pentru evitarea acestor tipuri de erori se poate alege folosirea unei sume de control Fletcher. Aceasta include o componentă pozițională, adăugând poziția si produsul datelor la sumă. Această metodă conferă o detecție mai puternică a schimbării poziționării datelor.

(Referință bibleografică 2)

**3.3 CRC (Control Redundant Ciclic)**

 Codurile ciclice sunt coduri bloc, având aceeași lungime a cuvintelor de cod, în care cele n simboluri ale cuvântului de cod sunt coeficienții unui polinom. Din combinațiile acestor coeficienți se pot forma polinoame diferite, iar numărul acestor polinoame este 2n. Pentru a realiza detecția și corecția erorilor, se aleg astfel pentru codare doar polinoamele divizibile printr-un polinom, numit polinom generator al codului.

 Principiul codurilor ciclice de detecție a erorilor constă în calcularea la emisie a restului împărțirii polinomului ce conține cuvântul de cod ce trebuie emis la polinomul generator. Acest rest al împărțiri constituie suma ciclică de control. Dacă în procesul de transmisiune nu s-au introdus erori, polinomul ce reprezintă cuvantul recepționat va fi divizibil prin polinomul generator, deci restul va fi zero. Faptul că există un rest diferit de zero, poate constitui un criteriu pentru detecția erorilor. Dacă pozițiile în care s-au introdus erorile pot fi specificate din structura restului, atunci corecția erorilor poate fi efectuată. În acest sens, se consideră polinoame generatoare primitive(ireductibile și divizibile prin xn SAU EXCLUSIV 1 ).

Codurile ciclice se clasifică în coduri sistematice și nesistematice. Codurile sistematice: simbolurile de informație pot fi amplasate grupat, la începutul sau sfârșitul cuvintelor de cod. Codurile nesistematice: simbolurile de informație și cele de control sunt amestecate

**Algoritmul de calculc CRC**

 Codurile polinomiale sunt bazate pe tratarea șirurilor de biți ca reprezentări de polinoame cu coeficienți 0 și 1. Ex.: 110001 = x5+x4+x0 Se va folosi aritmetica polinomială de tipul modulo 2, în care nu există transport la adunare și nici împrumut la scădere. Se va folosi, de asemenea, un polinom generator G(x). Acest polinom are atât bitul cel mai semnificativ cât și cel mai puțin semnificativ 1. Se dorește sa se adauge un număr de biți la sfărșitul unui cadru,ce are un polinom notat M(x)astfel încât M(x) divide G(x). Algoritmul pentru calculul sumei de control

* Fie r gradul lui G(x). Se adaugă r biți de 0 la capătul mai puțin semnificativ al cadrului astfel încât acesta

va corespunde polinomului xr M(x).

* Se împarte șirul de biți ce corespund lui G(x) într-un șir de biți corespunzător lui xr M(x), utilizând

împărțirea modulo 2.

* Se scade restul (care are întotdeauna r sau mai puțini biți) din șirul de biți corepunzător lui xr M(x),

utilizând scăderea modulo 2. Rezultatul este cadrul cu suma de control ce va fi transmis (acesta este divizibil modulo 2 cu G(x)).

* Exemplu:
	+ Cadru: 1101011011
	+ Generator: 10011
	+ Mesaj după adăugarea a 4 biți de 0: 1 1 0 1 0 1 1 0 1 10 0 0 0
	+ Cadru transmis: 1 1 0 1 0 1 1 01 1 1 1 1 0
	+ Anumite polinoame au devenit standarde internaționale. Cel folosit de IEEE 802 este: x32+x26+x23+x22+x16+x12+x11+x10+x8+x7+x5+x4+x2+x1+1 (Referință bibleografică 1)

**3. Protocoale ARQ**

 Detecţia erorilor se face cu coduri ciclice detectoare de erori, adică adăugând biţi de paritate la emisie, suma ciclica de control, CRC (Cyclic Redundancy Check), care se verifică la recepţie. Erorile sunt detectate la receptor prin verificarea sumei ciclice de control, CRC. Suma ciclica de control se calculează la emisie împărţind polinomul de informaţie cu un polinom, numit polinom generator, şi restul obţinut se ataşează informaţiei ca CRC. La recepţie, acest rest se recalculează din informaţia recepţionată şi se compară cu CRC-ul sosit, ataşat informaţiei. Dacă nu coincid, se transmite sursei decizia că există erori şi se cere retransmiterea pachetului. Corecţia erorilor se face prin retransmitere. S-au dezvoltat astfel tehnicile de cerere automată pentru retransmiterea pachetului detectat ca fiind eronat şi anume tehnicile ARQ (Automatic Repeat Request).

 **1 Transmiterea cu oprire şi aşteptare (Stop and Wait)**, la care transmiţătorul emite un cadru şi se opreşte aşteptând confirmarea recepţiei corecte a cadrului. Dacă soseşte o confirmare pozitivă, va transmite următorul cadru, sau va retransmite cadrul original dacă soseşte o confirmare negativă, cauzată de detectarea unui cadru eronat. Astfel, corecţia erorilor se face prin retransmitere. Metoda se poate folosi acolo unde timpul de propagare tur-retur este mic, altfel întroduce întârzieri mari şi eficienţa este redusă. Nu este necesară o numerotare a cadrelor. La emisie şi la recepţie este necesar să existe câte un buffer pentru memorarea pachetului de transmis, respectiv de recepţionat.

 **2 Transmiterea în cadrul unei ferestre de transmisie**, întoarcere înapoi la cadrul eronat, GBN (Go Back N) şi retransmiterea tuturor cadrelor incepând cu cel eronat. Transmiţătorul nu mai aşteaptă confirmarea fiecărui cadru, ci emite mai multe cadre, având numere de secvenţă dintr-o fereastră de transmisie, şi se opreşte când îşi epuizează fereastra, aşteptând confirmarea recepţiei cadrelor emise. Fereastra de transmisie este 3 2 17 cadre, la reţelele terestre. Dacă soseşte o confirmare pozitivă, fereastra avansează, şi se va transmite următorul cadru. Dacă soseşte o confirmare negativă, se vor retransmite toate cadrele începând cu cel eronat. Este necesară o numerotare a cadrelor; la emisie este necesar să existe un număr de buffere egal cu fereastra de transmisie, pentru memorarea pachetelor transmise în aşteptarea confirmării. La recepţie este necesar să existe un buffer, deoarece fereastra de recepţie este unitară.

 **3. Retransmiterea Selectivă**, SR (Selective Reject), este asemănătoare cu tehnica precedentă, GBN, dar se face retransmiterea doar a cadrului eronat, nu a tuturor cadrelor care urmează după acesta, deoarece timpul de propagare la reţelele prin satelit, unde se aplică tehnica, este mult mai mare (270 ms) decât la reţelele terestre (50 ms peste Europa). Este necesară o numerotare a cadrelor cu numere de secvenţă. Fereastra de transmisie este 7 2 1 127 , iar fereastra de recepţie trebuie să fie cel mult jumătate din fereastra de emisie, pentru a evita confuziile. Este necesar să existe la emisie şi la recepţie un număr de buffere egal cu fereastra de transmisie respectiv de recepţie. La recepţie trebuie să existe posibilitatea introducerii ordonate în şirul pachetelor recepţionate, a pachetului retransmis, conform numărului său de secvenţă. Detecţia erorilor se face cu coduri ciclice detectoare de erori, adică adăugând biţi de paritate la emisie, suma ciclica de control, CRC care se verifică la recepţie. Dacă au apărut erori se cere retransmiterea cadrului iar corecţia erorilor se face prin retransmitere. La oricare dintre ele, SW, GBN sau SR are loc retransmisia pachetului dacă timpul de asteptare a confirmării expiră (time-out). (Referință bibleografică 6)

****

Imagine: http://www.tc.etc.upt.ro/teaching/idd-cd/idd-cd-cap4.pdf

4. Coduri Corectoare de Erori

**4.1 Codul Hamming**

Un cadru constă dintr-un mesaj conținând m biți peste care se adaugă încă r biți redundanți, pentru corecția erorii. Intr-un bloc de cod, biturile r sunt văzute ca niște funcții a celor m biți din mesaj. Într-o schemă sistematică cei m biți sunt trimiși impreună cu biții r de control, în loc sa fie codați înainte să fie trimiși. Putem avea un cod de tip linear în care biții r sunt reprezentați de o funcție lineară a biților mesajului m care împreună cu operații logice de tip OR sau modulo2 fac posibilă codarea prin operații de matrici sau circuite logice simple.

Rata codului reprezintă fracțiunea din cuvântul de cod care conține informație ce nu este redundantă (m/n, unde n=m+r lungimea blocului transmis). In practică această fracțiune variază foarte mult. Poate sa fie ½ pentru un canal zgomotos pentru care jumătate din informația primită este redundantă . Sau fracțiunea poate fii egală cu 1 caz in care nu am avut foarte puțină informație redundantă si blocul primit este aproape identic cu mesajul transmis inițial.

Având două cuvinte de cod diferite putem determina câți biți diferă între ele aplicand un XOR si numărând câți de 1 avem în rezultat. Numărul de biți de poziție prin care două cuvinte cod diferă se numește distanța Hamming. Semnificația acesteia este: dacă doua cuvinte se află la o distanța d Hamming depărtare vom avea nevoie de d erori pe un singur bit pentru a le converti una în cealaltă.

Distanta Hamming între doi vectori de dimensiuni egale este data de numarul de pozitii în care acestia difera. Ea masoara astfel numarul de schimbari care trebuie facute într-un vector pentru a îl obtine pe celalalt, sau reformulat numarul de erori care transforma un vector în celalalt.

Având algoritmul pentru implementare a biturilor de control, este posibilă construirea unei liste întregi de cuvinte permise, iar din această listă să extragem cuvintele aflate la cea mai mică distanță Hamming.

In majoritatea aplicaților toate cele $2^{m}$ cuvinte sunt permise dar datorită modului de prelucrare a biturilor de control nu toate $2^{n}$ cuvinte pot fii folosite. În realitate doar o mică fractiune $2^{m}$/$2^{n}$ sau 1/$2^{r}$ din mesajul initial vor fii cuvinte admise.

Proprietăţile celor două tehnici depind această distanţă Hamming. Pentru detecţia a d erori este necesar un cod cu o distanţă Hamming d+1. Cu un asemenea cod nu se poate ca d erori pe un singur bit sa poată modifica un cuvant de cod corect intr+un nou cuvant de cod corect. In acest moment receptorul poate vedea toate cuvintele gresite si sa transmita mai departe faptul că s-a produs o eroare. (4+7)

Exemplu de detectie de erori singulare folosind distanta Hamming:

Fiecare bit este codat prin repetarea sa de doua ori. Am introdus o redundanta care va permite detectarea erori singulare.

Cand la receptie se obtin coduri 00 respectiv 11 se receptioneaza corect 0 respectiv 1 iar daca se obtine 01 sau 10 se detecteaza o eroare singulara (fara a putea decide însa nimic în sensul corectiei ei).

**Exemplu:**

mesaj initial: 0.1.0.0.1.0.1.1.0.1

mesaj codat: 00.11.00.00.11.00.11.11.00.11

mesaj receptionat cu eroare: 00.11.00.00.10.00.11.11.00.11 => detectie de eroare

Între oricare doua ”cuvinte de cod” valide se obține o distanta Hamming 2.

**4.2 Codul convultional**

 În 1965 a fost introdus un nou tip de coduri, ce se remarcă prin siguranta mărită pe care o asigură în transmiterea informației: codurile convolutionale. În acest caz, implementarea prin codificare a unui mesaj de lungime k se face urmărind și de codificările mesajelor anterioare. Companii foarte mari precum NASa utilizează o combinație între codurile convulționale și codurile Reed - Solomon. Mesajele de informatie sunt codificate

Într-o primă fază cu un cod de tip Reed Solomon, apoi cu un cod convolutional.

 Codul convulționalnu este un cod bloc. Acesta conține un codificator care procesează o serie de biți input și generează o serie de biți outpup. Spre deosebire de codurile bloc, aici nu avem o marime default pentru mesaje sau o codificare limită. Ieșirea depinde de biții actuali si anteriori de la intrare. Acest lucru este posibil deoarece codul are memorie. Numărul de biți anteriori de care depinde ieșirea se definește ca o constrângere a lungimii codului. Aceste coduri sunt larg răspândite în rețele de deploy: telefonie mobila GSM, în comunicațile prin satelit etc.

 Aceste coduri sunt folosite in nivelul legatura de date pentru a asigura transferul sigur a informatiei prin canale. Se pot transmite date digitale video, radio, de tip mobile si comunicatii de tip satelit. Aceste coduri deobicei sunt implementate impreuna cu un alt cod cum ar fii Reed Solomon. Codurile convulționale sunt eficiente în rezolvarea erorilor de biți izolați dar nu vor face față unei avalanșe de erori. Adăugând și un alt tip de cod precum Reed Solomon, erorile pot fii corectate chiar dacă sunt într-un nr foarte mare și consecutive. Codul final format din cele două coduri asigură o protecție foarte bună contra erorilor, atât singulare cât și în avalanșă.

 Acest cod se mai folosește și în nivelul fizic. De exemplu atunci când semnalul este foarte zgomots, se poate folosi QAM-64 pentru a transmite 6 biți pe simbol. Pentru a transmite cu un raport semnal zgomot foarte mic, se folosește QPSK cu o transmitere de 2 biti pe simbol. Informația poate fii mai întâi codată pentru corecția erorii cu un cod convulțional.

**Exemplu:**

Fie următorul circuit liniar:



Imagine și exemplul preluate din: http://www.galaxyng.com/adrian\_atanasiu/cursuri/coduri/cod16.pdf

Circuitul este reprezentat de un cod convoluțional binar. Liniaritatea este datorată circuitului, observăm o invarianță în timp echivalentă unei întârzieri de un tact pe intrare ce produce o întârziere de 2 tacte și la ieșire.

 De exemplu, pentru intrarea 1 se obține 11 la primul tact, 11 la al doilea tact și 01 la al treilea (după care urmează 0000 . . .).

 Deci C(1) = 111101 = 1 + X + X2 + X3 + X5 . Datorită invarianței în timp, avem C(X) = C(01) = 00111101, C(X2 ) = C(001) = 0000111101, . . . (s-a ținut cont de dualitatea de notare polinom - cuvânt).

 Datorită liniarității, răspunsul C(1) caracterizează complet codul. De exemplu, mesajul de intrare 101 se codifică prin C(101) = C(1 + X2 ) = C(1) + C(X2 ) = C(1) + C(001) = 11110100 . . . + 0000111101 . . . = 1111101101 = 1 + X + X2 + X3 + X4 + X6 + X7 + X9 .

(Referință bibleografică 5)

**4.3 Codul Reed-Solomon**

 Codurile Reed-Solomon (RS) sunt coduri corectoare de erori în bloc inventate în 1960 de Irving Reed şi Gustave Solomon. Aceste coduri au început să fie utilizate începând cu 1990, atunci când progresele tehnologice au făcut posibilă trimiterea datelor în cantităţi mari si la viteze ridicate. Actualmente aceste coduri sunt utilizate într-o gamă largă de echipamente electronice cum sunt: dispozitivele pentru stocarea datelor (CD, DVD, hard-disk), telefoanele mobile, echipamentele folosite în comunicaţiile prin satelit, televiziunea digitală, modemurile de mare viteză (ADSL, xDSL).

 Spre deosebire de codul Hamming care operează cu un singur bit o data, codul Reed Solomon poate prelucra mai multi biți simultan. Ideea de bază a courilor Reed este că fiecare polinom de gradul n este unic determinat de n+1 puncte. De exemplu o dreaptă având ecuația ax+b este determinată de două puncte. Orice punct în plus va fii redundant ceea ce va fii folositor pentru corectia eorilor. De exemplu, dacă avem o dreaptă reprezentată prin doua puncte putem trimitem acele două puncte plus încă două puncte de control ce aparțin aceleași linii. Dacă unul dintre puncte este eronat, tot mai putem să salvăm punctele inițiale trasând o linie prin punctele recepționate. Trei puncte se vor alinia cu dreapta inițială și unul (cel eronat) va rămâne nealiniat. Găsind linia, am corectat eroarea.

 Codurile Reed Solomon sunt definite ca polinoame ce pot prelucra niste arii finite, dar lucrează într-un mod similar. Pentru m simboluri biți, cuvintele de cod sunt simboluri de lungime 2m-1. De obicei se alege m=1 pentru a obține octeți de cuvinte. Un cuvânt de cod va avea o lungime de 255 bytes. Codul de 255-233 bytes este folosit foarte des. Se adaugă 32 de simboluri redundante la cele 233 de simboluri de date. (4)

 Un cuvânt de cod de tip Reed Solomon este generat folosind un polinom special. Toate cuvintele de cod valide sunt divizibile exact cu acest polinom.

Forma generală:

 g(x)=(x-ai)(x-ai+1)...(x-ai+2t)

Iar cuvântul de cod este construit folosind:

c(x)=g(x)i(x)

Unde g(x) reprezinta polinomul generator, i(x) reprezintă blocul informațional și c(x) reprezintă un cuvânt de cod valid și este referit ca un element primitiv.

 **Exemplu:**

 Polinom generator pentru RS(255,249)

 g(x)=(x-a0) (x-a1) (x-ai) (x-a2) (x-a3) (x-a4) (x-a5)

 g(x)=x6+g5x6+g5x5+g4x4+g3x3+g2x2+g1x1+g0

(Referință bibleografică 9)

 Decodificatorul RS se ocupă cu procesarea fiecărui bloc şi corectarea erorilor întâlnite şi recuperarea datelor trimise odată cu mesajul original. Un cod RS se notează cu cRS(n, k) cu s biţi. Această notație reprezintă faptul că, codificatorul procesează k biți de paritate astfel încât la final să rezulte un cuvânt de cod de dimensiunea n. Rezultă n - k biți de paritate, de câte s biţi fiecare. Un decodificator RS poate corecta până la t simboluri ce conţin erori, cu 2t = n - k.

Un cod Reed Solomon se obţine împărţind mesajul original în blocuri de lungime fixă.

Fiecare bloc este apoi împărţit în simboluri de m biţi.

Fiecare simbol are lungime fixă (între 3 si 8 biţi).

 **Exemplu**

 Un cod care este folosit uzual este RS(255, 233) cu informația reprezentată pe 8 biţi. Fiecare cuvânt de cod este format din 255 de simboluri din care 233 sunt de date şi 22 sunt de paritate.

 Putem scrie următoparele relații:

n = 25

k — 233

 s = 8,

 t= 16.

 Codurile Reed solomon sunt micșorate dacă la codificator unii biți iau valoarea 0 adica, nu se transmit dar sunt adăugaţi la decodificator.

 Codul RS(255, 233) poate fi scurtat la (200, 168)

 Pentru a realiza această micșorare codificatorul face următoarele operații:

 • se începe cu un bloc de 168 biţi de date;

• se adaugă 55 de biţi de zero = > cod (255, 233);

 • se transmit 168 biţi de date şi 32 biţi de paritate. Un decodificator RS poate să corecteze un număr de n erori şi până la 2t ştersături.

 La decodificarea unui cuvânt RS putem observa următoarele:

• dacă 2s + r < 2t atunci codul original transmis poate fi corectat în întregime;

• 2s+r>t nu se poate corecta codul

• decodificatorul va genera un cuvânt decodat cu erori şi nu va fi semnalat acest lucru.

(Referință bibleografică 8)

**4.4 Codurile de control al parităţii de mica densitate(Low denisity parity check)**

 Codurile LDPC sunt coduri bloc liniare ce au fost inventate de Robert Gallagher in teza sa de doctorat urmând mai apoi a fii reinventate in 1995 când a fost cu adevărat nevoie de ele in practică datorită evoluției tehnologiei calculatoarelor. Intr-un cod LDPC fiecare bit de iesire este format dintr-o fractiune a bitilor de la intrare. Prin acest procedeu se formeaza o matrice ce are foarte putine componente de 1. Cuvintele de cod receptionate sunt decodate printr-un algoritm de aproximare a cărui calitate crește iterativ pe măsură ce sunt recepționate datele. Astfel se realizează corecția erorii.

 Codurile LDPC sunt practice pentru blocuri de mărimi foarte mare și realizează o performanță foarte bună în corecția erorii comparativ cu alte coduri. Din acest motiv încep sa fie incluse în noi protocoale. (Referință bibleografică 4)

 Cea mai simplă schemă de codare LDPC este reprezentată de verificarea unui singur bit de paritate SPC. SPC-urile implică adăugarea unui singur bit de paritate la mesajul original, valoarea căruia depinde de biții din mesajul original. Într-un cod de paritate par, bitul adițional adăugat fiecărui mesaj, asigură un număr par de biți 1 în fiecare cuvânt de cod.

 **Exemplu**

Fie sirul de biți 1010011 care reprezintă litera S in ascii.

 Un bit de paritate este adăugat ca al 8-lea bit. Șirul S conținea deja un număr par de 1 și de aceea bitul de paritate va avea valoarea 0. Deci noul cuvânt de cod va fii 10100110.

 Presupunând că un cuvânt de cod a fost trimis pe un canal simetric binar, și că unul sau mai mulți biți au fost inversați.

 Pentru fiecare cuvânt de cod primit se poate detecta eroarea dacă nu satisfac ecuatia (

H\*yt=0( H- matricea de control al parității și y reprezintă cuvântul de cod.)

 **Exemplu**

Fie cuvântul de cod c=[101110] trimis printr-un canal binar simetric.

La iesire obtinem y=[101010]=>

 1 1 0 1 0 0

H= 0 1 1 0 1 0

 1 1 1 0 0 1

H\*yt=[1 0 0]t

Rezultatul este un non zero deci sirul y nu este un cuvânt de cod provenit provenit din mesajul original. Deci luăm decizia ca acest cuvânt de cod a fost eronat în timpul transmisie.

Vectorul s=Hyt se numește sindromul lui y.

Sindromul indică care constrângeri de paritate nu au fost îndeplinite de către y.

 Prin acest exemplu observăm și utilizarea blocului de cod pentru a detecta transmisia cu erori. Dacă presupunem ca acest canal avea și mai mult zgomot și erau defapt inversați 3 biți y=[0 0 1 0 1 1] observăm ca nu mai putem face detecția erorii atât de simplu. Putem folosi această metodă doar dacă blocul de cod cu erori nu ne schimbă cuvântul de biți într-un cuvânt de cod nou. (Referință bibleografică 10)

**5. CONCLUZII**

Metodele de detecție și corecție sunt variate și se bazează pe o varietate de coduri, fie ele ciclice, grup sau convoluționale. Codurile grup și cele convolutionale folosesc în general o codare binară a informației, tratând simbolurile prin intermediul reprezentării lor cu biți de 0 și 1, în timp ce codurile ciclice asociază câte un polinom fiecărui astfel de cod. Codurile ciclice se folosesc cel mai des în calcularea sumei de control (checksum în engleză), metodă foarte des folosită pentru detecția erorilor care, deși nu poate realiza și corectia, este destul de ușor de implementat și de utilizat.

Mecanismele de detectarea erorilor provoacă de obicei stoparea operațiilor cu memoria, înainte ca datele greșite să poată fi utilizate în sistem. Corecția erorilor este însă un proces mai complicat; el presupune mai întâi detectarea lor, dar apoi și reconstruirea informatiilor greșite, pe baza unor informații redundante, chiar în timpul functionării sistemului („în zbor”). În funcție de procedeul utilizat detectarea și corectarea erorilor pot repara multe din greșelile care au loc.

Bibleografie:

* 1. <http://ro.wikipedia.org/wiki/Cyclic_redundancy_check#Algoritmul_de_calcul_CRC>
	2. <http://en.wikipedia.org/wiki/Error_detection_and_correction>
	3. <http://ro.wikipedia.org/wiki/Cod_corector_de_erori>
	4. [Andrew Tanenbaum – "Computer Networks" editia V, Ed. Prentice Hall 2011](http://stst.elia.pub.ro/news/RC/Computer%20Networks%20-%20A%20Tanenbaum%20-%205th%20edition.pdf)
	5. <http://www.galaxyng.com/adrian_atanasiu/cursuri/coduri/cod16.pdf>
	6. <http://www.tc.etc.upt.ro/teaching/idd-cd/idd-cd-cap4.pdf>
	7. <http://computing.dcu.ie/~humphrys/Notes/Networks/data.hamming.html>
	8. <http://andrei.clubcisco.ro/cursuri/f/f-sym/3cn2/Laborator%202.pdf>
	9. <http://www.cs.cmu.edu/~guyb/realworld/reedsolomon/reed_solomon_codes.html>
	10. <http://sigpromu.org/sarah/SJohnsonLDPCintro.pdf>