CAPITOLUL V

CODURI DE TRANSLARE A DATELOR

V.1 Canale de comunicați cu constrângeri

Codurile de translare a datelor se mai numesc și **coduri cu constrângeri** întrucât elimină unele combinații de biți care nu sunt permise sau sunt dificil de transmis pe canalul de transmisie. Rolul lor este de a crește performanțele sistemului de comunicații prin adaptarea caracteristicilor semnalului transmis la cele ale canalului.

Canalul de comunicații este în general:

◆ dispersiv sau cu memorie (datorită benzii limitate);

◆ zgomotos (afectat de zgomote aditive și/sau multiplicative);

◆ cu constrângeri(restricționează secvențele de date de intrare în funcție de protocoalele de transmisie utilizate sau de caracteristicile fizice ale echipamentelor).

Aceste probleme se rezolvă în general separat prin cascadarea unor codoare care să minimizeze efectele zgomotelor (prin corecția erorilor), să compenseze memoria canalului (cu filtre digitale recursive sau nerecursive pentru eliminarea interferenței intersimboluri) și să adapteze semnalul de date la constrângerile impuse de canal (prin folosirea unor coduri de translare).

Se consideră în cele ce urmează un canal de transmisie discret, fără memorie, nezgomotos, cu constrângeri.

Constrângerile de transmisie pot fi determinate de:

◆ **lățimea benzii de frecvențe** a canalului de transmisie, care determină, la rate mari de transmisie, interferențe intersimboluri IIS (ISI-*Intersymbol Interference*) inacceptabil de mari pentru unele combinații de biți;

◆ **protocoalele de transmisie** care utilizează unele secvențe de date ca mesaje speciale de control a traficului și care nu pot să apară în semnalul informațional transmis;

◆ **natura fizică a canalului de transmisie** (magnetic, optic). În mediu magnetic, tranziții simultane apropiate spațial pot fi distructive pentru material datorită fenomenelor neliniare. În multe aplicații creșterea vitezei de transmisie este limitată de timpul necesar procesului de revenire a materialului la parametrii normali (reîncărcare cu sarcini electrice, răcire etc). Unele canale de transmisie rejectează componentele de joasă frecvență astfel încât transmisia unor secvențe lungi de biți identici (nivel constant de

semnal) nu se face în condiții bune, fiind necesar un control al evoluției în timp a spectrului semnalului transmis.

Funcționarea detectorului precum și a circuitului de sincronizare din receptor poate fi perturbată de unele combinații de biți.

În sistemele de înregistrări magnetice cu detecție de vârf [Mar92], obiectivele principale sunt:

◆ comprimarea benzii semnalului transmis;

◆ optimizarea compromisului făcut între IIS, lățimea ferestrei de decodare și densitatea datelor;

◆ menținerea în calare a circuitului PLL de sincronizare;

◆ minimizarea costului de implementare.

În cazul înregistrărilor de date pe discuri optice [Men95], cu sau fără posibilitate de ștergere, din considerente economice, același laser este utilizat atât pentru scriere, cât și pentru citire. Dacă se crește densitatea datelor înregistrate pe disc, ceea ce este echivalent cu creșterea vitezei pe canalele de transmisie, apare o asimetrie între secvențele de biți 0, respectiv de biți 1, întrucât lungimea markerilor nu poate fi oricât de mult micșorată din considerente practice, așa cum durata minimă a impulsului ce poate fi transmis pe un canal este limitată.

Folosirea codurilor de translare a datelor minimizează IIS, îmbunătățește funcționarea circuitelor de detecție și de sincronizare, minimizează distorsiunile semnalului transmis dar cel mai important fapt este acela că permit creșterea vitezei de transmisie respectiv a densității de înregistrare a datelor în condițiile menținerii acelorași performanțe ale sistemului digital de comunicații [Sch91].

Codurile de translare a datelor se mai numesc și coduri RLL (*Run-Length Limited*).

Secvențele de biți identici, așa-numitele secvențe de tip 'fugă' (*run sequence*), au lungimea limitată.

O familie largă de coduri RLL este cea care restricționează lungimea secvențelor de zerouri (*ZR - Zero Run Constraints*).

Codurile de tip ZR, **RLL(d;k)**, impun ca între doi biți consecutivi de 1 să existe cel puțin ***d*** și cel mult ***k*** zerouri.

Exemplu: Secvență codată RLL(1;3): 10100010100100100010100001.....

Alegerea finită a parametrului ***k*** este utilă pentru menținerea în calare a circuitului PLL din receptor. Parametrul ***d*** este ales nenul pentru minimizarea IIS în sistemele de transmisie simplu-curent.

Suplimentar se pot impune restricții privind valorile admise pentru lungimea sec- venței de zerouri dintre doi biți 1 consecutivi. Rezultă o altă clasă de coduri de translare, cu **spațiere multiplă: RLL(d;k;s).** Parametrul **s** reprezintă pasul de variație a lungimii 'fugilor' de zerouri și are valoarea tipică 2. Aceste coduri s-au dovedit utile în sistemele de înregistrări digitale magneto-optice care folosesc tehnici directe de ștergere și scriere cu circuite rezonante.

*Exemplu:*Codul RLL(2;6;2) admite ca între doi biți '1' să existe 2; 4 sau 6 zerouri.

În unele sisteme de comunicații pe fibră optică, se dovedesc utile **codurile cu constrângeri asimetrice ARLL(a;b;c;d)** care limitează lungimile 'fugilor' de zerouri ($a \le l_0 \le c$) și a secvențelor-fugă de biți 1 ($b \le l_1 \le d$).

Exemplu: Codul ARLL(1;1;2;3) admite în secvența codată maximum 2 biți '0' consecutivi și secvențe de maximum 3 biți '1': 0011101001101001 .....

Constrângeri simetrice pentru ambele secvențe-fugă sunt respectate prin folosirea **codurilor simetrice SRLL(k)** astfel încât în semnalul codat să nu existe mai mult de ***k*** biți identici consecutivi.

Exemplu: Codul SRLL(3) limitează lungimea oricărei secvențe de biți identici la maximum 3.

V.2 Descrierea matematică a codurilor RLL

Parametrii unui sistem de transmisie cu constrângeri sunt:

◆ **lungimile de constrângere** (a;b;c;d;k;s) depind de natura codului folosit.

Acestea pot fi ilustrate în graful orientat al canalului cu constrângeri, cu număr

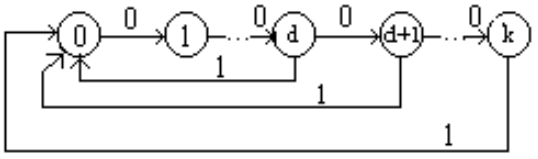

Fig.V.1 Graful RLL(d;k)

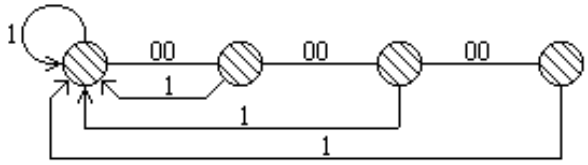

Fig. V.2 Graful cu constrângeri RLL(0;6;2)

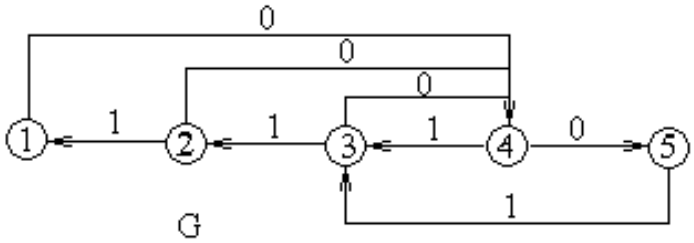

Fig.V.3 Graful ARLL(1,1, 2,3) cu un pas

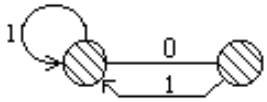

Fig. V.4 Graful cu constrângeri RLL(0;1)

finit de stări și săgeți orientate asociate cu un bit de ieșire (fig. V.1, V.2, V.3, V.4).

Observație: Vom exemplifica în continuare construcția unui cod RLL(0;1).

Analiza acestor diagrame se bazează pe teoria grafurilor de semnal și presupune scrierea matricii de adiacență a codului:

$A = [a_{ij}]$, $i, j = \overline{1:N}$

Elementul a_{ij} este egal cu numărul săgeților din graf care pleacă din starea i și ajung în starea j, pentru un graf cu N stări.

◆ **capacitatea canalului de transmisie** considerat nezgomotos și fără memorie:

$$C = \log_2 \lambda_{max} \qquad (V.1)$$

unde λ_{max} este rădăcina reală maximă a ecuației caracteristice a matricii de adiacență a codului [Mar92];

$$\det[A - \lambda I_N] = 0 \qquad (V.2)$$

Exemplu: Pentru graful cu constrângeri RLL(0;1), matricea de adiacență este:

$A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$; $N = 2$.

Din ecuația caracteristică: $\lambda^2 - \lambda - 1 = 0 \rightarrow \lambda_{max} = \frac{1+\sqrt{5}}{2} \rightarrow C \cong 0,694$.

◆ **rata de codare** (R), conform teoremei de codare a canalelor de transmisie enunțată de Shannon, satisface inegalitatea:

$$R \leq C \tag{V.3}$$

și se calculează ca raportul numărului variabilelor binare de intrare (***p***), respectiv a celor de ieșire (***q***) din codor:

$$R = \frac{p}{q} \tag{V.4}$$

◆ **eficiența de codare**:

$$\eta = \frac{R}{C} \cdot 100[\%] \tag{V.5}$$

Exemplu: Pentru canalul cu constrângeri considerat: se pot construi codurile:
1/2 RLL(0;1) cu $R = 1/2$; $\eta = 72,05\%$;
2/3 RLL(0;1) cu $R = 2/3$; $\eta = 96,06\%$.

Observații: Codurile cu rate mai mici de codare conduc la o valoare redusă a eficienței respectiv la o redundanță crescută a semnalului codat dar se implementează cu circuite de codare relativ simple. Prin alegerea unei rate de codare mari, se obține un semnal codat cu redundanță minimă dar cu structuri de codare și decodare mai complexe.

◆ **viteza de transmisie** a datelor (bps-*bits per seconds*);

Prezintă interes doar acele coduri care pot fi implementate ca automate cu număr finit de stări. De asemenea sunt utile codurile de tip finit, adică cele care au memorie și anticipație finită. Această ultimă condiție permite decodarea independentă de stare, pe baza unei **ferestre de decodare** care cuprinde mai multe grupuri de biți codați recepționați, ceea ce limitează fenomenul de propagare a erorilor de decodare.

Descrierea grafică a codurilor-bloc de translare a datelor se poate face cu **diagrame de tranziții** (stări) sau cu **diagrame de tip 'trellis'** desfășurate în timp. Graful asociat trebuie să fie determinist și ireductibil pentru ca să nu apară ambiguități la codare iar funcționarea codorului să nu se bucleze pe un număr restrâns de stări.

Fiecare cod RLL este descris de o diagramă de tranziții cu număr finit de stări, FSTD (*finite-state transition diagram*), care descrie constrângerile impuse de canalul de transmisie.

Teoremă: Dacă ***A*** este matricea de adiacență a unui graf ***G***, atunci A^q este matricea de adiacență a grafului ridicat la puterea ***q*** [Mar92].

Dacă în graful ***G*** sunt reprezentate tranzițiile dintre stări corespunzătoare unui singur bit de ieșire, prin ridicarea grafului la puterea ***q,*** se obțin toate tranzițiile dintre stările asociate cu segmente de cod de ***q*** biți.

Exemplu: Pentru a construi codul 2/3 RLL(0;1) este necesară ridicarea la puterea a treia a grafului din figura V.4, adică reprezentarea grafului G^3:

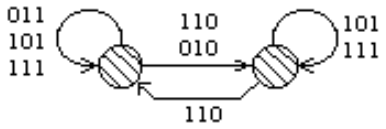

Fig. V.5 Graful G^3 cu constrângeri RLL(0;1)

Matricea de adiacență a noului graf este:

$$A' = A^3 = \begin{bmatrix} 3 & 2 \\ 2 & 1 \end{bmatrix}$$

Definiție: Numărul săgeților de ieșire dintr-un nod (stare) reprezintă **gradul de ieșire** a stării respective (*outdegree*).

Suma elementelor de pe linia ***i*** a matricii de adiacență a unui graf este egală cu gradul de ieșire al stării ***i***:

$$R(i) = \sum_j a_{ij} = O_i, \ \forall i = 1 : N \qquad \text{(V.6)}$$

Pentru ca un graf $\boldsymbol{G^q}$ să fie graf de codare binară cu rata ***p/q***, gradul de ieșire din fiecare stare trebuie să satisfacă inegalitatea:

$$O_i \geq 2^q, \ \forall i = 1 : N \qquad \text{(V.7)}$$

ceea ce este echivalent cu **condiția sumelor pe linie**:

$$R(i) \geq 2^q, \ \forall i = 1 : N \qquad \text{(V.8)}$$

unde ***R(i)*** reprezintă suma elementelor pe linia ***i*** din matricea de adiacență $\boldsymbol{A^q}$.

Exemplu: Pentru codul considerat se obțin:

$R(1) = O(1) = 5 > 2^2; R(2) = O(2) = 3 < 2^2$.

În cazul nerespectării condiției sumelor de linie, pentru una sau mai multe stări, se poate aplica **algoritmul de divizare** a unor stări din graf în vederea creșterii gradului de ieșire din stările critice. Întrucât pentru aplicarea acestuia este necesară determinarea **ponderilor** stărilor și săgeților din graf, este necesar ca în prealabil să se determine

vectorul propriu aproximativ minim al grafului G^q pentru valoarea proprie 2^p prin algoritmul lui Franazsek.

Definiție: Pentru o matrice $\boldsymbol{A}$ și un număr $\boldsymbol{L}$, $\bar{v}$ este un **vector propriu aproximativ** (AEV) asociat valorii proprii $\boldsymbol{L}$ dacă verifică inegalitatea:

$$A\bar{v} \geq L\bar{v} \tag{V.9}$$

Algoritmul de determinare a vectorilor proprii aproximativi (Franaszek)

❶ Se inițializează valoarea proprie $\boldsymbol{L} = 1$;

❷ Se inițializează contorul de pași: $\boldsymbol{k} = 0$;

❸ Se inițializează vectorul $v^{(0)} = [L; L; ...; L]$;

❹ Se calculează componentele vectorului la pasul următor:

$$v_i^{(k+1)} = \min\left(v_i^{(k)} ; \frac{1}{L}\left[\sum_j a_{ij} v_j^{(k)}\right]\right); \tag{V.10}$$

Notație: [x] semnifică partea întreagă a numărului x.

❺ Dacă $v^{(k+1)} \neq v^{(k)}$, atunci k = k+1 și se reia pasul 3;

❻ Dacă $v^{(k+1)} = v^{(k)}$, atunci $v = v^{(k)}$;

❼ Dacă $\boldsymbol{v}$ este identic nul atunci se incrementează L = L+1 și se reia algoritmul de la pasul 2.

Definiții:

Ponderea unei stări i (w(i) - *state weight*) dintr-un graf de codare-bloc binară $\boldsymbol{H}$, cu matricea de adiacență $\boldsymbol{B}$ și rata de codare $\boldsymbol{p/q}$, este egală cu valoarea componentei de pe poziția $\boldsymbol{i}$ din vectorul propriu aproximativ (VPA) al matricii $\boldsymbol{B}$ asociat valorii proprii $\boldsymbol{2^p}$.

Ponderea unei săgeți este egală cu ponderea stării terminale a acesteia:

$\bar{e} : s_i \rightarrow s_j \quad w(\bar{e}) = w(s_j)$.

Norma setului de săgeți de ieșire dintr-o stare dată este definită ca suma ponderilor săgeților componente.

Observație: Pentru ca divizarea stărilor să fie făcută eficient, în sensul creșterii maxime posibile a gradelor de ieșire din fiecare stare, trebuie calculate ponderile stărilor și divizate stările de pondere maximă. Nu este utilă divizarea stărilor de pondere unitară.

Exemplu: Să determinăm vectorul propriu aproximativ al matricii A^3 pentru valoarea proprie 4.

$$L = 2; v^{(0)} = \begin{Bmatrix} 2 \\ 2 \end{Bmatrix};$$

$$v^{(1)} = \left[\frac{1}{4}\begin{pmatrix} 3 & 2 \\ 2 & 1 \end{pmatrix}\begin{pmatrix} 2 \\ 2 \end{pmatrix}\right] = \left[\frac{1}{4}\begin{pmatrix} 10 \\ 6 \end{pmatrix}\right] = \begin{pmatrix} 2 \\ 1 \end{pmatrix} \neq v^{(0}$$

$$v^{(2)} = \left[\frac{1}{4}\begin{pmatrix} 3 & 2 \\ 2 & 1 \end{pmatrix}\begin{pmatrix} 2 \\ 1 \end{pmatrix}\right] = \left[\frac{1}{4}\begin{pmatrix} 8 \\ 5 \end{pmatrix}\right] = \begin{pmatrix} 2 \\ 1 \end{pmatrix} = v^{(1)} \neq ($$

Deci, $VPA(A^3; 2^2) = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$; $w_1 = v_1 = 2; w_2 = v_2 = 1$

Este necesară divizarea stării s_1.

Algoritmul de divizare a stărilor (*State-splitting algorithm*)

Într-un graf de codare ***G***, cu rată de codare ***p/q*** și vector propriu aproximativ ***v***, o stare de pondere $v_i \geq 2$ se poate diviza în două stări (***i1; i2***) prin împărțirea setului săgeților de ieșire;

$$\boldsymbol{E_i = E_{i1} \cup E_{i2}}$$

astfel încât normele celor două subseturi să verifice relațiile:

$$\|E_{i1}\| \geq 2^p ; \|E_{i2}\| \geq 2^p ; \|E_{i1}\| + \|E_{i2}\| = \|E_i\| = v \qquad \text{(V.11)}$$

Pentru construcția noului graf se respectă următoarele **reguli de divizare a stărilor**:

D1. Dacă ***e*** este o săgeată care pleacă dintr-o stare ***j*** și intră într-o stare ***k***, ambele neadiacente stării ***i***, atunci această săgeată rămâne neschimbată în noul graf:

$e : j \to k$

D2. Unei săgeți de intrare în starea ***i*** din starea ***j*** îi corespund în noul graf două săgeți:

$e_1 : j \to i_1$
$e_2 : j \to i_2$

D3. Pentru o săgeată de ieșire din starea ***i*** spre o stare ***j***, în funcție de subsetul de săgeți căreia îi aparține (***Ei1, Ei2***), va apare în noul graf o săgeată din starea ***i1*** sau ***i2*** spre starea ***j***:

- dacă $e \in E_{i1} \Rightarrow e : i_1 \to j$;
- dacă $e \in E_{i2} \Rightarrow e : i2 \to j$;

D4. În cazul buclelor din graf, se procedează astfel:

- dacă $e \in E_{i1}$ atunci apar $e_1 : i_1 \to i_1 ; e_2 : i_1 \to i_2$;
- dacă $e \in E_{i2}$ atunci apar $e_1 : i_2 \to i_1 ; e_2 : i_2 \to i_2$.

Exemplu: Pentru codul considerat, vom diviza starea s_1 în două stări denumite s_1 și s_3 cu seturile de săgeți de ieșire:

$E(1) = \{111; 101\}; \|E(1)\| = 2 + 2 = 4;$
$E(3) = \{011; 010; 110\}; \|E(3)\| = 2 + 1 + 1 = 4$

Rezultă graful H cu 3 stări din figura V.6. Observăm că în noul graf, toate stările au cel puțin 4 săgeți de ieșire, deci graful H poate fi folosit pentru codarea cu rată 2/3.

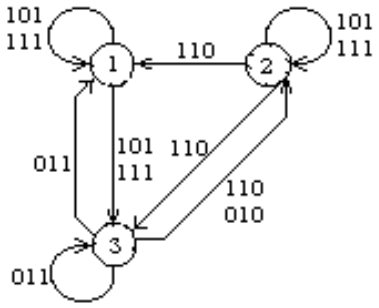

Fig. V.6 Graful H obținut prin divizarea stărilor

Observație: Reducerea complexității codorului RLL este posibilă prin scăderea numărului de stări din graful de codare aplicând **algoritmul de reunire a stărilor**. În acest mod scade numărul de biți necesar pentru exprimarea variabilei de stare. Condițiile și regulile de reunire a stărilor dintr-un graf sunt specificate în următorul algoritm:

Algoritmul de reunire a stărilor (*State-merging algorithm*)

Dacă setul săgeților de ieșire dintr-o stare ***i*** este inclus în setul de ieșire al altei stări ***j***, atunci se poate elimina starea ***j*** respectându-se următoarele reguli:

R1. Orice săgeată incidentă în starea ***j*** va deveni săgeată de intrare în starea ***i***.

R2. Buclele stării ***j*** devin bucle în starea ***i***.

R3. Se șterg toate săgețile de ieșire din starea ***j***.

R4. Se elimină starea ***j***.

Observație: Compararea săgeților de ieșire se face pe baza etichetei asociate și a stării terminale a fiecăreia.

Exemplu: Se observă în figura V.6 că săgețile de ieșire din starea s_1 sunt incluse în setul săgeților de ieșire din starea s_2, deci cele două stări pot fi reunite. Va rezulta un graf H' cu două stări (renotate cu un bit de stare). Pentru a obține graful de codare se vor elimina anumirte săgeți din graf astfel încât fiecare stare să aibă exact 2^p săgeți de ieșire

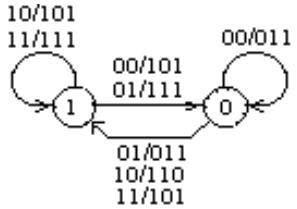

Fig. V.7 FSTD a codului 2/3 RLL(0;1)

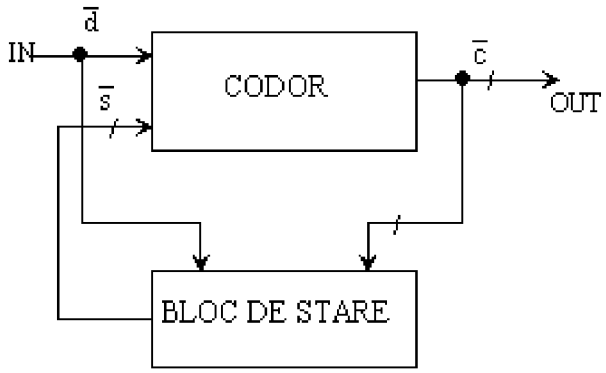

Fig.V.8 Schema de principiu a codorului RLL

care sunt apoi asociate cu cele 2^p combinații binare posibile de intrare în codor. Se obține astfel diagrama cu număr finit de stări a codului (FSTD) (fig.V.7).

Sinteza codecului RLL se realizează pe baza tabelelor de codare și de decodare.

Codorul RLL este compus dintr-un bloc secvențial de sinteză a variabilelor de stare și un bloc propriu-zis de codare combinațional (Fig.V.8). Se obține astfel un automat complex.

Exemplu: Graful de codare din figura V.7 este utilizat pentru scrierea tabelului de codare a codului 2/3 RLL(0;1) (Tabel V.1).

Ecuațiile de codare și de sinteză a stării codorului sunt scrise în formă minimizată, aplicând metoda Veitch-Karnaugh:

$c_{1n} = (s'_n d_{1n} d'_{2n})'$; $c_{2n} = (s_n d'_{2n})'$; $c_{3n} = ((s'_n d_{1n})'$; $s_{n+1} = (d'_{1n}(d_{2n} s'_n)')'$.

Structura codorului 2/3 RLL(0;1) este prezentată în figura V.9.

Tabelul V.1 Codarea 2/3 RLL(0;1)

$d_{1n}d_{2n}$	s_n	$c_{1n}c_{2n}c_{3n}$	s_{n+1}
00	0	011	0
	1	101	0
01	0	011	1
	1	111	0
10	0	110	1
	1	101	1
11	0	010	1
	1	111	1

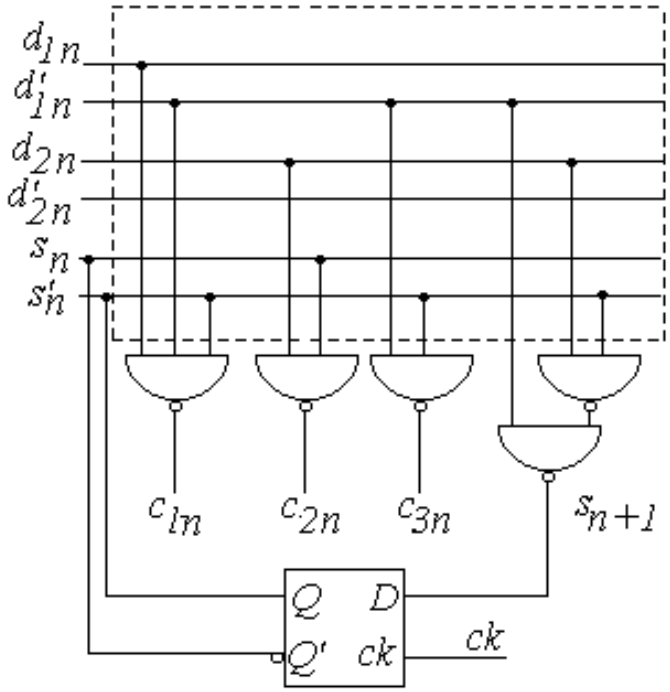

Fig.V.9 Codor 2/3 RLL(0;1)

Decodarea RLL se realizează independent de stare, pe baza unei 'ferestre de decodare' în care sunt incluse:

◆ ***m*** cuvinte de cod anterior recepționate (***m*** reprezintă **memoria** codului);

◆ cuvântul de cod curent;

◆ ***a*** cuvinte de cod ulterioare (***a*** reprezintă **anticipația** codului).

Acest tip de decodor (*sliding-block decoder*) limitează propagarea unei erori la cel mult $W = m + a + 1$ cuvinte decodate (W - lățimea ferestrei de decodare).

Exemplu: Decodorul 2/3 RLL(0;1) folosește o 'fereastră de decodare' de 2 cuvinte recepționate, respectiv 6 biți codați (Tabelul V.2), codul lucrând cu memorie 0 și anticipație 1. Sinteza decodorului (fig.V.10) s-a făcut pe baza ecuațiilor de decodare:

$d_{1n} = ((A_n B_n)'(C_n D_n)'(A_n D_n)')'$; $d_{2n} = c_{2n}(C'_n(c'_{1n}(c_{3n}D'_n)')')'$

cu

$A_n = (c'_{2n} + c'_{3n})'; B_n = c_{2(n+1)}; C_n = (c'_{1n} + c'_{3n})'; D_n = (c'_{1(n+1)} + c'_{3(n+1)})$.

Tabel V.2

Decodarea 2/3 RLL(0;1)

$c_{1n}c_{2n}c_{3n}$	$c_{1(n+1)}c_{2(n+1)}c_{3(n+1)}$	$d_{1n}d_{2n}$
010	---	11
011	101; 111	01
	010; 011; 110	00
101	101; 111	10
	010; 011; 110	00
110	---	10
111	101; 111	11
	010; 011; 110	01

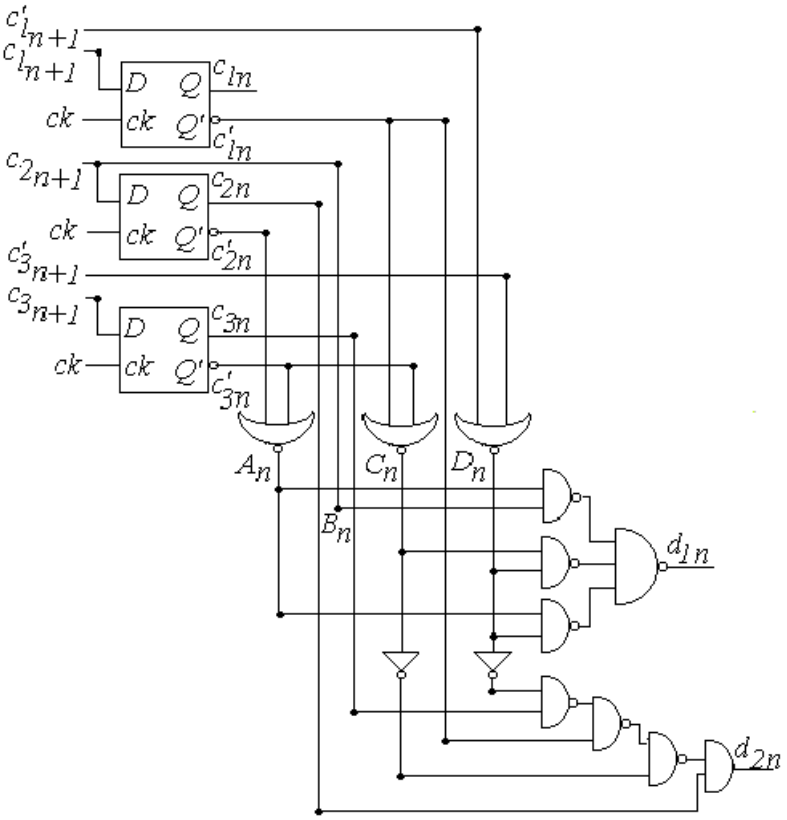

Fig.V.10 Decodor 2/3 RLL(0;1)

Algoritmul de construcție a codului RLL [Sch91]:

❶ Se construiește graful G asociat canalului conform constrângerilor impuse;

❷ Se scrie matricea de adiacență A a grafului și se soluționează ecuația caracteristică a acesteia;

❸ Se calculează capacitatea canalului cu constrângeri (C);

❹ Se alege rata de codare: $\mathbf{R = p/q \leq C}$;

❺ Se construiește graful G^q și i se calculează matricea de adiacență $B = A^q$;

❻ Se calculează vectorul propriu aproximativ VPA (AEV-*approximated eigenvalue vector*) minim nenul al matricii B corespunzător valorii proprii 2^p (**algoritmul lui Franaszek**) și se stabilesc ponderile stărilor;

❼ Se elimină stările de pondere nulă obținându-se un nou graf H;

❽ Dacă VPA nu este unitar, atunci se divizează stările de pondere maximă, pe baza **algoritmului de divizare a stărilor** (*state-splitting algorithm*). Se reia pasul 6.

❾ Dacă VPA asociat grafului H este identic cu vectorul-unitate atunci se păstrează exact 2^p săgeți de ieșire din fiecare stare care se asociază cu combinațiile binare de intrare de p biți, obținându-se astfel graful de codare. Eventual anumite stări pot fi eliminate din graf, dacă satisfac anumite relații de incluziune unele față de altele, pe baza **algoritmului de reunire a stărilor** (*state-merging algorithm*).

Observație: Minimizarea numărului de stări din graf conduce la micșorarea numărului de biți de stare, reducerea dimensiunilor tabelelor de codare/decodare și la scăderea complexității codec-ului.

În final, se face sinteza structurii logice de codare/decodare utilizând metode specifice de minimizare (Veitch-Karnaugh). Decodorul se implementează pe baza tabelului de decodare, cu memorie și anticipație finite, fără a utiliza variabile de stare.

V.3 Aplicații

P1. Construiți codul 1/2 RLL(0;6;2).

P2. Calculați capacitatea canalelor cu constrângeri simetrice pe maximum 4 biți.

P3. Analizați codul Miller ca un cod RLL.

P4. Cum se pot transforma secvențele de cod convoluțional în secvențe RLL? Analizați codul convoluțional binar 1/2 C(1110; 1101). Ce cod RLL obțineți?