Introducere în Matlab

Matlab-ul este un limbaj de nivel foarte înalt care prezintă performanțe deosebite în ceea ce priveşte calculul tehnic (Matlab reprezintă o prescurtare a cuvintelor “Matrix laboratory”). Pe lângă interpretorul de comenzi sunt prezente instrumente cum ar fi cele pentru vizualizarea datelor, prelucrarea imaginilor şi sunetelor, analiza circuitelor electrice, etc.

Spre deosebire de alte limbaje, elementele de bază cu care se lucrează sunt vectorii. Utilizatorul poate defini şi folosi vectori ai căror dimensiuni inițiale nu trebuie specificate. Problemele legate de gestionarea memoriei la operațiile care presupun creşterea dimensiunii unui vector se fac automat, transparent utilizatorului, ceea ce este un avantaj față de limbaje de programare cum ar fi C sau C++.

Funcțiile specifice unui anumit domeniu sunt grupate în colecții de funcții sau “toolboxes”. Acestea uşurează foarte mult folosirea programului în scop educațional sau de cercetare deoarece utilizatorul se poate concentra direct pe aplicarea unei serii de operații asupra unui set de date fără a se îngriji exclusiv de definirea acestor operații. Există mai multe colecții de funcții specifice domeniului electronicii cum ar fi cea pentru prelucrarea de semnale sau pentru domeniul comunicațiilor.

Fereastra principală a programului permite accesul direct la interpretorul de comenzi. Acesta este un instrument care execută o secvență de cod, linie cu linie. Secvența de cod poate fi introdusă direct de la tastatură, iar după fiecare linie se apasă tasta Enter sau poate fi scrisă într-un fişier de tip text, care se salvează cu extensia “.M” şi se execută prin simpla scriere a numelui fişierului.

Limbajul Matlab respectă principiile programării structurale, astfel că există o foarte mare asemănare între sintaxa şi structurile sale cu cea a limbajului C.

Noțiuni despre Simulink

Simulink este un pachet de programe pentru modelarea, simularea şi analizarea sistemelor dinamice. Pot fi simulate atât sisteme liniare cât şi neliniare, modelate în timp continuu, discret sau într-o combinație a celor două. Sistemele pot avea porțiuni eşantionate cu frecvențe de eşantionare diferite.

Pentru modelarea de sistem este furnizată o interfață grafică intuitivă. Blocurile sunt plasate şi interconectate cu ajutorul mouse-ului ceea ce reprezintă un mare avantaj (față de scrierea directă a ecuațiilor diferențiale ce definesc un sistem). Simulink oferă o colecție de blocuri cum ar fi: generatoare de semnal, instrumente de vizualizare, blocuri care realizează funcții matematice, componente liniare şi neliniare, etc. Setul de blocuri furnizat poate fi extins oricând cu noi blocuri – este furnizată documentație completă despre felul cum se poate crea un nou bloc.

Mai multe blocuri pot fi grupate oricând într-un bloc nou, oferind astfel posibilități extinse de analiză la un nivel superior de organizare. După definirea unui model nou, simularea se poate efectua atât în mod grafic cât şi cu ajutorul interpretorului. Cele două instrumente sunt legate între ele şi astfel se poate opta pentru orice modalitate de analiză.

Pentru lansarea programului Simulink se tastează în mediul Matlab comanda: '*simulink*'.

Fereastra care se deschide conține toate blocurile disponibile grupate pe categorii (figura 1.1 -pentru Matlab 6.0 sub sistemul de operare Windows). Dacă se deschide biblioteca "Simulink" se observă următoarele zone:

Continuous - blocuri ce furnizează funcții specifice circuitelor analogice: derivare, integrare, funcție de transfer, întârziere în domeniul timp, etc.;

Discrete - blocuri ce furnizează funcții specifice circuitelor discrete: funcție de transfer discretă, filtru discret, întârziere în domeniul timp cu un pas, integrator discret, etc.;

Function & Tables - blocuri care permit extinderea setului de blocuri existent cu blocuri create de utilizator;

Math Operations - funcții matematice de ordin general: sumă, produs, modul, amplificare, fază, funcții trigonometrice, etc.;

Nolinear - funcții specifice circuitelor neliniare;

Signal Routing - blocuri necesare pentru definirea semnalelor: masă, multiplexor de mai multe semnale, funcții pentru preluarea şi salvarea valorilor în spațiul Matlab, etc.;

Sinks - aparate de măsură: multimetru, osciloscop, graphic XY, etc.;

Sources - surse de semnal: generatoare de semnal sinusoidal, triunghiular, dreptunghiular, zgomot, rampă, pulsuri, etc.

Se poate obține o descriere detaliată a fiecărui bloc dacă se selectează şi se urmăreşte în partea de sus a ferestrei explicațiile aferente. O structurare asemănătoare se întâlneşte în cadrul fiecărui modul. Celelalte module introduc blocuri noi care realizează funcții complexe prin combinarea blocurilor elementare descrise anterior.

Pentru plasarea unui bloc nou în schemă, acesta este selectat în fereastra bibliotecilor şi se pozitionează în schemă folosind "drag and drop".

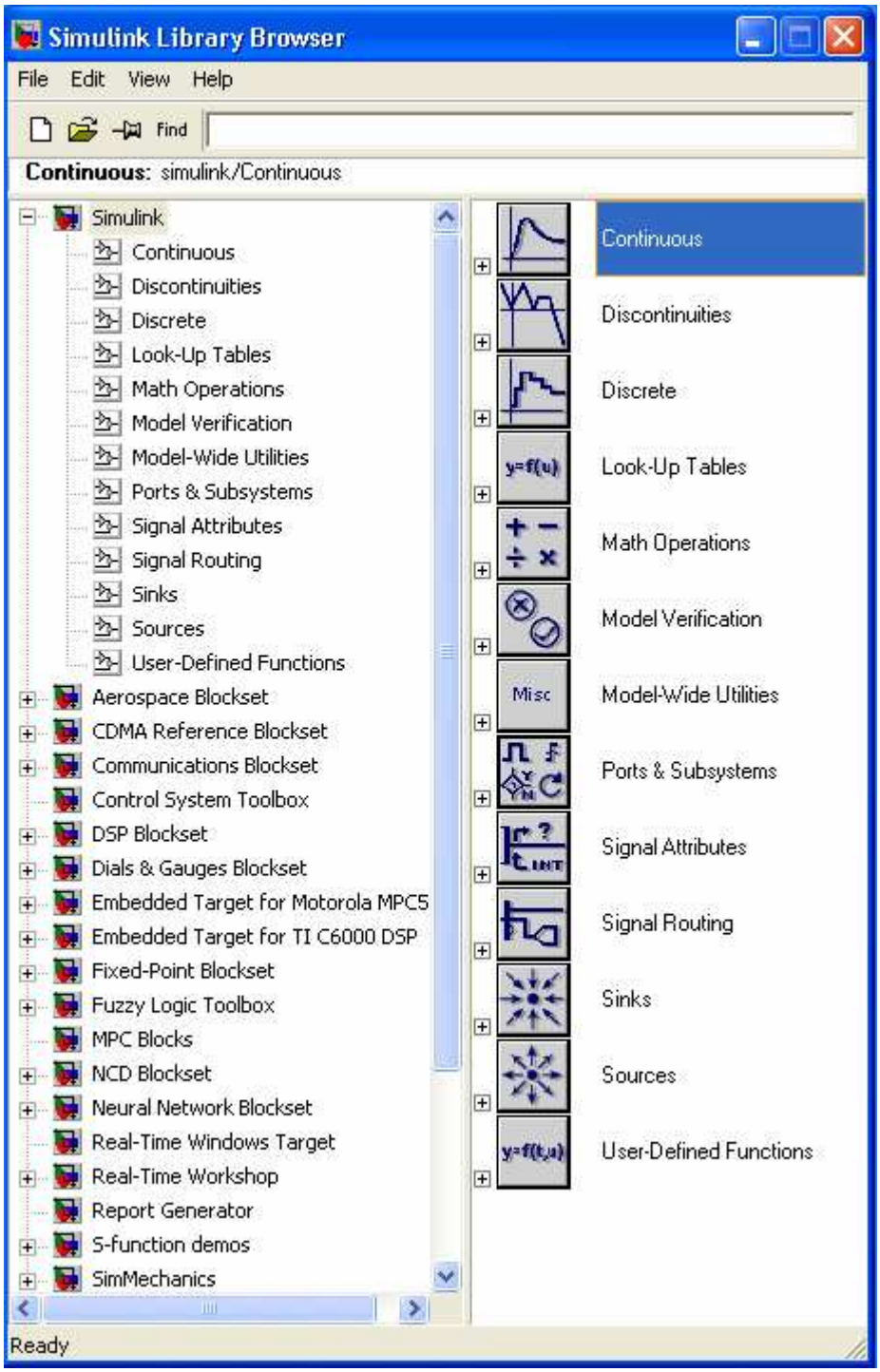

Fig. 1.1 Biblioteca de blocuri a Simulink-ului

Coduri Hamming

Codurile Hamming reprezintă o clasă aparte a codurilor bloc corectoare de erori. Aceste coduri şi variantele derivate din ele sunt larg utilizate pentru corecția erorilor în comunicații digitale și în sistemele de stocare a datelor.

Pentru orice întreg pozitiv $m \geq 3$, există un *cod Hamming* având urmatorii parametri:

- lungimea codului: $n = 2^m - 1;$
- numărul simbolurilor de informație: $k = 2^m - m - 1;$
- numărul simbolurilor de control: $n - k = m;$
- capacitatea de corecție: $t = 1 (d_{\min} = 3).$

Matricea H are următoarea formă:

$$H = [I_m Q],$$

unde I_m este matricea unitate de ordinul *m*.

Matricea generatoare a codului este:

$$G = [Q^T I_k],$$

unde I_k este matricea unitate de ordinul *k*.

Aplicație coduri Hamming

Codurile Hamming fac parte din categoria codurilor bloc. Aceste coduri sunt coduri perfecte, corectoare de o eroare, corectează toate structurile de erori simple dar nici o combinație de erori duble, şi detectează toate structurile de eroare cu două sau mai puține erori.

Pentru a arăta utilitatea codului Hamming utilizăm următorul model:

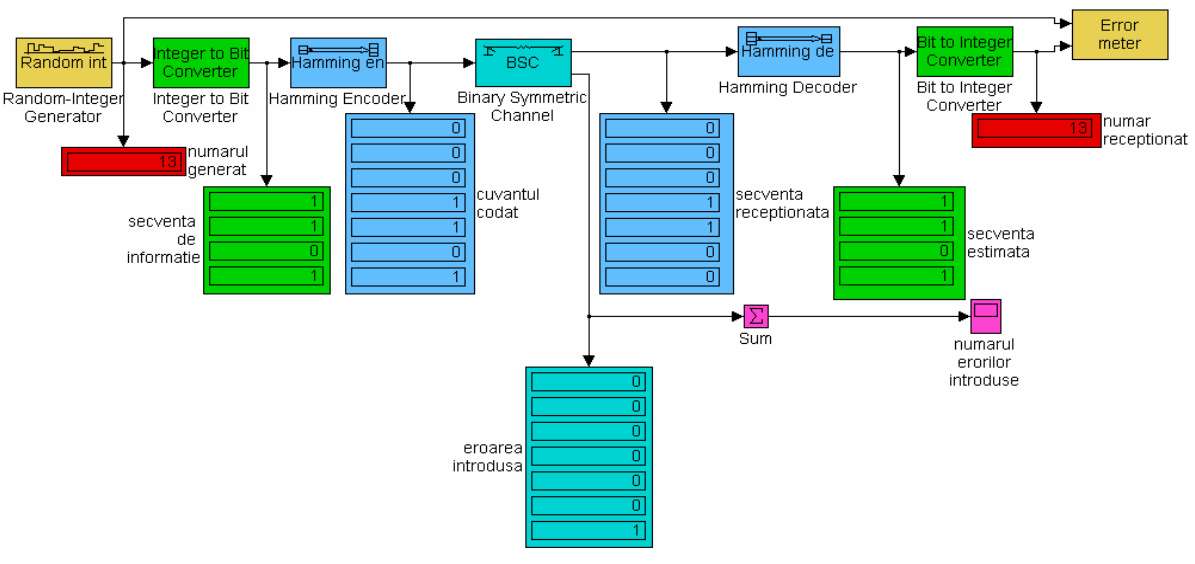

Fig. 1.2

Folosim urmatoarele blocuri:

- **Random-Integer Generator**: generează numere întregi distribuite în intervalul [0, M-1]. Parametrii blocului sunt:
  - ‘M-ary number’ este 2^4 deoarece codul Hamming utilizat este **C(7,4)** şi numerele generate sunt reprezentate în binar pe 4 biți.
  - ‘Initial seed’ este [12345]. Modificând acest parametru se modifică secvența de numere generate.
  - ‘Sample time’ este 1. Generează câte un număr la fiecare secundă.
- **Integer to Bit Converter**: transformă un vector de întregi într-un vector de biți. Parametrul blocului este:
  - ‘Number of bits per integer’ este 4.
- **Hamming Encoder**: crează un cod Hamming din datele vectorului binar. Parametrii blocului sunt:
  - ‘Codeword length N’ este 7 .
  - ‘Message length K’ este 4 deoarece se utilizează codul C(7,4).
- **Binary Symmetric Channel**: introduce erori binare. Parametrii blocului sunt:
  - ‘Error probability’ este 0.29, pentru a introduce o singură eroare.
  - ‘Input vector length’ este 7 deoarece cuvântul de cod cu care se adună este reprezentat pe 7 biți.
  - ‘Initial seed’ este 1234.
  - ‘Sample time’ este 1 pentru a se genera un eşantion la fiecare secundă.
- **Hamming Decoder**: decodează un cod Hamming pentru a reface vectorul binar transmis. Parametrii blocului sunt:
  - ‘Codeword length N’ este 7 .
  - ‘Message length K’ este 4 deoarece se utilizează codul C(7,4).

- **Bit to Integer Converter**: transformă un vector de biți într-un vector de întregi. Parametrul blocului este:
  - 'Number of bits per integer' este 4.
- **Error Meter**: compară semnalele de la intrare, le afişează şi evaluează rata de eroare. Parametrii blocului sunt:
  - 'Bit per symbol' este 4 deoarece utilizează 4 biți pentru fiecare simbol transmis.
  - 'Number of digits on display' este 20 deoarece afişează 20 de simboluri.
  - 'Delay between input (1st port) and output (2nd port)' este 0
  - 'Sample time' este 1 deoarece se consideră un eşantion la fiecare secundă.
- **Sum**: afişează suma elementelor de la intrare. Parametrii blocului sunt:
  - 'Icon shape' este rectangular.
  - 'List of signs' este |+.
- **Scope**: afişează numărul de erori.
- **Display**: afişează valoarea de la intrare.

Primul număr generat aleator este **13**.

Numărul 13 este transformat în binar devenind [**1101**] care reprezintă secvența de informație.

Codului C(7,4) îi corespunde matricea generatoare

$$G=\begin{bmatrix} g_0 \\ g_1 \\ g_2 \\ g_3 \end{bmatrix}=\left[\begin{array}{cccc:cccc} 1 & 1 & 0 & \vdots & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & \vdots & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & \vdots & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & \vdots & 0 & 0 & 0 & 1 \end{array}\right].$$

Calculăm cuvântul codat care este dat de: v=i*G

$$v=1\cdot g_0+1\cdot g_1+0\cdot g_2+1\cdot g_3=[1101000]+[0110100]+[1010001]=\left[\overbrace{\underbrace{000}_{n-k=3}:\underbrace{1101}_{k=4}}^{n=7}\right]$$

Blocul **Binary Symmetric Channel** introduce eroarea e**=[0000001]** care se adună la cuvântul codat formând secvența recepționată r=[**0001100]**.

Blocul **Hamming Decoder** decodează secvența recepționată astfel:

Pentru codul C(7,4), considerat, matricea de control va fi

$$H=\left[\begin{array}{cccc:cccc} 1 & 0 & 0 & \vdots & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & \vdots & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & \vdots & 0 & 1 & 1 & 1 \end{array}\right].$$

Sindromul este dat de :

$$s=Hr^T=[1 \quad 0 \quad 1]^T.$$

Fie cuvântul eroare $e=[e_0\ e_1\ e_2\ e_3\ e_4\ e_5\ e_6]$. Atunci :

$$s=Hr^T \Rightarrow \left\{\begin{array}{l} e_0+e_3+e_5+e_6=1 \\ e_1+e_3+e_4+e_5=0 \\ e_2+e_4+e_5+e_6=1 \end{array}\right\}$$. Rezultă că eroarea este e=[0000001].

Știind eroarea introdusă se reface secvența de informație transmisă.

$$v'=r+e=[0001100]+[0000001]=[0001101]=v.$$

Rezultă secvența estimată care este [1101].

Se reface astfel numărul 13.

Blocul **Error Meter** va afișa următoarele rezultate:

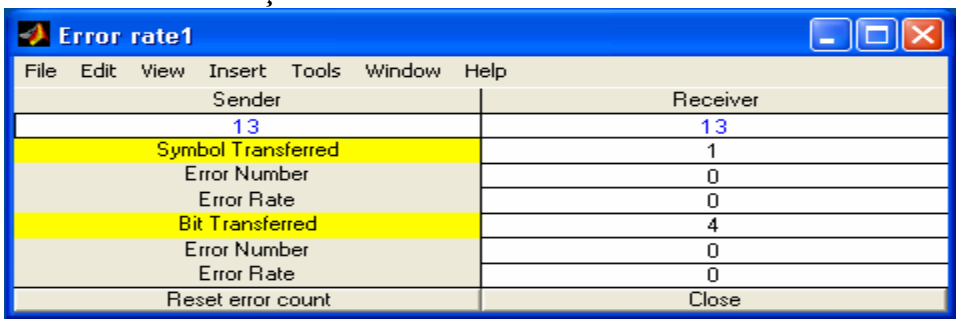

Fig. 1.3

Modificând în blocul **Binary Symmetric Channel** parametrul Error probability putem introduce două erori sau chiar nici una.

Pentru a nu introduce erori parametrul 'Error probability' este 0.2 .

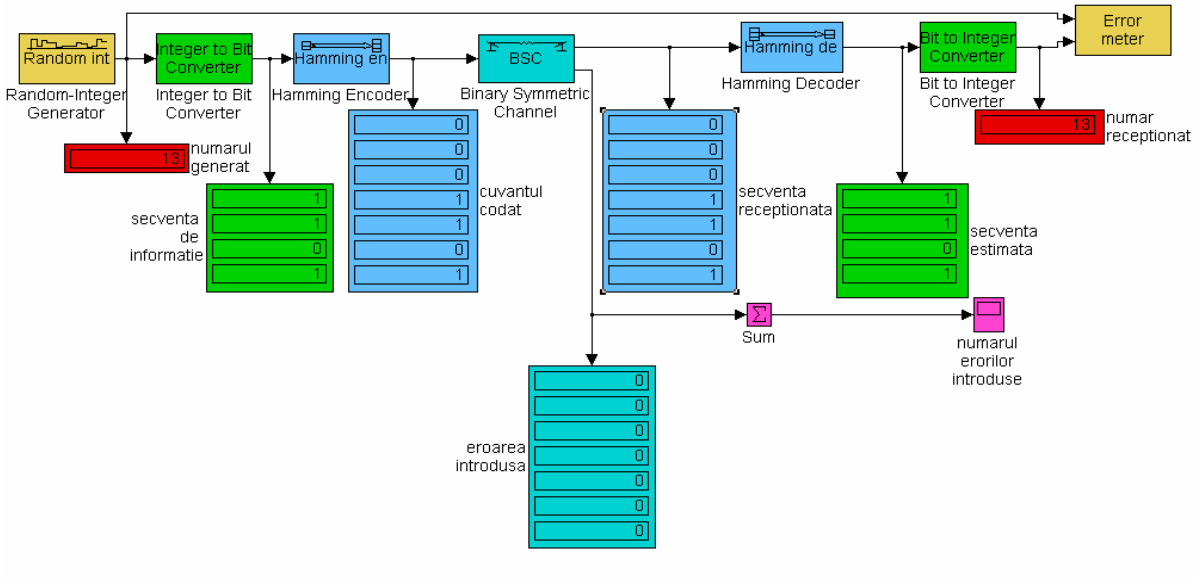

Fig. 1.4

În cazul modelului de mai sus nu a fost introdusă nici o eroare astfel încât sindromul calculat este s=0 de unde rezultă v = r (secvența de informație este egală cu secvența recepționată).

Pentru a introduce două erori parametrul ‘Error probability’ este 0.3 . În acest caz decodorul nu poate reface secvența de informație transmisă deoarece codul Hamming este cod corector de o eroare producându-se o eroare de decodare.

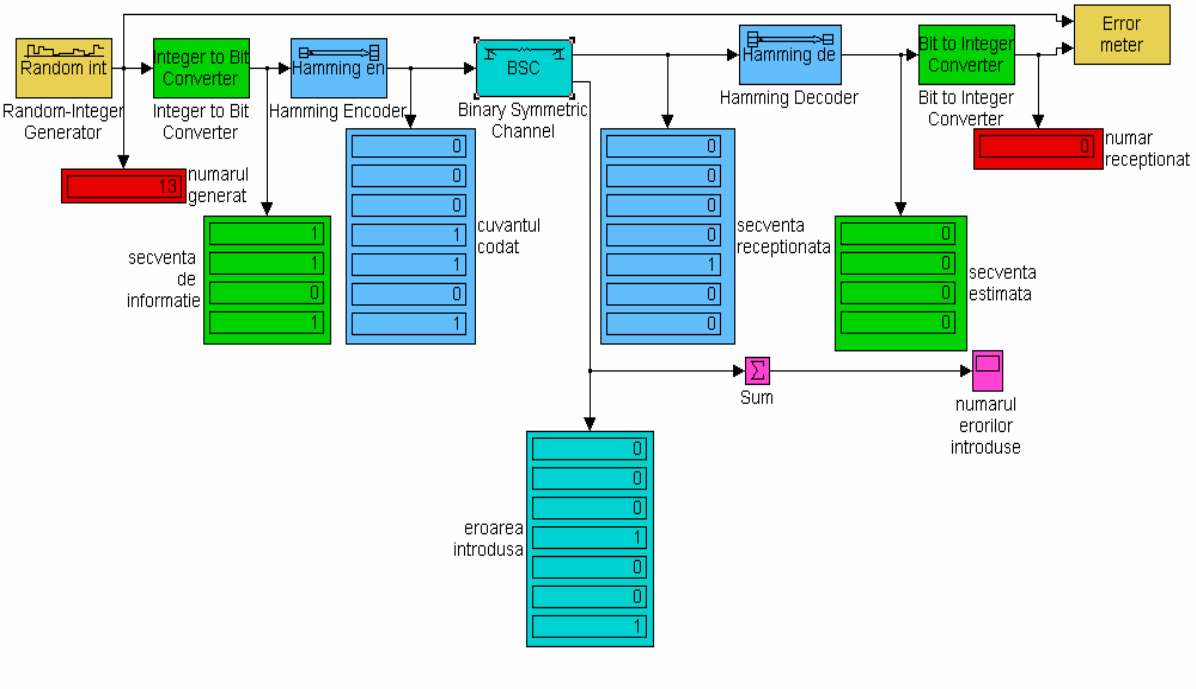

Fig. 1.5

Pentru ca **Random-Integer Generator** să genereze un singur simbol am stabilit timpul de simulare ca fiind de la ‘0’ la ‘0.5’ deoarece primul simbol se generează la ‘0’ şi al doilea la ‘1’.

Pentru a genera mai multe simboluri timpul de simulare este stabilit de la 0 la 19 generându-se astfel 20 de numere. Numărul erorilor introduse pentru cele 20 de numere sunt reprezentate pe diagrama blocului **Scope.**

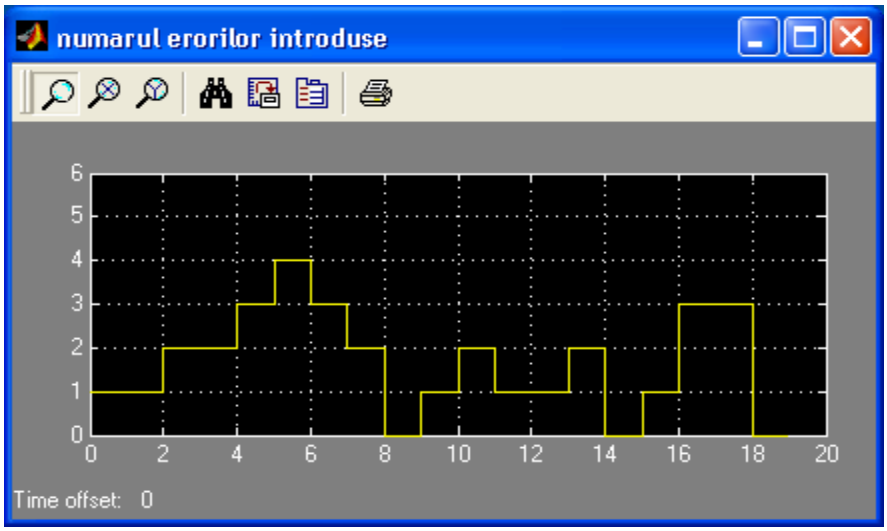

Fig. 1.6

În acest caz blocul **Error Meter** va afişa următoarele rezultate:

Error rate1

File Edit View Insert Tools Window Help

Sender	Receiver
13	13
10	10
10	9
5	3
10	12
8	9
10	1
15	11
6	6
2	2
1	9
14	14
1	1
10	7
13	13
7	7
4	10
5	0
3	3
2	2
Symbol Transferred	20
Error Number	10
Error Rate	0.5
Bit Transferred	80
Error Number	20
Error Rate	0.25
Reset error count	Close

Fig. 1.7

Comparând rezultatele reprezentate în fig. 1.6 şi 1.7 se observă că la transmiterea primului simbol, 13, a fost introdusă o singură eroare care a fost corectată; la transmiterea celui de al doilea simbol, 10, a fost introdusă tot o singură eroare care a

fost corectată; la transmiterea celui de al treilea simbol, 10, au fost introduse două erori care nu au putut fi corectate; la transmiterea celui de al patrulea simbol, 5, au fost introduse două erori care nu au putut fi corectate; ş.a.m.d.

Desfasurarea lucrarii:

1. Sa se simuleze schema de mai sus notand efectele acesteia la modificarea parametrului **Error probability** din blocul BCS si sa se specifice rolul acestuia in schema.
2. Sa se modifice numarul de simboluri pentru care se face simularea mentionand care este parametrul asupra caruia trebuie sa actionam.
3. Sa se repete simularea modificandu-se secventa de numere generate si sa se specifice parametrul care trebuie modificat.