**Obraz zawierający logo, tekst, Czcionka, Grafika

Opis wygenerowany automatycznie**

**Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie**

**WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI,**

**INFORMATYKI I INŻYNIERII BIOMEDYCZNEJ**

Raport

**Podstawy telekomunikacji**

Autor: Grzegorz Lis, Karolina Sawosz

Kierunek studiów: Mikroelektronika w Technice i Medycynie

# Kraków, 2024

**Laboratorium 7**

***Opis problemu***

Celem laboratoriów było zapoznanie się z zagadnieniem kodów konwolucyjnych oraz algorytmu Viterby’ego. Należało skonstruować enkoder oraz dekoder 3-, 4-bitowy. Zadaniem było także wykreślenie diagramu Trellis oraz dekodowanie przez diagram dla przypadku 3 bitów.

***Podstawy matematyczne rozwiązania***

Splot liniowy z odpowiedzią impulsową (kod splotowy):

x – wejściowy ciąg,

yj – wyjściowy ciąg z wyjścia j,

hj – odpowiedź impulsowa,

\* - splot

Zdolność poprawienia błędów:

d – minimalny dystans Hamminga między zakodowanymi sekwencjami

***Symulacje i obserwacje***

Enkoder i dekoder 3-bitowy

Tabela 1. Tabela stanów i wyjść enkodera.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Final state | |  |  |  |
|  | Initial State | |  |  |
| S1 (input) | S2 | S3 | X1 (xor S1 S2 S3) | X2 (S1 xor S3) |
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 |

Obraz zawierający diagram, tekst, linia, Plan

Opis wygenerowany automatycznie

Figure 1. Diagram stanu.

Tabela 2. Schemat stanów i wyjść dla wiadomości 10011.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Input S1 |  | 1 | 0 | 0 | 1 | 1 |  |  |
| S1-S2-S3 | 000 | 100 | 010 | 001 | 100 | 110 | 011 | 001 |
| Final State (S1 S2) | 00 | 10 | 01 | 00 | 10 | 11 | 01 | 00 |
| Output (X1 X2) | 00 | 11 | 10 | 11 | 11 | 01 | 01 | 11 |

Obraz zawierający diagram, Plan, linia, Rysunek techniczny

Opis wygenerowany automatycznie

Figure 2. Zakodowana wiadomość 10011 – schemat z simulink.

Obraz zawierający zrzut ekranu, linia

Opis wygenerowany automatycznie

Figure 3. Wynik symulacji simulink.

Listing 1. Implementacja enkodera w Matlabie.

d\_ff1 = 0;

d\_ff2 = 0;

d\_ff3 = 0;

i=0;

while ((d\_ff1~=0 || d\_ff2~=0 || d\_ff3~=0) || (i<length(input)))

i=i+1;

coded = append(coded,num2str(viterby\_code(d\_ff1,d\_ff2,d\_ff3)));

if i<=length(input)

d\_ff3 = d\_ff2;

d\_ff2 = d\_ff1;

d\_ff1 = input(i);

else

d\_ff3 = d\_ff2;

d\_ff2 = d\_ff1;

d\_ff1 = 0;

end

end

number = 10011;

input = dec2base(number,10) - '0';

coded = "";

coded = str2num(coded);

coded = dec2base(coded,10) - '0';

coded = int2bit(coded,2);

coded = reshape(coded,[14,1])' %% Zakodowana liczba

function y = xor3(a,b,c)

y = xor(a,xor(b,c));

end

function y = viterby\_code(d\_ff1,d\_ff2,d\_ff3)

x1 = xor3(d\_ff1,d\_ff2,d\_ff3);

x2 = xor(d\_ff1, d\_ff3);

y = x1\*2 + x2;

end

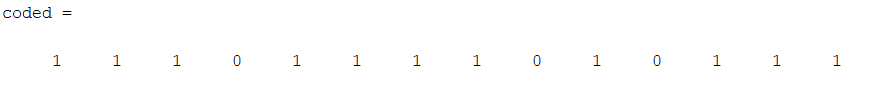


Figure 4. Zakodowana poprawnie wiadomość pzez algorytm Viterby'ego.

Obraz zawierający diagram, linia

Opis wygenerowany automatycznie

Figure 5. Diagram Trellis.

Obraz zawierający tekst, diagram, linia, Czcionka

Opis wygenerowany automatycznie

Figure 6. Dekodowanie za pomocą diagramu Trellis dla poprawnej wiadomości.

Listing 2. Dekodowanie wiadomości w Matlabie.

trellis = poly2trellis(3,[7 5]);

tb = 2;

decoded = vitdec(coded,trellis,tb,'trunc','hard') %%Zdekodowana liczba

Obraz zawierający tekst, Czcionka, zrzut ekranu, diagram

Opis wygenerowany automatycznie

Figure 7. Wiadomość zdekodowana z końcem 00.

Obraz zawierający diagram, mapa, tekst, linia

Opis wygenerowany automatycznie

Figure 8. Przykład dekodowania diagramem Trellis z błędem outputu.

Każdy przedstawiony sposób zwraca poprawny wynik. Dekodowanie diagramem Trellis polega na rozrysowaniu każdej możliwej ścieżki, a następnie, w przypadku występującego konfliktu, policzenie ile bitów zgadza się dla danej ścieżki. Ten sposób jest powtarzany iteracyjnie, aż do końca bitów wiadomości, wliczając w nią także „ogon” (czyli zera na końcu). Jest także możliwa inna interpretacja diagramu – liczone są wtedy bity, które się nie zgadzają, ale postępowanie jest podobne. Głównym komponentem układu do kodowania jest przerzutnik oraz bramka XOR.

Enkoder i dekoder 4-bitowy

Obraz zawierający diagram, linia, Plan, Rysunek techniczny

Opis wygenerowany automatycznie

Figure 9. Schemat układu.

Tabela 3. Tabela stanów i wyjść enkodera 4-bitowego.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Final state | |  |  |  |  |
|  | Initial State | |  |  |  |
| S1 (input) | S2 | S3 | S4 | X1 (xor S1 S2 S3) | X2 (xor S1 S2 S4) |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 |

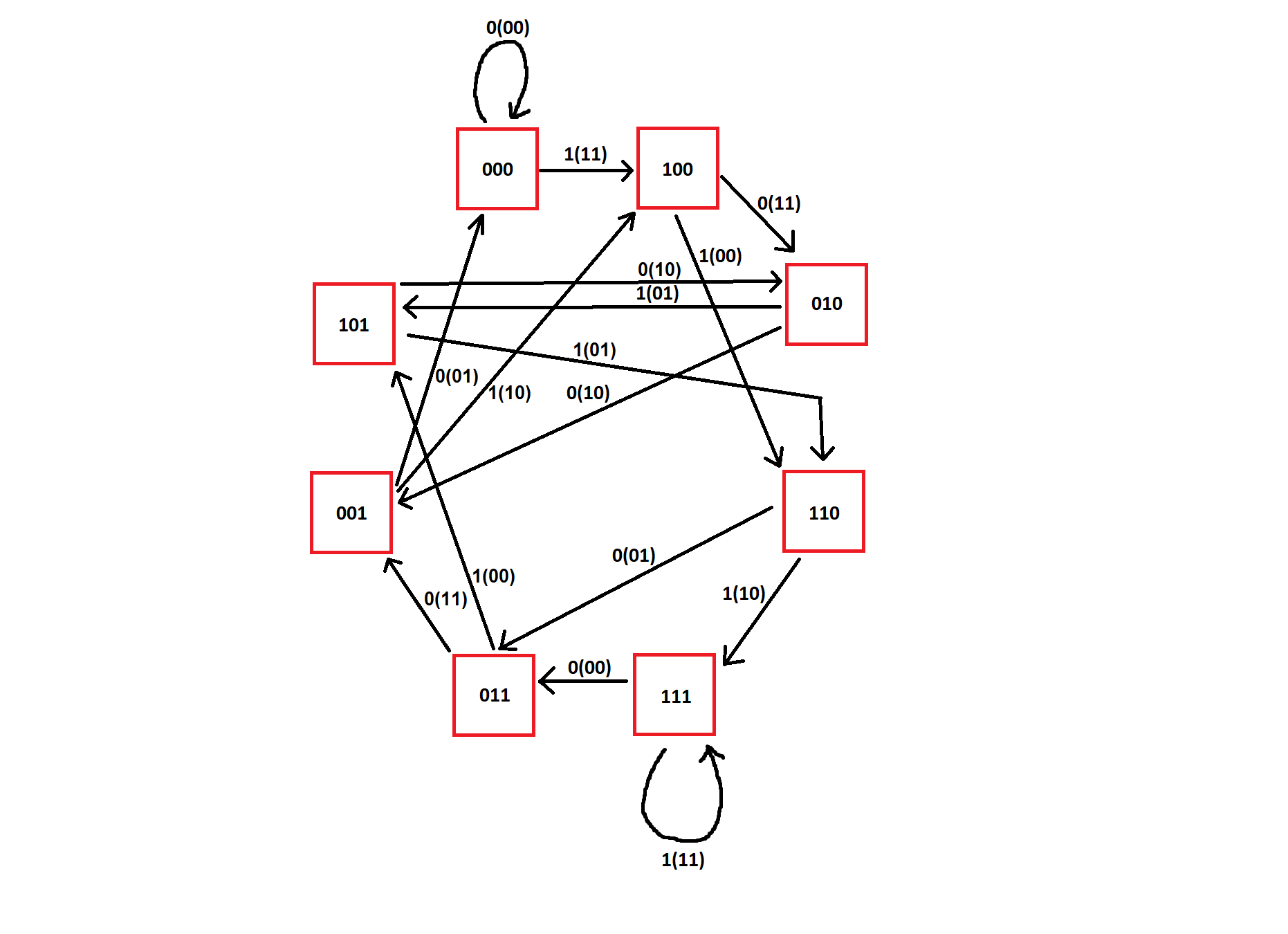


Figure 10. Diagram stanu.

Tabela 4. Schemat stanów i wyjść dla wiadomości 10011.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Input S1 |  | 1 | 0 | 0 | 1 | 1 |  |  |  |
| S1-S2-S3-S4 | 0000 | 1000 | 0100 | 0010 | 1001 | 1100 | 0110 | 0011 | 0001 |
| Final State (S1 S2 S3) | 000 | 100 | 010 | 001 | 100 | 110 | 011 | 001 | 000 |
| Output (X1 X2) | 00 | 11 | 11 | 10 | 10 | 00 | 01 | 11 | 01 |

Obraz zawierający diagram, linia, Wykres

Opis wygenerowany automatycznie

Figure 11. Diagram Trellis.

Obraz zawierający diagram, linia, Rysunek techniczny, Plan

Opis wygenerowany automatycznie

Figure 12. Zakodowana wiadomość 10011 – schemat z simulink.

Obraz zawierający zrzut ekranu, tekst, numer

Opis wygenerowany automatycznie

Figure 13. Wyniki symulacji - simulink.

Listing 1. Implementacja enkodera w Matlabie.

d\_ff1 = 0;

d\_ff2 = 0;

d\_ff3 = 0;

d\_ff4 = 0;

i=0;

number = 10011;

input = dec2base(number,10) - '0';

coded = "";

while ((d\_ff1~=0 || d\_ff2~=0 || d\_ff3~=0 || d\_ff4~=0) || (i<length(input)))

i=i+1;

coded = append(coded,num2str(viterby\_code(d\_ff1,d\_ff2,d\_ff3,d\_ff4)));

if i<=length(input)

d\_ff4 = d\_ff3;

d\_ff3 = d\_ff2;

d\_ff2 = d\_ff1;

d\_ff1 = input(i);

else

d\_ff4 = d\_ff3;

d\_ff3 = d\_ff2;

d\_ff2 = d\_ff1;

d\_ff1 = 0;

end

end

coded = str2double(coded);

coded = dec2base(coded,10) - '0';

coded = int2bit(coded,2);

coded = reshape(coded,[16,1])' %% Zakodowana liczba

function y = xor3(a,b,c)

y = xor(a,xor(b,c));

end

function y = viterby\_code(d\_ff1,d\_ff2,d\_ff3,d\_ff4)

x1 = xor3(d\_ff1,d\_ff2,d\_ff3);

x2 = xor3(d\_ff1,d\_ff2,d\_ff4);

y = x1\*2 + x2;

end

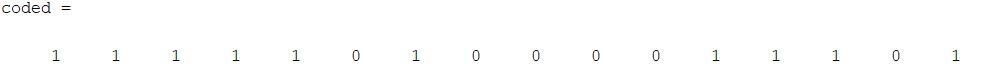


Figure 14. Zakodowana wiadomość przez algroytm Viterby'ego.

Listing 2. Dekodowanie wiadomości w Matlabie.

trellis = poly2trellis(4,[14 13]);

tb = 2;

decoded = vitdec(coded,trellis,tb,'trunc','hard')

Dla schematu enkodera 4-bitowego należy dodać jeden przerzutnik.

***Wnioski***

Algorytm Viterby’ego jest jednym z możliwych algorytmów dekodowania kodów splotowych. Działanie tego algorytmu oparte jest o wybór najbardziej prawdopodobnej ścieżki wiadomości. Proces ten można wyrazić iteracją. Jednym z zastosowań jest przywrócenie transmisji cyfrowych, pogorszonych przez zaszumiony kanał (na przykład transmisję radiową), polegający na odległości Hamminga w celu wydobycia najsłabszej miary między różnymi prawdopodobnymi wartościami.