|  |  |
| --- | --- |
| *Mikołaj Kisiel 242422*  *Mateusz Strumiłło 242539* | Rok akademicki *2022/23*  *wtorek, 14:00)* |

**Telekomunikacja i przetwarzanie sygnałów – LABORATORIUM**

Zadanie *1*

**Cel zadania**

*Opracowanie kodów korygujących dwa błędy w wiadomości ośmiobitowej jak i zaprogramowanie aplikacji wykorzystującej opracowany kod do zakodowania i odkodowania pliku.*

**Korygowanie pojedynczego błędu**

*W korekcji pojedynczego bitu / błędu posiadając 8 bitową wiadomość wystarczy użyć 4 bity parzystości. Po pierwotnym zakodowaniu wiadomość z 8 bitów zamienia się w wiadomość 12 bitową. Odkodowując wracamy do pierwotnej ilości bitów.*

*Reprezentujemy to za pomocą macierzy ‘H’, która:*

*- Nie posiada kolumny zerowej.*

*- Wiersze macierzy są różne względem siebie.*

*// 'word' type is 16-bits*

*// - thats because we need 8 bits for "messageBites" and*

*// 4 bits for parityBites, therefore*

*// 4 starting bits are useless.*

const size messageBites = 8;

const size parityBites = 4;

const array<const word, parityBites> matrixH {

0b0000'0111'0110'1000,

0b0000'1011'0011'0100,

0b0000'1101'1001'0010,

0b0000'1110'1100'0001,

};

**Korygowanie podwójnego błędu**

*Korygując dwa błędy / bity w wiadomości musimy przygotować do 8 bitowej wiadomości aż 8 bitów parzystości. Wynika to z zwiększenia odległości.*

*Powstała macierz dodatkowo względem do macierzy potrzebnej do zakodowania i odkodowania 1 błędu musi spełnić warunek:*

*- Zapewnione jest, że każdy wiersz jest liniowo niezależny.*

*// 'word' type is 16-bits*

*// - thats because we need 8 bits for "messageBites" and*

*// 8 bits for parityBites, therefore theres*

*// 0 starting bits that are useless. Look BitError1.*

const size messageBites = 8;

const size parityBites = 8;

const array<const word, parityBites> matrixH {

0b1110'0111'1000'0000,

0b1101'1111'0100'0000,

0b1010'1001'0010'0000,

0b0110'1011'0001'0000,

0b1001'0110'0000'1000,

0b0101'0101'0000'0100,

0b0011'0011'0000'0010,

0b0000'1111'0000'0001,

};

**Mechanizm zakodowania pojedynczego i podwójnego błędu**

*Mechanizm ten sprowadza się do przemnożenia ( modulo 2 ) przedstawionej macierzy przez wiadomość, którą chcemy wysłać. Otrzymamy w ten sposób 4 bitową wartość dla pojedynczego błędu bądź 8 bitową wartość dla podwójnego błędu. Bity te to obliczone wartości ‘parity’ dla bitów naszej wiadomości. Wystarczy jedynie umieścić obliczone bity za pierwotną wiadomością, aby ukończyć ich zakodowanie.*

**Mechanizm odkodowania dla pojedynczego błędu**

*Do odkodowania wiadomości zgodnie z logiką postepowania wystarczy ponownie przemnożyć*

*( modulo 2 ) macierz H przez naszą zakodowaną wiadomość. Otrzymana informacja cztero-bitowa jeśli przesłana bez uszkodzenia pojedynczego bitu wiadomości będzie równa ‘0’.*

*Jeżeli wynikiem mnożenia ( modulo 2 ) będzie inna liczba, to będzie ona w przypadku wystąpienia pojedynczego błędu informacją reprezentującą index bitu, który należy odpowiednio zamienić z 1 na 0, bądź z 0 na 1.*

**Mechanizm odkodowania dla podwójnego błędu**

*Do odkodowania wiadomości zgodnie z logiką postepowania wystarczy ponownie przemnożyć*

*( modulo 2 ) macierz H przez naszą zakodowaną wiadomość. Otrzymana informacja ośmio-bitowa jeśli przesłana bez uszkodzenia pojedynczego bitu wiadomości będzie równa ‘0’.*

*Jeżeli wynikiem mnożenia ( modulo 2 ) będzie inna liczba, to będzie ona w przypadku wystąpienia pojedynczego błędu informacją reprezentującą index bitu, który należy odpowiednio zamienić z 1 na 0, bądź z 0 na 1.*

*Jeżeli wynik operacji nie będzie równy transponowanemu wierszowi macierzy H. To zbadać możemy istnienie 2 błędów w wiadomości tworząc kolekcję sum wierszy macierzy i następnie porównując daną sumę z wynikiem operacji arytmetycznej. Jeżeli równość zajdzie dostaniemy sumę indeksów w naszej wiadomości które należy skorygować aby otrzymać poprawną końcową wiadomość.*

**Szczegóły Implementacji**

*Program pozwala na zakodowanie i odkodowanie pliku noceded.txt. Dodatkowo z poziomu konsolowego jest w stanie zasymulować pojawienie się pojedynczego, 2 jak i 3 błędów na każdym odczytanym word’dzie. Aby zbadać zachowanie algorytmu z innymi danymi wystarczy zmienić dane w podanym pliku. Program może zostać uruchomiony na 2 sposoby:*

*- Udostępniając GUI ( ./TASP\_Ex1.exe )*

*- W sposób działający w tle*

*- Zakodowanie ( ./TASP\_Ex1.exe c encode )*

*- Dekodowanie ( ./TASP\_Ex1.exe c decode )*

*- Dekodowanie z symulacją 1 błędu ( ./TASP\_Ex1.exe c decode single )*

*- Dekodowanie z symulacją 2 błędu ( ./TASP\_Ex1.exe c decode double )*

*- Dekodowanie z symulacją 3 błędu ( ./TASP\_Ex1.exe c decode triple )*