Государственное образовательное учреждение высшего профессионального образования

Московский институт электроники и математики Национального исследовательского университета «Высшая школа экономики»

Кафедра ИКТ

ЛАБОРАТОРНАЯ РАБОТА ПО ДИСЦИПЛИНЕ:

«ТЕОРИЯ КОДИРОВАНИЯ»

«Код Хэмминга в матричном виде»

Выполнил: студент группы СБ-35

Шумшурова Н.А.

Проверил: преподаватель

Гоманилова Н.Б.

Москва 2012

Оглавление

[Цель и задачи лабораторной работы 3](#_Toc340337531)

[Теоретические сведения 3](#_Toc340337532)

[Расширенный код Хэмминга. 4](#_Toc340337533)

[Синдромы: S, S0. Сравниваем. 4](#_Toc340337534)

[Принцип работы программы 5](#_Toc340337535)

[Пример выполнения 6](#_Toc340337536)

[Код программы 9](#_Toc340337537)

# Цель и задачи лабораторной работы

Целью данной лабораторной работы является практическое освоение кода Хэмминга в матричном виде.

В процессе работы была создана программа, способная закодировать любое двоичное число кодом Хемминга, декодировать число, закодированное этим кодом, найти ошибку и исправить ее.

# Теоретические сведения

Одними из наиболее распространенныхил инейных блоковых кодовявяются коды Хемминга с dmin = 3 и dmin = 4.

Линейный блоковый код (n,m) может быть задан так называемой проврочной матрицей Hn,k.

Каждая строка проверочной матрицы соответствует выражению для расчета одного контрольного разряда. Например:

Рассмотрим Код Хэмминга с dmin = 4. Такой код может корректировать одиночную ошибку и обнаруживать двоичную ошибку. Характерной особенностью кода Хэмминга заключается в том, что столбцы проверочной матрицы представляют все возможные двоичные комбинации кода.

Построим проверочную матрицу кода Хэмминга H84:

В этой матрице также имеются совпадения номеров разрядов в двоичной и десятичной форме.

Хэмминг предложил располагать контрольные разряды между информационными. В данной матрице это просто обнаружить, так как номера контрольных разрядов «еi», а информационных «аi». Впри таком располоении столбцов проверочной матрицы полученный синдром сразу указывает на номер искажённого разряда (операции сличения столбцов при этом выполнять не требуется).

## Расширенный код Хэмминга.

В расширенном коде Хэмминга при кодировании и декодировании кроме проверок по подмножествам производится проверка и всего кодового слова в целом. Для этого добавляется дополнительный контрольный разряд e0, куда при кодировании записывается дополнение до чётности всего кодового слова в целом. При записи расширенного кода в матричном виде был приписан нулевой столбец слева, разряд обозначается в таком случае e0, и, кроме того, была добавлена верхняя строку, состоящая из одних едениц.

При декодировании для определения номера искаженного разряда расчитывается синдром. Операция декодирования, как и операция кодирования выполняется с помощью проверочной матрицы Hn,k. Количество разрядов синдрома равно количеству контрольных разрядов. Каждый контрольный разряд синдрома вычисляется как сумма по модулю два принятого значения контрольного разряда и расчитанного на основании принятых значений информационных разрядов.

## Синдромы: S, S0. Сравниваем.

В программе реализован код Хемминга с d min=4. Это позволяет корректировать одиночную ошибку и обнаруживать ошибку в двух разрядах.

В данном коде Хэмминга вычисленный синдром не указывает точно номер искаженного разряда, как это было ранее. Для определения искаженного разряда необходимо текже учитывать возможность задания двоичной ошибки.

При декодировании рассчитывается синдром S и одиночный разряд синдрома S0. Сопостовляя значения синдромов S0 и S делается вывод о характере ошибок.

* Если во всех контрольных разрядах регистра ошибок находятся нули, то в передаваемой кодовой комбинации не произошло ни одиночных, ни двоичных ошибок.
* Если в контрольных разрядах регистра ошибок записано ненулевое двоичное число и в дополнительном контрольном разряде еденица, то произошла одиночная ошибка, которая корректируется.
* Если в контрольных разрядах регистра ошибок записано ненулевое двоичное число, а в дополнительном контрольном разряде ноль, то произошла двоичная ошибка, которая не корректируется.
* Если во всех контрольных разрядах регистра ошибок находятся все нули, а в дополнительном контрольном разряде еденица, то произошла ошибка в дополнительном контрольном разряде.

Синдром не зависит от вида передаваемой кодовой комбинации, он полностью определяется характером возникших ошибок.

# Принцип работы программы

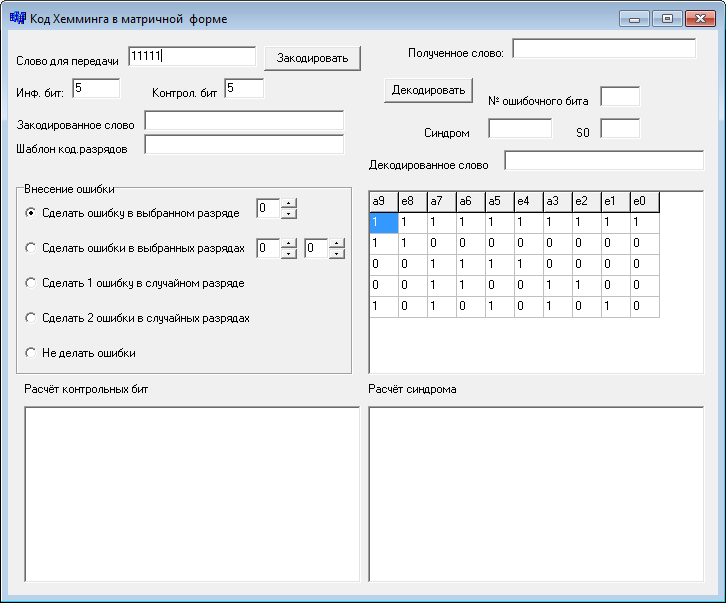
Программа написана на языке C++ в среде Borland Builder 6. После запуска программы появляется окно стандартного вида. Необходимо ввести в отдельное поле число, представленное в двоичной системе счисления, и нажать на кнопку «Закодировать», после чего будет введено на экран количество контрольных разрядов, также число информационных разрядов и сформирована проверочная матрица, выводимая в отдельное поле окна. Также будет выведено закодированное число и то как считались контрольные разряды.

Далее требуется указать номер разряда или нескольких разрядов, в которых произошла ошибка для того, чтобы преобразовать число, сымитировав ошибку при передаче закодированного числа. Возможно, дополнительно, рандомное задание номеров разрядов в которых будет сделана ошибка. Также можно выбрать пункт списка «Внесение ошибки», позволяющий не делать ошибки в кодируемом слове.

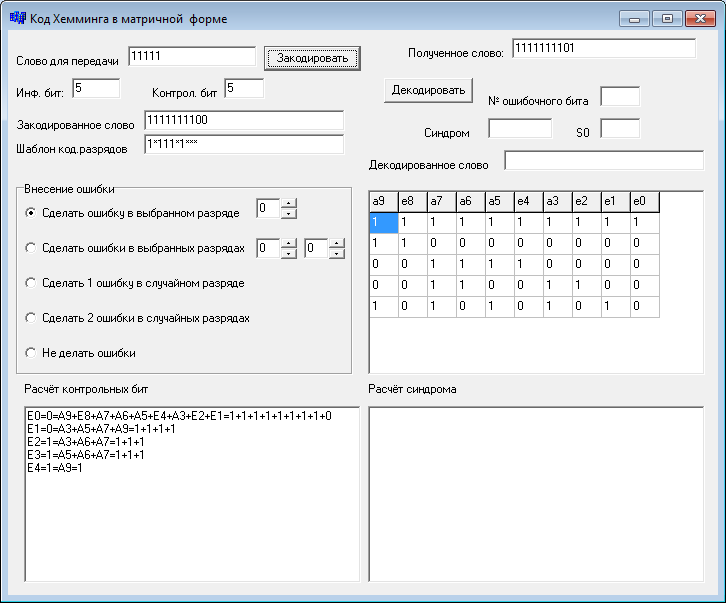
После искажения разряда, будет произведена операция декодирования. Ччтобы ее произвести необходимо нахать на кнопку «Декодировать» и на экран будет выведен синдром и его расчет, номер разряда в котором произошла ошибка и дополнительно надпись если ошибок было две в поле «№ ошибочного бита» или исправленное число, если ошибка была сделана в одном разряде.

# Пример выполнения

1. Вводим информационные биты в форму «Слово для передачи». Появляется проверочная матрица и количество информационных и контрольных бит.

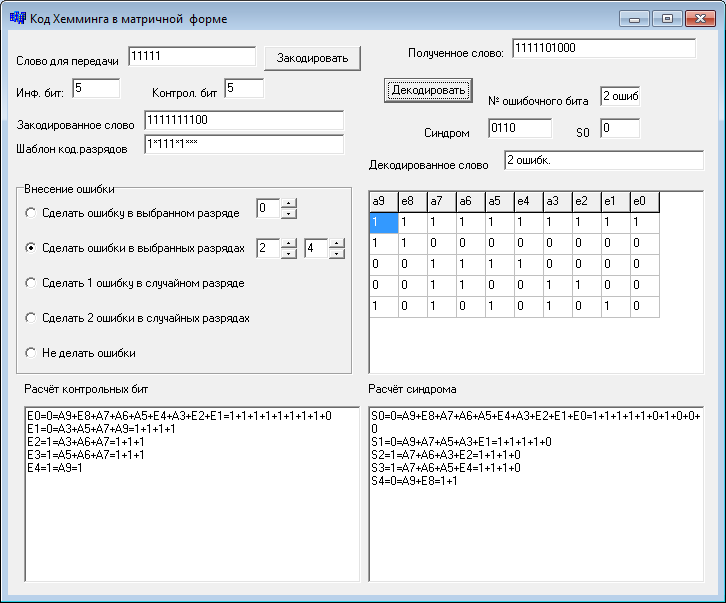


1. Была нажата кнопка «Закодировать». В окне отобразились: расчет контрольных разрядов, закодированное слово, шаблон кодовых разрядов, отображающй позиции на которых в закодированном слове находятся кодовые разряды.



1. Было выбрано количество ошибок и разряды в которых ошибки будут сымитированны, также была нажата кнопка «Декодировать».

Программа вывела синдром и его расчет, и определила то, что было сделано две ошибки в переданном слове.

****

# 

# Код программы

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

int p\_max,d\_len;

unsigned char s\_in[30]={0,1,0,1,0,1,0,1,0,1,0,1},s\_cod[30],s\_bag[30],s\_corr[30];

unsigned char matrix[25][200];

void \_\_fastcall TForm1::FormCreate(TObject \*Sender)

{

Edit1->OnChange(Sender);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)//кодируем слово

{

char buf[100];

AnsiString t1,t2;

unsigned char t;

int p=0,i,j,k;

buf[p\_max+d\_len]='\*';

Memo1->Text="";

Memo2->Text="";

for(i=1;i<=p\_max+d\_len;i++)

{ if(i==(1<<p))

{

buf[p\_max+d\_len-i]='\*';p++;

s\_cod[p\_max+d\_len-i]=0;

}

else

{

buf[p\_max+d\_len-i]=s\_in[d\_len-i+p]+'0';

s\_cod[p\_max+d\_len-i]=s\_in[d\_len-i+p];

}

}

buf[p\_max+d\_len+1]=0;

for(p=0;p<p\_max;p++)

{

t=0; t1="";t2="";

for(int j=0;j<p\_max+d\_len;j++)

{

t^=matrix[p\_max-p][j]&s\_cod[j];

for(k=0;k<=p\_max;k++)

if((1<<k)==(p\_max+d\_len-j)) break;

if((k>p\_max)&&matrix[p\_max-p][j])

{

if(t1!="")

{

t1="+"+t1;t2="+"+t2;

}

t1=(AnsiString)"A"+(p\_max+d\_len-j)+t1;

t2=(AnsiString)(s\_cod[j])+t2;

}

}

s\_cod[p\_max+d\_len-(1<<p)]=t;

Memo2->Text=Memo2->Text+"E"+(p+1)+"="+t+"="+t1+"="+t2+"\r\n";

}

deb->Text=buf;

s\_cod[p\_max+d\_len]=0;

for(int j=0;j<d\_len+p\_max;j++)

s\_cod[p\_max+d\_len]^=s\_cod[j];

for(i=0;i<=p\_max+d\_len;i++) buf[i]=s\_cod[i]+'0';

buf[p\_max+d\_len+1]=0;

Edit3->Text=buf;

t1="";t2="";

for(t=0,j=0;j<p\_max+d\_len;j++)

{

t^=s\_cod[j];

if(t1!="") {t1=t1+"+"; t2=t2+"+"; }

for(i=0;i<p\_max;i++)

{

if(p\_max+d\_len-j==1<<i) break;

}

if(i<p\_max||j==p\_max+d\_len) t1=t1+"E"; else t1=t1+"A";

t1=t1+(AnsiString)+(p\_max+d\_len-j);

t2=t2+(AnsiString)s\_cod[j];

}

Memo2->Text=(AnsiString)"E0="+(t)+"="+t1+"="+t2+"\r\n"+Memo2->Text;

RadioGroup1Click(Sender);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::RadioGroup1Click(TObject \*Sender)//Установка количества ошибок

{

char buf[30];

int i;

for(i=0;i<=p\_max+d\_len;i++)

s\_bag[i]=s\_cod[i];

if(RadioGroup1->ItemIndex==1)

{

s\_bag[p\_max+d\_len-Edit4->Text.ToInt()]^=1;

s\_bag[p\_max+d\_len-Edit5->Text.ToInt()]^=1;

}

if(RadioGroup1->ItemIndex==0)

{

s\_bag[p\_max+d\_len-Edit6->Text.ToInt()]^=1;

}

if(RadioGroup1->ItemIndex==2)

{

s\_bag[rand()%(p\_max+d\_len+1)]^=1;

}

if(RadioGroup1->ItemIndex==3)

{

s\_bag[rand()%(p\_max+d\_len+1)]^=1;

s\_bag[rand()%(p\_max+d\_len+1)]^=1;

}

for(i=0;i<=p\_max+d\_len;i++)

buf[i]=s\_bag[i]+'0';

buf[i]=0;

Edit7->Text=buf;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::UpDown4Changing(TObject \*Sender, bool &AllowChange)

{ RadioGroup1->OnClick(Sender);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::UpDown4Click(TObject \*Sender, TUDBtnType Button)

{

RadioGroup1->OnClick(Sender);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Edit1Change(TObject \*Sender)//ввод числа

{

int i=0;

Edit1->Color=clWhite;

for(i=0;i<Edit1->Text.Length();i++)

{

if((Edit1->Text[i+1]!='0')&&(Edit1->Text[i+1]!='1'))

{

i=-1;

break;

}

s\_in[i]=Edit1->Text[i+1]-'0';

}

Edit2->Text=i;

Edit3->Text="";

if(i!=-1)

{

int j,k;

d\_len=i;

for(p\_max=0;d\_len>=(1<<p\_max)-p\_max;p\_max++);

Edit10->Text=p\_max+1;

StringGrid1->RowCount=p\_max+1+1;

StringGrid1->ColCount=p\_max+1+d\_len;

for(j=0;j<=p\_max+1+d\_len;j++)

matrix[0][j]=1;

for(j=0;j<=p\_max+d\_len;j++)

{

for(i=0;i<p\_max;i++)

{

if(p\_max+d\_len-j==1<<i) break;

}

if((i<p\_max)||(j==p\_max+d\_len))

StringGrid1->Cells[j][0]=(AnsiString)"e"+(p\_max+d\_len-j);

else

StringGrid1->Cells[j][0]=(AnsiString)"a"+(p\_max+d\_len-j);

}

for(i=0;i<=(d\_len+p\_max);i++)

for(j=0;j<p\_max;j++)

{

matrix[p\_max-j][d\_len+p\_max-i]=((i>>j)&1)?1:0;

}

for(i=0;i<=p\_max;i++)

for(j=0;j<=p\_max+d\_len;j++)

StringGrid1->Cells[j][i]=matrix[i][j];

}

else

{

Edit1->Color=clRed;

Edit2->Text="";

Edit3->Text="Ошибочные данные";

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button2Click(TObject \*Sender)//Декодируем

{

unsigned char t=0;

int i,j,k;

AnsiString t1,t2;

Memo1->Clear();

k=0;

Edit11->Text="";

for(t=0,i=0;i<=p\_max+d\_len;i++)

t^=s\_bag[i];

t1="";t2="";

for(t=0,j=0;j<=p\_max+d\_len;j++)

{

t^=s\_bag[j];

if(t1!="") {t1=t1+"+"; t2=t2+"+"; }

for(i=0;i<p\_max;i++)

{

if(p\_max+d\_len-j==1<<i)

break;

}

if(i<p\_max||j==p\_max+d\_len)

t1=t1+"E";

else

t1=t1+"A";

t1=t1+(AnsiString)+(p\_max+d\_len-j);

t2=t2+(AnsiString)s\_bag[j];

}

Memo1->Text=(AnsiString)"S0="+t+"="+t1+"="+t2;

for(i=0;i<p\_max;i++)

{t1="";t2="";

for(t=0,j=0;j<p\_max+d\_len;j++)

{t^=matrix[p\_max-i][j]&s\_bag[j];

if(matrix[p\_max-i][j])

{ if(t1!="")

{t1=t1+"+";

t2=t2+"+";

}

if(p\_max+d\_len-j==1<<i) t1=t1+"E";

else t1=t1+"A";

t1=t1+(AnsiString)+(p\_max+d\_len-j);

t2=t2+(AnsiString)s\_bag[j];

}

}

k+=(t<<i);

if(t) Edit11->Text=(AnsiString)'1'+Edit11->Text;

else Edit11->Text=(AnsiString)'0'+Edit11->Text;

Memo1->Text=Memo1->Text+"\r\nS"+(i+1)+"="+t+"="+t1+"="+t2;

}

for(t=0,i=0;i<=p\_max+d\_len;i++) t^=s\_bag[i];

Edit12->Text=(AnsiString)t;

// если хаффман говорит что есть ошибка а бит чётности что нет. то две ошибки

if(t) //бит чётности говорит что есть ошибка тогда ошибка в k-ом бите.

{Edit9->Text=k;

Edit8->Text="";

for(j=0;j<=p\_max+d\_len;j++)

{

if(j==k)

Edit8->Text=(AnsiString)(1-s\_bag[p\_max+d\_len-j])+Edit8->Text;

else

Edit8->Text=(AnsiString)(s\_bag[p\_max+d\_len-j])+Edit8->Text;

}

}

else

{

if(k!=0) {Edit9->Text="2 ошибк.";Edit8->Text="2 ошибк.";}

else {Edit9->Text="нет.ошибк.";Edit8->Text=Edit7->Text;}

}

}

//---------------------------------------------------------------------------