**Національний технічний університет України «КПІ»**

**Інститут прикладного системного аналізу**

**Кафедра Системного проектування**

Лабораторна робота №5

за темою:

«Алгоритм циклічні коди.»

з дисципліни: Теорія інформації та кодування.

Виконала:

студентка групи ДА-42

факультету ННК «ІПСА»

Хмарська С. С.

Київ – 2015

Мета роботи: Мета роботи - вивчити методи побудови циклічних код і їх програмну реалізацію на ЕОМ.

**Завдання**

**4.2. Вміст роботи**

1. По таблиці. 1 або 2 вибрати створюючий многочлен для побудови циклічної коди і загальну довжину n кодової комбінації циклічної коди. Таблиця. 4.1 призначена для студентів академгрупп з непарним номером, а таблиця. 4.2 - з парним.Таблиця 2

Варіанти завдань для студентів академгрупп з парним номером

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Номер студен­та за списком | Створюючий многочлен | ***n*** | Номер студен­та за списком | Створюючий многочлен | ***n*** |
| I | 1001101 | 31 | 16 | 1101011 | 24 |
| 2 | 1001111 | 15 | 17 | 1101011 | 12 |
| 3 | 1010001 | 28 | 18 | 1100011 | 20 |
| 4 | 1010011 | 21 | 19 | 1110001 | 31 |
| 5 | 1010101 | 24 | 20 | 1110111 | 24 |
| 6 | 1010111 | 21 | 21 | 1110101 | 21 |
| 7 | 1011001 | 31 | 22 | 1110111 | 12 |
| 8 | 1011101 | 15 | 23 | 1111001 | 15 |
| 9 | 1011101 | 30 | 24 | 1111011 | 31 |
| 10 | 1011111 | 30 | 25 | 1111101 | 30 |
| 11 | 1101001 | 28 | 26 | 1111111 | 21 |
| 12 | 1100011 | 20 | 27 | 11100111 | 30 |
| 13 | 1100101 | 21 | 28 | 101001011 | 21 |
| 14 | 1100011 | 30 | 29 | 100111001 | 17 |
| 15 | 1101001 | 28 | 30 | 111010111 | 17 |

2. Ознайомитися з методикою побудови циклічних код де­лением вихідної кодової комбінації на створюючий многочлен [5].

3. Розробити програму, що реалізовує алгоритм кодування діленням вихідної кодової комбінації на створюючий многочлен. Вимоги до програми:

а) мова програмування по розсуду студента;

б) вихідна кодова комбінація повинна вводитися в діалоговому режимі безпосередньо з клавіатури терміналу.

4. Ввести програму в ЕОМ і провести її відладку.

5. Для довільних десяти інформаційних кодових комбінацій за допомогою розробленої програми отримати відповідні цикличе­ские коди (спочатку на екрані терміналу, а потім надруковані на принтері). Оцінити час кодування.

6. Виконати операції кодування уручну і порівняти получен­ные кодові комбінації з отриманими при виконанні п.4.2.5.

7. Модифікувати програму, розроблену при виконанні п.4.2.3, додавши в неї модулі визначення мінімальної кодової відстані, побудови матриць, що породжують і перевірочної, для заданого створюючого многочлена. Відлагодити програму і визначити мінімальну кодову відстань для досліджуваної коди, вивести на печатьпорождающую і перевірочну матриці . Зробити виводи про можливості коди, що коректують.

8. Розробити структурну схему кодера на базі сдвигового регістра. Оформити звіт про ла­бораторной роботу.

1. Ознайомитися з методикою корекції помилок в циклічних кодах.
2. Скласти алгоритм корекції і розробити програму для його реалізації, в результаті виконання якої має бути отримана виправлена кодова комбінація, виведено повідомлення про наявність помилки і вказано її положення у вихідній комбінації.
3. Перевірити працездатність розробленої програми за допомогою кодових комбінацій циклічної коди, отриманих в лабораторній роботі 4. Для цього ввести однократні помилки у вказані комбінації, і за результатами автоматичної корекції, отриманим за допомогою розробленої програми, оцінити правильність корекції.
4. Визначити максимальне число коректованих помилок для даної коди. Для цього повторити попередній пункт для помилок кратністю 2, 3 і так далі, до тих пір, поки корекція здійснюватиметься правильно.

***Теоретичні відомості.***

Циклічні коди допускають досить просту технічну реалізацію і вельми ефективні при виявленні і виправленні помилок.

Побудову і дослідження будь-якої циклічної коди можна звести до операцій над поліномами. Кожному двійково кодованому n-разрядному слову може бути поставлений у відповідність поліном (n-1) -степени. При цьому говорять, що слово представляється поліномом змінної x, причому коефіцієнтами полінома є значення відповідних розрядів слова.

Наприклад, слово A=1011 може бути представлене поліномом третьої міри PA(X)=x3+x+1.

Введені поліноми описуються законами звичайної алгебри, за винятком того, що операція складання проводиться по модулю 2.

Циклічний код може бути заданий за допомогою створюючого полінома P(x)=prxr+pr-1xr-1+...+p1x+p0. При цьому поліном F(x) належить циклічному коду в тому і лише в тому випадку, якщо він ділиться без залишку на P(x), тобто для полінома F(x) справедливе порівняння F(x)=0 по модулю P(x).

Як створюючі поліноми використовують поліноми, що не наводяться, які не можуть бути представлені у вигляді твору поліномів нижчих мір. Такі поліноми діляться лише на одиницю і на самих себе.

У загальному випадку кодову комбінацію циклічної n-значного коди можна отримати двома способами:

1. множенням вихідної комбінації U(x) на одночлен Xr і додаванням до цього твору залишку R(x), отриманого в результаті ділення твору U(x)\*xr на створюючий поліном P(x);
2. множенням вихідної комбінації U(x) на створюючий поліном.

При користуванні першого способу в отриманих кодових комбінаціях розташування інформаційних символів строго впорядковане - вони займають до розрядів, а останні (n-к) розрядів відводяться під контрольних, при другому способі виходять нероздільні коди. На практиці частіше використовують роздільні циклічні коди, тобто коди, що отримуються першим способом.

Хай, наприклад, необхідно отримати роздільний циклічний код для інформаційної кодової комбінації U=1101 за допомогою створюючого полінома P(x)=x3+x+1. В даному випадку число інформаційних розрядів k=4, міра створюючого полінома r=3, і загальна довжина кодової комбінації отриманої циклічної коди n=k+r. Представимо кодову комбінацію U=1101 у вигляді полінома U(x)=x3+x2+1. Знайдемо твір U(x) xr=(x3+x2+1) x3=x6+x5+x3, а потім виконаємо ділення

.

Отримаємо залишок R(x)=1. Відповідно до правила кодування отримуємо комбінацію циклічної коди F(x)=U(x)x3+R(x)=x6+x5+x3+1 або в двійковій формі: .

На практиці контрольну частину коди отримують послідовним відніманням (складанням) по модулю 2 із зрушеної на n розрядів вліво інформаційній частині слова іншого слова, відповідного створюючому поліному P(x).

Процес знаходження контрольної частини слова доцільно преставить у вигляді таблиці:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Інформаційна*  *частина слова* | | | | | *Контрольна*  *частина слова* | | | |
| ***1*** | ***1*** | ***0*** | ***1*** |  | ***0*** | ***0*** | ***0*** |
| ***1*** | ***0*** | ***1*** | ***1*** |  | ***0*** | ***0*** | ***0*** |
|  | ***1*** | ***1*** | ***0*** |  | ***0*** | ***0*** | ***0*** |
|  | ***1*** | ***0*** | ***1*** |  | ***1*** |  |  |
|  |  | ***1*** | ***1*** |  | ***1*** | ***0*** | ***0*** |
|  |  | ***1*** | ***0*** |  | ***1*** | ***1*** | ***0*** |
|  |  |  | ***1*** |  | ***0*** | ***1*** | ***0*** |
|  |  |  | ***1*** |  | ***0*** | ***1*** | ***1*** |
|  |  |  |  |  | ***0*** | ***0*** | ***1*** | Контрольна частина слова | |

Штриховою лінією виділено слово, відповідне поліному P(x).

Оскільки циклічні коди - окремий випадок лінійних, то для їх опису зручно використовувати матричну виставу, тобто матриці, що породжують і перевірочні.

Для формування рядків матриці циклічної роздільної коди, що породжує, використовуються комбінації коди U(x), які містять одиницю лише в одному розряді. Ці комбінації умножають на xr, а потім знаходять залишок від ділення Ui(x) xr/P(x), рівний Ri(x). Відповідний рядок матриці, що проводить, записують у вигляді Ui(x) xr+Ri(x). При цьому матрицю будують у вигляді Gn,k=[Ik|P], де Ik - одинична підматриця з розмірністю до X до; P - підматриця розмірністю до X r, рядки якої є залишками Ri(x).

Матриця, що породжує, дозволяє безпосередньо отримати перші до комбінацій коди, а останні 2k-до-1 комбінацій можна отримати підсумовуванням по модулю 2 рядків матриці Gn,k у всіх можливих поєднаннях. Остання комбінація коди - нульова.

Знайдемо матрицю, що породжує, для коди із створюючим поліномом P(x)=x3+x+1 (P(1,0)=1011) при n=7 і k=4. Оскільки k=4, то для побудови G7,4 необхідно вибрати одиничні вектори U1(x)=0001, U2(x)=0010, U3(x)=0100, U4(x)=1000. Тоді з урахуванням того, що r=n-k=3, для кодової комбінації U1(x) отримаємо, і, виконавши ділення, знайдемо R1(x)=011, тобто перший рядок матриці, що породжує, дорівнюватиме 0001011. Аналогічно знайдемо залишки для кодових комбінацій U2(x), U3(x), U4(x): R2(x)=110, R3(x)=111, R4(x)=101.

Таким чином, матриця, що породжує, може бути записана в наступному вигляді:



або, переставивши місцями рядки, отримаємо її в канонічній формі:



Використовуючи знайдену матрицю, легко знайти кодову комбінацію циклічної коди, соотвествующую інформаційної кодової комбінації, розглянутої в наведеному раніше прикладі (U1(x)=1101).

Для комбінацій циклічної коди F справедливе матричне вираження F=UG, з якого виходить, що шукані комбінації є лінійними комбінаціями рядків матриці G. У даному прикладі необхідно підсумувати по модулю 2 1-й, 2-й і 4-й рядки матриці, оскільки одиниці в U(x)=1101 стоять на 1-ій, 2-ій і 4-ій позиціях, вважаючи зліва, тобто шукана комбінація



Як видимий, знайдена кодова комбінація F=1101001 збігається з отриманою раніше діленням на створюючий поліном.

**Виправлення помилок.**

Виявлення помилок в циклічних кодах засноване на тому, що як кодові комбінації циклічної коди використовують такі комбінації, поліноми яких діляться без залишку на створюючий поліном. Якщо при діленні кодової комбінації циклічної коди на створюючий поліном виходить залишок R(x), то це є ознакою помилки. Таким чином, залишки R(x) є пізнавачами помилок, але не вказують безпосередньо на місце розташування помилки в кодовій комбінації.

Для виправлення помилок виконують наступні операції:

1. Прийняту кодову комбінацію ділять на створюючий поліном.
2. Підраховують кількість одиниць в залишку, тобто знаходять вагу залишку W. Якщо W=Sн, де Sн - допустиме число помилок, що виправляються кодом, то прийняту комбінацію складають по модулю 2 із залишком. Сума дає виправлену комбінацію. Якщо W>Sн, то виконують п.3.
3. Проводять циклічне зрушення кодової комбінації на один розряд вліво, ділять її на створюючий поліном і визначають вагу отриманого залишку. Якщо знайдена вага залишку, то ділиме підсумовують із залишком по модулю 2 і переходять до п.4.
4. Проводять циклічне зрушення результату підсумовування управо на один розряд. Отримують комбінацію, що не містить помилок. Якщо після першого циклічного зрушення вліво на один розряд і ділення W>Sн, то выплняют п.5.
5. Повторюють операції по п.3. до тих пір, поки не буде отриманий залишок з . Потім комбінацію, отриману в результаті останнього циклічного зрушення вліво, підсумовують з цим залишком по модулю 2 і виконують п.6.
6. Проводять циклічне зрушення управо на стільки розрядів, на скільки була зрушена вліво підсумовувана з останнім залишком комбінація відносно початкової. В результаті повчають виправлену комбінацію.

Властивості циклічної коди, що коректують, залежать від вибору створюючого полінома. У даній лабораторній роботі розглядаються питання корекції однократних помилок, для яких число контрольних (r) і інформаційних (k) розрядів повинне задовольняти наступній нерівності: . Оскільки загальна довжина кодової комбінації n=k+r, то при заданому n число контрольних розрядів може бути знайдене з нерівності .

Створюючий поліном зазвичай вибирають по таблицях. Його слід вибирати як можна коротшим, але міра його має бути не менше числа контрольних розрядів r, а число ненульових членів - не менше мінімальної кодової відстані.

1. Помилка у четвертому розряді

Code: 110101

Code with control sum: 110101011000

Code with mistake: 110001011000

Mistake in bit number 4

Code without mistake: 110101011000

Process returned 0 (0x0) execution time : 0.020 s

Press any key to continue.

1. Помилка в 3 розряді

Code: 110101

Code with control sum: 110101011000

Code with mistake: 110100011000

Mistake in bit number 6

Code without mistake: 110101011000

Process returned 0 (0x0) execution time : 0.031 s

Press any key to continue.

1. Помилка у 7 розряді

Code: 110101

Code with control sum: 110101011000

Code with mistake: 110101111000

Mistake in bit number 7

Code without mistake: 110101011000

Process returned 0 (0x0) execution time : 0.035 s

Press any key to continue.

**Висновки.** Виправляє одну помилку. Пошук помилки здійснюється за рахунок перевіряння помилки в кожному біті по черзі. Перевірка здійснюється за допомогою виключного «або».

Умовою відсутності помилки є результат із всіх нулів, якщо присутня хоча б одна одиниця, то є помилка.

**Код програми.**

#include <iostream>

#include <fstream>

#include <cstring>

using namespace std;

char xor1(char a, char b)

{

if (a==b)

return '0';

else

return '1';

}

string devide(string code, string work\_str)

{

string buff;

int i=0, j=0;

while(i<work\_str.length())

{

buff.push\_back(code[i]);

i++;

}

j=i;

i=0;

while(code[j-1])

{

i=0;

if(buff[i]=='0')

{

while(buff[i]=='0')

{

buff.push\_back(code[j]);

j++;

i++;

if(j==code.length())

break;

Code: 110101

Code with control sum: 110101011000

}

int k=0;

while (i<=buff.length())

{

buff[k]=buff[i];

i++;

k++;

}

while(buff.length()!=work\_str.length())

{

buff.pop\_back();

}

}

if(j==code.length())

{

break;

}

i=0;

while(true)

{

buff[i]=xor1(buff[i+1], work\_str[i+1]);

//cout<<"i^"<<buff[i]<<endl;

i++;

if (i==buff.length())

{

buff[i-1]=code[j];

j++;

i=0;

break;

}

}

}

return buff;

}

int find\_mistake(string code, string polynome)

{

int i=0, flag=0, q=0, j;

string buff;

buff=devide(code, polynome);

for(int j=0; j<buff.length(); j++)

{

if(buff[j]=='1')

flag++;

}

if(flag==0)

{

cout<<"No mistakes!!!"<<endl;

return 0;

}

while(true)

{

q=0;

while(q<code.length())

{

if(i==q)

code[q]=xor1(code[q],'1');

else

code[q]=xor1(code[q],'0');

q++;

}

buff=devide(code, polynome);

j=0;

flag=0;

while(j<buff.length())

{

if(buff[j]=='1')

flag++;

j++;

}

if(flag==0)

{

cout<<"Mistake in bit number "<<i+1<<endl;

cout<<"Code without mistake: "<<code<<endl;

break;

}

q=0;

while(q<code.length())

{

if(i==q)

code[q]=xor1(code[q],'1');

else

code[q]=xor1(code[q],'0');

q++;

}

i++;

}

return 0;

}

int main()

{

string code, polynome, check, buff;

int q=1;

code="110101";

polynome="1110111";

cout<<"Code: "<<code<<endl;

buff=devide(code+"000000", polynome);

while(q<buff.length())

{

code.push\_back(buff[q]);

q++;

}

cout<<"Code with control sum: "<<code<<endl;

code[6]='1';

cout<<"Code with mistake: "<<code<<endl;

find\_mistake(code,polynome);

return 0;

}