# МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

# федеральное государственное автономное образовательное учреждение высшего образования

# «САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

# АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА №25

ОТЧЕТ

ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | А. А. Бурков |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |
|  |  |  |  |  |

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №1

ИСПОЛЬЗОВАНИЕ ЦИКЛИЧЕСКИХ КОДОВ ДЛЯ ОБНАРУЖЕНИЯ ОШИБОК В СЕТЯХ ПЕРЕДАЧИ ДАННЫХ

по курсу: сети и системы передачи информации

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | | 3031 |  | ­ |  | А. В. Гончаров |
|  | номер группы | |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург

2023

# Цель работы

Разработка программы, наглядно демонстрирующую работу кодера и декодера для типового алгоритма формирования циклических кодов.

# Описание моделируемой системы

В современных системах передачи данных используется метод добавления контрольной суммы к передаваемым данным для обнаружения ошибок. Контрольная сумма вычисляется на основе данных и включается в сообщение, которое передается по каналу связи. Такой подход позволяет определить наличие ошибок при передаче сообщения.На рис. 1 изображена структурная схема рассматриваемой в лабораторной работе системы передачи данных.



Рисунок 1 – Структурная схема системы передачи данных

На вход кодера поступает некоторое информационное сообщение m, состоящее из нулей и единиц. Кодер по некоторому алгоритму вычисляет контрольную сумму, дописывает ее к передаваемому сообщению и таким образом формирует закодированное сообщение a так же состоящее из 0 и 1. В канале могут произойти ошибки, в результате которых некоторые биты сообщения инвертируются (0 становится 1 или 1 становится 0). Вектор ошибок показывает на каких позициях произошла ошибка, при этом канал может быть описан как операция XOR передаваемого сообщения и вектора ошибок. Пример: 𝑎̅ = [101], 𝑒̅ = [001], 𝑏̅ = [100]. Декодер по некоторому

алгоритму проверяет контрольную сумму в принятом сообщении и принимает одно из следующих решений:

Рассматривается модель двоично-симметричного канала (ДСК) без памяти представленного на рис. 2. Как видно из рисунка 2 с вероятностью p происходит ошибка (0 становится 1 или 1 становиться 0). Канал является двоичным, поэтому возможны только два значения битов на входе и выходе канала: {0,1}. Канал называется симметричным ввиду того, что вероятность ошибки для обоих значений битов одинакова. Модель ДСК приведена на рис.

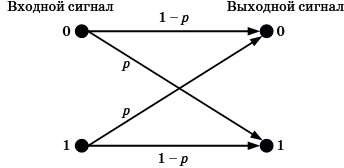
1. Канал без памяти характеризуется тем, что случайные события, связанные с ошибками в канале независимы для разных моментов времени.

Рисунок 2 – Модель двоично -симметричного канала

# Описание задания для варианта

На вход программы подается порождающий многочлен g x( ), число k, вектор ошибки e и информационная последовательность из l бит (l может быть как меньше, так и больше k ). Программа формирует кодовое слово. На основе него и вектора ошибки формируется последовательность на выходе канала. По принятой последовательности принимается решение о наличии ошибок в канале. В программе должна быть предусмотрена возможность вывода всех промежуточных значений, которые формируются как при работе кодера, так и декодера.

Алгоритм работы Кодера:

Передаваемое сообщение рассматривается как вектор длины k. Для каждого сообщения (𝑚¯) кодер выполняет следующие действия:

* 1. На основе вектора m формируется многочлен m(x). Степень многочлена m(x) при этом может быть, как равна k – 1, так и превышать это значение;
  2. Вычисляется многочлен c(x) = m(x)xr mod g(x);
  3. Вычисляется многочлен a(x) = m(x)xr + c(x);
  4. На основе многочлена a(x) формируется вектор a, длина которого n бит, где n = k + r.

Алгоритм работы Декодера:

1. Принятое сообщение 𝑏¯ = 𝑎¯ + 𝑒̅ переводится в многочлен b(x);
2. Вычисляется синдром: s(x) = b(x) mod g(x);
3. Если s(x) ≠ 0, то декодер выносит решение, что произошли ошибки (E

= 1), иначе декодер выносит решение, что ошибки не произошли (E = 0).

# Блок схема алгоритма

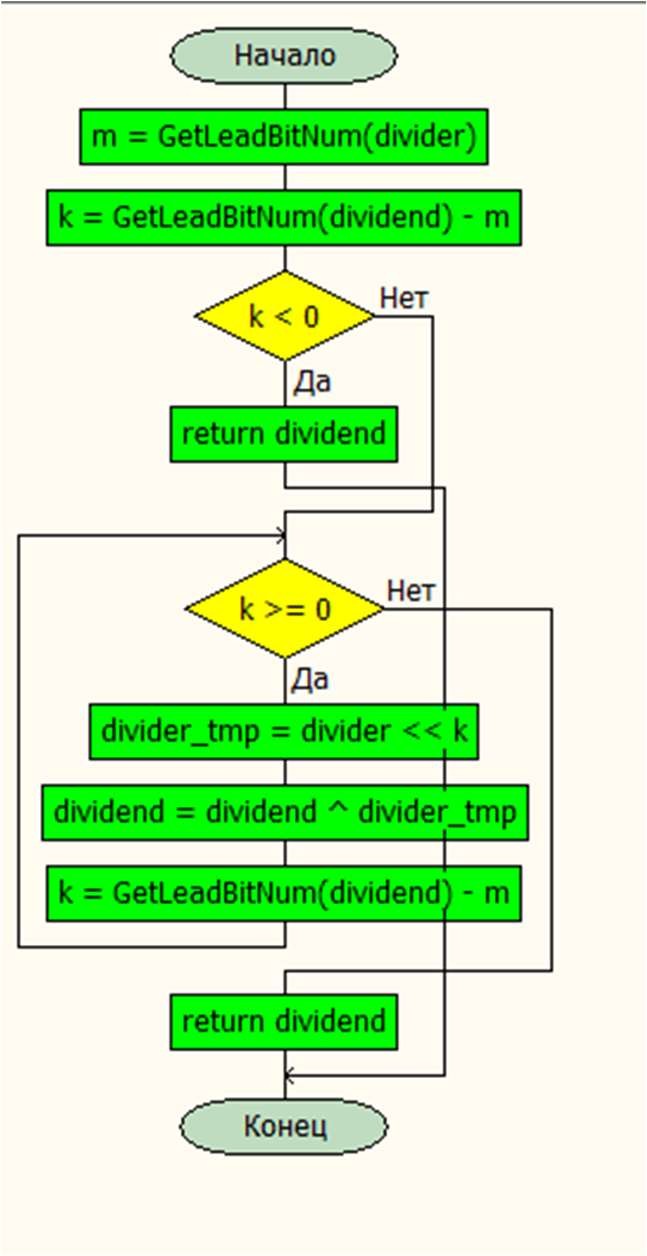


Рис. 3. – Блок схема

# Контрольный пример (Visual Studio – C#)

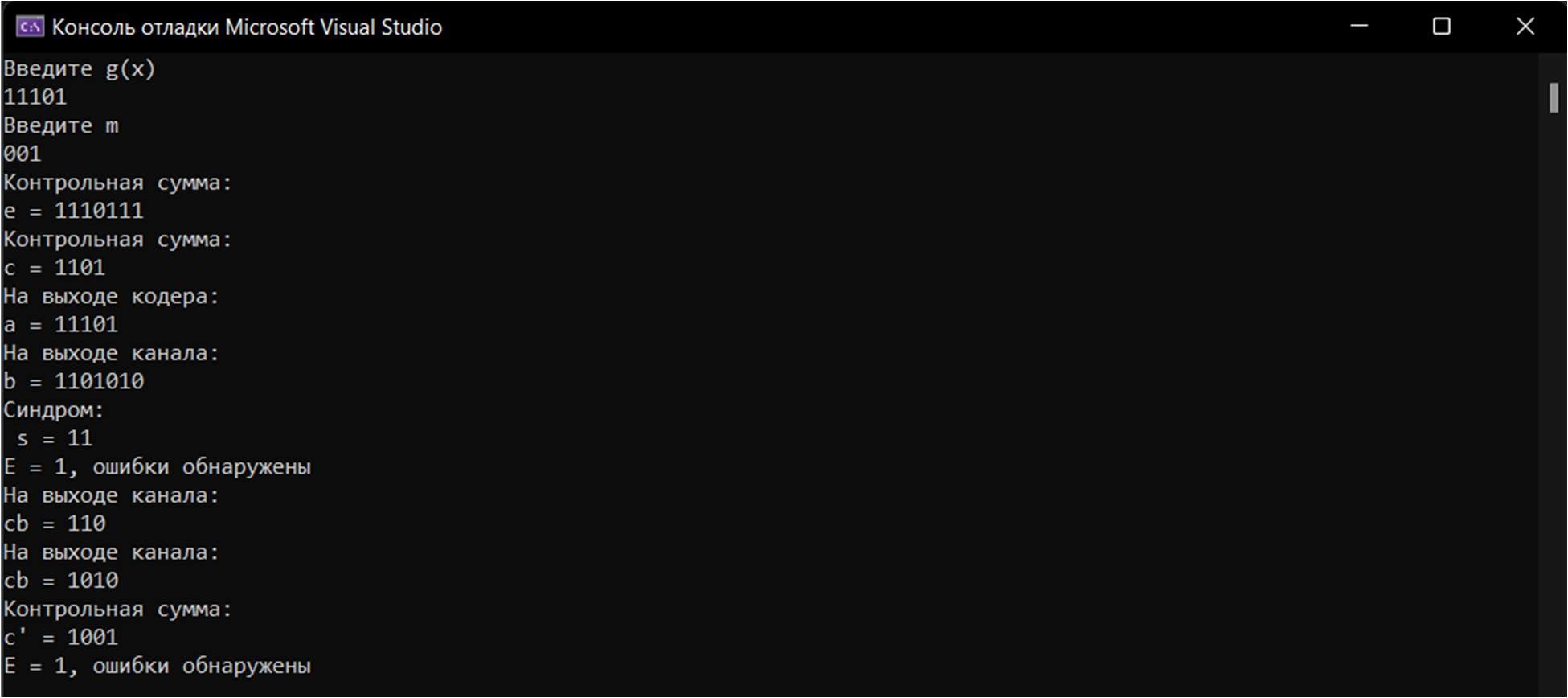


Рис. 4. – Пример 1.

# Дополнительное задание пример (Visual Studio – C#)

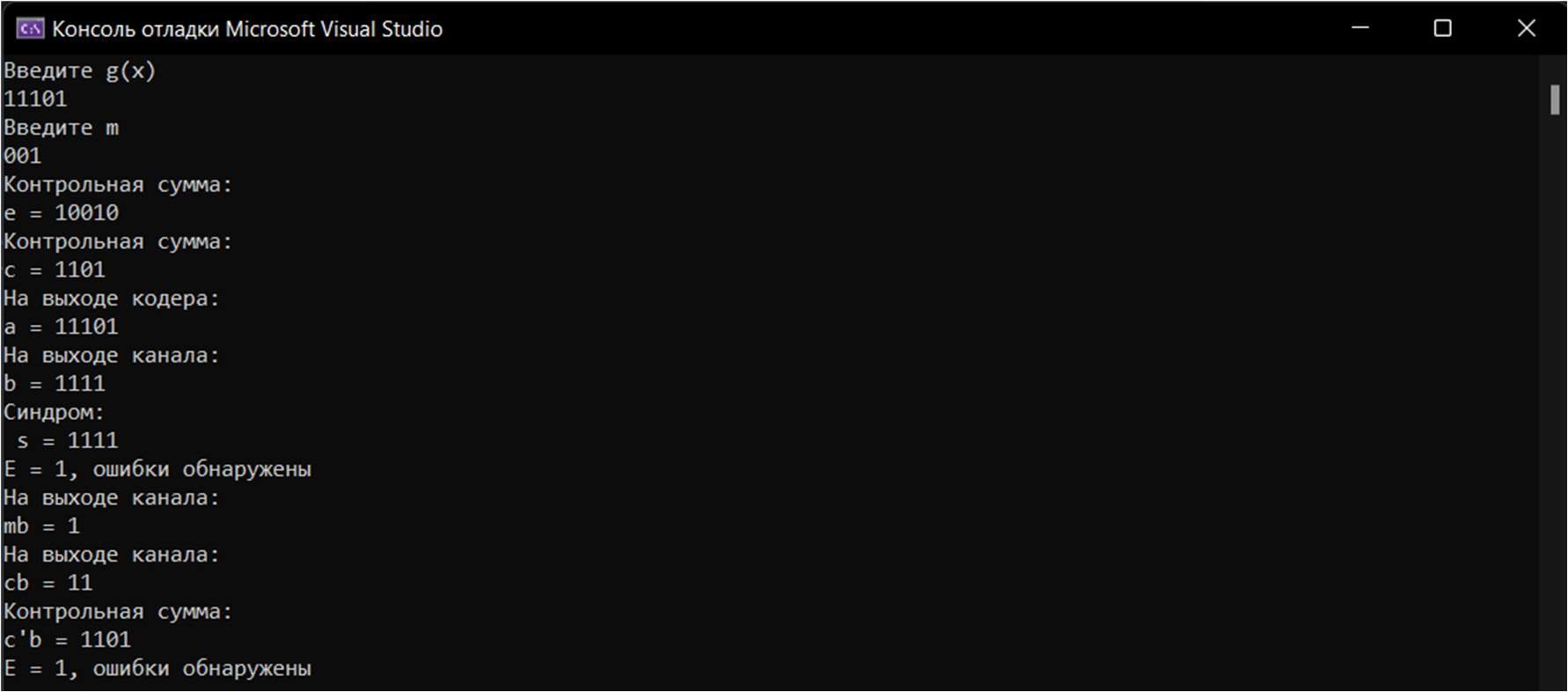


Рис. 5. – Пример 1.

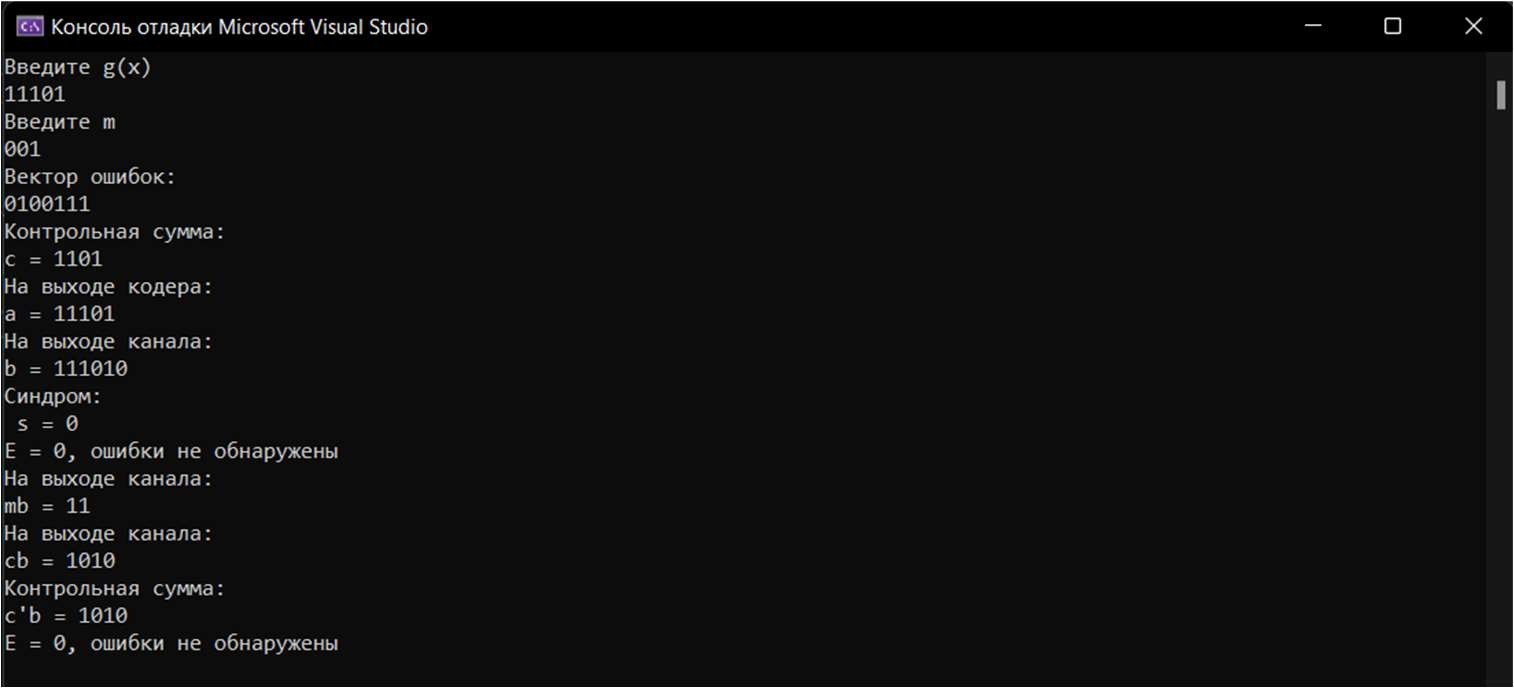


Рис. 6. – Пример 2.

# Выводы

Для реализации алгоритма построения контрольной суммы была разработана программа, которая осуществляет кодирование и декодирование сообщения, а также выявляет наличие ошибки. Этот подход позволяет обеспечить надежность передачи данных и гарантировать доставку информации без потерь и искажений.

# Листинг программы

class Program

{

stat c nt GetLeadB tNum( nt Val)

{

nt B tNum = 31;

u nt CmpVal = 1u << B tNum; wh le (Val < CmpVal)

{

CmpVal >>= 1; B tNum--;

}

return B tNum;

}

stat c nt D v dePolynomMod( nt d v dend, nt d v der)

{

nt m = GetLeadB tNum(d v der);

nt k = GetLeadB tNum(d v dend) - m; nt rema nder;

f (k < 0)

{

return d v dend;

}

wh le (k >= 0)

{

nt d v der\_tmp = d v der << k;

d v dend = d v dend ^ d v der\_tmp; k = GetLeadB tNum(d v dend) - m;

}

return d v dend;

}

stat c vo d Ma n()

{

nt g, e, m, c, s, tlen, tc\_, tm\_, c\_; str ng tb, tc, tm;

Console.Wr teL ne("Введите g(x)");

g = Convert.ToInt32(Console.ReadL ne(), 2); Console.Wr teL ne("Введите m");

m = Convert.ToInt32(Console.ReadL ne(), 2);

// Генерация вектора ошибки Random rnd = new Random(); e = rnd.Next(0, 127);

Console.Wr teL ne("Контрольная сумма:\ne = " + Convert.ToStr ng(e, 2));

/\* Console.Wr teL ne("Вектор ошибок: ");

e = Convert.ToInt32(Console.ReadL ne(), 2);\*/

// Console.Wr teL ne("Вектор ошибок:\ne = " + e); m = m << GetLeadB tNum(g);

c = D v dePolynomMod(m, g);

Console.Wr teL ne("Контрольная сумма:\nс = " + Convert.ToStr ng(c, 2)); m = m | c;

Console.Wr teL ne("На выходе кодера:\na = " + Convert.ToStr ng(m, 2)); m = m ^ e;

tb = Convert.ToStr ng(m, 2); tlen = (tb.Length / 2) - 1;

tm = new str ng(tb.Take(tlen).ToArray()); tc = new str ng(tb.Take(2..^0).ToArray());

Console.Wr teL ne("На выходе канала:\nb = " + Convert.ToStr ng(m, 2));

s = D v dePolynomMod(m, g);

Console.Wr teL ne("Синдром:\n s = " + Convert.ToStr ng(s, 2)); f (s == 0)

{

Console.Wr teL ne("E = 0, ошибки не обнаружены");

}

else

{

Console.Wr teL ne("E = 1, ошибки обнаружены");

}

//Доп задание

Console.Wr teL ne("На выходе канала:\nmb = " + Convert.ToStr ng(tm)); Console.Wr teL ne("На выходе канала:\ncb = " + Convert.ToStr ng(tc)); tc\_ = Convert.ToInt32(tc, 2);

tm\_ = Convert.ToInt32(tm, 2);

tm\_ = tm\_ << GetLeadB tNum(g); c\_ = D v dePolynomMod(tm\_, g);

Console.Wr teL ne("Контрольная сумма:\nс'b = " + Convert.ToStr ng(c\_, 2));

f (tc\_ == c\_)

{

Console.Wr teL ne("E = 0, ошибки не обнаружены");

}

else

{

Console.Wr teL ne("E = 1, ошибки обнаружены");

}

}

}