Федеральное государственное бюджетное образовательное учреждение высшего образования «Сибирский государственный университет телекоммуникаций и информатики»

Лабораторная работа №5

**по дисциплине «Основы систем мобильной связи»**

**«Циклический избыточный код. CRC»**

Выполнил: Дембирел Д.Т. Группа: ИА-232

Проверил: Дроздова В.Г.

Вариант: 6 GitHub: https://github.com/Dembirel/laba5



Новосибирск 2024

**Содержание**

[Цель работы 3](#_bookmark0)

[Краткие теоретические сведения 4](#_bookmark1)

[Этапы выполнения работы 9](#_bookmark2)

[Контрольные вопросы 12](#_bookmark3)

[Заключение 14](#_bookmark4)

## Цель работы

Получить представление о том, как осуществляется проверка на наличие ошибок в пакетах с данными в современных системах связи (Error detection) посредством использования циклического избыточного кода CRC (Cyclic Redundancy Check).

**Задачи работы:**

* Написать программу на языке С/С++ для вычисления CRC для пакета данных длиной N бит и определения факта наличия ошибки при передаче пакета по каналу связи;
* Добавить полученный остаток от деления на G к пакету исходными данными;
* На приемной стороне повторно вычислить остаток от деления пакета с данными + CRC на полином G;
* Определить есть ли ошибка в принятом пакете;
* Вывести в окно терминала полученное значение CRC и отчет об ошибках в принятом пакете;
* Проделать предыдущие шаги при N = 250;
* Сделать цикл из 250 + CRC\_length итераций и в этом цикле по очереди искажать по одному биту;
* Проверить обнаружена ли ошибка на приемной стороне и выполнить подсчет того сколько раз за этот цикл приемник обнаружил и не обнаружил ошибки.

# Краткие теоретические сведения

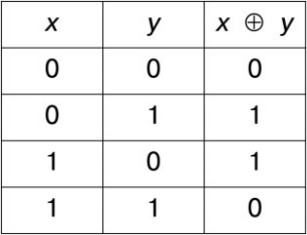
### Псевдослучайные двоичные последовательности

CRC — циклический избыточный код, иногда называемый также контрольным кодом или контрольной суммой. CRC – это добавочная порция избыточных бит, вычисляемых по заранее известному алгоритму на основе исходного передаваемого пакета данных (информационной битовой последовательности), которое передаѐтся вместе с самим пакетом по каналам связи (добавляется после информационных битов) и служит для контроля его безошибочной передачи.

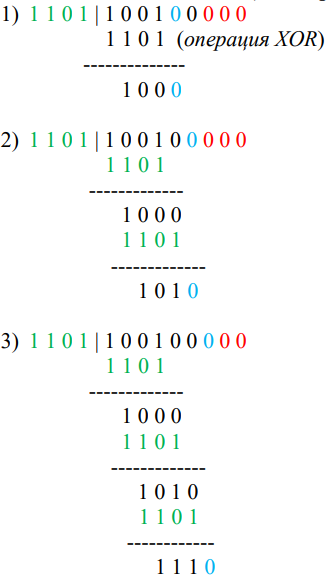
Простыми словами, CRC – это остаток от двоичного деления оригинального пакета с данными на какое-то двоичное n-разрядное число (порождающий полином), и его длина будет равна n-1 бит. Рассмотрим пример, где имеется 7 бит данных: 100100 и 4-битный порождающий полином 1101. Требуется определить CRC. Для того, чтобы выполнить деление этих битовых последовательностей нужно в конце последовательности с данными добавить n-1 нулей, как показано ниже, где n=4, для нашего случая.

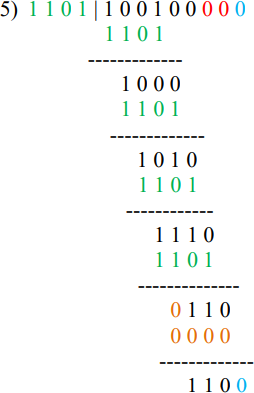
Делитель - 1 1 0 1 | 1 0 0 1 0 0 0 0 0 - Делимое (данные+n-1 нулей).

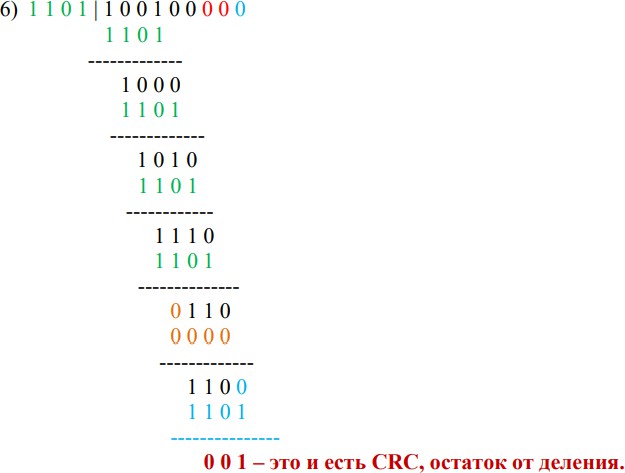
Основной операцией, используемой при делении бинарных чисел, является исключающее ИЛИ (XOR). Ниже показана таблица истинности для данной операции.



Пошаговое вычисление CRC (на стороне передатчика):







Делитель принято записывать в виде полинома. Если считать, что каждый разряд делителя — это коэффициент полинома, то этот полином будет иметь вид:

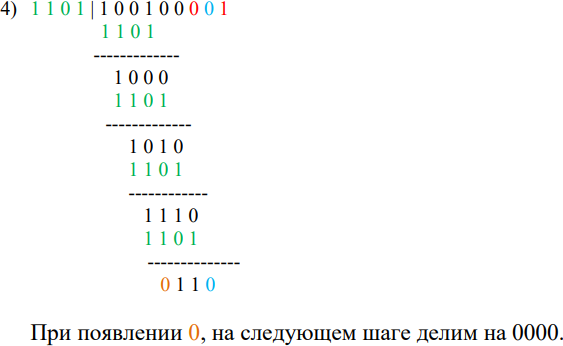
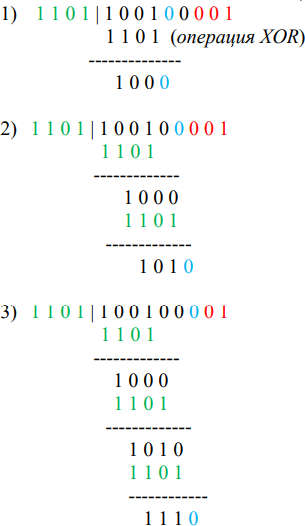


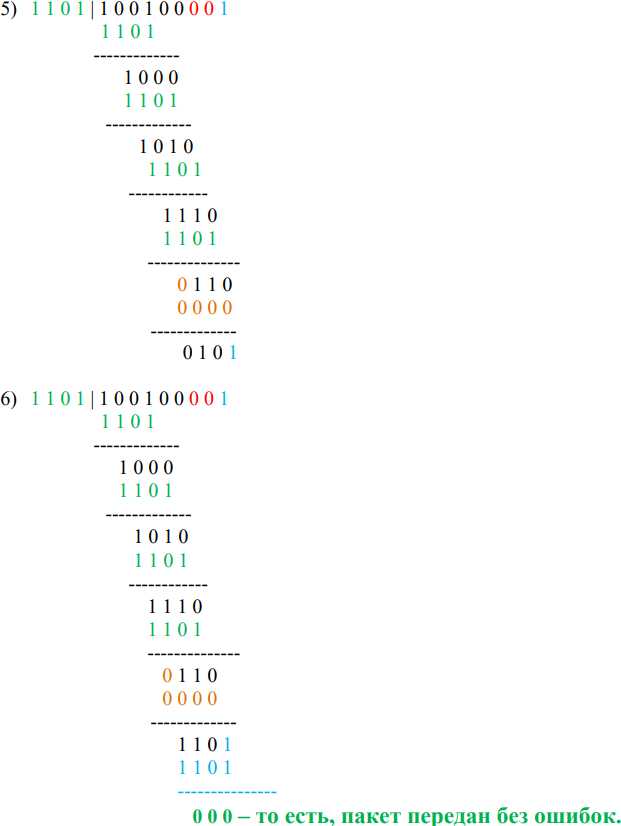
Таким образом, делитель из примера выше можно записать в виде полинома как: 1\*x3+1\*x2+0\*x1+1\*х0, или сокращенно как: x3+x2+1 = 1101.

Полученный остаток от деления CRC добавляется на передающей стороне к исходным данным и уже эта битовая последовательность, преобразованная в радиосигнал, передается в канал связи: 1 0 0 1 0 0 0 0 1.

На приемной стороне для обнаружения ошибки (или ее отсутствия) с полученным пакетом осуществляется ровно такая же процедура – деление на порождающий CRC полином. Если полученный в результате данного деления остаток будет ненулевым, то фиксируется факт наличия ошибки.

Пошаговое вычисление CRC (на стороне приемника):





# Этапы выполнения работы

### Исходные данные:

N = 20 + 6 = 26

G = x7 + x6 + x5 + x4 + x3 + x2 + 1

1. Первым делом требуется написать программу на языке С/С++ для вычисления CRC для пакета данных длиной N бит и определения факта наличия ошибки при передаче пакета по каналу связи. Для этого будет использован язык C++. Напишем функцию computeCRC, которая будет принимать в себя два вектора: вектор пакета и вектор полинома:

vector<int> computeCRC(const vector<int>& packet, const vector<int>& polynomial){ vector<int> packet\_zeros(packet.begin(), packet.end()); packet\_zeros.resize(packet.size() + polynomial.size() - 1, 0);

for (int i = 0; i <= packet.size(); ++i){ if (packet\_zeros[i] == 1) {

for (int j = 0; j < polynomial.size(); ++j) { packet\_zeros[i + j] ^= polynomial[j];

}

}

}

vector<int> CRC(packet\_zeros.end() - (polynomial.size() - 1), packet\_zeros.end()); return CRC;

}

Результатом выполнения функции является вектор, который содержит в себе значение вычисленного CRC для пакета с данными.

Второй функцией является проверка ошибок в пакете при передаче по каналу связи:

bool checkPacket(const vector<int>& receivedPacket, const vector<int>& polynomial){ vector<int> result = computeCRC(receivedPacket, polynomial);

for (int bit : result) { if (bit != 0){

return false;

}

}

return true;

}

Она также принимает в себя значения двух векторов: принятого пакета и полинома. Для обнаружения ошибки выполняется абсолютно такая же процедура, как и при расчѐте CRC. Поэтому здесь и используется функция computeCRC. Если в результате деления остаток будет нулевым, то ошибок в принятом пакете нет.

1. Сгенерируем случайный пакет данных длиной 32 бита. Для этого используем написанную функцию generate\_random\_packet:

vector<int> generate\_random\_packet(int size){ vector<int> packet;

for (int i = 0; i < size; i++){ packet.push\_back(rand() % 2);

}

return packet;

}

В результате получаем следующий пакет данных:



1. Вычислим для пакета данных значение CRC при помощи функции computeCRC:



Далее необходимо добавить получившееся CRC к нашему пакету и на приѐмной стороне вычислить факт наличия ошибок. Для этого используется следующий код:

vector<int> transmittedPacket(packet); transmittedPacket.insert(transmittedPacket.end(), CRC.begin(), CRC.end()); cout << "Пакет для передачи: ";

for (int i : transmittedPacket){ cout << i;

}

cout << endl;

if (checkPacket(transmittedPacket, G)) {

cout << "Пакет передан без ошибок." << endl;

} else {

cout << "Обнаружена ошибка в пакете." << endl;

}

После чего получаем:



1. Повторяем все прошлые шаги при длине пакета 250 бит. Генерируем пакет данных:



Вычисляем для него CRC:



Добавляем значение CRC в конец пакета и проверяем его на наличие ошибок:



1. Теперь создадим цикл из 250 + CRC итераций, и в нѐм, по очереди, будем искажать по одному биту, проверяя каждый раз пакет на наличие ошибки. Также будем вести подсчѐт того сколько раз за этот цикл приемник обнаружил и не обнаружил ошибки.

int errorsDetected = 0, errorsMissed = 0;

for (int i = 0; i < transmittedPacket\_250.size(); ++i){ vector<int> distortedPacket(transmittedPacket\_250); distortedPacket[i] ^= 1;

if (!checkPacket(distortedPacket, G)) { errorsDetected++;

} else {

errorsMissed++;

}

}

cout << "Обнаружено ошибок: " << errorsDetected << endl; cout << "Пропущено ошибок: " << errorsMissed << endl;

# Контрольные вопросы

1. **Для чего в мобильных сетях используются CRC-проверки?**

CRC – это добавочная порция избыточных бит, вычисляемых по заранее известному алгоритму на основе исходного передаваемого пакета данных (информационной битовой последовательности), которое передаѐтся вместе с самим пакетом по каналам связи (добавляется после информационных битов) и служит для контроля его безошибочной передачи.

Проще говоря, в мобильных сетях CRC-проверки используются для обнаружения случайных изменений в передаваемых данных. CRC позволяет проверить контрольную сумму пакета данных и выявить любые нежелательные изменения в переданных или сохранѐнных данных.

1. **Что такое порождающий полином?**

Порождающий полином **—** это предварительно специальным образом подобранный полином, на который впоследствии будет делиться информационный полином для вычисления контрольного кода**.**

Порождающий полином используется, например, в полиномиальных кодах, которые применяются для проверки целостности данных и при передаче данных.

Делитель принято записывать в виде полинома. Если считать, что каждый разряд делителя — это коэффициент полинома, то этот полином будет иметь вид:



Таким образом, делитель из примера выше можно записать в виде полинома как: 1\*x3+1\*x2+0\*x1+1\*х0, или сокращенно как: x3+x2+1 = 1101.

1. **Как вычислить CRC для пакета с данными?**

CRC – это остаток от двоичного деления оригинального пакета с данными на какое-то двоичное n-разрядное число (порождающий полином), и его длина будет равна n-1 бит. Для того, чтобы выполнить деление битовых

последовательностей нужно в конце последовательности с данными добавить n-1 нулей.

Основной операцией, используемой при делении бинарных чисел, является исключающее ИЛИ (XOR).

# Заключение

В ходе выполнения лабораторной работы мы узнали, как современные системы связи используют циклический избыточный код CRC (Cyclic Redundancy Check) для обнаружения ошибок в пакетах данных.

Мы разработали программу на языке C++, которая вычисляет значение CRC для каждого пакета и проверяет его на наличие ошибок после передачи по каналу связи.

В этой лабораторной работе мы протестировали пакеты данных различной длины на наличие ошибок. Кроме того, мы намеренно искажали пакет, а затем снова проверяли его на наличие ошибок, чтобы увидеть, как работает система обнаружения и исправления ошибок.