Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Отчет

К лабораторной работе №8

«Преобразования сигнала»

Выполнила:

студентка 2 курса 7 группы

Каракозова Р. В.

Руководитель:

доцент Буснюк Н.Н.

***Задание 1.***

Создать компьютерную программу реализации следующего задания.

1. Исходная строка – *фамилия, имя, отчество студента без пробелов, символ подчеркивания, номер группы, символы подчеркивания* **(**всего 32 байта). Пример: **СидоровСидорСидорович\_группа8\_\_\_** . Строку распечатать (печать).
2. Сгенерировать маску из 8 первых цифр (печать).
3. Выполнить перемежение (печать).
4. Произошла потеря 9 бит (затенение) на интервале N\*2 – N\*2+8 в соответствии с вариантом задания N. Развернуть исходную последовательность, заменив потерянные символы пробелами (печать).
5. Укорачивать область затенения с конца на 1 символ до тех пор, пока в восстановленной последовательности не окажется подряд двух затертых бит (символов).

**Выполнение задания 1:**

Строка после перемежения

Что произошло: выбираем первые 8 символов исходной строки, затем следующие 8 и так ещё 2 раза (т.к. 8\*4 = 32 - длина строки).

К этим подстрокам по 8 символов применяем маску и в соответствии с ней меняем последовательность символов в этих подстроках.

Например, первые 8 символов = Каракозо

индексы для каждого символа: К-0, а-1, р-2, а-3, к-4, о-5, з-6, о-7

**маска = 25410367**

Смотрим, какой символ подстроки стоит под индексом равным первой цифре из маски – видим, что под индексом 2 стоит символ «р», затем под индексом 5 (это у нас «о») и т.д.

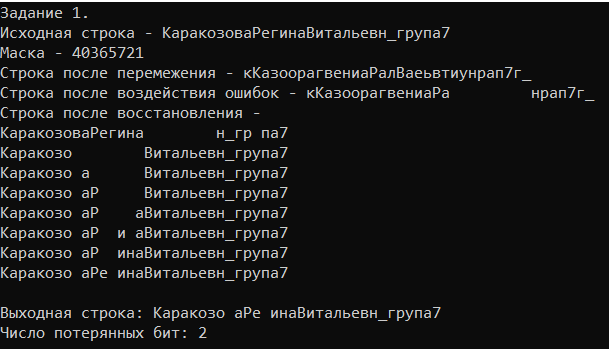
В итоге перемежения получаем первых 8 перемешанных символов.

То же самое повторяем для остальных 8-мисимвольных последовательностей.

Укорачивать область затенения с конца на 1 символ до тех пор, пока в восстановленной последовательности не окажется подряд двух затертых бит (символов).

То есть в последовательности не должно быть 2 и более затертых бит, идущих подряд.

Если сначала по условию 4 задания требовалась потеря 9 бит, то сейчас мы последовательно уменьшаем это число, т.е. происходит потеря 8 бит, 7, 6 и т.д. Пока мы не получим последовательность, в которой не будет двух и более подряд идущих пробелов (затертых бит).



***Задание 2.***

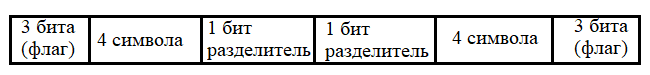
1. Исходная строка – *фамилия, имя, отчество студента без пробелов, символ подчеркивания, номер группы, символы подчеркивания* **(**всего 32 байта). Пример: **СидоровСидорСидорович\_группа8\_\_\_** (печать).
2. Выполнить первый уровень перемежения, разбив на 8 кадров по 4 символа (печать).
3. Создать пакеты из 16 символов, добавив по краям флаги из 3 символов, каждый символ – это номер пакета (цифры от 1 до 4); в середине в качестве разделителей использовать символ нижнего подчеркивания (печать).
4. Выполнить второй уровень перемежения (печать).

Произошла потеря 11 бит в соответствии с вариантом N на интервале:

N\*2 ––– N\*2+10. Развернуть исходную последовательность, заменив потерянные символы пробелами (печать).

***Выполнение задания 2:***

*1)На первом уровне перемежения* разбиваем последовательность на 8 кадров по 4 символа и из этих кадров формируем пакеты:



В нашем случае флаги – наборы цифр от 1 до 4, разделители – символы дефисов, между разделителями должна быть тренировочная последовательность, но у нас её нет (по заданию).

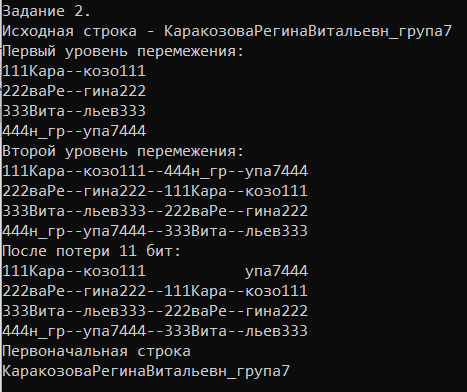
*2)На втором уровне объединяем пакеты*

На первом уровне перемежения мы получили 4 пакета.Допустим, A – 0 пакет, B – 1 пакет, C – 2 пакет, D – 3 пакет.Объединяем их через разделители таким образом: A--D, B--A, C--B, D--C.

*3)Далее потеря 11 бит* (вместо потерянных бит пробелы)

*4)После этого пытаемся восстановить исходную последовательность*

Последовательность удалось восстановить полностью, ибо группу потерянных символов из одного 16-тисимвольного пакета (пакет A--D) можно найти в другом 16-тисимвольном пакете (пакет D--C).



**Контрольные вопросы**

1. **Что такое «перемежение» и для чего оно предназначено?**

Перемежение — перестановка байтов по определенным правилам для устранения группирования ошибок.

Перемежения предназначено для борьбы с групповыми ошибками.

1. **Что такое «пакет нормального типа»?**

Нормальный пакет NB используется для передачи информации по каналам связи. Он состоит из:

·       3+3 бита флаги;

·       1+1 бита разделители полей;

·       57+57 битов информация;

·       26 битов тренировочная последовательность.

Изображение выглядит как стол

Автоматически созданное описание

1. **Что такое «кадр» при передаче сигнала в электросвязи? Чему он равен в стандарте GSM, как он формируется?**

Кадр – массив битов, служащий для хранения информации в том виде, чтобы её можно было перемежить. В GSM в одном кадре 57 битов. Формируется в блоке SGSN - узел обслуживания абонентов, выступающий точкой соединения между системой базовых станций и базовой сетью.

1. **Что такое «канальный интервал»?**

Канальный интервал - это частотный диапазон, который выделяется для передачи одного канала в беспроводной связи. Это означает, что когда вы используете беспроводную сеть, вы используете канальный интервал, чтобы передать данные между вашим устройством и точкой доступа.

1. **Из каких блоков состоит мобильная станция?**

В ее состав входит 3 основных блока: блок управления, приемопередающий блок, антенный блок.

*Блок управления* включает в себя микротелефонную трубку (микрофон и динамик), клавиатуру и дисплей.

*Приемопередающий блок* состоит из передатчика, приемника, синтезатора частот и логического блока.

*Антенный блок* включает в себя антенну и коммутатор прием/передача.

1. **Какие узлы включает в себя приемопередающий блок?**

Приемопередающий блок состоит из:

·       передатчика;

·       приёмника;

·       синтезатора частот;

**Синтезатор частот** (СЧ) - это устройство, формирующее из входного сигнала стабильной частоты выходной сигнал в требуемом частотном диапазоне.

·       логического блока.

**Логический блок** – это микрокомпьютер, осуществляющий управление работой приемо-передающей станцией.

1. **Какие виды преобразования сигнала в передатчике-приемнике еще используются? В каких узлах это происходит?**

Используется преобразование из аналогового (волны) сигнала в цифровой (биты). За это отвечают АЦП (аналого-цифровой преобразователь) и ЦАП (цифро-аналоговый преобразователь).

Перемежение сигнала осуществляется в блоке *кодера канала*.

1. **Что такое «эквалайзер», для чего он применяется в мобильной связи?**

Эквалайзер — это адаптивный фильтр, настраиваемый таким образом, чтобы сигнал на его выходе был в возможно большей степени очищен от межсимвольных искажений, содержащихся в принимаемом входном сигнале; блок эквалайзера не является функционально необходимым и в некоторых случаях может отсутствовать.

1. **Для чего служит «синтезатор» в мобильной станции?**

Синтезатор является источником колебаний несущей частоты, используемой для передачи информации по радиоканалу.

1. **Что такое «детектор речевой активности»?**

Детектор речевой активности - это компонент кодека, который используется для определения наличия речи во время телефонного разговора.

Он используется для сжатия аудио-сигнала в целях экономии пропускной способности сети и позволяет снизить задержку в передаче звука и улучшить качество воспроизведения речи.

Детектор анализирует звукового сигнала, который передается по телефонной линии. Он определяет, происходит ли речь в текущий момент времени или нет, и передает эту информацию в кодек для дальнейшей обработки.

Он позволяет сжимать аудио-сигнал только тогда, когда на линии нет речи, что позволяет сэкономить пропускную способность сети.

Когда речь начинается, детектор речевой активности отключает сжатие, чтобы сохранить качество аудио-сигнала.

1. **Где расположен «транскодер» и для чего он предназначен?**

**Транскодер** – это элемент, который выполняет функцию перекодировки речевого сигнала.

Данный элемент обеспечивает преобразование выходных сигналов канала передачи речи и данных ЦК (64 кбит/с) к виду, соответствующему рекомендациям GSM по радиоинтерфейсу (13 кбит/с). Транскодер обычно располагается вместе с ЦК.

В центре коммутации мобильной связи (MSC) устанавливается преобразователь-транскодер ТСЕ, который преобразует выборки речевого сигнала из одного цифрового формата в другой.

1. **Что такое «групповые ошибки»?**

Групповые ошибки – ошибки, которые сгруппированы и действуют последовательно на каждый бит передаваемой информации.

В лб объяснение как решать такие проблемы(перемещение).

1. **Какие еще способы противодействия негативным воздействиям на радиосигнал существуют?**

Перемежение, разнесенный прием, перескоки по частоте, помехоустойчивое кодирование, управление мощностью.

Разнесенный прием работает следующим образом:

1. Несколько антенн размещаются на разных удаленных устройствах.
2. Сигнал, принятый каждой антенной, передается на центральный узел для дальнейшей обработки.
3. На центральном узле сигналы, полученные от разных антенн, объединяются и обрабатываются с целью улучшения

Перескоки по частоте работают следующим образом:

1. Устройство, которое передает сигнал, выбирает набор частот, которые будут использоваться для передачи данных.
2. Устройство переключает частоту передачи сигнала с высокой скоростью в соответствии с заданным набором частот.
3. Устройство, которое принимает сигнал, также переключает частоту приема сигнала в соответствии с заданным набором частот.

Помехоустойчивое кодирование работает следующим образом:

1. Исходные данные разбиваются на блоки (например, байты) и кодируются с помощью специального кода, который добавляет дополнительные биты (называемые проверочными битами) к каждому блоку данных.
2. При передаче данных по каналу связи, возникающие помехи могут привести к изменению некоторых битов данных. Проверочные биты используются для обнаружения ошибок в блоке данных.
3. Если обнаружены ошибки, то с помощью специального алгоритма исправления ошибок проверочные биты позволяют определить, какие биты данных были изменены, и восстановить исходные данные.

Управление мощностью работает следующим образом:

1. Устройство определяет текущую нагрузку и потребление мощности.
2. Если устройство не используется или используется в минимальном режиме, то управление мощностью может перевести его в спящий режим или режим глубокого сна, чтобы уменьшить потребление энергии.
3. Если устройство используется в полную мощность, то управление мощностью может уменьшить потребление энергии путем снижения частоты процессора, яркости экрана, подключенных устройств и других параметров.
4. Управление мощностью может также использовать различные методы для оптимизации потребления энергии, такие как управление питанием компонентов, управление зарядом батареи и т.д.

**ПРОГРАММННЫЙ КОД**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text.RegularExpressions;

namespace laba8

{

static class Task1

{

public static void task1(string input)

{

string mask = RandMask("01234567");

Console.WriteLine($"Маска - {mask}");

string afterInput = InterLeaving(input, mask);

Console.WriteLine($"Строка после перемежения - {afterInput}");

string afterErrors = ErrorsGenerator(afterInput, 8, 8);

Console.WriteLine($"Строка после воздействия ошибок - {afterErrors}");

string afterAntiInput = AntiInterLeaving(afterErrors, mask);

Console.WriteLine($"Строка после восстановления - {afterAntiInput}");

LessErorrs(afterInput, mask, 4);

}

static string RandMask(string str)

{

// перемешиваем сиволы строки str

List<char> randArr = new List<char>();

Random random = new Random();

while (randArr.Count != str.Length)

{

char rand = random.Next(0, str.Length).ToString()[0];

if (!randArr.Contains(rand))

randArr.Add(rand);

}

return new string(randArr.ToArray());

}

static string InterLeaving(string str, string mask) //перемежение

{

string output = "";

for (int i = 0; i < 4; i++)

{

foreach (char m in mask)

{

output += str.Substring(i \* 8, 8)[(int)Char.GetNumericValue(m)];

}

}

return output;

}

// возвращение исходной последовательности символов перемеженной строки

static string AntiInterLeaving(string str, string mask)

{

char[] antiArr = new char[32];

for (int i = 0; i < 4; i++)

{

foreach (char m in mask)

{

antiArr[(int)Char.GetNumericValue(m) + 8 \* i] = str.Substring(i \* 8, 8)[mask.IndexOf(m)];

}

}

return new string(antiArr);

}

static string ErrorsGenerator(string str, int variant, int length)

{

// для замены символов в строке делаем массив char символов этой строки

char[] errors = str.ToCharArray();

string final = "";

// символы с "variant \* 2" (вкл) по "variant \* 2 + length" (не вкл) в строке заменяем на пробелы

for (int i = variant \* 2; i < variant \* 2 + length; i++)

{

errors[i] = ' ';

}

for (int i = 0; i < errors.Length; i++)

final += errors[i];

return final;

}

// before - строка сразу после перемежения

static string LessErorrs(string before, string mask, int variant)

{

int losing\_bits = 8; // кол-во потерянных бит после функции ErrorsGenerator примененной к перемеженной строке

string after\_losing = before;

for (int i = 6; i != 0; i--)

{

// применяем функцию ErrorsGenerator потери бит (сначала 8, потом 7 и т.д. до 1 бита) в перемеженной строке before

// затем результат этой функции отправляем на восстановление последовательности бит

// то есть, чтобы порядок символов стал, как в строке до перемежения

after\_losing = AntiInterLeaving(ErrorsGenerator(before, variant, i), mask);

Console.WriteLine(after\_losing);

losing\_bits = i;

// если полученная строка after\_losing не имеет последовательностей двух и более подряд идущих пробелов,

// то она подходит и мы прерываем цикл, и выводим эту строку

bool validate = true;

for (int j = 1; j < after\_losing.Length - 1; j++)

{

// если в строке идет два пробела подряд, то строка не прошла валидацию

if (after\_losing[j] == ' ' && after\_losing[j + 1] == ' ')

{

validate = false;

}

}

// если в предыдущем цикле не было ситуаций с подруд идущими двумя и более пробелами, то искомая строка найдена

if (validate)

break;

}

Console.WriteLine("\nВыходная строка: " + after\_losing);

Console.WriteLine("Число потерянных бит: " + losing\_bits);

return after\_losing;

}

}

static class Task2

{

public static void task2(string str)

{

string[] firstlvl = FirstLevel(str);

Console.WriteLine("Первый уровень перемежения:");

foreach (string el in firstlvl)

{

Console.WriteLine(el);

}

Console.WriteLine("Второй уровень перемежения:");

string[] secondlvl = SecondLevel(firstlvl);

foreach (string el in secondlvl)

{

Console.WriteLine(el);

}

Console.WriteLine("После потери 11 бит:");

string[] afterErrors = Errors(secondlvl, 8);

foreach (string el in afterErrors)

{

Console.WriteLine(el);

}

Console.WriteLine("Первоначальная строка");

Console.WriteLine(InitialString(afterErrors));

}

static string[] FirstLevel(string str)

{

// 8 подстрок по 4 символа

string[] sublvl = new string[8];

for(int i = 0; i < 8; i++)

{

sublvl[i] = str.Substring(i \* 4, 4);

}

string[] output = new string[4];

for (int i = 0; i < 4; i++)

{

// флаги из трех символов, первая подстрока sublvl, "--", вторая подстрока sublvl, флаги из трех символов

output[i] = $"{i + 1}{i + 1}{i + 1}{sublvl[i \* 2]}--{sublvl[i \* 2 + 1]}{i + 1}{i + 1}{i + 1}";

}

return output;

}

static string[] SecondLevel(string[] str)

{

// на первом уровне перемежения мы получили 4 пакета

// допустим, A - 0 пакет, B - 1 пакет, C - 2 пакет, D - 3 пакет

// объединяем их через разделители таким образом: A--D, B--A, C--B, D--C

string[] output = new string[4];

output[0] = $"{str[0]}--{str[3]}";

for (int i = 0; i < str[0].Length; i++)

output[0] += ' ';

for (int i = 1; i < 4; i++)

{

output[i] = $"{str[i]}--{str[i-1]}";

}

return output;

}

static string[] Errors(string[] str, int variant)

{

string[] output = new string[4];

// помещаем в temp массив строк str, полученных на втором этапе перемежения

string temp = "";

foreach (string i in str)

{

temp += i;

}

temp = temp.Substring(0, temp.Length / 4 - 4) + temp.Substring(temp.Length / 4 + 12); // такм какие-то пробелы после первого кадра, убираем их

// символы с variant \* 2 (вкл) по variant \* 2 + 10 (вкл) заменяем на пробел

char[] errors = temp.ToCharArray();

for (int i = variant \* 2; i <= variant \* 2 + 10; i++)

{

errors[i] = ' ';

}

for (int i = 0; i < 4; i++)

{

output[i] = new string(errors).Substring(errors.Length/4 \* i, errors.Length / 4);

}

return output;

}

static string InitialString(string[] str)

{

string[] temp = {str[1], str[2], str[3] };

string tempStr = "";

foreach (string i in temp)

{

tempStr += i;

}

tempStr = tempStr.Substring(16);

tempStr = tempStr.Substring(0, tempStr.Length - 16);

temp = tempStr.Split('-');

tempStr = "";

foreach (string i in temp)

{

tempStr += i;

}

return tempStr.Substring(3, 8) + tempStr.Substring(31, 8) + tempStr.Substring(17, 8) + tempStr.Substring(45, 8);

}

}

class Program

{

static void Main()

{

string input = "КаракозоваРегинаВитальевн\_група7"; // 32 бита

Console.WriteLine($"Задание 1.\nИсходная строка - {input}");

Task1.task1(input);

Console.ReadKey();

Console.Clear();

Console.WriteLine($"Задание 2.\nИсходная строка - {input}");

Task2.task2(input);

Console.ReadKey();

}

}

}