Московский государственный технический университет им. Н.Э. Баумана
Кафедра «Системы обработки информации и управления»

УТВЕРЖДАЮ:

_____ Галкин В. А.

«__» _____ 2017 г.

Расчётно-пояснительная записка
к курсовой работе
«Локальная безадаптерная сеть»
(вариант № 26а)
по курсу «Сетевые технологии в АСОИУ»

ИСПОЛНИТЕЛИ:

_____ Лещев А. О., ИУ5-64

_____ Мельников К. И., ИУ5-64

«__» _____ 2017 г.

Москва — 2017 г.

Содержание

1. Введение

Наименование программы: «Локальная безадаптерная сеть». Основанием для разработки является учебный план кафедры ИУ5 «Системы обработки информации и управления» МГТУ им. Н.Э. Баумана на 6 семестр. Исполнителями являются студенты МГТУ им. Н.Э. Баумана группы ИУ5-64 Лещев Артем Олегович и Мельников Константин Игоревич. Программа выполнена на языке программирования C# и работает под управлением операционной системы Microsoft Windows версии 10 или выше.

Программа «Локальная безадаптерная сеть», созданная в рамках курсовой работы по курсу «Сетевые технологии в АСОИУ» кафедры ИУ5 «Системы обработки информации и управления» МГТУ им. Н.Э. Баумана, позволяет передавать файлы по локальной сети с возможностью докачки после восстановления прерванной связи, состоящей из двух персональных компьютеров, соединённых через интерфейс RS-232C нуль-модемным кабелем.

1.1. Требования к программе

К программе предъявляются следующие требования:

1. Программа должна реализовывать разработанные протоколы взаимодействия объектов пользовательского, канального и физического уровней локальной сети.

2. Программа должна защищать передаваемую информацию циклическим кодом [15, 11].

3. Программа должна реализовать функцию передачи файлов между двумя персональными компьютерами с возможностью докачки после восстановления прерванной связи.

2. Определение структуры программного продукта

При взаимодействии компьютеров между собой выделяются несколько уровней: нижний уровень должен обеспечивать передачу, а верхний — управление передачей и обеспечение интерфейса пользователя. Программа разбивается на три уровня: физический, канальный и пользовательский.

Физический уровень предназначен для сопряжения компьютера со средой передачи и непосредственной передачи данных по ней.

Канальный уровень занимается формированием и проверкой кадров, оборачивающих пакеты верхнего уровня.

Пользовательский уровень занимается формированием пакетов и квитанций, обеспечивает изменение интерфейса, выполняет основную задачу программы.

3. Физический уровень

3.1. Функции физического уровня

Основными функциями физического уровня являются:

1. установление параметров COM-порта;

2. установление, поддержание и разъединение физического канала;

3. непосредственная передача данных по каналу.

3.2. Описание физического уровня

На физическом уровне поддерживается передача по единственной линии с последовательной передачей данных в дуплексном канале. При этом для синхронизации группе битов данных предшествует специальный стартовый бит, далее после группы битов следует один стоповый бит.

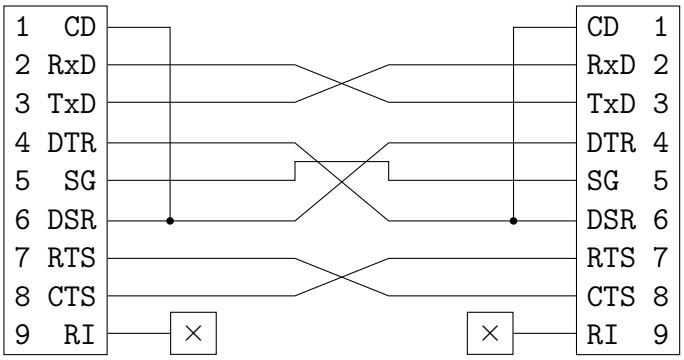
Стартовый бит сигнализирует о начале передачи данных. Далее передаются биты данных. В самом конце передается один стоповый бит, завершающий передачу байта.

Для корректной передачи данных настройка портов должна быть одинакова как на стороне приемника, так и на стороне отправителя. Соответственно, формат данных на обоих портах должен быть настроен одинаково.

Другая важная характеристика — скорость передачи данных. Она также должна быть одинаковой для передатчика и приемника. Измеряется в бодах.

Интерфейс RS-232C описывает несимметричный интерфейс, работающий в режиме последовательного обмена двоичными данными. Интерфейс поддерживает как асинхронный, так и синхронный режимы работы.

Последовательная передача данных означает, что данные передаются по единственной линии. При этом биты байта данных передаются по очереди с использованием одного провода. Интерфейс называется несимметричным, если для всех цепей обмена интерфейса используется один общий возвратный провод — сигнальная «земля». Для определения состояния абонента на другой стороне кабеля используются другие контакты.



Контакт DTR (Data Terminal Ready) — готовность к приёму данных. Означает готовность к приему данных (физическое соединение).

Контакт RTS (Request to Send) — запрос на передачу. Означает готовность к приёму данных.

Контакт DSR (Data Set Ready) — готовность к передаче данных.

Контакт CTS (Clear to Send) — готовность передачи.

Контакт DCD (Carrier Detect) — наличие несущей.

3.3. Защита передаваемой информации

Физический уровень получает массив бит, которые следует передать. Для осуществления надёжной передачи на данном уровне, массив делится на группы 11 бит или меньше, которые защищаются циклическим кодом [15, 11].

Отобранные 11 бит подвергаются сдвигу влево на 4 бита с заполнением нулями. Далее полученный массив бит подвергается циклическому делению с помощью сложения по модулю 2 с образующим полиномом $x^4 + x + 1$, полученный остаток записывается в 4 последних бита.

Обратная операция аналогична. Если остаток (синдром ошибки) от циклического деления с помощью сложения по модулю 2 с образующим полиномом больше 0, то считается, что обнаружена ошибка. Защита циклическим кодом может исправить 1 бит. Если ошибка больше одного бита, это будет видно по контрольной сумме на канальном уровне.

4. Настройка COM-порта средствами C#

Пространство имен `System.IO.Ports` стандартной библиотеки платформы .NET Framework языка программирования C# предлагает широкие возможности по настройке COM-порта.

4.1. Описание класса `SerialPort`

Данный класс используется для управления файловым ресурсом последовательного порта. Данный класс предоставляет возможности управления вводом-выводом в синхронном режиме или на основе событий, доступа к состоянию линии и состоянию разрыва, а также доступа к свойствам последовательного драйвера.

Методы класса

Имя	**Описание**
`Close`	Закрывает соединение порта, присваивает свойству `IsOpen` значение `false` и уничтожает внутренний объект `Stream`.
`GetPortNames`	Получает массив имен последовательных портов для текущего компьютера.
`Open`	Открывает новое соединение последовательного порта.
`Read(System.Byte[], System.Int32, System.Int32)`	Считывает из входного буфера `SerialPort` определенное число байтов и записывает их в байтовый массив, начиная с указанной позиции.
`ReadByte`	Считывает из входного буфера `SerialPort` один байт в синхронном режиме.
`DiscardInBuffer`	Удаляет данные из буфера приема последовательного драйвера
`DiscardOutBuffer`	Удаляет данные из буфера передачи последовательного драйвера
`Write(System.String)`	Записывает указанную строку в последовательный порт.

Свойства класса

Имя	**Описание**
`BaudRate`	Получает или задает скорость передачи для последовательного порта (в бодах).

Имя	Описание
BreakState	Получает или задает состояние сигнала разрыва.
BytesToRead	Получает число байтов данных, находящихся в буфере приема.
BytesToWrite	Получает число байтов данных, находящихся в буфере отправки.
CDHolding	Получает состояние линии обнаружения несущей для порта.
CtsHolding	Получает состояние линии готовности к приему.
DataBits	Получает или задает стандартное число битов данных в байте.
DsrHolding	Получает или задает состояние сигнала готовности данных (DSR).
DtrEnable	Получает или задает значение, включающее поддержку сигнала готовности терминала (DTR) в сеансе последовательной связи.
Encoding	Получает или задает кодировку байтов для преобразования текста до и после передачи.
Events	Возвращает список обработчиков событий, которые прикреплены к этому объекту Component (унаследовано от Component).
IsOpen	Получает значение, указывающее состояние объекта SerialPort — открыт или закрыт.
Parity	Получает или задает протокол контроля четности.
PortName	Получает или задает последовательный порт, в частности, любой из доступных портов COM.
ReadBufferSize	Получает или задает размер входного буфера SerialPort.
RtsEnable	Получает или задает значение, показывающее, включен ли сигнал запроса передачи (RTS) в сеансе последовательной связи.
StopBits	Получает или задает стандартное число стоповых битов в байте.
WriteBufferSize	Получает или задает размер выходного буфера последовательного порта.

События класса

Имя	Описание
DataReceived	Представляет метод обработки события получения данных для объекта SerialPort.
ErrorReceived	Представляет метод обработки ошибок объекта SerialPort.
PinChanged	Представляет метод обработки события изменения состояния контактов для объекта SerialPort.

4.2. Описание класса Physical

Данный класс был разработан в ходе работы над данной программы для реализации её физического уровня.

Поля класса

Имя	Описание
`_serialPort`	Экземпляр объекта `SerialPort`.
`connected`	Состояние подключения.
`failList`	Список синдромов ошибок циклического кода [15, 11].

События класса

Имя	Описание
`OnCheck`	Событие, срабатывающее при изменении состояния COM-порта.
`UICheck`	Событие, отправляющее состояние порта при попытке отправки группы байт.

Методы класса

Имя	Описание
`SetRts`	Позволяет выставлять контакт RTS.
`StatusCheck`	Обработчик изменения состояния COM-порта.
`Connect`	Метод подключения к COM-порту.
`Disconnect`	Метод отключения от COM-порта.
`DataReceivedHandler`	Обработчик события получения байт.
`DeCycle`	Метод для декодирования циклического кода [15, 11].
`Send`	Метод для отправки кадра.
`GetCycled`	Метод, кодирующий данные циклическим кодом [15, 11].

5. Канальный уровень

5.1. Функции канального уровня

На канальном уровне выполняются следующие функции:

1. управление передачей кадров;
2. обеспечение необходимой последовательности блоков данных, передаваемых через межуровневый интерфейс;
3. контроль и исправление ошибок.

Канальный уровень управляет передачей кадров, но не гарантирует их доставку. Управление соединением осуществляется через пакеты на пользовательском уровне.

Канальный уровень получает массив байт (пакет), экранирует все байты `0xFE` и `0xFE`, считает проверочную сумму, добавляет её в начало кадра. Добавляет стартовый и стоповый байты кадра. После этого кадр отправляется на физический уровень.

5.2. Описание класса DataLink

Данный класс был разработан в ходе работы над данной программы для реализации её канального уровня.

Поля класса

Имя	Описание
currentPacket	Пакет для отправки.
checkSumm	Проверочная сумма, посчитанная на основе пакета.
length	Длина пакета.
firstTrigger	Триггер, показывающий, работает ли в данный момент канальный уровень с пакетом.
secondTrigger	Триггер, показывающий, получен ли пакет полностью.
screenTrigger	Триггер, показывающий, был ли предыдущий байт экранирован.
recievedPacket	Список байтов, полученных от физического уровня.
recievedPacketBuffer	Промежуточный список битов, полученных от физического уровня.
debugBuffer	Отладочный буфер битов, полученных от физического уровня.

Методы класса

Имя	Описание
DataLink	Конструктор класса.
TimerListener	Функция, определяющая, необходимо ли очистить буфер по истечении таймера.
RecievePacket	Функция, вызываемая физическим уровнем для обработки полученных бит.
SendPacket	Функция, формирующая из пакета кадр и отправляющая его на физический уровень.

5.3. Формат кадра

Стартовый байт	Контрольная сумма	Пакет	Стоповый байт
11111111	$\sum_{i=1}^{m}$ newPacket$[i]$	newPacket	11111111
1 байт	4 байта	m байтов	1 байт

6. Пользовательский уровень

Функции пользовательского уровня обеспечивает интерфейс программы через форму. Пользовательский уровень предоставляет нижнему уровню пакет для отправки.

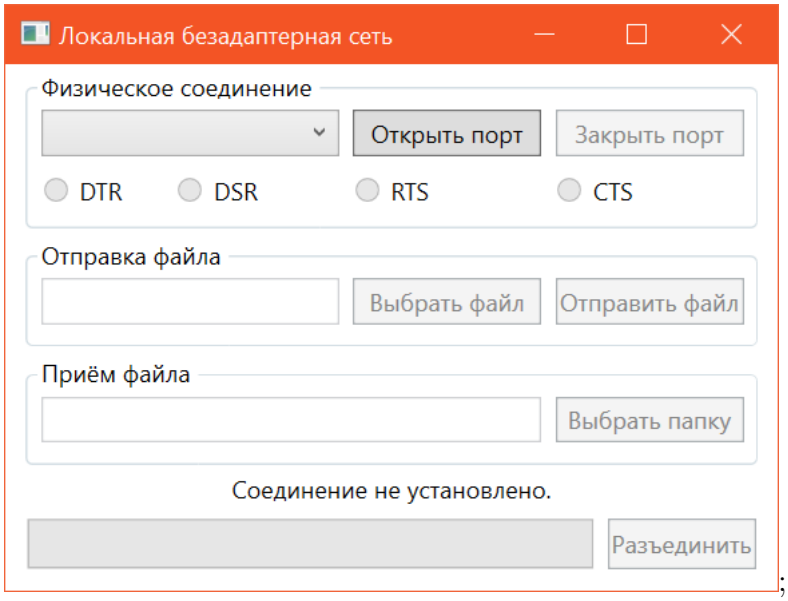
На данном уровне обеспечивается выбор файла и папки, отображение статуса COM-порта и передачи файла, вывод системных сообщений.

6.1. Описание интерфейса

Пользовательский интерфейс выполнен в среде Microsoft Visual Studio с использованием библиотеки Windows Presentation Foundation.

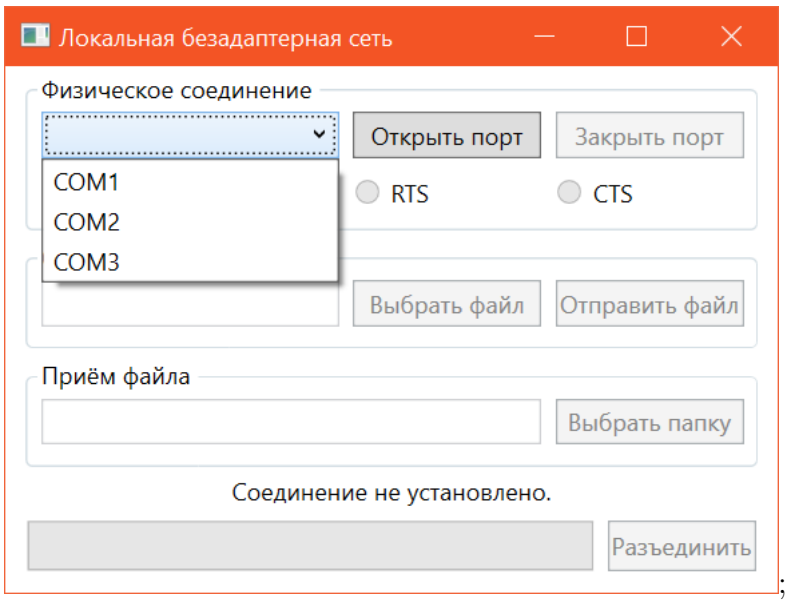
Основным окном программы является окно «Локальная безадаптерная сеть». В данной форме есть следующие возможности:

- выбор COM-порта;
- открытие COM-порта;
- закрытие COM-порта;
- мониторинг состояния контактов COM-порта;
- выбор файла для отправки;
- отправка файла;
- выбор папки для приёма файла;
- мониторинг состояния передачи файла;
- разъединение логического соединения.



;

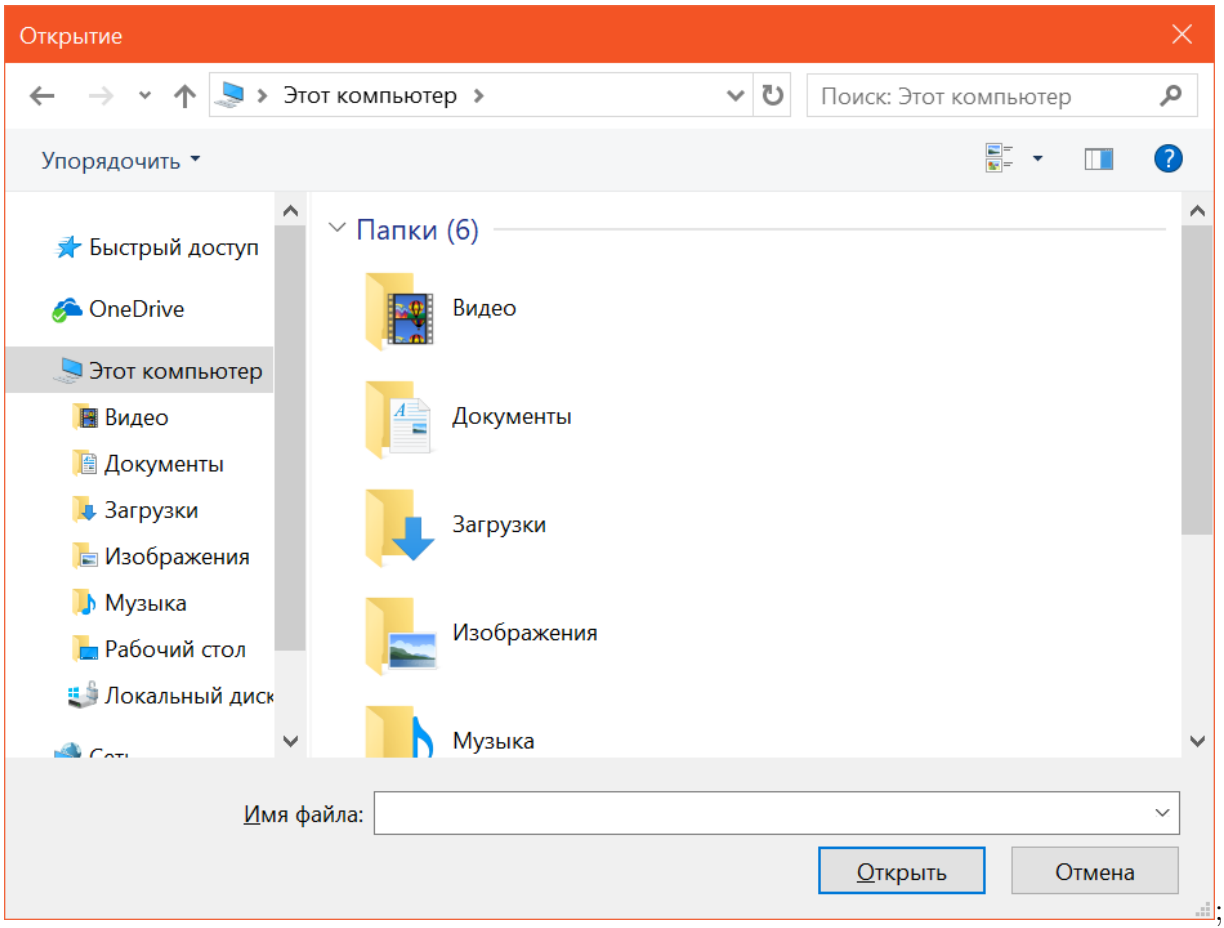
При открытии списка отображается список доступных COM-портов.



;

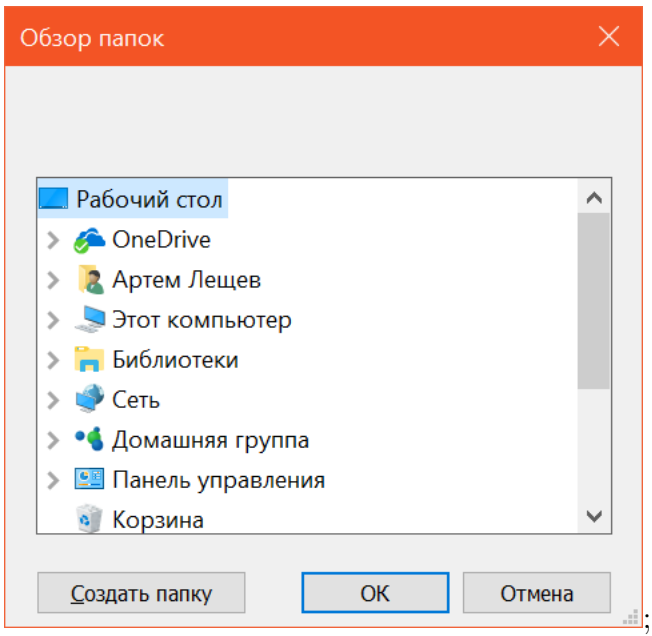
При нажатии на кнопку «Открыть порт» происходит открытие порта. При нажатии кнопки «Закрыть порт» происходит закрытие порта.

При нажатии на кнопку «Выбрать файл», открывается стандартное окно выбора файла.

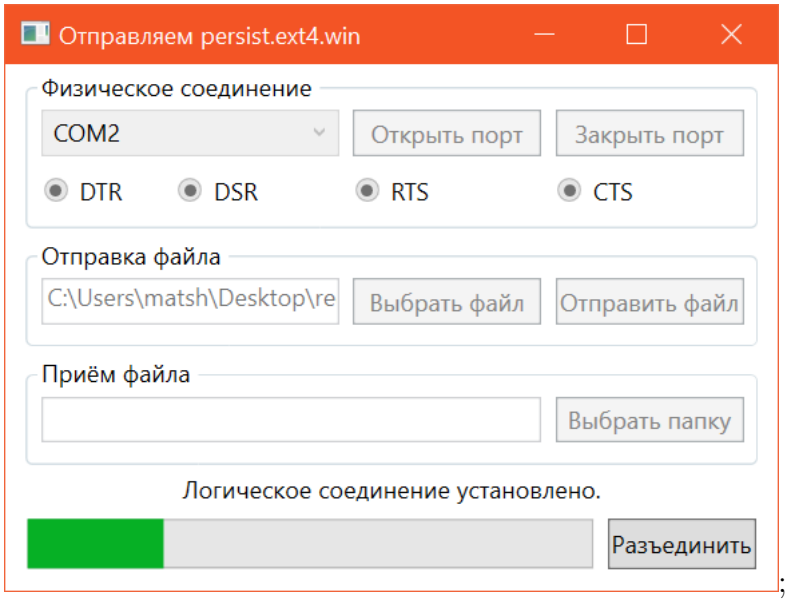


;

При нажатии на кнопку «Выбрать папку», открывается стандартное окно выбора папки.



После нажатия на кнопку «Отправить файл» происходит отправка файла. Процесс отправки отображается индикатором.



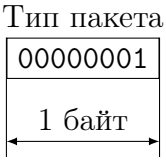
Нажатием кнопки «Разъединить» можно досрочно прервать передачу файла. После этого можно возобновить передачу, снова нажав кнопку «Отправить файл».

6.2. Форматы пакетов

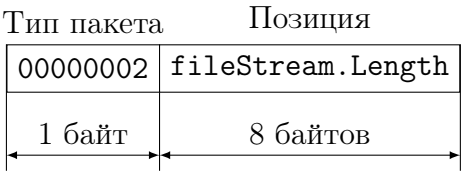
1. Пакет данных о файле

Тип пакета	Имя файла	Длина файла	Контрольная сумма
`00000000`	`fileDialog.SafeFileName`	`fileStream.Length`	SHA512(`fileStream`)
1 байт	$(k + \lfloor \log_{128} k \rfloor + 1)$ байтов	8 байтов	64 байта

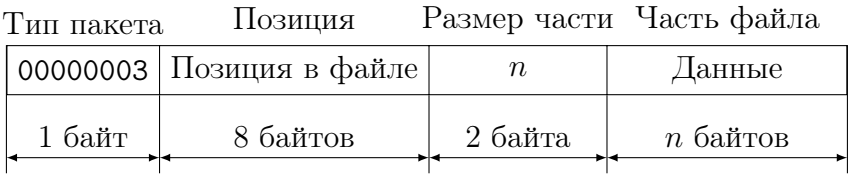
2. Пакет неготовности к получению файла



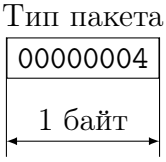
3. Пакет запроса части файла



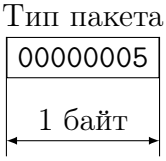
4. Пакет части файла



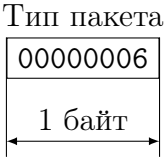
5. Пакет квитанции о получении файла



6. Пакет квитанции о завершении передачи



7. Пакет разрыва соединения



8. Пакет квитанции о разрыве соединения