Московский государственный технический университет имени Н. Э. Баумана.

Факультет “Информатика и системы управления”.

Кафедра “Системы обработки информации и управления”.

|  |  |
| --- | --- |
| Утверждаю: |  |
| Галкин В.А. | "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2016г. |

Курсовая работа

по курсу

Сетевые технологии

##### «Программа пересылки файлов»

Расчетно-пояснительная записка

(вид документа)

бумага А4

(вид носителя)

15

(количество листов)

Вариант 25

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛИ: |  |
| студенты группы ИУ5-64 |  |
| Вострокнутов И.Н. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Егоров А.С. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Мусин Т.М. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2016 г. |

Москва 2016

Оглавление

[1. ФИЗИЧЕСКИЙ УРОВЕНЬ. 3](#_Toc450576736)

[1.1. Интерфейс RS-232C. 3](#_Toc450576737)

[1.2. Физические параметры интерфейса RS-232C. 3](#_Toc450576738)

[1.3. Асинхронная передача данных. 5](#_Toc450576739)

[1.4. Реализация физического уровня. 7](#_Toc450576740)

[1.4.1. Открытие порта. 7](#_Toc450576741)

[1.4.2. Закрытие порта. 8](#_Toc450576742)

[1.4.3. Передача и прием данных 8](#_Toc450576743)

[2. КАНАЛЬНЫЙ УРОВЕНЬ. 8](#_Toc450576744)

[2.1. Защита передаваемой информации. 8](#_Toc450576745)

[2.2. Передача данных. 9](#_Toc450576746)

[2.3. Функции кодирования/декодирования. 10](#_Toc450576747)

[2.4. Форматы кадров. 10](#_Toc450576748)

[3. ПОЛЬЗОВАТЕЛЬСКИЙ УРОВЕНЬ 11](#_Toc450576749)

[3.1. Формы 11](#_Toc450576750)

[3.2. Работа форм по порядку 13](#_Toc450576751)

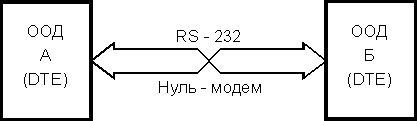
# 1. ФИЗИЧЕСКИЙ УРОВЕНЬ.

## Интерфейс RS-232C.

Интерфейс RS-232C (официальное название "EIA/TIA–232–E") является наиболее широко распространенной стандартной последовательной связью между микрокомпьютерами и периферийными устройствами. Интерфейс, определенный стандартом Ассоциации электронной промышленности (EIA), подразумевает наличие оборудования двух видов: терминального DTE и связного DCE.

Терминальное оборудование, например микрокомпьютер, может посылать или принимать данные по последовательному интерфейсу. Оно как бы оканчивает (terminate) последовательную линию. Связное оборудование - устройства, которые могут упростить передачу данных совместно с терминальным оборудованием. Наглядным примером связного оборудования служит модем (модулятор-демодулятор). Он оказывается соединительным звеном в последовательной цепочке между компьютером и телефонной линией.

Конечной целью подключения является соединение двух устройств DTE. Полная схема соединения включает в себя устройства DCE соединённые с DTE через интерфейс RS-232 и линию удалённой связи. Интерфейс позволяет исключить канал удаленной связи вместе с парой устройств DCE (модемов), соединив устройства непосредственно с помощью нуль-модемного кабеля (рис. 1).

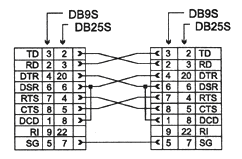


**Рис. 1.** Соединение по RS-232C нуль-модемным кабелем

## Физические параметры интерфейса RS-232C.

Стандарт RS-232C регламентирует типы применяемых разъемов, что обеспечивает высокий уровень совместимости аппаратуры различных производителей. На аппаратуре DTE (в том числе, и на COM-портах PC) принято устанавливать вилки (male - "папа") DB25-P или DB9-P. Девятиштырьковые разъемы не имеют контактов для дополнительных сигналов, необходимых для синхронного режима.

В случае когда аппаратура DTE соединяется без модемов ("Короткозамкнутая петля"), то разъемы устройств (вилки) соединяются между собой нуль-модемным кабелем (Zero modem или Z-modem), имеющим на обоих концах розетки, контакты которых соединяются перекрестно схеме, приведенной на рис. 2.

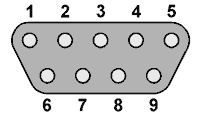


**Рис. 2.** Полный нуль-модемный кабель.

В таблице 1 приведено назначение контактов разъемов COM-прортов (и любой другой аппаратуры DTE). Назначение контактов разъема DB9S (рис. 3) определено стандартом EIA/TIA-574.

**Таблица 1.** Разъемы и сигналы интерфейса RS-232C.

|  |  |  |  |
| --- | --- | --- | --- |
| **Обозначе-ние цепи** | **Контакт разъема** | **Напра-вление** | **Название цепи** |
| **RS232** | **DB9S** | **Вход/Выход** |  |
| PG | - | - | Protect Ground - Защитная земля |
| TD | 3 | Выход | Transmit Data - Передаваемые данные |
| RD | 2 | Вход | Receive Data - Принимаемые данные |
| RTS | 7 | Выход | Request To Send - Запрос на передачу |
| CTS | 8 | Вход | Clear To Send - Готовность модема к приему данных для передачи |
| DSR | 6 | Вход | Data Set Ready - Готовность модема к работе |
| SG | 5 | - | Signal Ground - Схемная земля |
| DCD | 1 | Вход | Data Carrier Detect - Несущая обнаружена |
| DTR | 4 | Выход | Data Terminal Ready - Готовность терминала (PC) к работе |
| RI | 9 | Вход | Ring Indicator - Индикатор вызова |



**Рис. 3.** Назначение контактов разъема DB9.

1. Установкой DTR компьютер указывает на желание использовать модем.
2. Установкой DSR модем сигнализирует о своей готовности и установлении соединения.
3. Сигналом RTS компьютер запрашивает разрешение на передачу и заявляет о своей готовности принимать данные от модема.
4. Сигналом CTS модем уведомляет о своей готовности к приему данных от компьютера и передаче их в линию.
5. Снятием CTS модем сигнализирует о невозможности дальнейшего приема (например, буфер заполнен) — компьютер должен приостановить передачу данных.
6. Сигналом CTS модем разрешает компьютеру продолжить передачу (в буфере появилось место).
7. Снятие RTS может означать как заполнение буфера компьютера (модем должен приостановить передачу данных в компьютер), так и отсутствие данных для передачи в модем. Обычно в этом случае модем прекращает пересылку данных в компьютер.
8. Модем подтверждает снятие RTS сбросом CTS.
9. Компьютер повторно устанавливает RTS для возобновления передачи.
10. Модем подтверждает готовность к этим действиям.
11. Компьютер указывает на завершение обмена.
12. Модем отвечает подтверждением.
13. Компьютер снимает DTR, что обычно является сигналом на разрыв соединения (“повесить трубку”).
14. Модем сбросом DSR сигнализирует о разрыве соединения.

## Асинхронная передача данных.

Асинхронный режим передачи является байт-ориентированным (символьно-ориентированным): минимальная пересылаемая единица информации — один байт (один символ). Формат посылки байта иллюстрирует рис. 4. Передача каждого байта начинается со старт-бита, сигнализирующего приемнику о начале посылки, за которым следуют биты данных и, возможно, бит четности (Parity). Завершает посылку стоп-бит, гарантирующий паузу между посылками. Старт-бит следующего байта посылается в любой момент после стоп-бита, то есть между передачами возможны паузы произвольной длительности. Старт-бит, имеющий всегда строго определенное значение, обеспечивает простой механизм синхронизации приемника по сигналу от передатчика. Подразумевается, что приемник и передатчик работают на одной скорости обмена. Внутренний генератор синхронизации приемника использует счетчик-делитель опорной частоты, обнуляемый в момент приема начала старт-бита. Этот счетчик генерирует внутренние стробы, по которым приемник фиксирует последующие принимаемые биты. В идеале стробы располагаются в середине битовых интервалов, что позволяет принимать данные и при незначительном рассогласовании скоростей приемника и передатчика. Очевидно, что при передаче 8 бит данных, одного контрольного и одного стоп-бита предельно допустимое рассогласование скоростей, при котором данные будут распознаны верно, не может превышать 5 %. С учетом фазовых искажений и дискретности работы внутреннего счетчика синхронизации реально допустимо меньшее отклонение частот. Чем меньше коэффициент деления опорной частоты внутреннего генератора (чем выше частота передачи), тем больше погрешность привязки стробов к середине битового интервала, и требования к согласованности частот становятся более строгие. Чем выше частота передачи, тем больше влияние искажений фронтов на фазу принимаемого сигнала. Взаимодействие этих факторов приводит к повышению требований к согласованности частот приемника и передатчика с ростом частоты обмена.



**Рис. 4.** Формат асинхронной передачи RS-232C

Формат асинхронной посылки позволяет выявлять возможные ошибки передачи.

* Если принят перепад, сигнализирующий о начале посылки, а по стробу старт-бита зафиксирован уровень логической единицы, старт-бит считается ложным и приемник снова переходит в состояние ожидания. Об этой ошибке приемник может не сообщать.
* Если во время, отведенное под стоп-бит, обнаружен уровень логического нуля, фиксируется ошибка стоп-бита.
* Если применяется контроль четности, то после посылки бит данных передается контрольный бит. Этот бит дополняет количество единичных бит данных до четного или нечетного в зависимости от принятого соглашения. Прием байта с неверным значением контрольного бита приводит к фиксации ошибки.
* Контроль формата позволяет обнаруживать обрыв линии: как правило, при обрыве приемник “видит” логический нуль, который сначала трактуется как старт-бит и нулевые биты данных, но потом срабатывает контроль стоп-бита.

Для асинхронного режима принят ряд стандартных скоростей обмена: 50, 75, 110, 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600 и 115200 бит/с (в своей работе мы использовали 9600). Иногда вместо единицы измерения “бит/с” используют “бод” (baud), но при рассмотрении двоичных передаваемых сигналов это некорректно. В бодах принято измерять частоту изменения состояния линии, а при недвоичном способе кодирования (широко применяемом в современных модемах) в канале связи скорости передачи бит (бит/с) и изменения сигнала (бод) могут отличаться в несколько раз.

Количество бит данных может составлять 5, 6, 7 или 8 (5- и 6-битные форматы распространены незначительно). Количество стоп-бит может быть 1, 1,5 или 2 (“полтора бита” означает только длительность стопового интервала).

## Реализация физического уровня.

### 1.4.1. Открытие порта.

В ОС Windows доступ к COM-портам предоставляется посредством файловых интерфейсов. Для работы с портом - функции пространства имён **System.IO.Ports** из библиотеки классов .NET FRAMEWORK.

**ComPort** - объект класса SerialPort, который используется для определения COM-порта.

**ComPort.Open() –** функция открытия COM-порта.

После открытия порта производится его сброс. Порт очищается сам при считывании всех байтов с помощью функции **COMPort.Read.**

Вызов этой функции позволяет решить две задачи: очистить очереди приема/передачи в драйвере и завершить все находящиеся в ожидании запросы ввода/вывода.

Установка параметров COM-порта осуществляется с помощью функций:

**COMPort.BaudRate() –** устанавливаем скорость передачи данных;

**ComPort.PortName()** - имя порта;

**ComPort.ReadBufferSize()** – устанавливается размер буфера чтения;

**ComPort.WriteBufferSize**() - устанавливается размер буфера записи;

**ComPort.Handshake()** - протокол установления связи;

Далее производится настройка тайм-аутов чтения и записи порта с помощью функций:

**ComPort.ReadTimeout ()** - таймаут чтения;

**ComPort.WriteTimeout()** - таймаут записи;

Тайм-ауты необходимы для правильной работы функций чтения из порта и записи в порт.

### 1.4.2. Закрытие порта.

Перед закрытием порта производится его сброс. Порт очищается сам при считывании всех байтов с помощью функции **COMPort.Read()**.

Закрытие порта осуществляется с помощью функции **ComPort.Close()**.

### 1.4.3. Передача и прием данных

Для передачи/приема данных нет специальных функций - выполняются по логике программы с помощью операций записи/чтения из буферов порта.

Функция для передачи данных **- ComPort.Write (byte[] InfBytes, int offset, int count)**;

Функция приема данных – **byte** **ComPort.Read();**

# КАНАЛЬНЫЙ УРОВЕНЬ.

## Защита передаваемой информации.

При передаче данных по линиям, входящим в коммутируемую сеть, чаще всего возникают ошибки, обусловленные электрическими помехами. Эти помехи в свою очередь могут вызвать ошибки в цепочке или пакете последовательных битов.

Для обнаружения ошибок применяют разнообразные корректирующие коды. Например: линейный код, код Хемминга, циклический код, логический код 4B/5B.

Для практической реализации циклических кодов могут быть использованы схемы с регистрами сдвига и обратными связями, или таблицы.

В рамках данной курсовой работы необходима реализация алгоритма кодирования Хэмминга кодом.

Построение кодов Хэмминга основано на принципе проверки на четность числа единичных символов: к последовательности добавляется такой элемент, чтобы число единичных символов в получившейся последовательности было четным.

 ~r_1 = i_1 \oplus  i_2 \oplus  ... \oplus  i_k., знак ~ \oplus  здесь означает [сложение по модулю 2](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BF%D0%BE_%D0%BC%D0%BE%D0%B4%D1%83%D0%BB%D1%8E_2).

~ S = i_1 \oplus  i_2 \oplus  ... \oplus  i_n \oplus  r_1 

~ S = 0  — ошибки нет, ~ s = 1  однократная ошибка.

Такой код называется ~ (k+1,k)  или ~ (n,n-1) . Первое число — количество элементов последовательности, второе — количество информационных символов.

Для каждого числа проверочных символов ~ r = 3,4,5..  существует классический код Хэмминга с маркировкой ~(n,k)=(2^r-1,2^r-1-r) то есть — ~(7,4),(15,11),(31,26) . При иных значениях k получается так называемый усеченный код, например международный телеграфный код МТК-2, у которого ~ k = 5 . Для него необходим код Хэмминга ~ (9,5) , который является усеченным от классического ~ (15,11) .

## Передача данных.

В программе можно назначить станцию, инициирующую передачу. Компьютер – передатчик после выбора пользователем файла для передачи передаёт компьютеру-приёмнику информационные кадры, содержащие имя файла, информационные байты файла. Компьютер – приёмник принимает эти кадры и компьютеру – передатчику отправляет положительную квитанцию на каждый из этих кадров. В случае ошибки передачи компьютер – приёмник передаёт компьютеру – передатчику отрицательную квитанцию и передатчик повторяет передачу.

В случае прихода положительной квитанций на получение заголовка файла и согласия на прием, компьютер-передатчик начинает поблочную передачу файлов, дожидаясь прихода положительной квитанции после отсылки каждого блока. В случае ошибки передачи компьютер – приёмник передаёт компьютеру – передатчику отрицательную квитанцию и передатчик повторяет передачу.

После успешной передачи файлов программа возвращается в исходное состояние.

## Функции кодирования/декодирования.

Кодирование и декодирование данных в программе осуществляется циклическим кодом с помощью функций:

Кодирование - **static void Code(ref byte[])**

Декодирование - **static void Decode(ref byte[])**

## Форматы кадров.

Типы кадров:

I – информационный кадр – содержит полезную информацию;

A – квитанция об успешной/неуспешной передаче;

H – содержит информацию об имени файла;

Y – пользователь согласен и готов принимать файл;

N – пользователь отклонил прием файла;

F – конец передачи.

**Информационный кадр.**

Структура:

[стартовый байт][тип кадра][длина информационной части (в байтах)][индекс][максимальное количество индексов][информационные байты][стоповый байт]

Служит для передачи файла на компьютер-клиент.

**Служебные кадры.**

Структура ACK-кадра:

[стартовый байт][тип кадра][индекс][стоповый байт]

ACK-кадр служит для корректного поддержания передачи файла.

Структура HEAD-кадра:

[стартовый байт][тип кадра][имя файла][стоповый байт]

Является стартовым кадром, содержит информацию о передаваемом файле.

Структура FIN-кадра:

[стартовый байт][тип кадра][этап финализации][стоповый байт]

Является стоповым кадром – сигнализирует об успешном окончании передачи данных.

Структура Y/N-кадров:

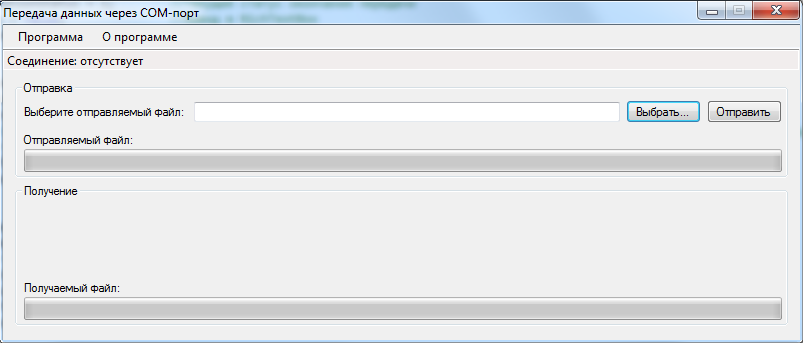
[стартовый байт][тип кадра][стоповый байт]

Сигнализируют о принятии или отклонении файла.

1. ПОЛЬЗОВАТЕЛЬСКИЙ УРОВЕНЬ

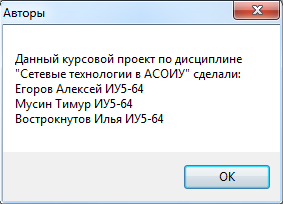
## 3.1. Формы

Главная форма:

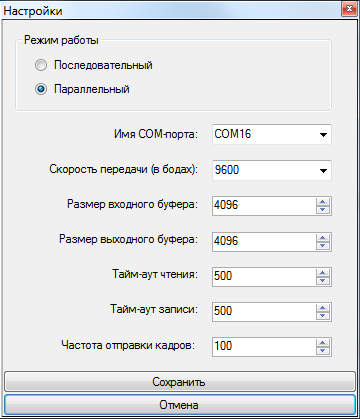


На главной форме присутствуют следующие элементы:

1. Надписи
   1. PathSendLabel – “Выберите отправляемый файл:”
   2. ConnectionStatusTSSL - Надпись состояния подключения («Соединение: Отсутствует», «Соединение: Активно», «Соединение: прервано», «Соединение: превышено ожидание»)
   3. SendNameLabel – Надпись, которая показывает путь к отправляемому файлу
   4. GetLabel – Надпись, которая показывает путь к принимаемому файлу
   5. GetMessageLabel - Надпись, которая показывает имя принимаемого файла до принятия решения о получении
2. Кнопки
   1. ChooseFileButton – «Выбрать…» - для выбора файла
   2. SendFileButton - «Отправить» - для отправки файла
   3. ApplyButton – «Подтвердить» - для принятия файла
   4. DeclineButton - «Отклонить» - для отклонения файла
3. Текстовое поле – SendPathTextBox – для записи пути к файлу
4. Шкалы прогресса
   1. GetProgressBar – Шкала прогресса, показывающая, сколько осталось до полного принятия файла
   2. SendProgressBar - Шкала прогресса, показывающая, сколько осталось до полной передачи файла
5. Меню
   1. “Программа”
      1. Выбрать файл – открывает диалог для выбора отправляемого файла
      2. Свойства соединения – открывает окно настроек COM-порта
      3. Выход – закрывает окно программы
   2. “О программе”
      1. Об авторах – выдает диалоговое сообщение с информацией о разработчиках программы



Форма настроек COM-порта:



На данной форме присутствуют следующие элементы:

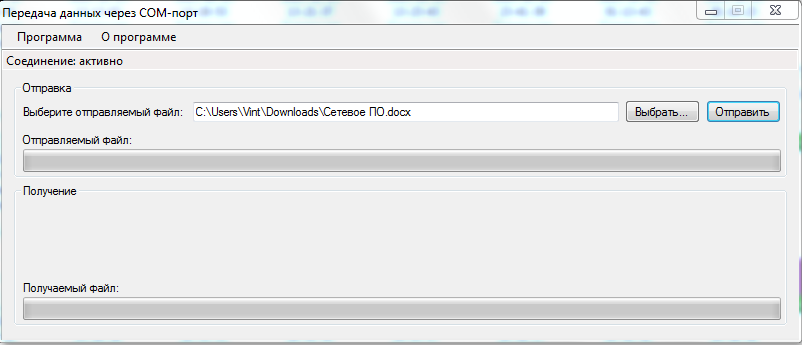
1. Надписи – не несут функциональной нагрузки, поэтому перечислению не подлежат.
2. Комбинированные списки
   1. ComPortComboBox - Имя COM-порта
   2. BaudRateComboBox - Скорость передачи (в бодах)
3. Регуляторы целочисленных значений
   1. InBufferNumericUpDown - Размер входного буфера
   2. OutBufferNumericUpDown - Размер выходного буфера
   3. ReadNumericUpDown - Тайм-аут чтения
   4. WriteNumericUpDown - Тайм-аут записи
   5. FrequencyNUD - Частота отправки кадров
4. Переключатель
   1. SequentialRB – Переключение в последовательный режим передачи
   2. ParallelRB – Переключение в параллельный режим передачи
5. Кнопки
   1. OKButton – Сохранить изменения
   2. CancelButton – Отменить изменения

## 3.2. Работа форм по порядку

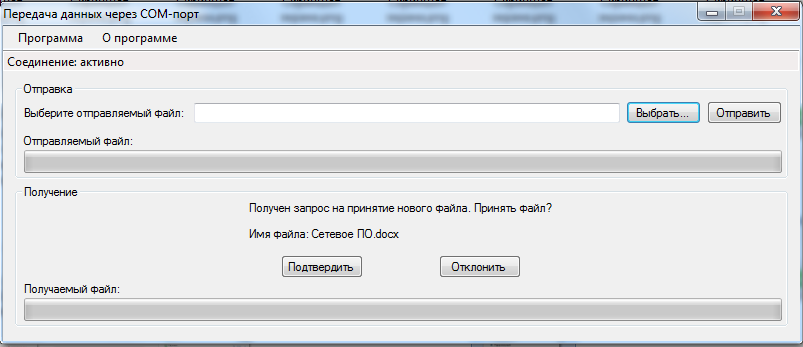
При первом запуске программа просит настроить COM-порт для дальнейшей работы программы. Первой формой, которую увидит пользователь в данном случае – это форма настройки программы.

В случае, если программе не удается подключиться к запрашиваемому COM-порту, то появляется предложение о перенастройке программы. В данном случае вызывается форма настроек.

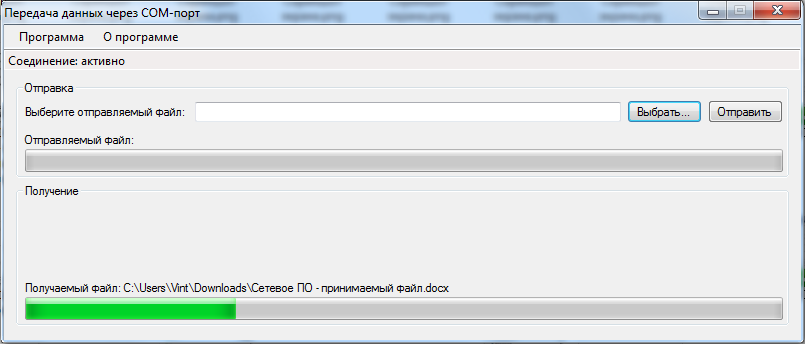
Далее, возникает основная форма. После выбора файла нажимаем кнопку “Отправить”. Ждем ответа клиента.



На машине-клиенте появляются надпись, в которой можно увидеть имя файла, а также две кнопки “Подтвердить” и “Отклонить”.



Затем, в зависимости от выбора пользователя, передача либо начинается, либо не начнется. На скриншоте далее приведен пример окна клиента при односторонней передаче файла в параллельном режиме.



Далее, при двухсторонней передаче, мы выполняем аналогичные шаги на компьютере-клиенте. Примеры окон первого компьютера (компьютера сервера в первой передаче) при двухсторонней передаче файла в параллельном режиме.

