ТЕХНОЛОГИИ ФИЗИЧЕСКОГО УРОВНЯ ПЕРЕДАЧИ ДАННЫХ
Занятие №19
Общие принципы построения сетей

1. Связь компьютера с периферийными устройствами
2. Простейший случай взаимодействия двух компьютеров
3. Сетевые службы и приложения
4. Вопросы

Связь компьютера с периферийными устройствами

Для организации связи между компьютером и периферийным устройством (ПУ) в обоих этих устройствах предусмотрены внешние физические интерфейсы.

Физический интерфейс (называемый также **портом**) - определяется набором электрических связей и характеристиками сигналов.

Обычно он представляет собой разъем с набором контактов, каждый из которых имеет определенное назначение, например, это может быть группа контактов для передачи данных, контакт синхронизации данных и т. п. Пара разъемов соединяется кабелем, который состоит из набора проводов, каждый из которых соединяет соответствующие контакты (рис. 1).

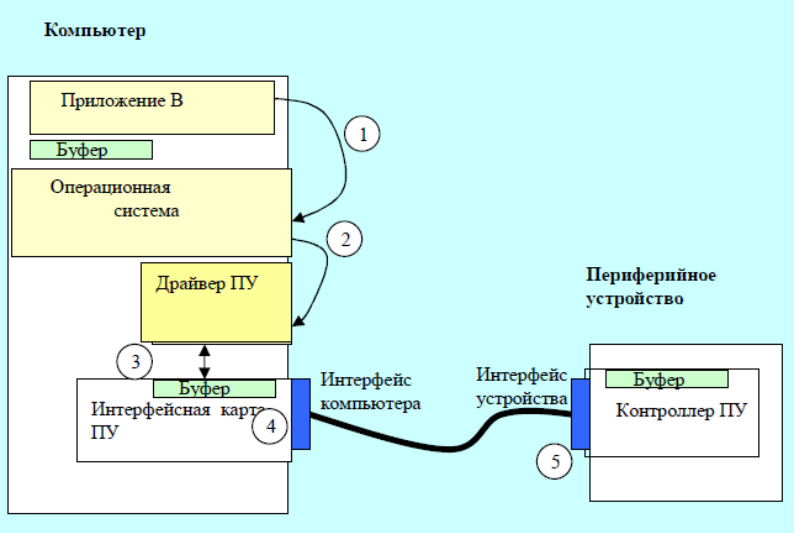

Рис. 1. Взаимодействие компьютера с периферийным устройством.

1- запрос приложения к ОС, 2 – вызов драйвера, 3 – загрузка в буфер интерфейсной карты команды или данных, 4 – побитная передача информации в линию связи, 5 - прием битов и размещение их в буфере.

Логический интерфейс — это набор информационных сообщений определенного формата, которыми обмениваются два устройства или две программы, а также набор правил, определяющих логику обмена этими сообщениями.

Примерами стандартных интерфейсов, используемых в компьютерах, являются параллельный (передающий данные байтами) интерфейс Centronics, предназначенный, как правило, для подключения принтеров, и последовательный интерфейс (передающий данные битами) RS-232C (известный также как COM-порт), который имеет более универсальное назначение — он поддерживается не только принтерами, но и графопостроителями, манипуляторами типа «мышь» и многими другими устройствами. Существуют также специализированные интерфейсы, которые предназначены для подключения уникальных периферийных устройств, например сложной физической экспериментальной установки. В настоящее время широкое распространение нашел интерфейс **USB** (Universal Serial Bus - универсальная последовательная шина).

В ПУ интерфейс чаще всего полностью реализуется аппаратным устройством - **контроллером**, хотя встречаются и программно-управляемые контроллеры для управления современными принтерами, обладающими более сложной логикой.

Периферийные устройства могут принимать от компьютера как данные, например байты информации, которую нужно распечатать на бумаге, так и команды управления, в ответ на которые контроллер ПУ может выполнять специальные действии (перевести головку диска на требуемую дорожку, вытолкнуть лист бумаги из принтера и т. д.). Контроллер принтера, например, поддерживает некоторый набор достаточно простых команд, таких как «Печать символа», «Перевод строки», «Возврат каретки» и т. п., которые он получает от компьютера по интерфейсу и отрабатывает, управляя электромеханическими частями принтера.

Итак, рассмотрим порядок действий, в результате которых приложение распечатывает данные на принтере.

1. Приложение обращается с запросом на выполнение операции ввода-вывода к операционной системе. В запросе указываются адрес данных в оперативной памяти, идентифицирующая информация о периферийном устройстве и операция, которую надо выполнить.

2. Получив запрос, операционная система запускает драйвер принтера. Дальнейшие действия по выполнению операции ввода-вывода со стороны компьютера реализуются интерфейсной картой, работающей под управлением драйвера.

3. Драйвер принтера оперирует командами, понятными контроллеру принтера, то есть командами «Печать символа», «Перевод строки», «Возврат каретки». Драйвер в определенной последовательности помещает коды этих команд в регистр интерфейсной карты, которая побайтно передает их по линиям связи контроллеру периферийного устройства. Для одного и тот же контроллера можно разработать различные драйверы, которые с помощью одного набора команд будут реализовывать разные алгоритмы управления ПУ.

4. Интерфейсная карта выполняет низкоуровневую работу, она не вдастся в смысл данных и команд, передаваемых ей драйвером, считая их однородным потоком битов. После получения от драйвера очередного байта интерфейсная карта просто последовательно передает биты в линию связи, представляя каждый бит электрическим сигналом.

Чтобы контроллеру ПУ стало понятно, что начинается передача байта, перед передачей первого бита интерфейсная карта формирует **стартовый сигнал** специфической формы, а после передачи последнего информационного бита - **стоповый сигнал**. Эти сигналы *синхронизируют* передачу байта. Контроллер, опознав стартовый бит, начинает принимать информационные биты, формируя из них байт и своем приемном буфере.

Помимо информационных битов карта может передавать бит контроля четности для определения достоверности обмена. При корректно выполненной передаче в регистре контроллера устанавливается соответствующий признак.

5. Получив очередной байт, контроллер интерпретирует его и запускает заданную операцию принтера. Закончив работу по печати всех символов документа, контроллер принтера сообщает об этом драйверу компьютера. Драйвер передает операционной системе сообщение о выполнении запроса, а та, в свою очередь, сигнализирует об этом событии приложению.

Простейший случай взаимодействия двух компьютеров

Вернемся к исходному вопросу: как пользователю, работающему с некоторым приложением на компьютере А. распечатать текст на принтере компьютера В (рис. 2).

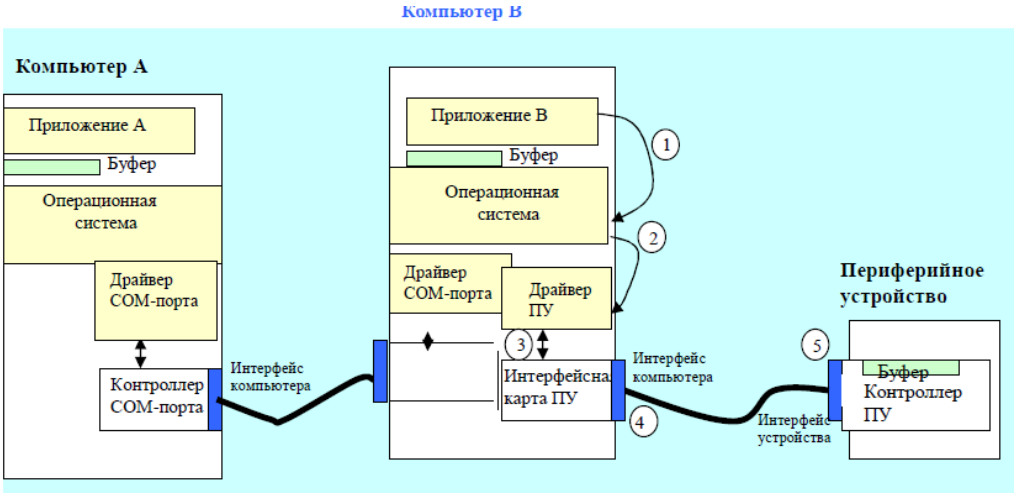

Рис. 2. Совместное использование периферийного устройства.

Приложение А не может получить непосредственный доступ к ресурсам компьютера В — его дискам, файлам, принтеру. Оно может только «попросить» об этом другую программу, выполняемую на том компьютере, которому принадлежат эти ресурсы. Эти «просьбы» выражаются в виде сообщений, передаваемых по каналам связи между компьютерами.

Сообщения могут содержать как команды на выполнение некоторых действий («открыть файл»), так и собственно информационные данные (содержимое некоторого файла).

Сообщение – блок данных стандартного формата. Формат сообщения: заголовок, данные.

Механизмы взаимодействия компьютеров в сети многое позаимствовали у схемы взаимодействия компьютера с периферийными устройствами. В самом простом случае связь компьютеров может быть реализована с помощью тех же самых средств, которые используются для связи компьютера с периферией. Пусть для определенности связь между компьютерами будет осуществляться через последовательный интерфейс — COM-порт. С каждой стороны контроллер COM-порт работает под управлением драйвера COM-порта. Вместе они обеспечивают передачу по кабелю между компьютерами одного байта информации.

В «настоящих» локальных сетях подобные функции передачи данных в линию связи выполняются сетевыми интерфейсными картами (Network Interface Card, NIC), называемыми также сетевыми адаптерами, и их драйверами.

Итак, механизм обмена байтами между двумя компьютерами определен. Однако этого еще недостаточно для решения поставленной задачи — распечатки текста на «чужом» принтере. В частности, необходимо, чтобы компьютер В «понял», какую операцию он должен выполнить с передаваемыми данными, на каком из имеющихся в его распоряжении устройств, в каком виде должен быть распечатан текст и т. п. Обо всем этом должны договориться приложения А и В путем обмена сообщениями.

Чтобы приложения могли «понимать» получаемую друг от друга информацию, программисты, разрабатывавшие приложения А и В, должны *строго оговорить* форматы сообщений, которыми будут обмениваться приложения, и их семантику. Например, они могут договориться о том, что любое выполнение удаленной операции печати начинается с передачи сообщения, запрашивающего информацию о готовности приложения В; что в следующем сообщении идут идентификаторы компьютера и пользователя, сделавшего запрос; что признаком срочного завершения печати является определенная кодовая комбинация и т. п. **Тем самым определяется протокол взаимодействия приложений.**

Протокол — это набор информационных сообщений определенного формата, которыми обмениваются два устройства или две программы, а также набор правил, определяющих логику обмена этими сообщениями.

Рассмотрим взаимодействие всех элементов этой небольшой сети, которые позволят приложению на компьютере А распечатать текст на принтере компьютера В

1. Приложение А формирует сообщение-запрос для приложения В на печать текста и помещает его в свой буфер. Чтобы передать данный запрос компьютеру В, приложение А обращается к локальной ОС, которая запускает драйвер COM-порта компьютера и сообщает ему адрес буфера, где хранится запрос. Затем, по ранее описанной схеме, драйвер и контроллер COM-порта компьютера А, взаимодействуя с драйвером и контроллером COM-порта компьютера В, передают сообщение байт за байтом в компьютер В.

2. Драйвер COM-порта компьютера В постоянно находится в режиме ожидания прихода информации из внешнего мира. В некоторых случаях драйвер вызывается асинхронно, по прерываниям от контроллера. Получив очередной байт и убедившись в его корректности, драйвер помещает его в буфер приложения В.

3. Приложение В принимает сообщение, интерпретирует его и формирует запрос к локальной ОС на выполнение тех или иных действии с принтером. В ходе печати могут возникнуть ситуации, о которых необходимо сообщить приложению А. В этом случае используется симметричная схема: теперь запрос на передачу сообщения поступает от приложения В к локальной ОС компьютера В. Драйверы и контроллеры COM-портов обоих компьютеров организуют побайтную передачу сообщения, которое затем помещается в буфер приложения А.

Потребность в доступе к удаленным файлам может возникать у пользователей многих других приложений: текстового редактора, графического редактора, системы управления базой данных (СУБД). Очевидно, нерационально включать рассмотренные универсальные функции по организации ввода-вывода в состав каждого приложения. Более эффективно решают задачу пара специализированных программных модулей.

Клиент – системный программный модуль, предназначенный для формирования сообщений-запросов к удаленной машине от разных приложений, а затем приема результатов и передачи их соответствующим приложениям.

Сервер – системный программный модуль, который постоянно ожидает прихода по сети запросов от клиентов и, приняв запрос, пытается его выполнить, возможно, с участием локальной ОС. Один сервер может выполнять запросы сразу нескольких клиентов (последовательно или одновременно)

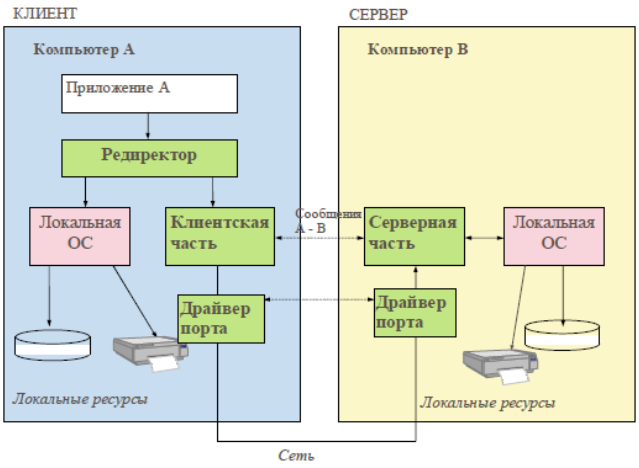

Рис. 3. Взаимодействие программных компонентов при связи двух компьютеров

Очень удобной и полезной функцией клиентской программы является способность отличить запрос к **удаленному** ресурсу от запроса к **локальному** ресурсу. Если клиентская программа умеет это делать, то приложения не должны заботиться о том, с каким принтером они работают (локальным или удаленным), клиентская программа сама распознает и **перенаправляет** (redirect) запрос к удаленной машине. Отсюда и название, часто используемое для клиентского модуля, — **редиректор**.

Клиент и сервер выполняют системные функции по обслуживанию запросов всех приложений компьютера А на удаленный доступ к ресурсу (принтеру, файлам, факсу) компьютера В. Чтобы приложения компьютера В могли пользоваться ресурсами компьютера А описанную схему нужно симметрично дополнить клиентом для компьютера В и сервером для компьютера А.

Схема взаимодействия клиента и сервера с приложениями и локальной операционной системой приведена на **рис. 3.**

Взаимодействие между компьютерами сети происходит за счет передачи сообщений через сетевые адаптеры и каналы связи. С помощью этих сообщений один компьютер обычно запрашивает доступ к локальным ресурсам другого компьютера. Такими ресурсами могут быть как данные, хранящиеся на диске, так и разнообразные периферийные устройства – принтеры, плоттеры, факс-аппараты и т.д.

Разделение локальных ресурсов каждого компьютера между всеми пользователями сети – основная цель создания вычислительной сети.

Несмотря на то, что мы рассмотрели очень простую схему связи только двух компьютеров, функции программ, обеспечивающих удаленный доступ к принтеру, во многом совпадают с функциями сетевой операционной системы, работающей в сети с более сложными аппаратными связями компьютеров.

К основным функциональным компонентам сетевой ОС относятся **средства управления локальными ресурсами** и **сетевые средства.**

Сетевые средства можно разделить на три компоненты:

1. Средства предоставления локальных ресурсов и услуг в общее пользование – серверная часть ОС.
2. Средства запроса доступа к удаленным ресурсам – клиентская часть ОС.
3. Транспортные средства ОС, которые с коммуникационной системой обеспечивают передачу сообщений между компьютерами сети.

В связи с этим, локальные сети можно подразделить в зависимости от используемой сетевой операционной системы на:

- **Серверные** сети
- **Одноранговые** сети
- **Комбинированные** сети.

Термины «клиент» и «сервер» используются для обозначения не только программных модулей, но и компьютеров, подключенных к сети. Если компьютер преимущественно предоставляет свои ресурсы другим компьютерам сети, то он называется **сервером**, а если он их потребляет — **клиентом**. Иногда один и тот же компьютер может одновременно играть роли и сервера, и клиента.

Сетевые службы и приложения

Предоставление пользователям совместного доступа к определенному типу ресурсов, например к файлам, называют также предоставлением сервиса.

Обычно сетевая операционная система поддерживает несколько видов сетевых сервисов для своих пользователей — файловый сервис, сервис печати, сервис электронной почты, сервис удаленного доступа и т. п. Программы, реализующие сетевые сервисы, относятся к классу распределенных программ.

Распределенная программа – это программа, которая состоит из нескольких взаимодействующих частей (в приведенном на **рис. 4** примере – из двух), причем каждая часть, как правило, выполняется на отдельном компьютере сети.

Рис. 4. Взаимодействие частей распределенного приложения

Однако в сети могут выполняться и распределенные пользовательские приложения.

Распределенное приложение также состоит из нескольких частей, каждая из которых выполняет какую-то определенную законченную работу по решению прикладной задачи. Например, одна часть приложения, выполняющаяся на компьютере пользователя, может поддерживать специализированный графический интерфейс, вторая — работать на мощном выделенном компьютере и заниматься статистической обработкой введенных пользователем данных, третья — заносить полученные результаты в базу данных на компьютере с установленной стандартной СУБД. Распределенные приложения в полной мере используют потенциальные возможности распределенной обработки, предоставляемые вычислительной сетью, и поэтому часто называются **сетевыми приложениями.**

Не всякое приложение, выполняемое в сети, является распределенным. Значительная часть истории локальных сетей связана как раз с использованием таких нераспределенных приложений. Рассмотрим, например, как происходила работа пользователя с известной в свое время СУБД dBase. Файлы базы данных, с которыми работали все пользователи сети, располагались на файловом сервере. Сама же СУБД хранилась на каждом клиентском компьютере в пиле единого программного модуля. Программа dBase была рассчитана только на обработку данных, расположенных на том же компьютере, что и сама программа. Пользователь запускал dBase на своем компьютере, и программа искала данные на локальном диске, совершенно не принимая во внимание существование сети. Чтобы обрабатывать с помощью dBase данные, расположенные на удаленном компьютере, пользователь обращался к услугам файловой службы, которая доставляла данные с сервера на клиентский компьютер и создавала для СУБД эффект их локального хранения.

Большинство приложений, используемых в компьютерных сетях в середине 80-х годов, являлись обычными нераспределенными приложениями. И это понятно — они были написаны для автономных компьютеров, а потом просто были перенесены в сетевую среду. Создание же распределенных приложений, хотя и сулило много преимуществ (снижение сетевого трафика, специализация компьютеров), оказалось делом совсем не простым. Нужно было решать множество дополнительных проблем:

- на сколько частей разбить приложение,
- какие функции возложить на каждую часть,
- как организовать взаимодействие этих частей, чтобы в случае сбоев и отказов
- оставшиеся части корректно завершали работу и т. д.

Поэтому до сих пор только небольшая часть приложений являются распределенными, хотя очевидно, что именно за этим классом приложений будущее, так как они в полной мере могут использовать потенциальные возможности сетей, по распараллеливанию вычислений.

Вопросы

1. Какая информация перелается по каналу, связывающему внешние интерфейсы компьютера и периферийного устройства?
2. Какие компоненты включает интерфейс устройства?
3. Какие задачи решает ОС при обмене с периферийным устройством?
4. Какие функции возлагаются на драйвер периферийного устройства?