9.1 Принципы аппаратуры ввода-вывода

9.1.1 Устройства ввода-вывода

Устройства делят на две категории (некоторые не попадают ни в одну):

- блочные устройства - информация считывается и записывается по блокам, блоки имеют свой адрес (диски)
- символьные устройства - информация считывается и записывается посимвольно (принтер, сетевые карты, мыши)

9.1.2 Контроллеры устройств

Устройства ввода-вывода обычно состоят из двух частей:

- механическая (не надо понимать дословно) - диск, принтер, монитор
- электронная - **контроллер** или **адаптер**

Если интерфейс между контроллером и устройством стандартизован (ANSI, IEEE или ISO), то независимые производители могут выпускать совместимые как контроллеры, так и устройства. Например: диски IDE или SCSI.

Операционная система обычно имеет дело не с устройством, а с контроллером. Контроллер, как правило, выполняет простые функции, например, при считывании с диска, преобразует поток бит в блоки, состоящие из байт, и осуществляют контроль и исправление ошибок, проверяется контрольная сумма блока, если она совпадает с указанной в заголовке сектора, то блок считан без ошибок, если нет, то считывается заново.

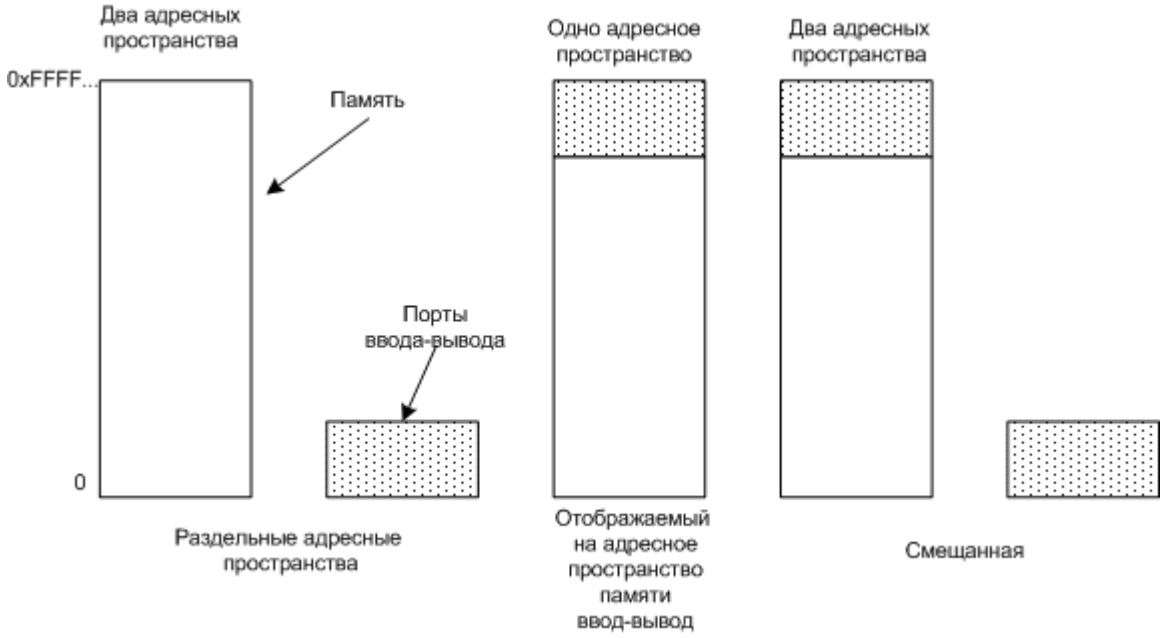
9.1.3 Отображаемый на адресное пространство памяти ввод-вывод

Каждый контроллер имеет несколько регистров, которые используются для взаимодействия с центральным процессором. При помощи этих регистров ОС управляет (считывает, пишет, включает и т.д.) и определяет состояние (готовность) устройства.

У многих устройств есть буфер данных (например: видеопамять).

Реализации доступа к управляющим регистрам и буферам:

- **номер порта ввода-вывода** - назначается каждому управляющему регистру 8- или 16-рзрядное целое число. Адресные пространства ОЗУ и устройства ввода-вывода в этой схеме не пересекаются.
  Недостатки
  - для чтения и записи применяются специальные команды, например: IN и OUT
  - необходим специальный механизм защиты от процессов
  - необходимо сначала считать регистр устройства в регистр процессора
- **отображаемый на адресное пространство памяти ввод-вывод** - регистры отображаются на адресное пространство памяти.
  Недостатки
  - при кэшировании памяти, могут кэшироваться и регистры устройств
  - все устройства должны проверять все обращения к памяти, чтобы определить, на какие им реагировать. На одной общей шине это реализуется легко, но на нескольких будут проблемы.
- **смешанная реализация** - используется в x86 и Pentium,
  от 0 до 64К отводится портам,
  от 640 до 1М зарезервировано под буферы данных.



Способы реализации доступа к управляющим регистрам и буферам

9.1.4 Прямой доступ к памяти (DMA - Direct Memory Access)

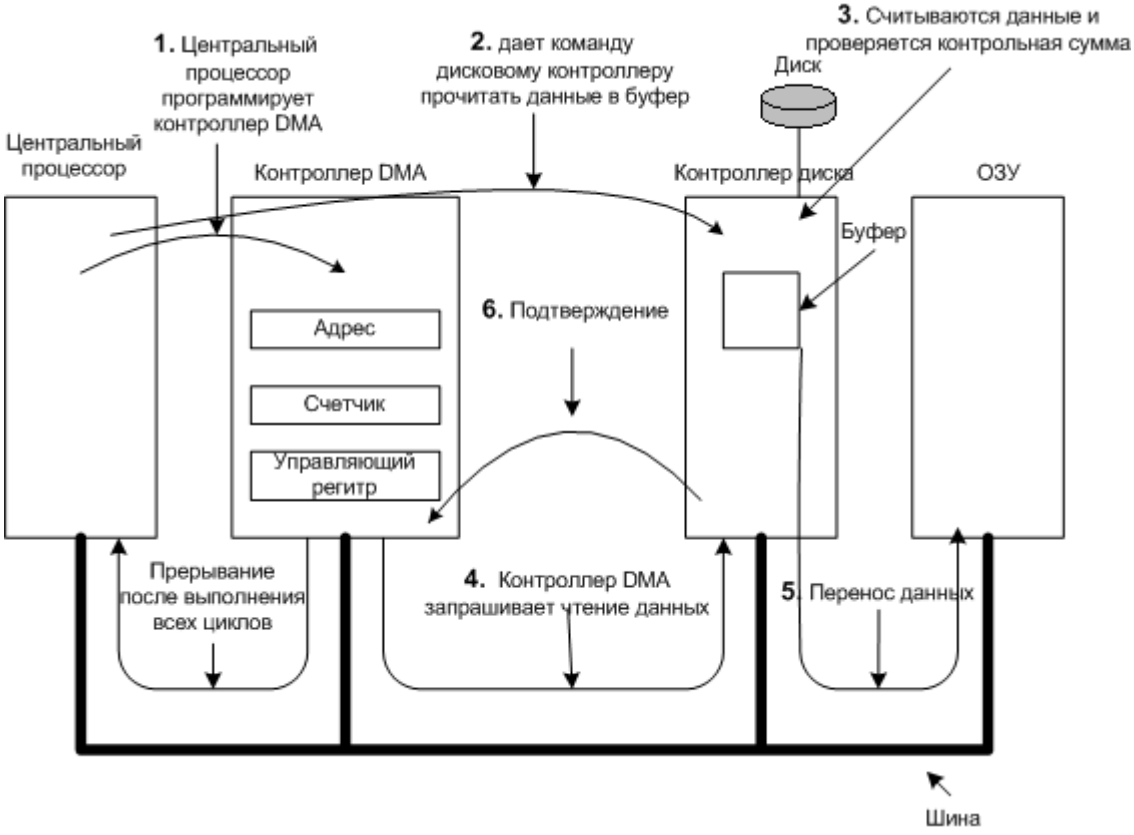
Прямой доступ к памяти реализуется с помощью **DMA - контроллера**.

Контроллер содержит несколько регистров:

- регистр адреса памяти
- счетчик байтов
- управляющие регистры, могут содержать:
  - порт ввода-вывода
  - чтение или запись
  - единицы переноса (побайтно или пословно)

Без контроллера происходит следующее:

1. Процессор дает команду дисковому контроллеру прочитать данные в буфер,
2. Считываются данные в буфер, контроллер проверяет контрольную сумму считанных данных (проверка на ошибки). Процессор, до прерывания, переключается на другие задания.
3. Контроллер диска инициирует прерывание
4. Операционная система начинает работать и может считывать из буфера данные в память



Работа DMA - контроллера

С контроллером происходит следующее:

1. Процессор программирует контроллер (какие данные и куда переместить)
2. Процессор дает команду дисковому контроллеру прочитать данные в буфер

3. Считываются данные в буфер, контроллер диска проверяет контрольную сумму считанных данных, (процессор, до прерывания, переключается на другие задания).
4. Контроллер DMA посылает запрос на чтение дисковому контроллеру
5. Контроллер диска поставляет данные на шину, адрес памяти уже находится на шине, происходит запись данных в память
6. Когда запись закончена, контроллер диска посылает подтверждение DMA контроллеру
7. DMA контроллер увеличивает используемый адрес и уменьшает значение счетчика байтов
8. Все повторяется с пункта 4, пока значение счетчика не станет равной нулю.
9. Контроллер DMA инициирует прерывание

Операционной системе не нужно копировать данные в память, они уже там.

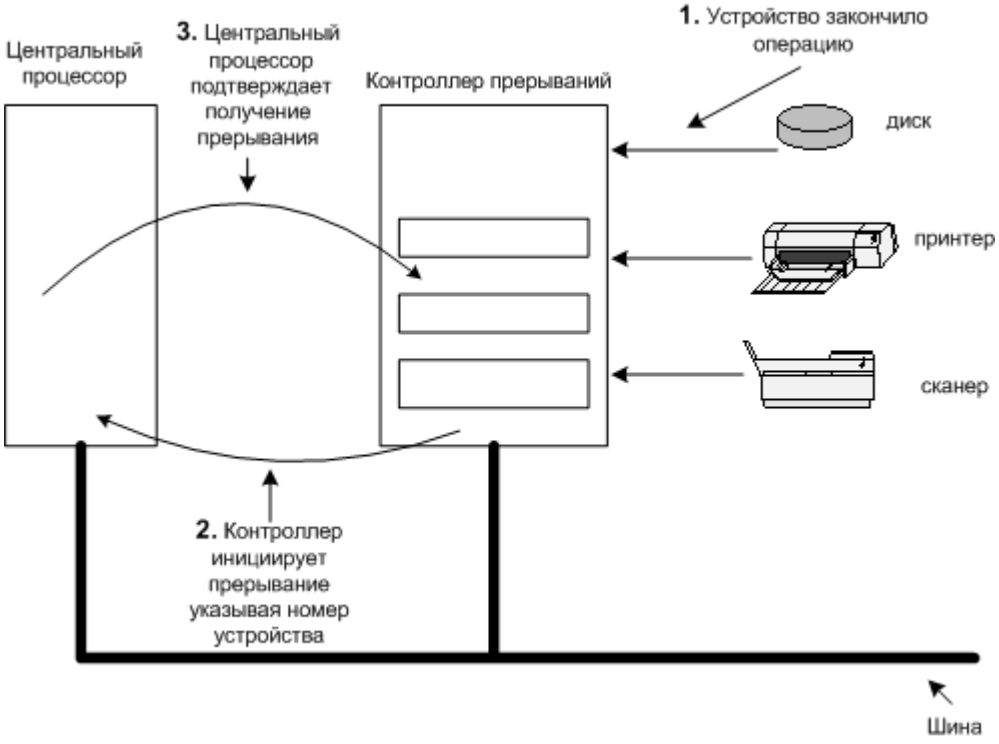
9.1.5 Прерывания

После того как устройство ввода-вывода начало работу, процессор переключается на другие задачи.

Чтобы сигнализировать процессору об окончании работы, устройство инициализирует прерывание, выставляя сигнал на выделенную устройству линию шины (а не выделенный провод).

Контроллер прерываний - обслуживает поступающие прерывания от устройств.

1. Если необработанных прерываний нет, прерывание выполняется немедленно.
2. Если необработанных прерываний есть, контроллер игнорирует прерывание. Но устройство продолжает удерживать сигнал прерывания на шине до тех пор, пока оно не будет обработано.



Работа прерываний

Алгоритм работы:

- Устройство выставляет сигнал прерывания
- Контроллер прерываний инициирует прерывание, указывая номер устройства
- Процессор начинает выполнять обработку прерывания, вызывая процедуру
- Эта процедура подтверждает получение прерывания контроллеру прерываний

9.2 Принципы программного обеспечения ввода-вывода

9.2.1 Задачи программного обеспечения ввода-вывода

Основные задачи, которые должно решать программное обеспечение ввода-вывода:

- Независимость от устройств - например, программа, читающая данные из файла не должна задумываться с чего она читает (CD, HDD и др.). Все проблемы должна решать ОС.
- Единообразное именование - имя файла или устройства не должны отличаться. (В системах UNIX выполняется дословно).
- Обработка ошибок - ошибки могут быть отловлены на уровне контроллера, драйвера и т.д.
- Перенос данных - **синхронный и асинхронный** (в последнем случае процессор запускает перенос данных, и переключается на другие задачи до прерывания).
- Буферизация
- Проблема выделенных (принтер) и невыделенных (диск) устройств - принтер должен предоставляться только одному пользователю, а диск многим. ОС должна решать все возникающие проблемы.

Три основных способа осуществления операций ввода-вывода:

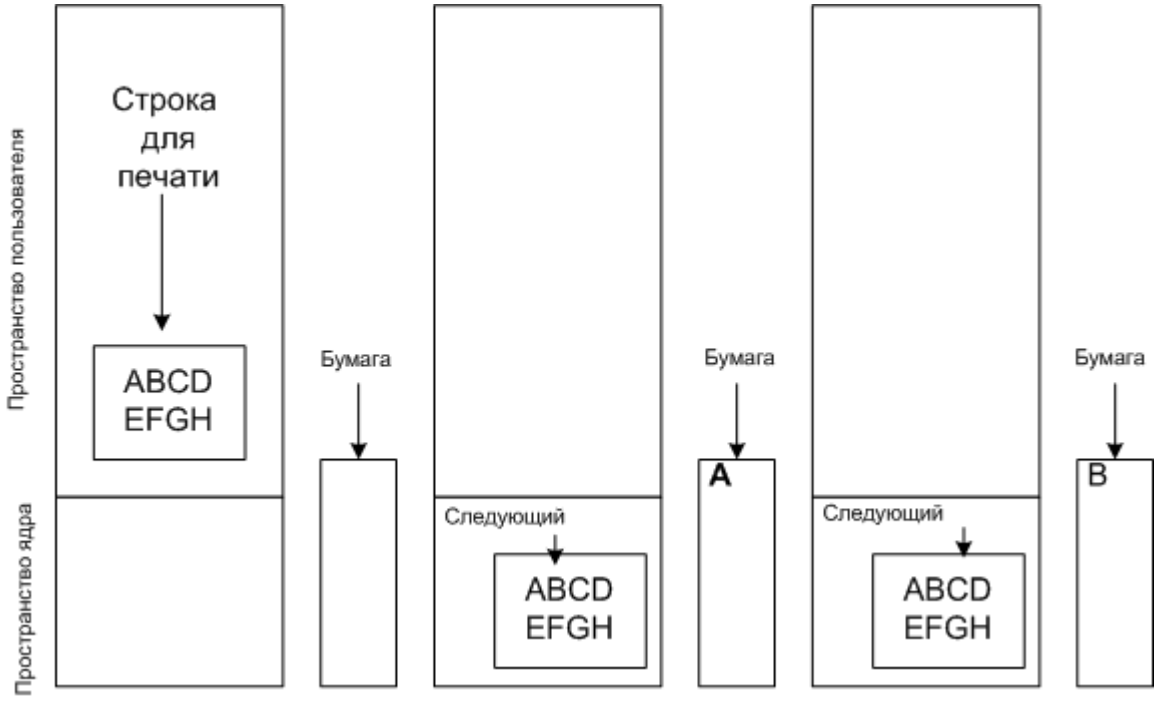
- Программный ввод-вывод
- Управляемый прерываниями ввод-вывод
- Ввод-вывод с использованием DMA

Рассмотрим их подробнее.

9.2.2 Программный ввод-вывод

В этом случае всю работу выполняет центральный процессор.

Рассмотрим процесс печати строки ABCDEFGH этим способом.



Этапы печати строки ABCDEFGH

Алгоритм печати:

1. Строка для печати собирается в пространстве пользователя.
2. Обращаясь к системному вызову, процесс получает принтер.
3. Обращаясь к системному вызову, процесс просит распечатать строку на принтере.
4. Операционная система копирует строку в массив, расположенный в режиме ядра.
5. ОС копирует первый символ в регистр данных принтера, который отображен на памяти.
6. Символ печатается на бумаге.
7. Указатель устанавливается на следующий символ.
8. Процессор ждет, когда бит готовности принтера выставится в готовность.
9. Все повторяется.

При использовании буфера принтера, сначала вся строка копируется в буфер, после этого начинается печать.

9.2.3 Управляемый прерываниями ввод-вывод

Если в предыдущем примере буфер не используется, а принтер печатает 100 символов в секунду, то на каждый символ будет уходить 10мс, в это время процессор будет простаивать, ожидая готовности принтера.

Рассмотрим тот же пример, но с небольшим усовершенствованием.

Алгоритм печати:

1. До пункта 8 тоже самое.
2. Процессор не ждет готовности принтера, а вызывает планировщик и переключается на другую задачу. Печатающий процесс блокируется.
3. Когда принтер будет готов, он посылает прерывание процессору.
4. Процессор переключается на печатающий процесс.

9.2.4 Ввод-вывод с использованием DMA

Недостаток предыдущего метода в том, что прерывание происходит при печати каждого символа.

Алгоритм не отличается, но всю работу на себя берет контроллер DMA.

9.3 Программные уровни и функции ввода-вывода

Четыре уровня ввода-вывода:

Уровни ввода-вывода

9.3.1 Обработчики прерываний

Прерывания должны быть скрыты как можно глубже в недрах операционной системы, чтобы как можно меньшая часть ОС имела с ними дело. Лучше всего блокировать драйвер, начавший ввод-вывод.

Алгоритм:

1. Драйвер начинает операцию ввод-вывод.
2. Драйвер блокирует сам себя,
   - выполнив на семафоре процедуру down
   - выполнив на переменной состояния процедуру wait
   - выполнив на сообщении процедуру receive
3. Происходит прерывание
4. Обработчик прерываний начинает работу
5. Обработчик прерываний может разблокировать драйвер (например, выполнив на семафоре процедуру up)

9.3.2 Драйвера устройств

Драйвер устройства - необходим для каждого устройства. Для разных ОС нужны разные драйверы.

Драйверы должны быть частью ядра (в монолитной системе), что бы получить доступ к регистрам контроллера.

Это одна из основных причин приводящих к краху операционных систем. Потому что драйверы, как правило, пишутся производителями устройств, и вставляются в ОС.

Логическое расположение драйверов устройств. На самом деле обмен данными между контроллерами и драйверами идет по шине.

Драйвера должны взаимодействовать с ОС через стандартные интерфейсы.

Стандартные интерфейсы, которые должны поддерживать драйвера:

- Для блочных устройств
- Для символьных устройств

Раньше для установки ядра приходилось перекомпилировать ядра системы.

Сейчас в основном ОС загружают драйверы. Некоторые драйверы могут быть загружены в горячем режиме.

Функции, которые выполняют драйвера:

- обработка запросов чтения или записи
- инициализация устройства
- управление энергопотреблением устройства
- прогрев устройства (сканера)
- включение устройства или запуска двигателя

9.3.3 Независимое от устройств программное обеспечение ввода-вывода

Функции независимого от устройств программного обеспечения ввода-вывода:

- Единообразный интерфейс для драйверов устройств,
- Буферизация
- Сообщения об ошибках
- Захват и освобождение выделенных устройств (блокирование)
- Размер блока, не зависящий от устройств

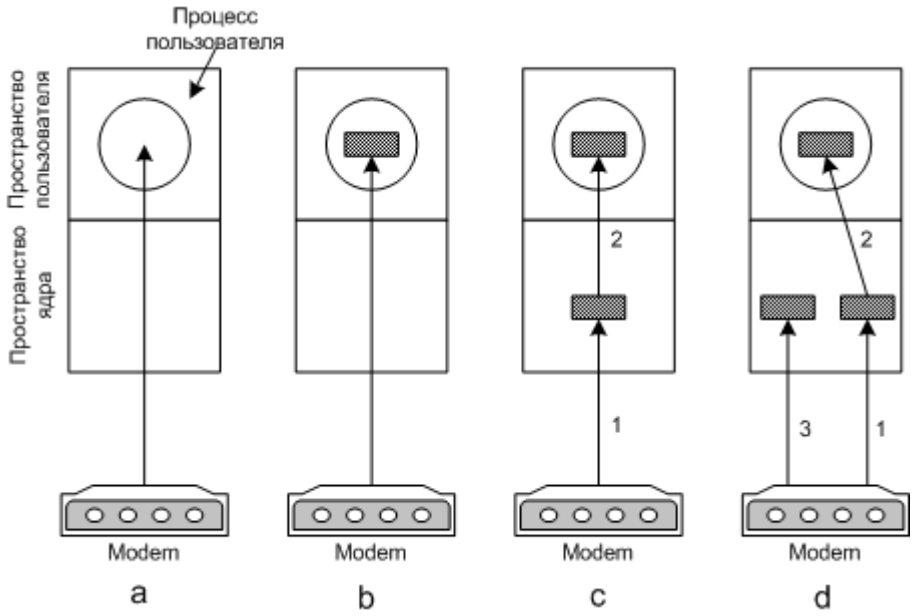
Единообразный интерфейс для драйверов устройств

Кроме интерфейса, в него также входят проблемы,

- именование устройств
- защита устройств

Буферизация

Рассмотрим несколько примеров буферизации.



a) Не буферизованный ввод - после ввода каждого символа происходит прерывание

b) Буферизация в пространстве пользователя - приходится держать загруженными необходимые страницы памяти в физической памяти.

c) Буферизация в ядре с копированием в пространство пользователя - страница загружается только когда буфер ядра полный, данные из буфера ядра в буфер пользователя копируется за одну операцию. Проблема может возникнуть, когда буфер ядра полный, а страница буфера пользователя еще не загружена.

d) Двойная буферизация в ядре - если один буфер заполнен, и пока он выгружается, символы пишутся во второй буфер.

Сообщения об ошибках

Наибольшее число ошибок возникает именно от операции ввода-вывода, поэтому их нужно определять как можно раньше. Ошибки могут быть очень разные в зависимости от устройств.

Захват и освобождение выделенных устройств

Для устройств (принтер) с которыми должен работать в одно время только один процесс, необходима возможность захвата и освобождения устройств. Когда один процесс занял устройство, остальные встают в очередь.

Независимый от устройств размер блока

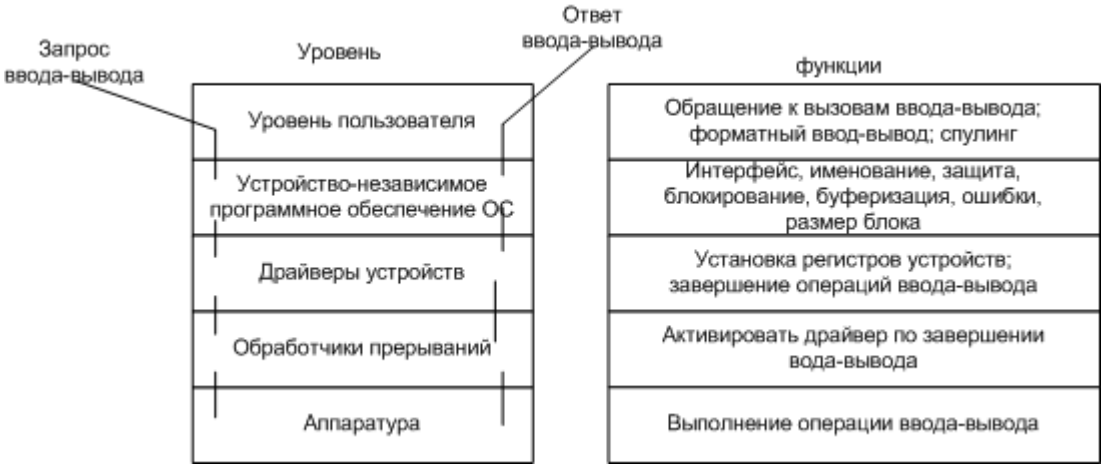
Размер блока должен быть одинаковый для верхних уровней, и не зависеть от устройств (размеров секторов на диске).

9.3.4 Программное обеспечение ввода-вывода пространства пользователя

Функции этого обеспечения:

- Обращение к системным вызовам ввода-вывода (через библиотечные процедуры).
- Форматный ввод-вывод (меняют формат, например, в ASCII)
- Спулинг (для выделенных устройств) - создается процесс (например, демон печати) и каталог спулера.

9.3.5 Обобщение уровней и функций ввода-вывода



Уровни и основные функции системы ввода-вывода