# Определение DMA

DMA, или **прямой доступ к памяти**, представляет собой метод управления обменом данными между основной памятью и контроллерами ввода-вывода без непосредственного участия центрального процессора (ЦП). Этот процесс включает несколько ключевых этапов:

1. **Запрос на передачу данных:**

- Процессор отправляет запрос модулю DMA для передачи блока данных.
- DMA управляет обменом данных, что позволяет процессору выполнять другие задачи во время передачи.

2. **Перенос данных:**

- Модуль DMA осуществляет перенос данных между памятью и контроллером ввода-вывода.
- Прерывание происходит только после завершения передачи всего блока данных, что увеличивает эффективность работы системы.

3. **Управление памятью:**

- DMA управляет адресами и счетчиками, что позволяет ему эффективно взаимодействовать с памятью и контроллерами.

**Преимущества DMA:**
- **Увеличение производительности:** Процессор может выполнять другие операции, пока данные передаются, что снижает время ожидания.
- **Снижение нагрузки на процессор:** Уменьшается количество прерываний, что позволяет процессору сосредоточиться на более важных задачах.

**Применение DMA:**

- Используется в системах, где требуется высокая скорость передачи данных, таких как видеокарты, звуковые карты и другие устройства ввода-вывода.

# Классы защищенности операционных систем

1. **Класс «А»**:

- Операционные системы общего назначения, предназначенные для работы на стандартных вычислительных устройствах, таких как АРМ, серверы, смартфоны и

планшеты.

2. **Класс «Б»**:

- Встраиваемые операционные системы, которые прошиты в специализированные устройства для выполнения заранее определенных задач.

3. **Класс «В»**:

- Операционные системы реального времени, которые обеспечивают реагирование на события в рамках заданных временных ограничений.

# Связывание адресов

Связывание адресов включает три основных типа: **физическое**, **логическое** и **виртуальное**.

### 1. Физическое связывание:

- Это процесс, при котором логические адреса, используемые программой, преобразуются в физические адреса в оперативной памяти.
- Физическое связывание происходит на этапе выполнения, когда процессор и операционная система отображают ссылки в коде программы в реальные физические адреса, соответствующие текущему расположению программы в памяти.

### 2. Логическое связывание:

- Логическое связывание происходит на этапе компиляции, загрузки или выполнения программы.
- На этапе компиляции, если известно точное место размещения процесса в памяти, генерируются физические адреса.
- На этапе загрузки, если информация о размещении программы отсутствует, компилятор создает перемещаемый код.
- На этапе выполнения, если процесс может быть перемещен, связывание откладывается до момента выполнения.

### 3. Виртуальное связывание:

- Виртуальное связывание используется в системах с виртуальной памятью, где логические адреса преобразуются в физические адреса с помощью таблиц страниц.
- Виртуальные адреса делятся на страницы, которые соответствуют единицам в физической памяти, называемым страничными кадрами. Менеджер памяти преобразует виртуальный адрес в упорядоченную пару (p, d), где p – номер страницы в виртуальной памяти, а d – смещение в рамках страницы.

# Средства межпроцессного взаимодействия

Средства межпроцессного взаимодействия можно классифицировать по нескольким категориям, каждая из которых имеет свои особенности и механизмы работы:

### 1. Разделяемая память:

- Позволяет двум или более процессам совместно использовать область адресного пространства.
- Обеспечивает максимальную возможность обмена информацией и влияние на поведение других процессов, но требует осторожности в использовании.
- Реализуется с помощью обычных языков программирования и специальных системных вызовов для сигнальных и канальных средств.

### 2. Канальные средства:

- Передача информации осуществляется через линии связи, предоставленные операционной системой.
- Напоминает общение людей по телефону или с помощью записок, с ограниченной пропускной способностью.

### 3. Сигнальные средства:

- Передают минимальное количество информации (один бит), обычно для извещения процесса о наступлении события.
- Степень воздействия на поведение процесса минимальна, и неправильная реакция может привести к серьезным последствиям.

### 4. Прямая и непрямая адресация:

- **Прямая адресация:** Взаимодействующие процессы напрямую обмениваются данными, указывая имя или номер процесса, с которым происходит обмен.
- **Непрямая адресация:** Данные помещаются в промежуточный объект, откуда они могут быть извлечены другим процессом.

# FIFO как средство межпроцессного взаимодействия

FIFO (First In, First Out) представляет собой механизм межпроцессного взаимодействия, который позволяет процессам обмениваться данными через именованные каналы (pipe). Основные характеристики FIFO включают:

### 1. Доступность для всех процессов:

- Вход и выход FIFO видимы для всех процессов, что позволяет им взаимодействовать через общий канал.
- FIFO зарегистрирован в операционной системе под определенным именем, что упрощает его использование.

**2. Связь между родственными процессами:**

- FIFO может использоваться для связи между процессами, которые имеют общего предка, создавшего данный канал связи.
- Это ограничение делает FIFO подходящим для кооперативных процессов, которые влияют на поведение друг друга.

**3. Преимущества использования FIFO:**

- FIFO позволяет организовать упорядоченный обмен данными, где данные, поступившие первыми, будут обработаны первыми.
- Это упрощает синхронизацию процессов и уменьшает вероятность потери данных.

**4. Логическая организация:**

- FIFO относится к категории средств обмена информацией, где передача данных происходит через линии связи, предоставленные операционной системой, что напоминает общение людей по телефону.

**5. Применение в системах:**

- FIFO может быть использован в различных сценариях, включая обмен данными между процессами, работающими на одной или разных вычислительных системах.

# Определение прерывания

**Прерывание** — это механизм, который позволяет процессору временно приостановить выполнение текущей задачи для обработки более приоритетной задачи. Основные аспекты прерываний включают:

**1. Процесс обработки прерывания:**
- **Приостановка процесса:** Когда происходит прерывание, работа текущего процесса приостанавливается.
- **Сохранение контекста:** Процессор сохраняет состояние выполнения, включая счетчик команд и регистры, в стеке исполняемого процесса.
- **Передача управления:** Управление передается специальному адресу, где находится обработчик прерывания.

**2. Обработка прерывания:**
- **Сохранение контекста:** Операционная система сохраняет динамическую часть системного контекста процесса в его PCB (Process Control Block) и переводит процесс в состояние готовности.
- **Выполнение действий:** Обработчик прерывания выполняет необходимые действия, связанные с возникшим прерыванием.

**3. Виды прерываний:**
- **Аппаратные прерывания:** Возникают из-за событий, происходящих в аппаратном обеспечении, таких как завершение операции ввода-вывода.
- **Программные прерывания:** Инициируются программным обеспечением, например, при выполнении системных вызовов.

**4. Прерывания и DMA:**
- **Прямой доступ к памяти (DMA):** Прерывания также могут быть связаны с модулями DMA, которые управляют обменом данных между основной памятью и контроллерами ввода-вывода, минимизируя нагрузку на процессор.

# Определение NTFS

NTFS (**New Technology File System**) — это файловая система, используемая в операционных системах Windows.
**Структура тома:**

- Основой структуры тома NTFS является **главная таблица файлов (MFT)**, которая содержит запись для каждого файла тома, включая саму себя.
- Каждая запись MFT имеет фиксированную длину, зависящую от объема диска (1, 2 или 4 Кбайт).
- Файлы идентифицируются номером файла, который определяется позицией файла в MFT.

**Физическая организация:**

- Поддержка больших файлов и дисков объемом до **264 байт**.
- Восстанавливаемость после сбоев и отказов программ и аппаратуры.
- Высокая скорость операций, включая работу с большими дисками.
- Низкий уровень фрагментации.

**Безопасность и управление доступом:**

- NTFS использует **Access Control List (ACL)** для управления доступом к файлам и папкам, что позволяет контролировать, какие пользователи и группы имеют доступ к

ресурсам.

**Самовосстановление:**

- Файловая система включает журнал транзакций, который помогает в восстановлении данных после сбоев.

# Определение драйвера

Драйвер — это программный модуль, который выполняет различные функции в ОС.

**Основные характеристики драйвера:**
- Драйвер работает в привилегированном режиме и является частью ядра ОС.
- Он непосредственно управляет внешним устройством, взаимодействуя с его контроллером через команды ввода-вывода компьютера.
- Драйвер обрабатывает прерывания от контроллера устройства

- Он предоставляет прикладному программисту логически удобный интерфейс для работы с устройством, скрывая низкоуровневые детали управления и организации данных устройства.
  - Драйвер взаимодействует с другими модулями ядра ОС через строго оговоренный интерфейс, который описывает формат передаваемых данных, структуру буферов и способы вызова драйвера.

**Функции драйвера:**
1. Обработка запросов на запись и чтение от программного обеспечения управления устройствами.
2. **Проверка параметров:** Проверка входных параметров запросов и обработка ошибок.
3. **Инициализация устройства:** Инициализация устройства и проверка его статуса.
4. **Управление энергопотреблением:** Управление энергопотреблением устройства.
5. **Регистрация событий:** Регистрация событий в устройстве.
6. **Выдача команд:** Выдача команд устройству и ожидание их выполнения, возможно, в блокированном состоянии до поступления прерывания от устройства.
7. **Проверка завершения операций:** Проверка правильности завершения операции.
8. **Передача данных:** Передача запрошенных данных и статуса завершенной операции.
9. **Обработка новых запросов:** Обработка нового запроса при незавершенном предыдущем запросе (для реентерабельных драйверов).

# Определение виртуальной памяти

Виртуальная память - это технология, которая позволяет компьютеру создавать у каждой программы иллюзию большого и непрерывного пространства памяти, даже если

физическая оперативная память (ОЗУ) ограничена. Она также защищает процессы друг от друга, предотвращая ошибки и сбои системы.

## Как это работает простыми словами

1. **Виртуальные адреса вместо физических**
   Каждая программа работает с виртуальными адресами - это как собственная карта памяти, которая не связана напрямую с реальными физическими ячейками ОЗУ. Операционная система (ОС) и специальное аппаратное устройство (блок управления памятью) переводят эти виртуальные адреса в реальные физические адреса в памяти компьютера
2. **Разделение памяти на страницы и фреймы**
   Вся память разбивается на маленькие блоки одинакового размера - виртуальная память на *страницы* (обычно 4 или 8 КБ), а физическая память - на *фреймы* такого же размера. ОС следит за соответствием страниц виртуальной памяти и фреймов физической памяти с помощью таблиц страниц
3. **Использование файла подкачки (swap)**
   Если физической памяти не хватает, ОС временно сохраняет неиспользуемые страницы на жёсткий диск в специальный файл подкачки. Это позволяет программам работать с объёмом памяти, который превышает реальный размер ОЗУ. Когда данные снова нужны, они загружаются обратно в ОЗУ
4. **Изоляция процессов**
   Каждому процессу выделяется своё виртуальное адресное пространство, которое не пересекается с другими. Это защищает программы от случайного или злонамеренного доступа к памяти друг друга и предотвращает сбои
5. **Оптимизация доступа**
   Для ускорения преобразования виртуальных адресов в физические используется кэш специальных таблиц - Translation Lookaside Buffer (TLB). Это помогает быстро находить нужные соответствия и ускоряет работу программ

## Итог

- Виртуальная память даёт программам больше памяти, чем физически есть в компьютере, за счёт использования жёсткого диска.
- Каждая программа работает в своём изолированном пространстве, что повышает безопасность и стабильность.
- ОС автоматически управляет переносом данных между ОЗУ и диском, делая этот процесс незаметным для пользователя и программиста.