

# Шины, периферийные устройства, ввод/вывод и прерывания

Луцив Дмитрий Вадимович

Кафедра системного программирования СПбГУ

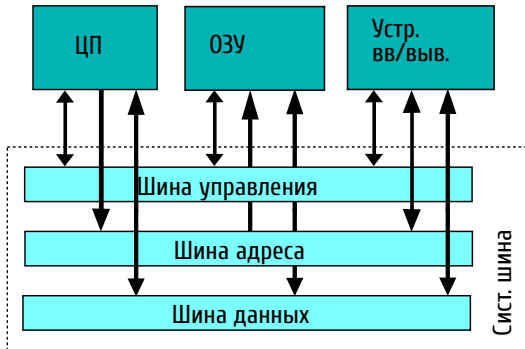


- 1 Шины и периферийные устройства
  - Периферийные устройства и контроллеры
- 2 Прерывания
  - Основы аппаратных прерываний
  - Другие способы использования прерываний
- 3 DMA
  - DMA для высокопроизводительных устройств
  - DMA для устройств реального времени
- 4 Настройка устройств
- 5 Исторически значимые синхронные шины
  - Устаревшие шины ISA и PCI
  - Проблемы

# Шины и периферийные устройства

- Периферийные устройства и контроллеры

# Вспоминаем: архитектура фон Неймана



- **Контроллер** — устройство в составе ЭВМ, обеспечивающее связь системной шины с внешним устройством
  - Например, контроллер жёсткого диска, контроллер порта USB
- **Драйвер устройства** — ПО, предоставляемое производителем устройства или ОС
  - Реализует низкоуровневые операции работы с устройством
  - Позволяет абстрагироваться от модели оборудования. Например: для прикладного ПО и ОС все принтеры «одинаковые», т.к. разные драйвера принтеров реализует один и тот же стандартный программный интерфейс

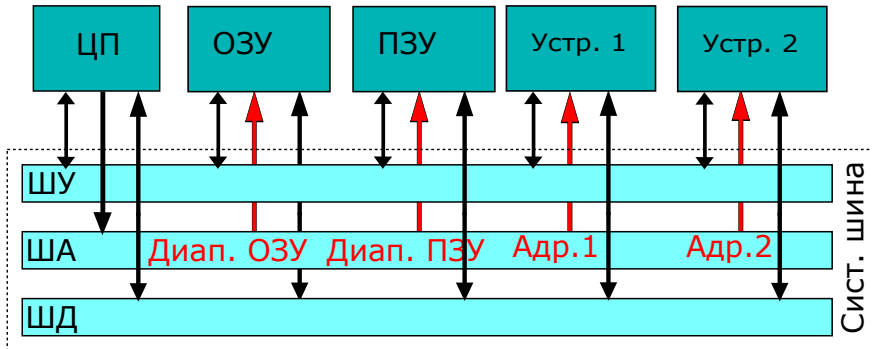
Устройство соединено с системной шиной


- реагирует на сигнал шины управления «работа с устройствами»
- проверяет, совпадает ли его идентификатор со значением на шине адреса
- по команде на шине управления может прочитать данные с шины данных или записать на неё
- может генерировать *аппаратные прерывания* (позже), сообщая об этом через шину управления

**Адрес устройства** — его идентификатор. Не является адресом в смысле адреса данных в ОЗУ, но передаётся по шине адреса

- Соединение устройства с шиной — порт (port)
- Инструкция записи в порт — out адрес, регистр
- Инструкция чтения из порта — in адрес, регистр

**Порт** — аппаратная и программная абстракция: механизм сопряжения контроллера устройства с системной шиной и механизм программного обращения к контроллеру. Обычно говорят «номер порта» или «адрес порта». У устройства может быть и несколько портов.



- В ранних и простых ЭВМ типичным было подключение устройств непосредственно к системной шине. Поэтому «порт» — разъём на корпусе и «порт» — способ доступа к устройству были практически синонимами
  - Пример: ZX Spectrum **Interface 1**  — фактически это был контроллер
- Сейчас разъём на корпусе — обычно способ присоединить устройство к контроллеру в ПК, а в устройстве «говорить» с контроллером в ПК будет ответный контроллер
  - Пример: универсальный USB-контроллер в ПК и USB-клавиатура со своим внутренним контроллером



## Прерывания

- Основы аппаратных прерываний
- Другие способы использования прерываний

## «Системные горячие клавиши»

Ctrl+Alt+Del на PC:

- немедленная перезагрузка (DOS)
- завершение и перезагрузка (Linux)
- вызова системного меню (Windows)

При этом ни ОС, ни, тем более, пользовательская программа, не проверяют всё время: не нажали ли Ctrl+Alt+Del?

## «Системные горячие клавиши»

Ctrl+Alt+Del на PC:

- немедленная перезагрузка (DOS)
- завершение и перезагрузка (Linux)
- вызова системного меню (Windows)

При этом ни ОС, ни, тем более, пользовательская программа, не проверяют всё время: не нажали ли Ctrl+Alt+Del?

- Значит прерывание по Ctrl+Alt+Del на USB-клавиатуре генерируется программно драйвером клавиатуры

## «Системные горячие клавиши»

Ctrl+Alt+Del на PC:

- немедленная перезагрузка (DOS)
- завершение и перезагрузка (Linux)
- вызова системного меню (Windows)

При этом ни ОС, ни, тем более, пользовательская программа, не проверяют всё время: не нажали ли Ctrl+Alt+Del?

- Значит прерывание по Ctrl+Alt+Del на USB-клавиатуре генерируется программно драйвером клавиатуры

## Мышь

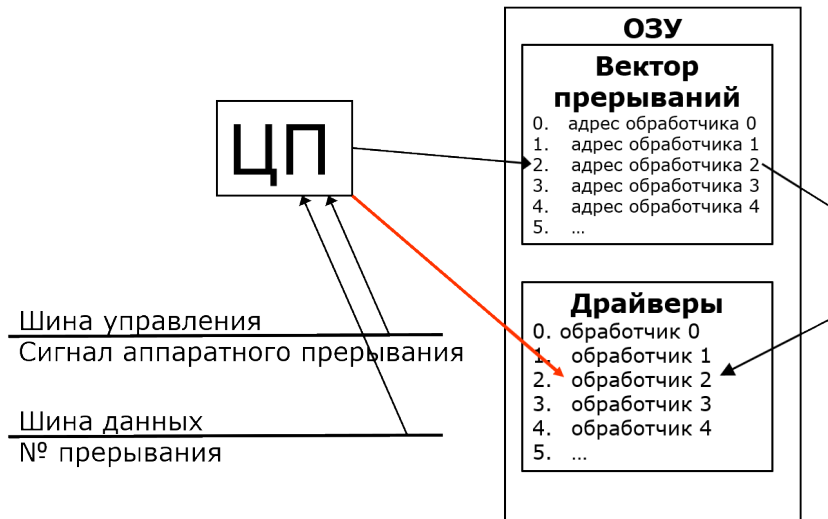
- Ни прикладные программы, ни ОС не опрашивают мышь постоянно
- Тем не менее «что-то» реагирует на движение мыши и отображает на экране движущийся указатель

**Аппаратное прерывание** — сигнал, сообщающий ЦП о возникновении ситуации, требующей немедленной обработки

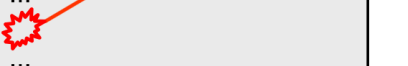
- Обычно этот сигнал генерируют контроллеры внешних устройств
- Обработка ситуации выполняется программно
- Текущая выполняемая программа «не замечает» того, что процессор «отвлёкся», и после обработки ситуации продолжается

- Запускается и работает ОС
  - ОС регистрирует адреса специальных процедур — обработчиков прерываний; обработчик прерывания — часть драйвера устройства
  - Прерываний от разных устройств множество (они имеют номера), адреса обработчиков помещаются в специальный массив в ОЗУ — вектор прерываний
- Запускается и работает прикладное ПО
  - Выполняется программа
  - Получив сигнал от устройства (например, мыши), контроллер посылает сигнал ЦП о необходимости его обработки (проверить, на сколько переместили мышь, какие кнопки нажали)
  - ЦП заканчивает выполнение очередной инструкции, а вместо следующей производит вызов процедуры — обработчика прерывания. Эта процедура — не часть прикладной программы
  - После завершения обработчика прерывания продолжается выполнение прикладной программы

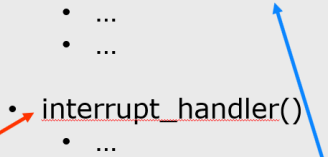
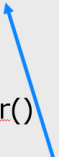
# Вектор прерываний



## Прикладное ПО

- `double calculate(...)`
    - ...
    - ...
    - ...
    - ...
  - `int main()`
    - ...
    - `double r = calculate(...)`
    - ...
    - ...
    - ...
- 

## Драйвер мыши

- `read_mouse_state()`
    - ...
    - ...
  - `interrupt_handler()`
    - ...
    - `read_mouse_state()`
    - ...
- 
- 



# Косвенный вызов API ОС (в недавнем прошлом)

- Прерывание генерируется программно, на PC — машинная команда `int <номер>`
- Вектор прерываний фактически хранит адреса части системных функций
- Это позволяет менять адреса системного кода, не меняя машинного кода пользовательских программ

## Где это использовалось?

- Вызов API BIOS для управления графикой (`int 10h`) и DOS (`int 21h`)
  - Пример

```
mov ah, 0eh      ; function number = 0Eh : Display Character
mov al, '!'      ; AL = code of character to display
int 10h          ; call INT 10h, BIOS video service
```
- Вызов API ядра Windows (`int 2Eh`)
- Вызов API ядра Linux (`int 80h`)

Этот способ удобный, но механизм прерываний небыстрый

- Windows и Linux используют специально разработанные инструкции (`syscall`, `sysenter`) и техники (vDSO). [Подробнее здесь](#) ↗

Вызов драйвера виртуальной памяти, когда страница памяти выгружена или не проинициализирована

Вызов драйвера виртуальной памяти, когда страница памяти выгружена или не проинициализирована  
0 виртуальной памяти позже

## DMA

- DMA для высокопроизводительных устройств
- DMA для устройств реального времени

Внешние устройства могут передавать значительные объёмы информации. Основные способы взаимодействия:

- Чтение и запись в порты. Обмен небольшими порциями данных загружает ЦП
- Выделение контроллеру устройства области памяти и выдача команд, которые выполняются отложено

Внешние устройства могут передавать значительные объёмы информации. Основные способы взаимодействия:

- Чтение и запись в порты. Обмен небольшими порциями данных загружает ЦП
- Выделение контроллеру устройства области памяти и выдача команд, которые выполняются отложено

**DMA (Direct Memory Access)** — механизм прямого обмена данными между оперативной памятью и контроллерами устройств

Механизм поддерживается многими контроллерами устройств, предназначенными для передачи значительных объёмов данных. Для небольших данных (например, работа с системными часами) необходимости его использовать нет.

Для того чтобы сообщить о завершении операции, контроллер генерирует аппаратное прерывание. Обработчик прерывания находится в драйвере соответствующего устройства



А когда Sound Blaster «доиграет» фрагмент звука, он сгенерирует прерывание, и будет ждать, пока ЦП не выдаст ему ещё данных?..

# DMA для потоковых устройств реального времени

А когда Sound Blaster «доиграет» фрагмент звука, он сгенерирует прерывание, и будет ждать, пока ЦП не выдаст ему ещё данных?..



# Настройка устройств

# Как настраивалось оборудование до середины 1990-х?

Классический способ — jumpers, DIP switches

Computer Jumper



ComputerHope.com

Настраивались *системные ресурсы* — адреса портов, характеристики DMA, номера аппаратных прерываний. Пример Sound Blaster для DOS: Port 220, IRQ 7, DMA 1

# Как настраивается оборудование сейчас?

- С середины 1980-х — разные технологии для передачи метаданных по системной шине
- Первая широко внедрённая — Plug-n-Play. Первая широко использующая ОС для PC — Windows 95 (жаргон конца 1990-х — Plug and Pray)

# Как настраивается оборудование сейчас?

- С середины 1980-х — разные технологии для передачи метаданных по системной шине
- Первая широко внедрённая — Plug-n-Play. Первая широко использующая ОС для PC — Windows 95 (жаргон конца 1990-х — Plug and Pray)

Посмотреть, какие ресурсы выделены устройствам в популярных ОС можно:

- В Windows — при помощи Диспетчера устройств
- В Linux — при помощи `lsdev` (собирает информацию из `/proc/interrupts`, `/proc/ioports` и `/proc/dma`)

## Исторически значимые синхронные шины

- Устаревшие шины ISA и PCI
- Проблемы

- «Родная» шина IBM PC: **разъёмы ISA** ↗ позволяли непосредственно подключить устройство к системной шине
- 8, затем 16-разрядная
- В компьютерах с 32-битной шиной данных (процессоры 80386 и новее) для общения с 16- и 8-разрядными устройствами шина переходила в специальный режим
- Поздние версии уже поддерживали Plug-and-Play, хотя и с трудом




- «Родная» шина IBM PC: **разъёмы ISA** ↗ позволяли непосредственно подключить устройство к системной шине
- 8, затем 16-разрядная
- В компьютерах с 32-битной шиной данных (процессоры 80386 и новее) для общения с 16- и 8-разрядными устройствами шина переходила в специальный режим
- Поздние версии уже поддерживали Plug-and-Play, хотя и с трудом
- Изредка используется до сих пор

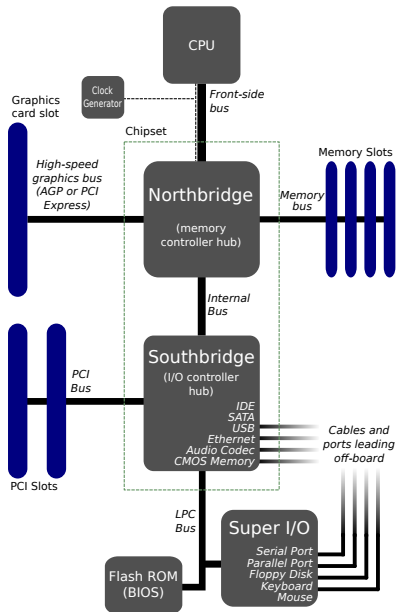
- 32-разрядная ↗ (немногие реализации — 64-разрядные)
- Отделена от процессора:
  - Взаимодействие с устройствами по специальному протоколу
  - Поддерживаются *арбитраж*, связь нескольких шин друг с другом

- 32-разрядная [↗](#) (немногие реализации — 64-разрядные)
- Отделена от процессора:
  - Взаимодействие с устройствами по специальному протоколу
  - Поддерживаются *арбитраж*, связь нескольких шин друг с другом
- Есть вариант разъёма для ноутбуков — Mini PCI [↗](#)

# Северный и южный мосты в машине с шиной PCI

Доосмысление принципа фон Неймана:  
иерархия запоминающих устройств →  
иерархия шин :

- Ближе к процессору — через северный мост, быстро и немного
- Дальше от процессора — через южный мост, медленно, много, и «зоопарк»



# Дорожки разной длины — это серьёзно!

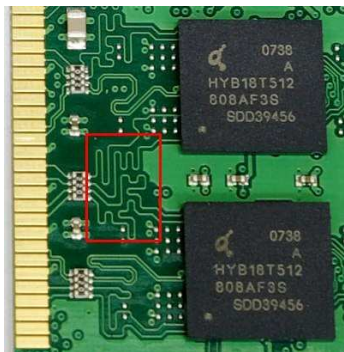
❶ 1 наносекунда — это много или мало? Нам поможет [Грейс Хоппер](#)

# Дорожки разной длины — это серьёзно!

- 1 1 наносекунда — это много или мало? Нам поможет [Грейс Хоппер](#)
- 2 Но длина дорожек на платах сравнима с таким расстоянием. Что же делать?

# Дорожки разной длины — это серьёзно!

- ❶ 1 наносекунда — это много или мало? [Нам поможет](#) [Грейс Хоппер](#)
- ❷ Но длина дорожек на платах сравнима с таким расстоянием. Что же делать?
- ❸ **Выворнять длину!**



## Упражнения

- Выберите несколько внутренних контроллеров своего ПК, выясните, какие системные ресурсы им выделены
- Идентифицируйте внутренние разъёмы расширения системной платы своего ПК

## Вопросы

- Что такое аппаратное прерывание?
- Что такое драйвер, контроллер и порт?
- Что такое вектор и обработчик прерываний?
- Опишите принцип работы механизма DMA
- Каковы особенности DMA для устройств реального времени?
- Что такое северный и южный мосты?
- Что такое Root Complex?



# Вопросы



[EDU.DLUCIV.NAME](https://edu.dluciv.name) 