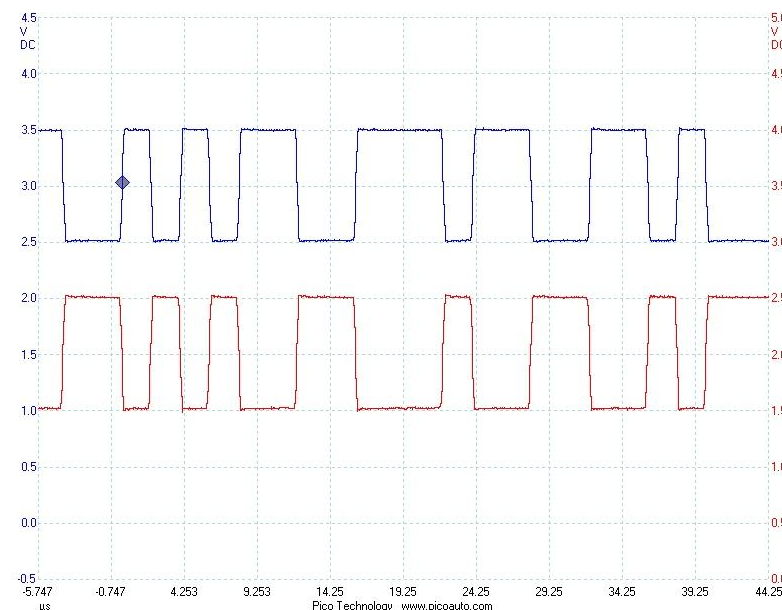
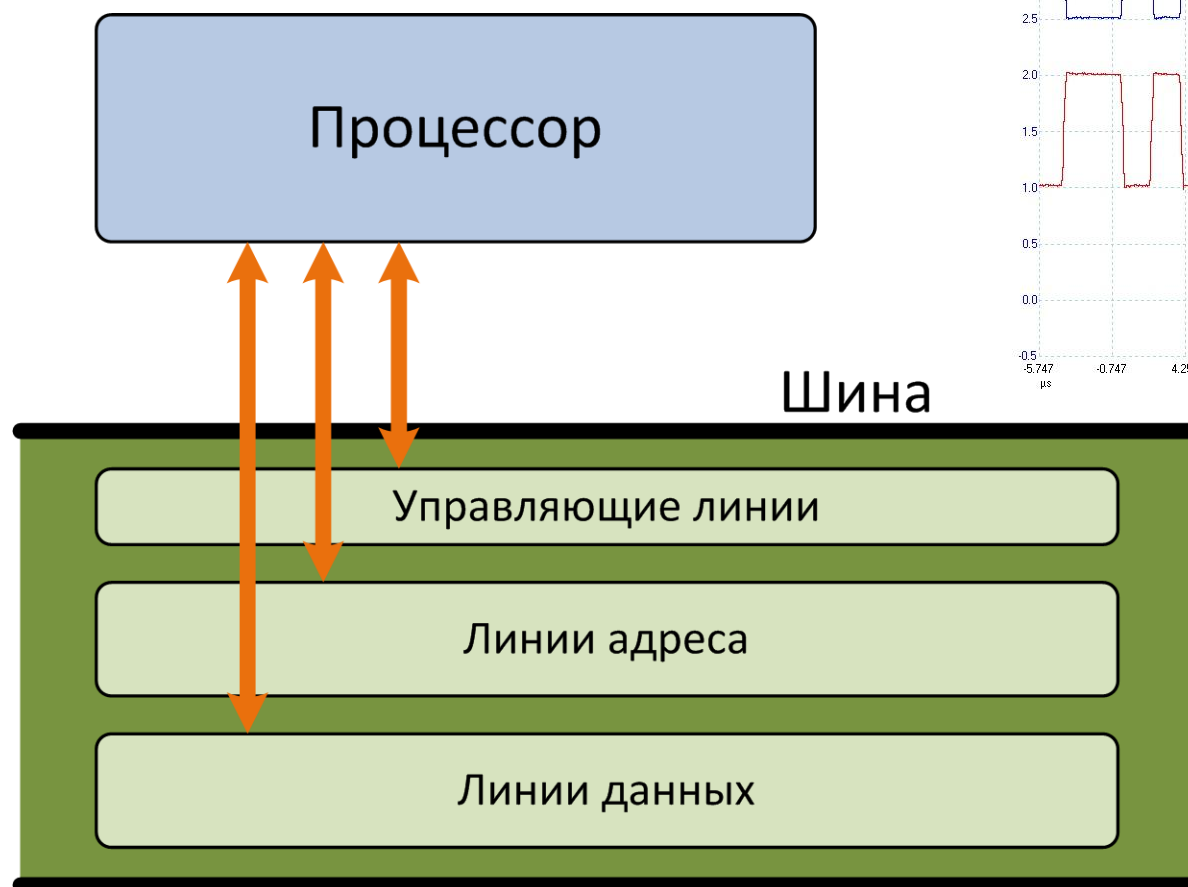


Лекция 0x16

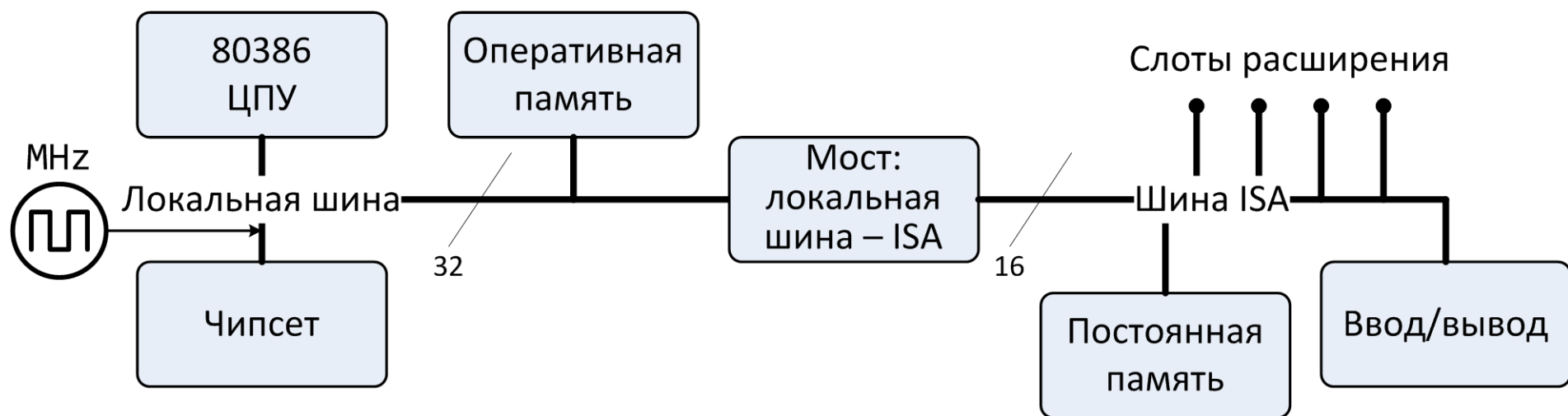
8 мая

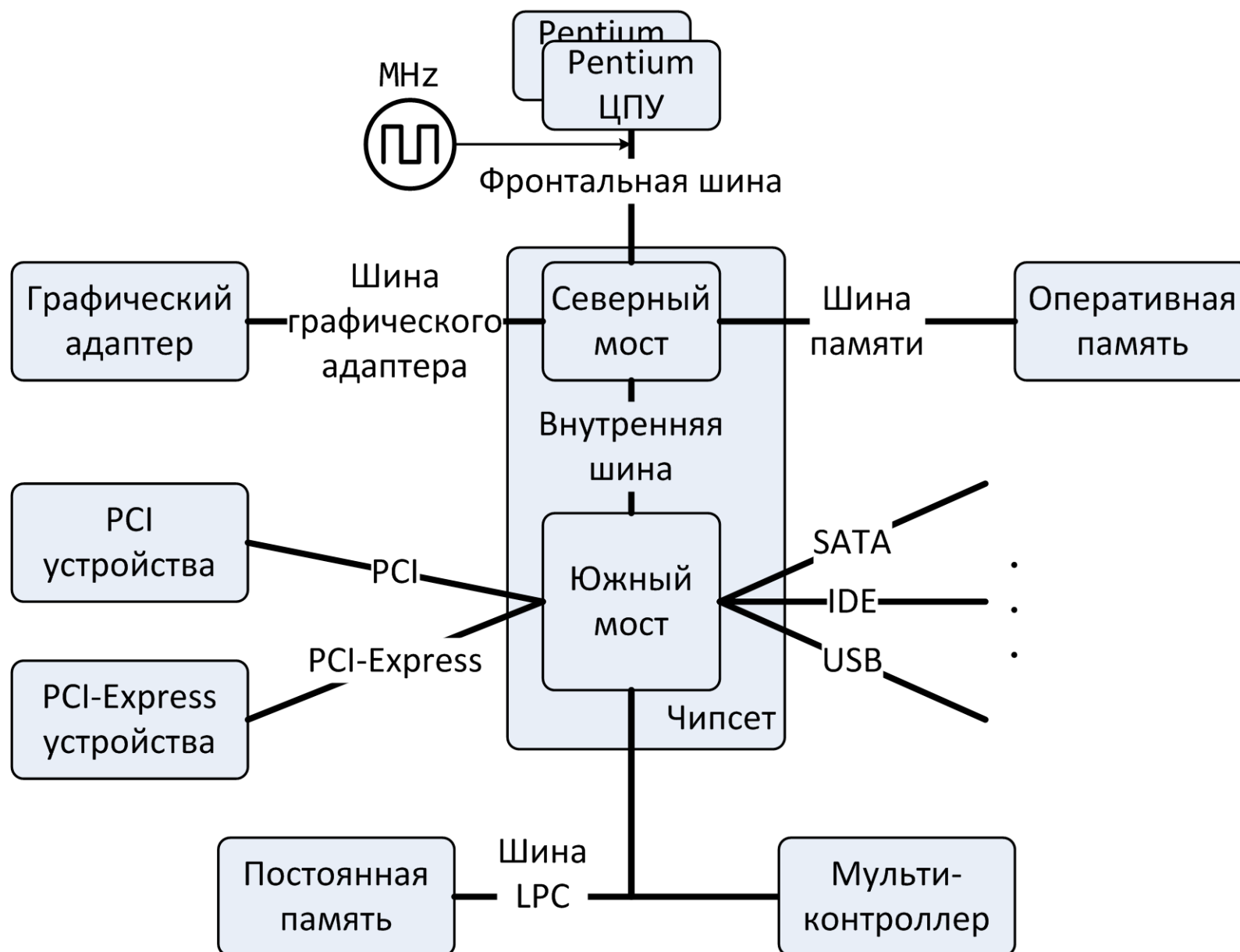
Шина: точка зрения разработчика аппаратуры

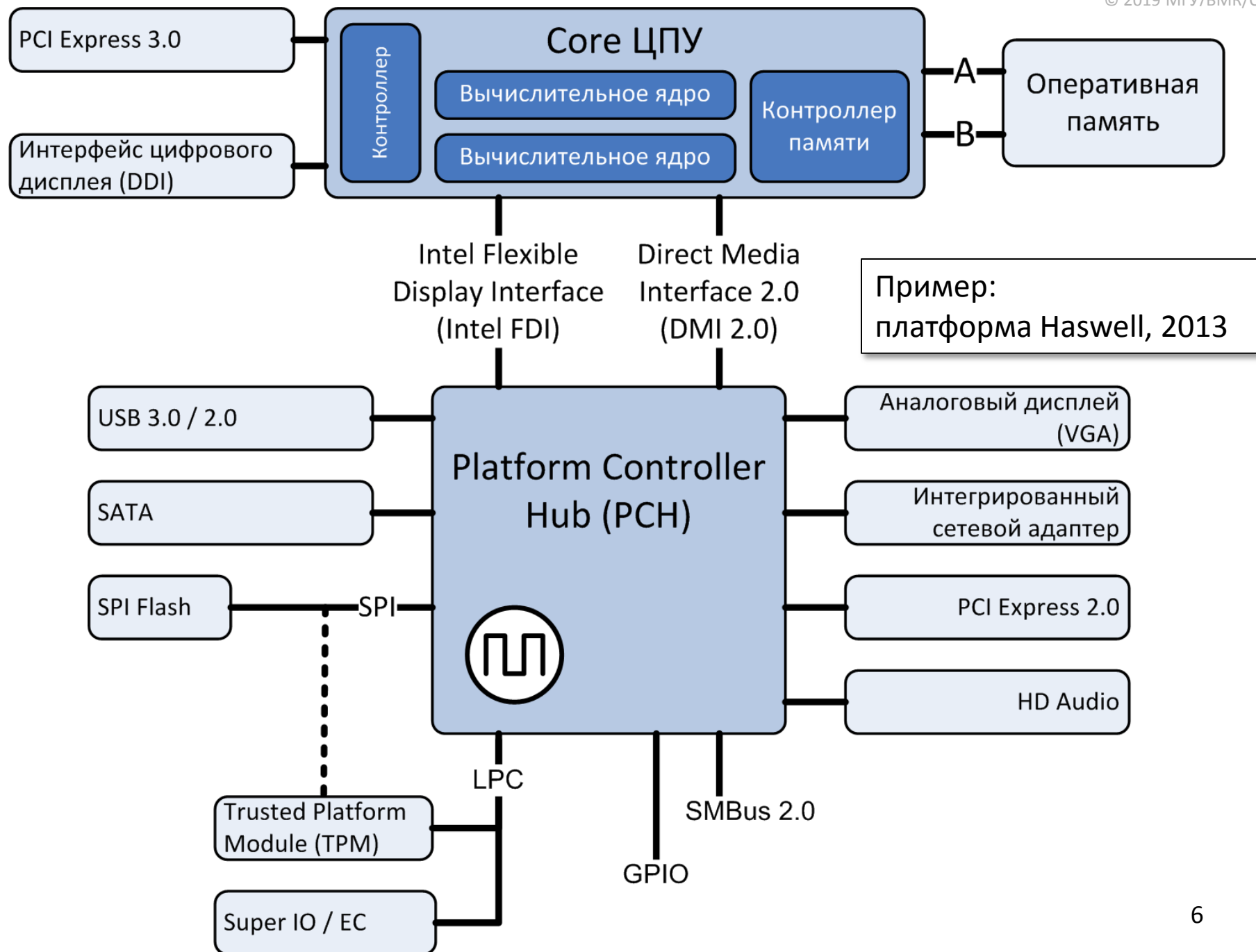


Характеристики шин

- Ширина
 - Количество линий, линии адреса, линии данных, мультиплексирование адресов и данных
- Частота
- Пиковая пропускная способность
 - Ширина (байты или биты) × Частота (1/сек)
- Арбитраж
 - Централизованный
 - Линии запроса и захвата шины
 - Децентрализованный
- Возможность горячей замены устройства
- Физическая организация
 - Выделенные линии данных, адресов, команд
 - Мультиплексированные линии
 - Топология связей
 - Синхронная/асинхронная
 - Ограничения по длине линии





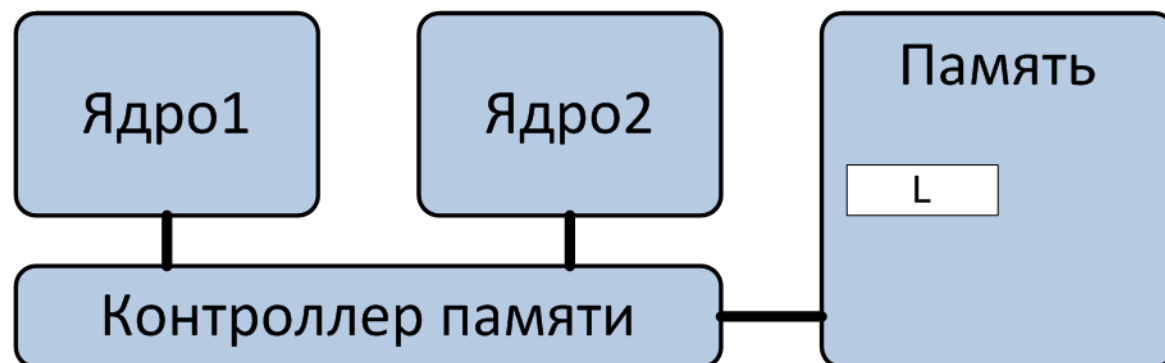


Примеры шин (1/3)

- Фронтальная шина
 - HyperTransport (HT), апрель 2001, AMD. Открытый стандарт - HyperTransport Technology
 - 2 – 32 разряда, двунаправленная
 - 200 – 2600 MHz, DDR
 - QuickPath Interconnect (QPI), ноябрь 2008, Intel
 - 20 линий, двунаправленная, 4 такта = 64 бита
 - 2.4, 2.93, 3.2 GHz, DDR
 - Intel Ultra Path Interconnect (UPI) в процессорах Skylake EX/EP Xeon,
 - Соединение точка-точка.
 - Гарантированные физические каналы: один отправитель – один приемник.
 - Не требуется арбитраж.

Синхронизация обращений к памяти

- Захват блокировки
- Работа с общими (разделяемыми) переменными
- Освобождение блокировки



Замок – статическая переменная L (младший разряд)

- 0 – доступ открыт
- 1 – доступ закрыт

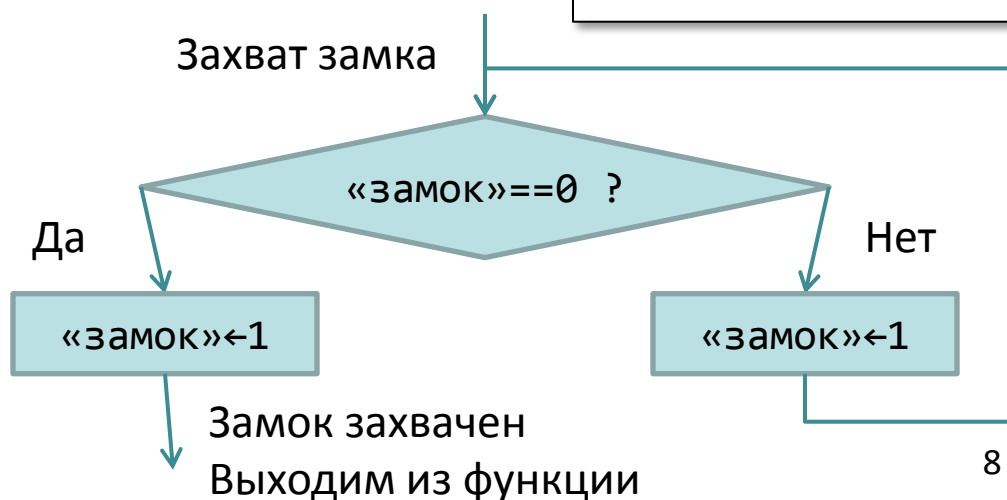
Начальное состояние замка – 0

```

acquireLock:
.retry:
    bts byte [L], 0
    jc .retry
    ret
  
```

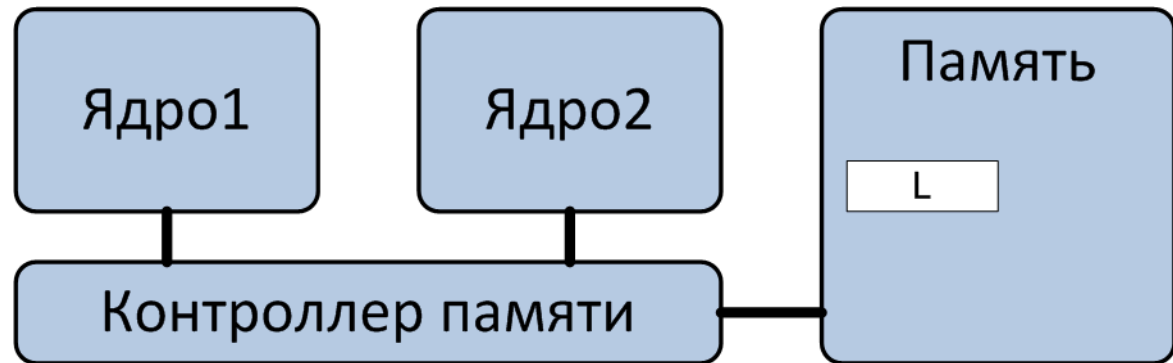
```

releaseLock:
    btr byte [L], 0
    ret
  
```



Синхронизация обращений к памяти

- Захват блокировки
- Работа с общими (разделяемыми) переменными
- Освобождение блокировки



Замок – статическая переменная L (младший разряд)

- 0 – доступ открыт
- 1 – доступ закрыт

Начальное состояние замка – 0

```

acquireLock:
.retry:
    bts byte [L], 0
    jc .retry
    ret
  
```

```

releaseLock:
    btr byte [L], 0
    ret
  
```

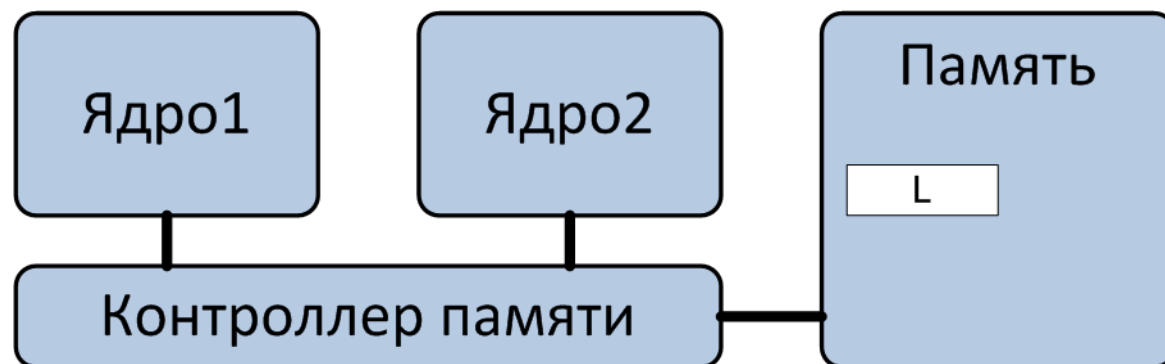
Освобождение замка

«замок» ← 0

Замок освобожден
Выходим из функции

Синхронизация обращений к памяти

- Захват блокировки
- Работа с общими (разделяемыми) переменными
- Освобождение блокировки



Оба ядра попытались захватить замок «*почти*» одновременно

```

acquireLock:
.retry:
    bts byte [L], 0
    jc .retry
    ret
  
```

```

releaseLock:
    btr byte [L], 0
    ret
  
```

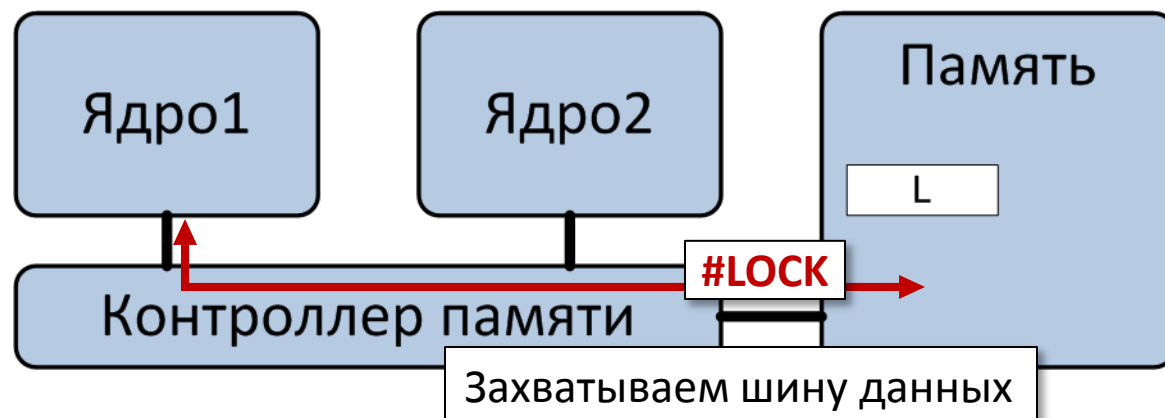
Может возникнуть
Состояние Гонки

Ядро1		Ядро2	
ЦПУ ← память	В		
CF ← ЦПУ:L 0	Т	ЦПУ ← память	В
память:L ← 1	С	CF ← ЦПУ:L 0	Т
		память:L ← 1	С

↓
Время

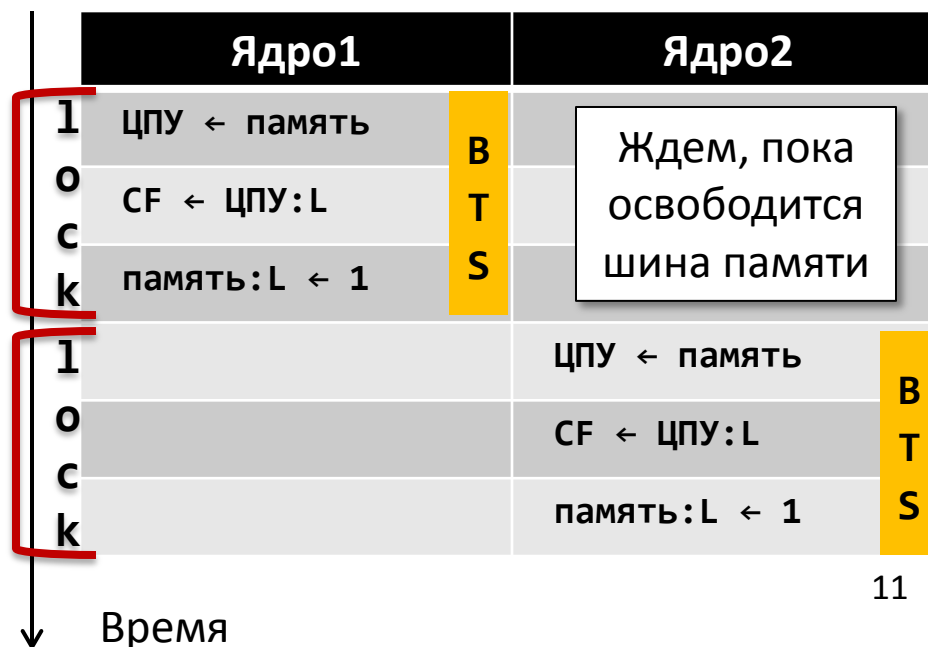
Синхронизация обращений к памяти

- Захват блокировки
- Работа с общими (разделяемыми) переменными
- Освобождение блокировки



```
acquireLock:
.retry:
    lock bts byte [L], 0
    jc .retry
    ret

releaseLock:
    lock btr byte [L], 0
    ret
```



Синхронизация обращений к памяти

- Явно указываемый префикс lock
 - Применим только к некоторым командам ADD, AND, BTC, BTR, BTS, CMPXCHG, ...
 - Первый операнд команды – память
- Все время выполнения команды процессор удерживает шину памяти, посылая на нее сигнал #LOCK
- Постоянный захват шины негативно сказывается на производительности

```
acquireLock:
    lock bts byte [L], 0
    jc .retry
    ret
.retry:
    pause
    lock bts byte [L], 0
    jc .retry
    ret
```

В современных процессорах команда `pause` используется как подсказка, что выполнение находится в цикле активного ожидания (busy wait)

ОС Linux использует для реализации активного ожидания не `bts/btr`, а гораздо более быструю команду `cmpxchg`

<http://heather.cs.ucdavis.edu/~matloff/50/PLN/lock.pdf> (на английском)

Примеры шин (2/3)

- Общая шина ввода/вывода для подключения периферийных устройств Peripheral component interconnect (PCI), 1992, Intel, открытый стандарт.
 - PCI 1.0 / 2.0
 - Топология - общая шина, децентрализованный арбитраж
 - 32 линии, общие для адресов и данных
 - Передача данных транзакциями, возможна приостановка
 - Частота 33 MHz
 - Расширения
 - PCI 64, PCI 66, PCI 64/66, PCI-X (266 и 533 МГц)
 - PCI Express (PCI-E), июль 2002, Intel
 - Топология – звезда, подключение устройств через двунаправленные соединения различной ширины. x1 – 4 проводника (дифф. пара)
 - x1, x2, x4, x8, x12, x16, x32
 - Скорость работы соединения варьируется
 - 2.5, 5.0, 8.0 и 16GT/s (32GT/s в PCI 5.0) , T/s - передач по шине в секунду
 - Избыточное кодирование данных, 128b/130b в PCI 3.0 и более новых, 8b/10b – в старых версиях шины
- Идентификация PCI-устройств
 - $256 \text{ шин} \times 32 \text{ устройства} \times 8 \text{ функций} = 65536 \text{ PCI-устройств}$

Примеры шин (2/3)

- Конфигурационное пространство регистров PCI-устройства составляет суммарно 256 байт
- Регистры первых 64 байт стандартизированы
- Чтение и запись в регистры
 - Через порты ввода/вывода 0xCF8 (адрес) и 0xCFC (данные) командами IN и OUT
 - Через отображение регистров PCI-устройства в 4КБ адресного пространства памяти (зависит от реализации...)
В адресном пространстве «теряется» 256 МБ памяти

31		16 15		0
Device ID		Vendor ID		00h
Status		Command		04h
Class Code			Revision ID	08h
BIST	Header Type	Lat. Timer	Cache Line S.	0Ch
Base Address Registers				10h
				14h
				18h
				1Ch
				20h
				24h
Cardbus CIS Pointer				28h
Subsystem ID		Subsystem Vendor ID		2Ch
Expansion ROM Base Address				30h
Reserved			Cap. Pointer	34h
Reserved				38h
Max Lat.	Min Gnt.	Interrupt Pin	Interrupt Line	3Ch

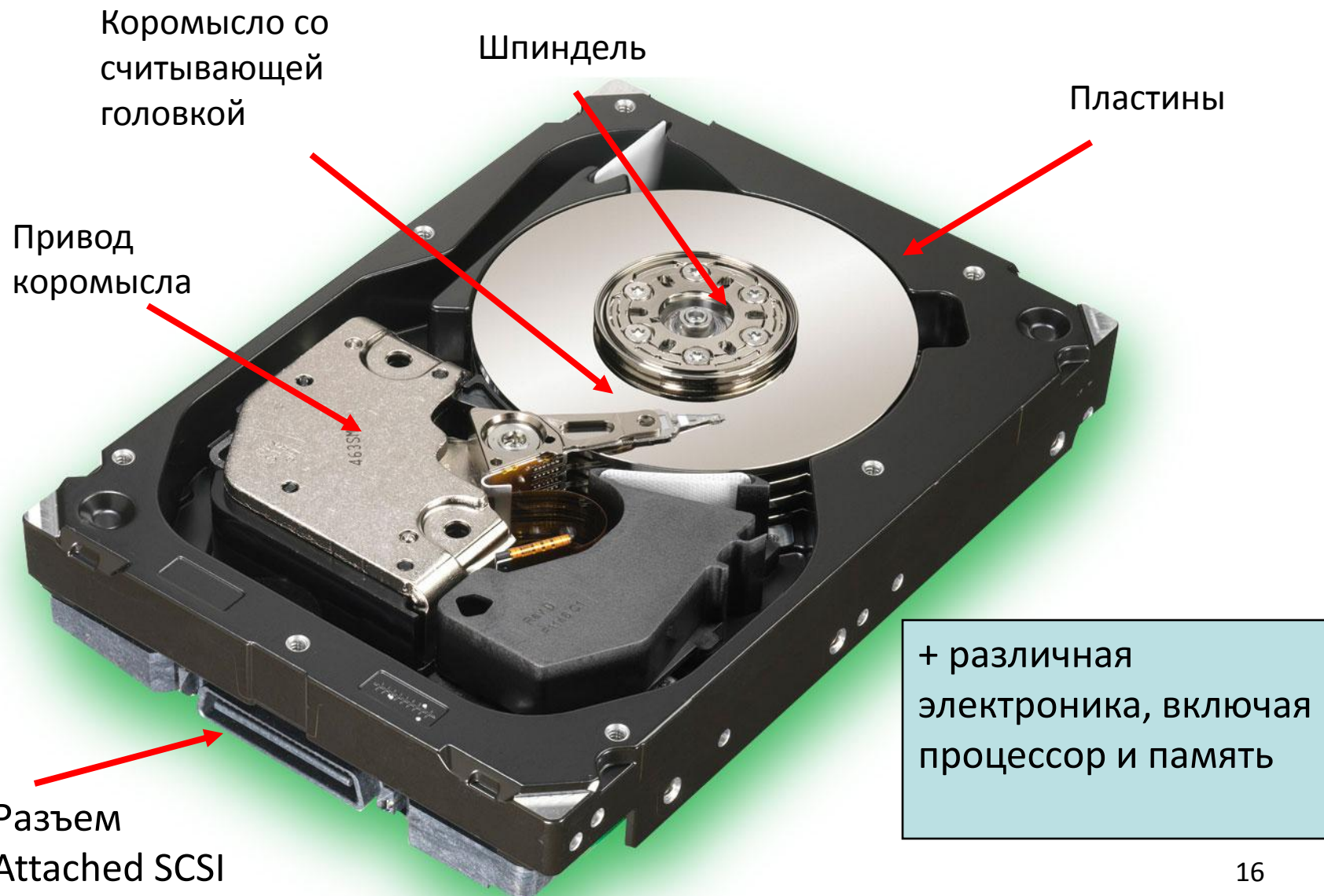
Дополнительный технический материал (на английском)

[https://pcisig.com/sites/default/files/files/PCI Express Basics Background.pdf](https://pcisig.com/sites/default/files/files/PCI%20Express%20Basics%20Background.pdf)

Примеры шин (3/3)

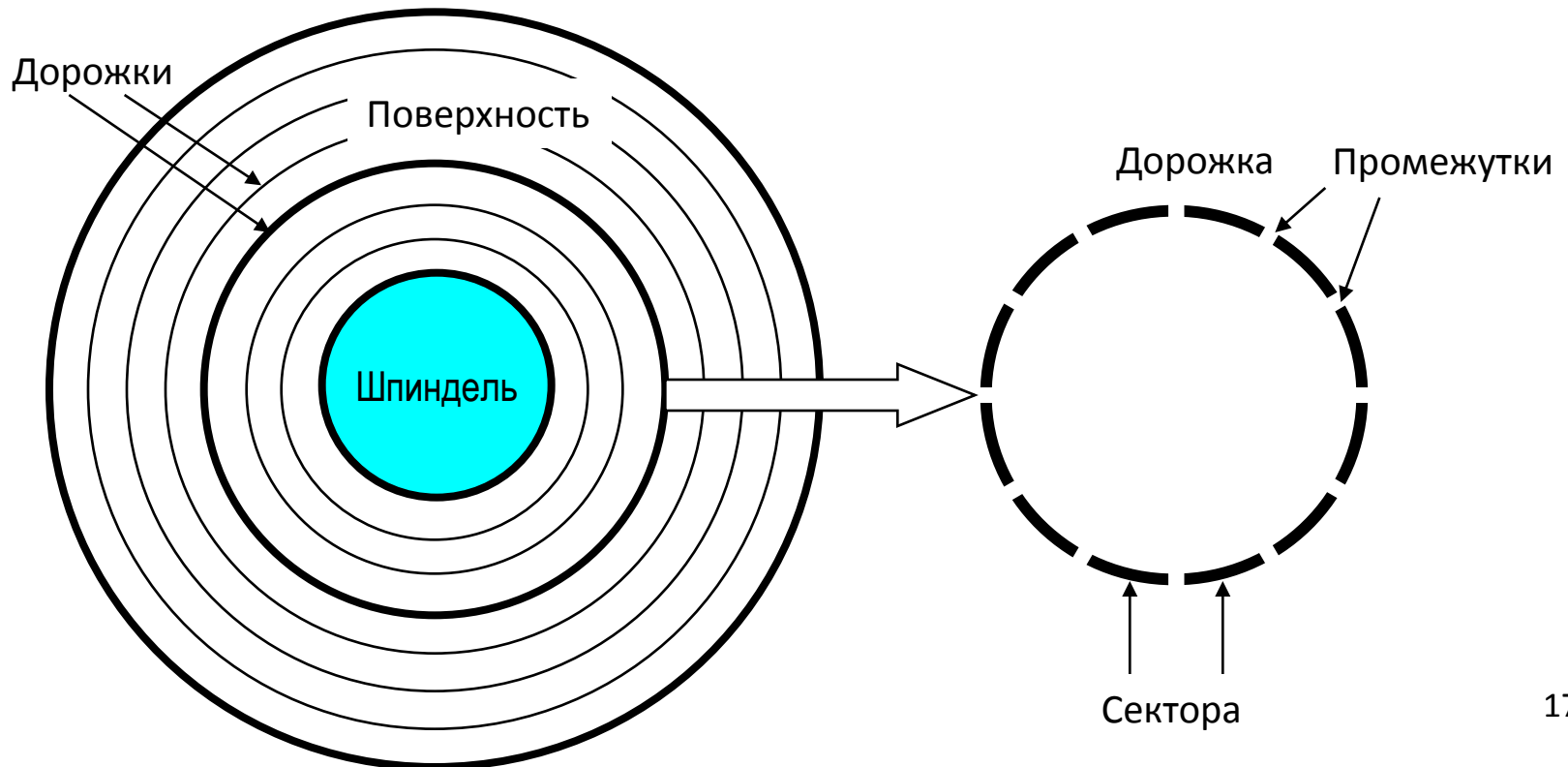
- AGP, 1996
 - Непосредственный доступ к оперативной памяти со стороны видеоадаптера, GART (graphics address remapping table)
- USB, 1996
 - Топология – звезда, до 127 устройств, разветвители, оконечные точки (15/15 + 1/1 управляющие).
 - 4 типа передач: управляющие, поточные, прерывания, изохронные
 - Open Host Controller Interface
 - Сам контроллер является PCI-устройством
- Serial ATA, 2000
 - 7 линий, 2 – прием, 2 – передача.
 - 1.5, 3, 6 GBit/s
 - Возможность горячей замены

Внутреннее устройство HDD



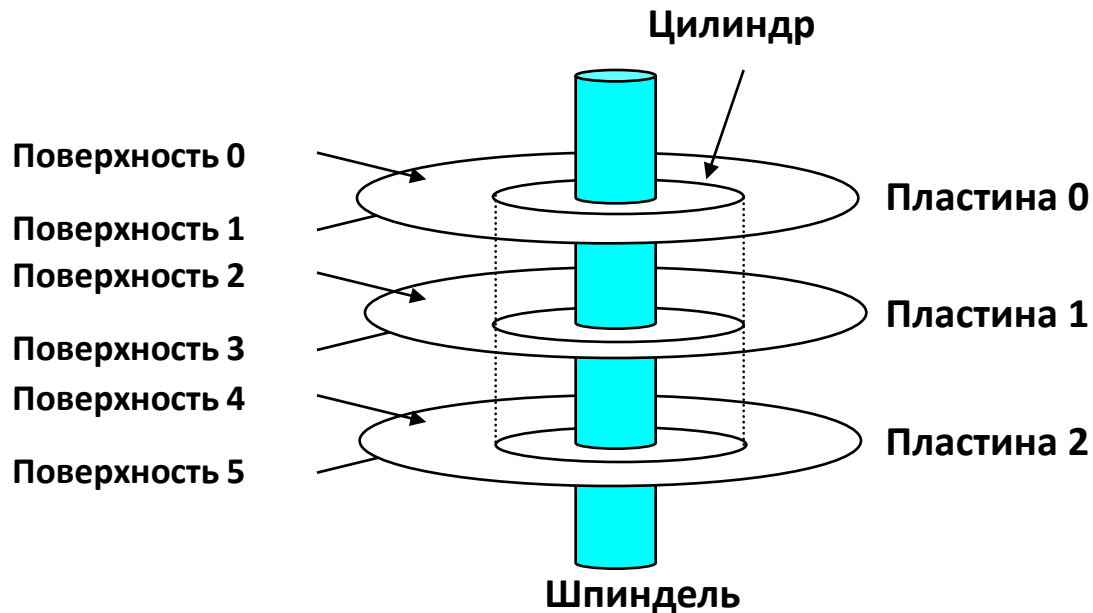
Геометрия диска

- Диск состоит из **пластин**, каждая обладает двумя **поверхностями**.
- Каждая поверхность состоит из концентрических кругов – **дорожек**.
- Каждая дорожка состоит из **секторов**, разделенных **промежутками**.



Геометрия диска (несколько пластин)

- Равноудаленные от шпинделя дорожки образуют **цилиндр**.



Емкость диска

- **Емкость:** максимальное количество сохраняемых бит.
 - Производители выражают емкость в «необычных» гигабайтах, $1 \text{ ГБ} = 10^9$ байтам.
 - Различают десятичные (СИ) и двоичные приставки (МЭК)
 - Разница: КБ и КиБ $\approx 2.4\%$, МБ и МиБ $\approx 4.9\%$, ТБ и ТиБ $\approx 9.95\%$
- Емкость определяется следующими технологическими факторами:
 - Плотность записи / линейная плотность (биты/дюймы – BPI): сколько битов может быть размещено на одном дюйме дорожки.
 - Трековая плотность (треки/дюйм – TPI): сколько треков может быть размещено на одном дюйме радиуса.
 - Поверхностная плотность (биты/дюймы²): произведение линейной плотности на трековую плотность.
- Современные диски группируют дорожки в несколько зон записи
 - Каждая дорожка в зоне состоит из одного и того же количества секторов, определяемого длиной самой короткой дорожки.
 - У каждой зоны различное количество дорожек/секторов

Вычисление емкости диска

Емкость = (#байт/сектор) x (среднее # сектор/дорожка) x
(# дорожка/поверхность) x (# поверхность/пластина) x
(# пластина/диск)

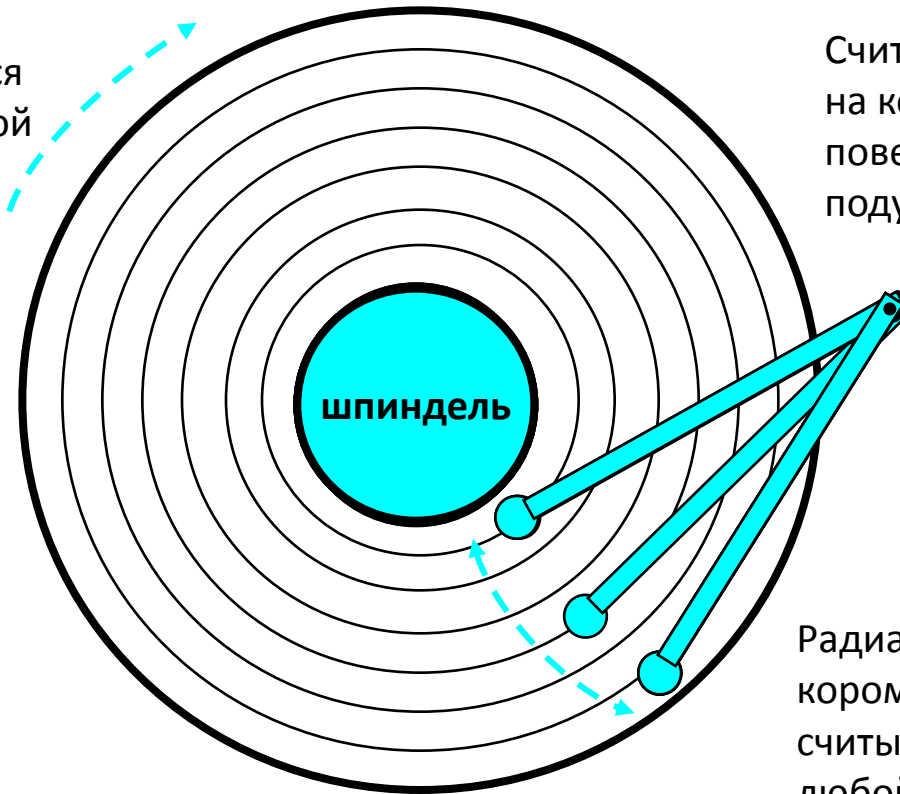
Пример:

- 512 байт/сектор
- 300 сектор/дорожка (в среднем)
- 20,000 дорожка/поверхность
- 2 поверхность/пластина
- 5 пластина/диск

Емкость = $512 \times 300 \times 20000 \times 2 \times 5 = 30,720,000,000 = 30.72\text{ГБ}$ ₂₀

Работа с диском (одна пластина)

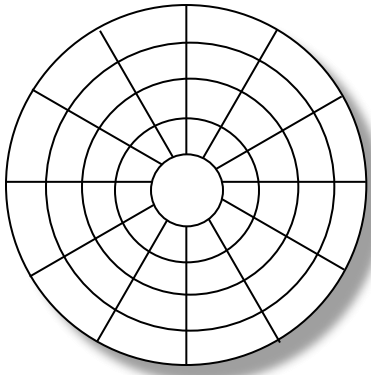
Поверхность диска вращается с фиксированной скоростью.



Считывающая головка закреплена на конце коромысла и парит над поверхностью на тонкой воздушной подушке.

Радиально перемещаясь, коромысло может выставить считывающую головку над любой дорожкой.

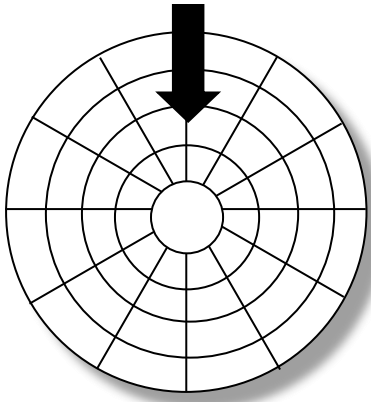
Структура диска – вид сверху на одну пластину



Поверхность разбита на дорожки

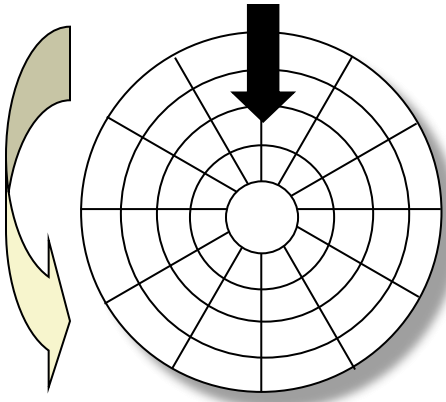
Дорожки разделены на сектора

Доступ к диску



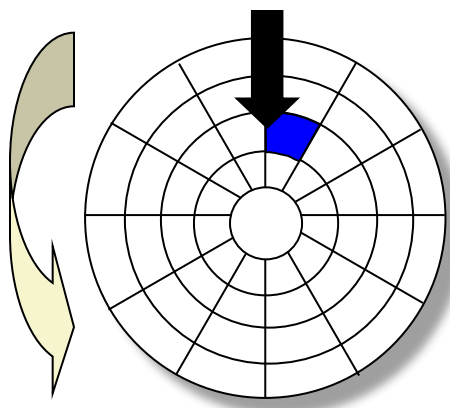
Считывающая головка в указанной
позиции над диском

Доступ к диску



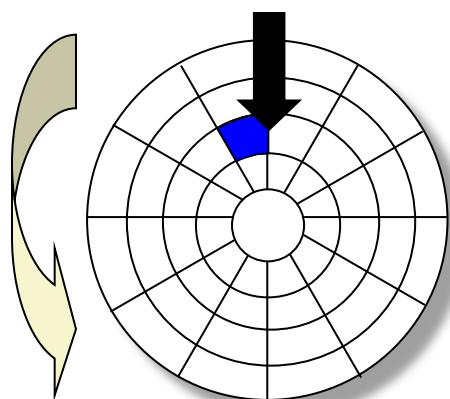
Направление вращения – против
часовой стрелки

Доступ к диску – Чтение



Перед чтением синего сектора

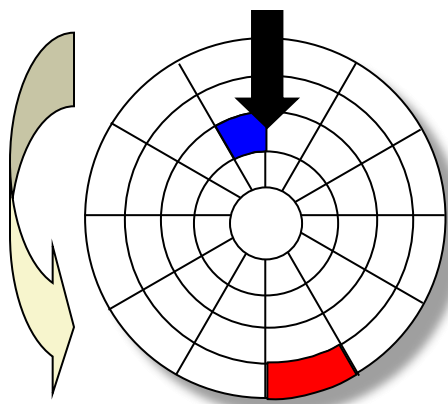
Доступ к диску – Чтение



Синий сектор
считан

После чтения синего сектора

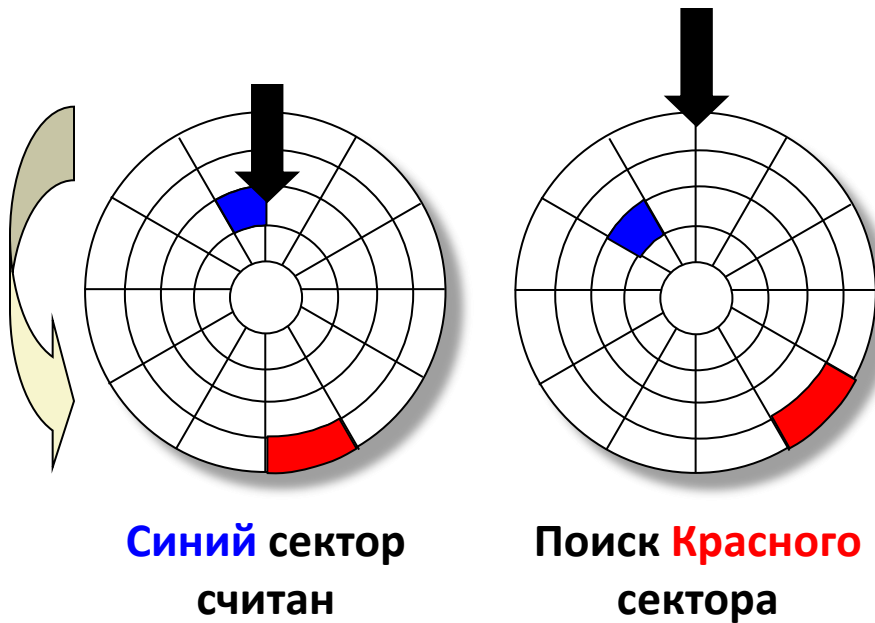
Доступ к диску – Чтение



Синий сектор
считан

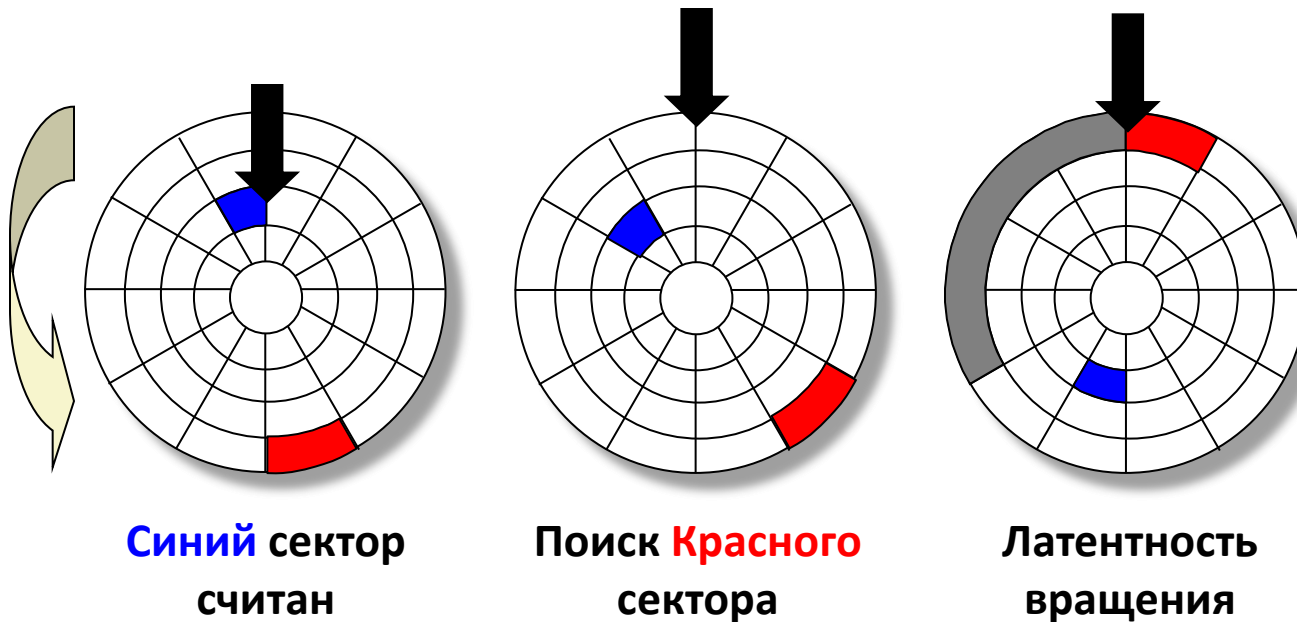
Поступил запрос на чтение красного
сектора

Доступ к диску – Поиск



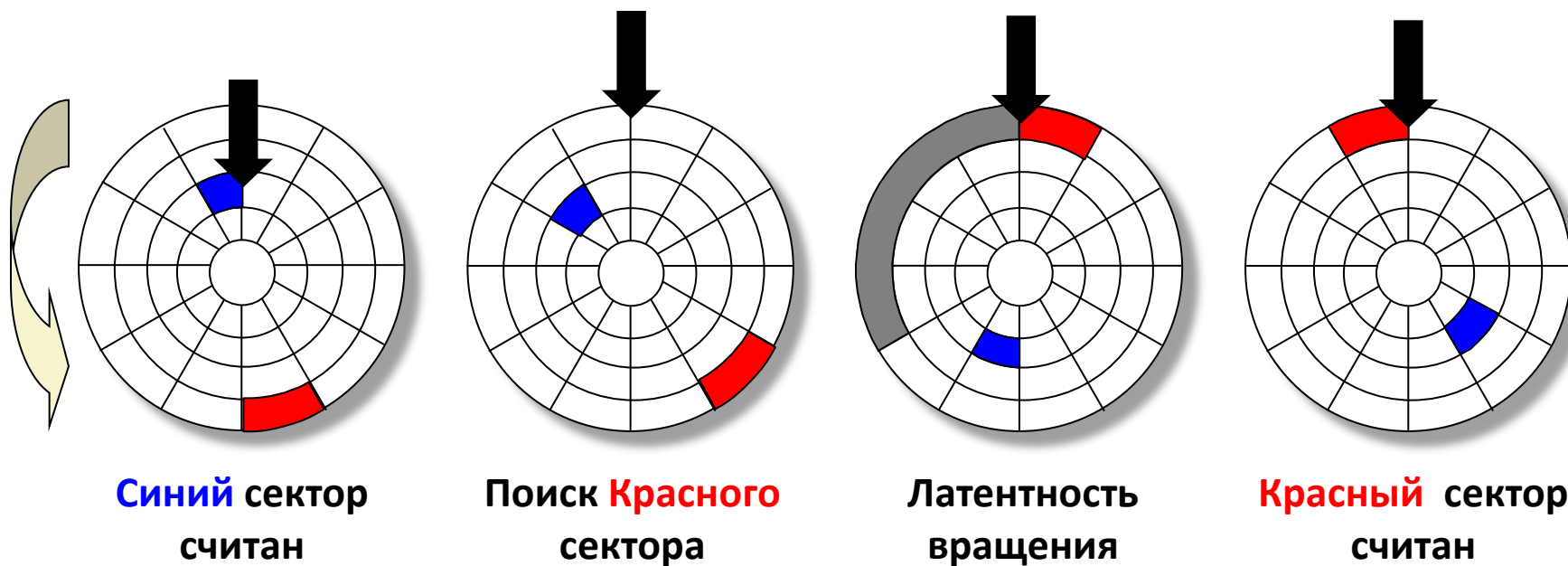
Ищем дорожку на которой расположен красный сектор

Доступ к диску – временная задержка из-за вращения



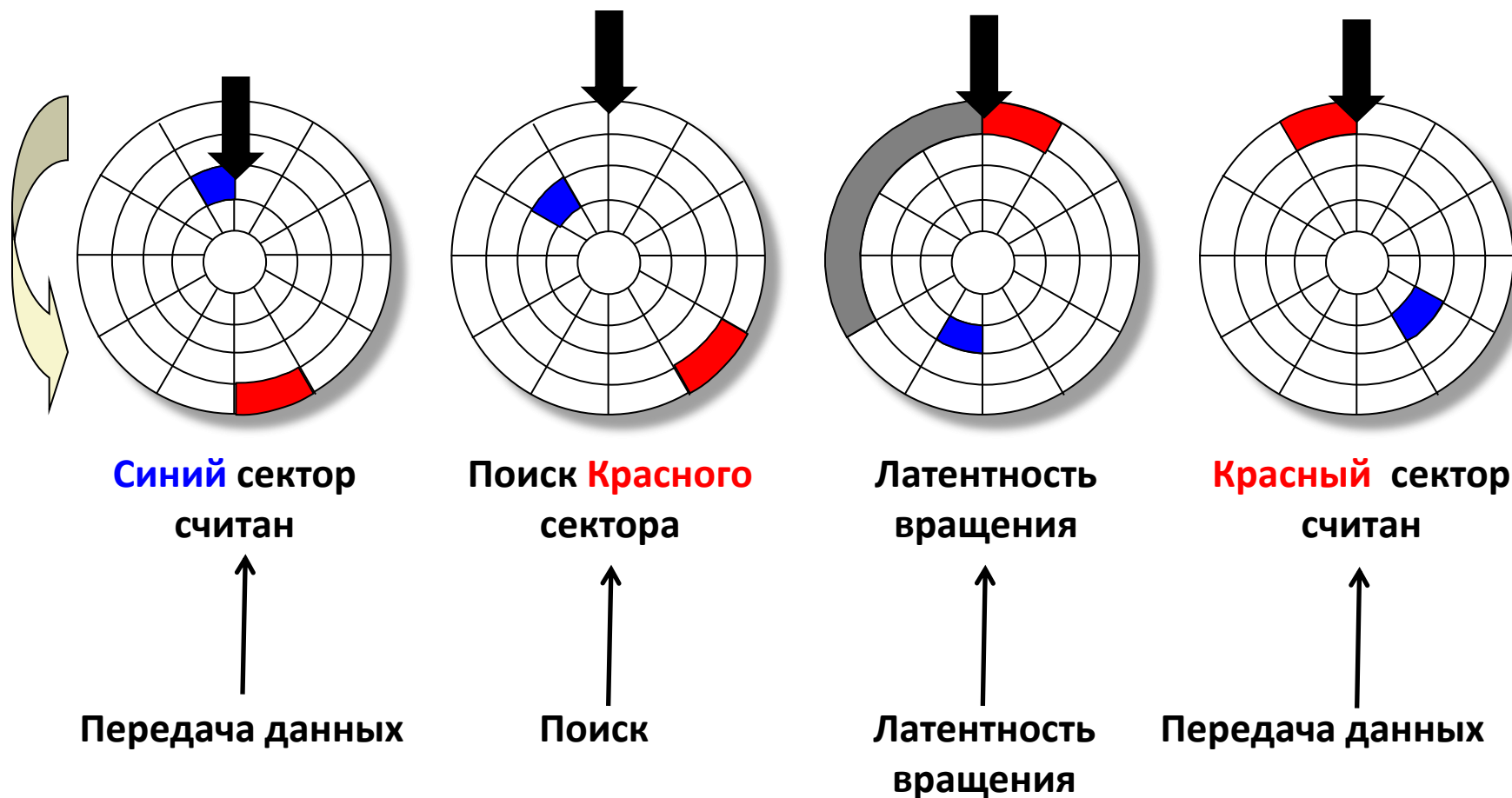
Вынужденное ожидание того момента, когда красный сектор достигнет считывающей головки

Доступ к диску – Чтение



Чтение красного сектора завершено

Доступ к диску – распределение времени



Время доступа к диску

- $T_{\text{доступа}} = T_{\text{ср. поиск}} + T_{\text{ср. вращения}} + T_{\text{ср. передача}}$
- **Время поиска** ($T_{\text{ср. поиск}}$)
 - Время, требуемое для перемещения считывающей головки в цилиндр, содержащий требуемый сектор.
 - Как правило $T_{\text{ср. поиск}}$ занимает 3—9 мс.
- **Латентность вращения** ($T_{\text{ср. вращения}}$)
 - Время ожидания момента, когда первый бит запрашиваемого сектора достигнет считывающей головки.
 - $T_{\text{ср. вращения}} = 1/2 \times 1/\text{RPM} \times 60 \text{ с} / 1 \text{ мин}$
 - Типичная скорость вращения – 7200 RPM. $T_{\text{ср. вращения}} \approx 4 \text{ мс.}$
- **Время передачи** ($T_{\text{ср. передача}}$)
 - Время чтения содержимого сектора.
 - $T_{\text{ср. передача}} = 1/\text{RPM} \times 1/(\text{ср. \# секторов на дорожке}) \times 60 \text{ с./1 мин.}$

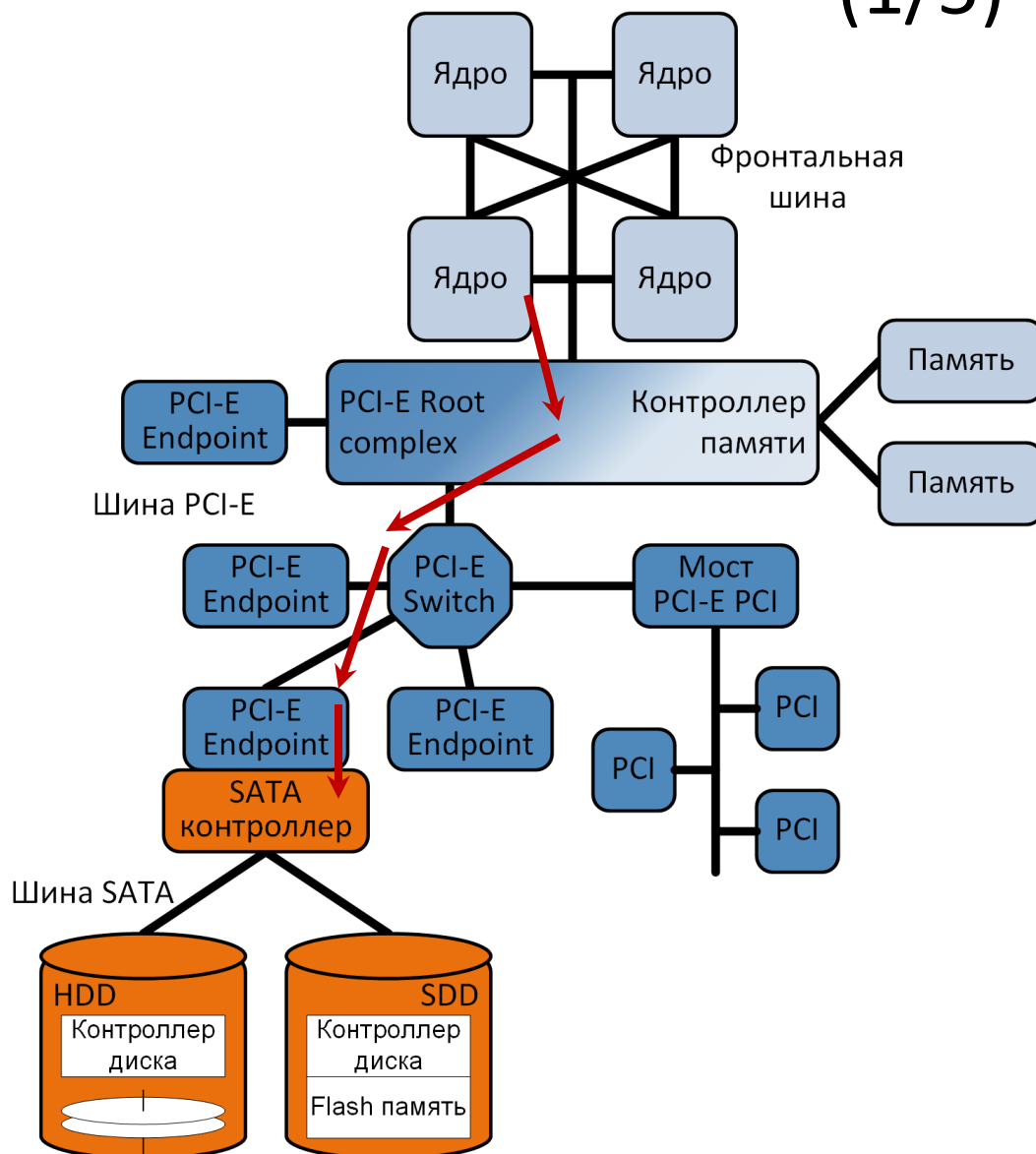
Пример оценки времени доступа

- Исходные характеристики:
 - Скорость вращения = 7,200 RPM
 - Среднее время поиска = 9 мс.
 - Среднее # секторов на дорожке = 400.
- Оцениваем слагаемые и общую сумму:
 - $T_{\text{ср. вращения}} = 1/2 \times (60 \text{ с}/7200 \text{ RPM}) \times 1000 \text{ мс} = 4 \text{ мс}.$
 - $T_{\text{ср. передача}} = 60/7200 \text{ RPM} \times 1/400 \text{ с/дорожка} \times 1000 \text{ мс} = 0.02 \text{ мс}$
 - $T_{\text{доступа}} = 9 \text{ мс} + 4 \text{ мс} + 0.02 \text{ мс}$
- Выводы:
 - Время передачи существенно меньше остальных слагаемых.
 - Считать первый бит из сектора – «дорогая» операция, считывание остальных битов – «дешево».
 - Время доступа SRAM $\approx 4 \text{ нс}$ для двойного слова, DRAM $\approx 60 \text{ нс}$
 - Диск медленнее SRAM в 40,000 раз, и ...
 - ... в 2,500 раз, чем DRAM.

Логические блоки

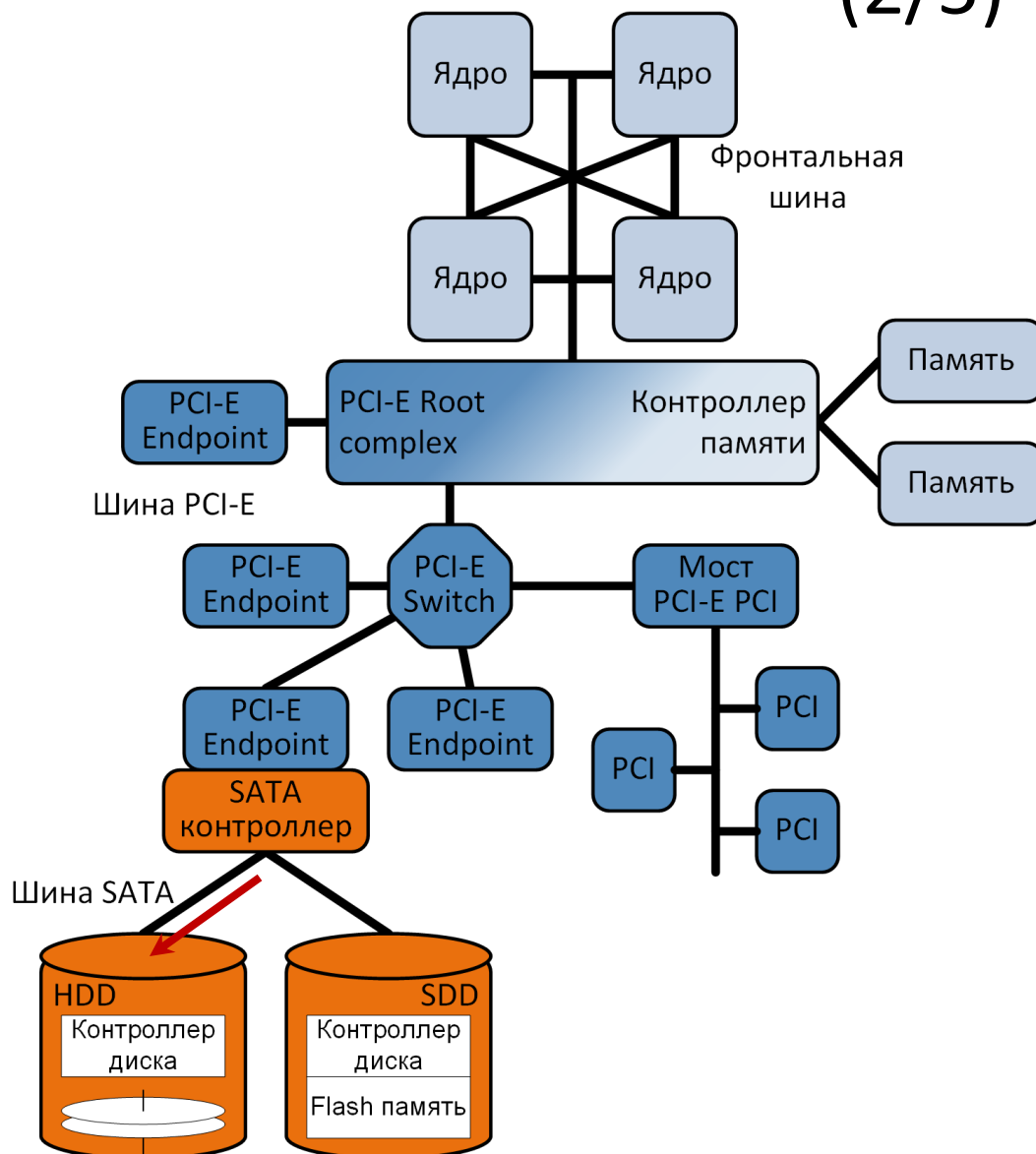
- Первоначальный способ задания сектора: $\langle C, H, S \rangle$
 - В разных зонах – разное число секторов на дорожке
 - Для обращения к диску необходимо знать его геометрию
- Более простой метод обращения к данным:
 - Сектора рассматриваются как последовательность **логических блоков** (0, 1, 2, ...)
- Соответствие между логическими блоками и (физическими) секторами
 - Отображение поддерживается аппаратурой = контроллер диска + «прошивка»
 - Номер логического блока \rightarrow (поверхность, дорожка, сектор).
- Защита от выхода из строя отдельных цилиндров. Каждая зона записи содержит запасные цилиндры.
 - Размер диска после форматирования становится ощутимо меньше. 34

Как работает ввод/вывод SATA-устройств (1/5)



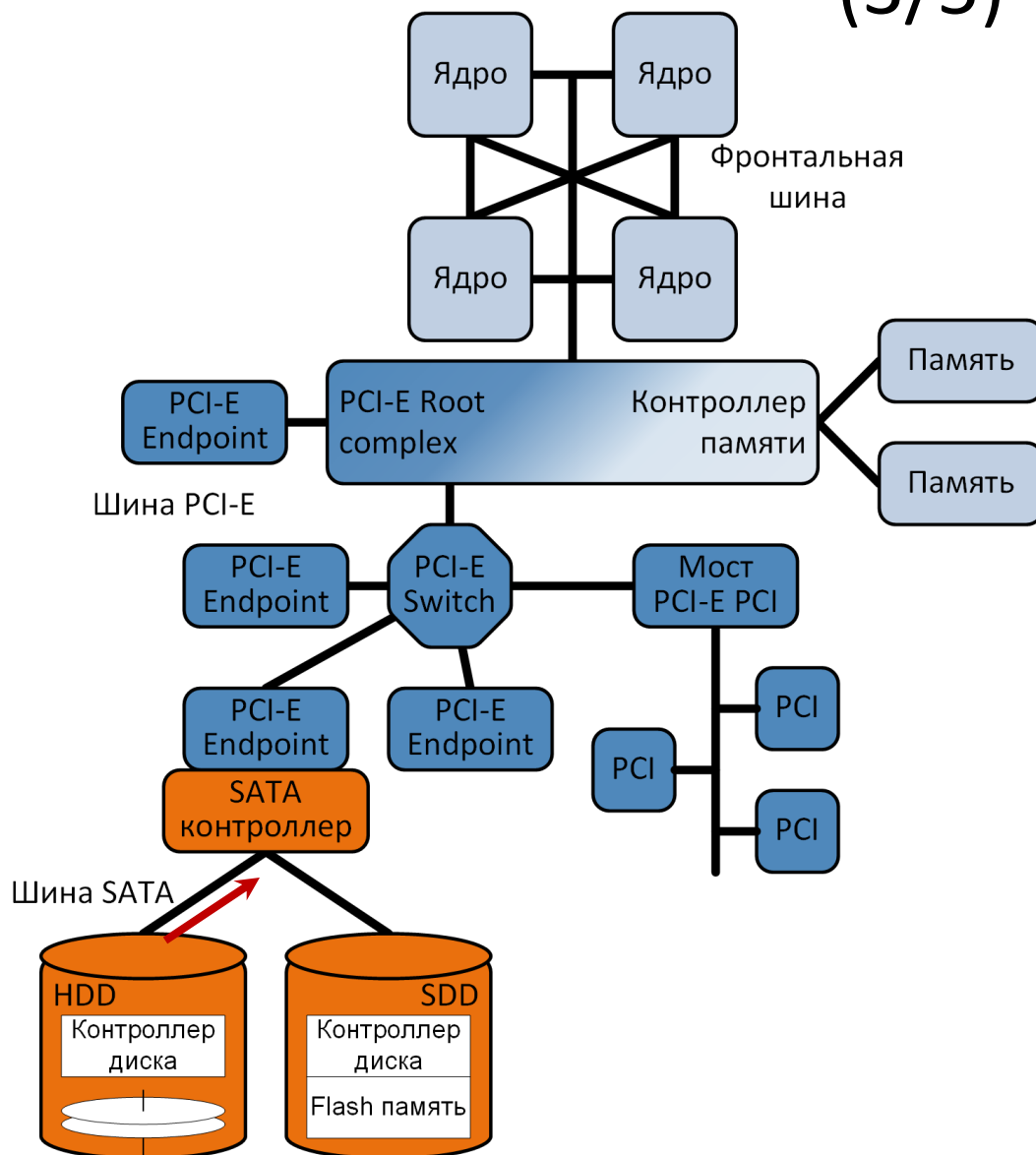
- ЦПУ (некоторое вычислительное ядро) запускает чтение сектора диска, записав по определенному адресу оперативной памяти (адрес связан с SATA-контроллером) команду «чтение», номер логического блока, адрес буфера памяти, в который необходимо поместить содержимое сектора.

Как работает ввод/вывод SATA-устройств (2/5)



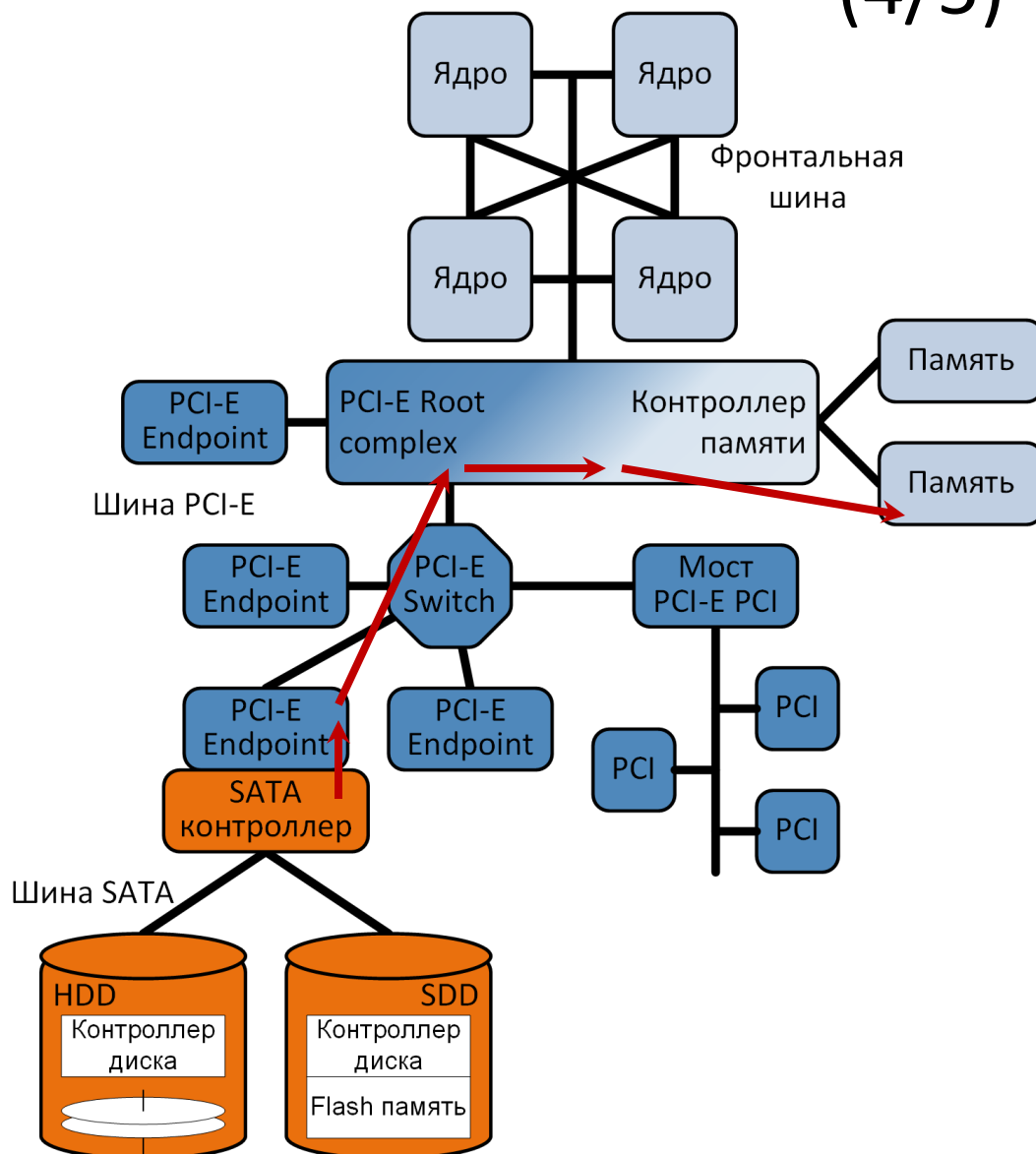
- Контроллер через интерфейс SATA отправляет запрос (команду) на чтение сектора соответствующему диску

Как работает ввод/вывод SATA-устройств (3/5)



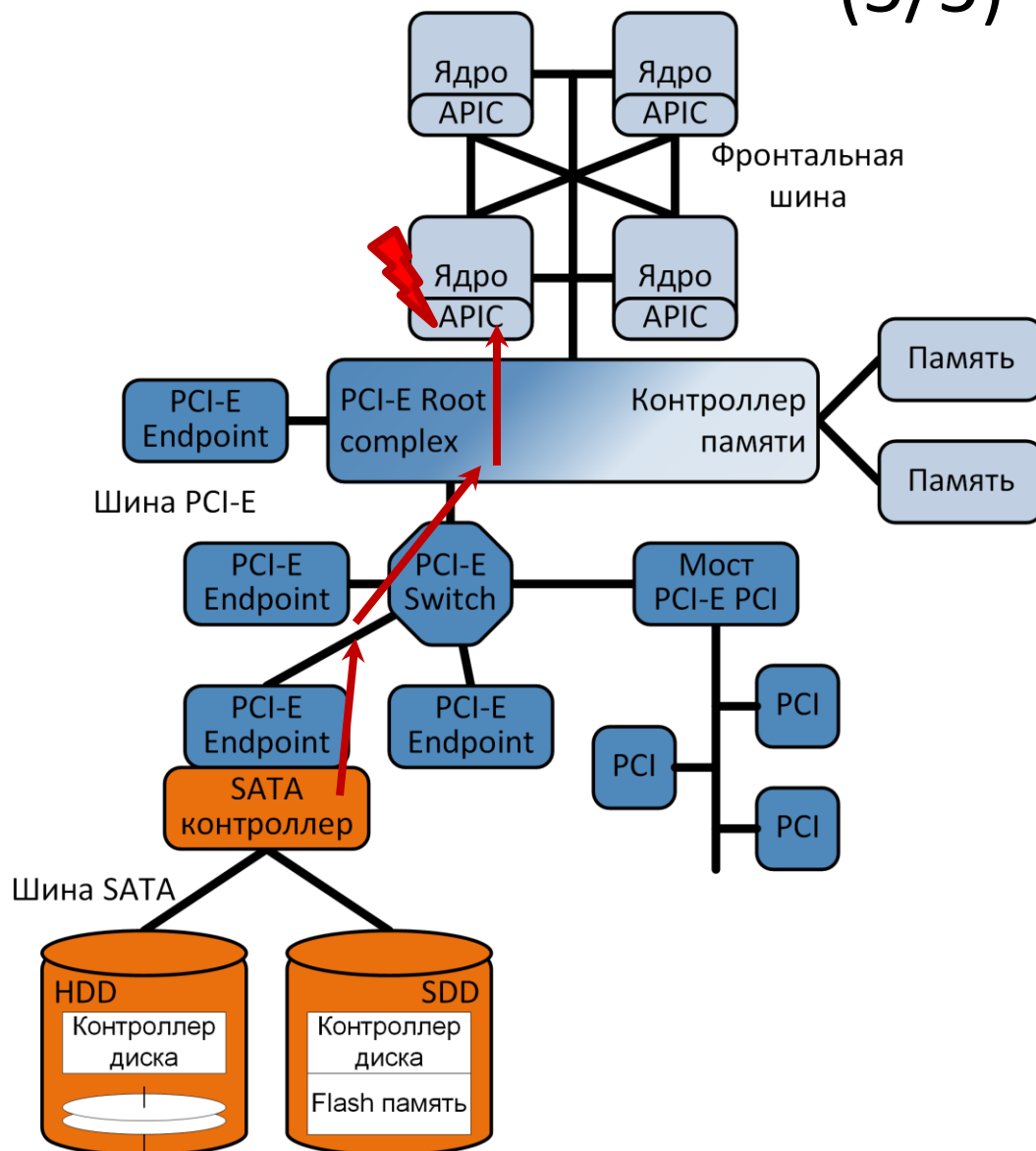
- Контроллер диска пересчитывает LBA (линейный адрес) в тройку CHS (поверхность, дорожка, сектор), считывает содержимое сектора с пластин и отправляет его SATA-контроллеру
- Если запрашиваемый сектор уже считывался некоторое время назад, его содержимое может храниться в микросхеме DRAM-памяти диска. В таком случае данные можно считать из нее, а не с пластины.

Как работает ввод/вывод SATA-устройств (4/5)



- SATA-контроллер инициирует передачу содержимого сектора напрямую в память через шину PCI в виде серии PCI-транзакций записи

Как работает ввод/вывод SATA-устройств (5/5)



- После завершения записи данных в память SATA-контроллер должен как-то сообщить процессору, что все данные уже помещены в память.
- SATA-контроллер через шину PCI пишет по некоторому адресу памяти (адрес связан с локальным контроллером прерываний вычислительного ядра) сообщение – команду вызвать аппаратное прерывание в ядре (механизм Message Signaled Interrupts)
- Прерывание в процессоре передает управление функции-обработчику