Лекция 0x15

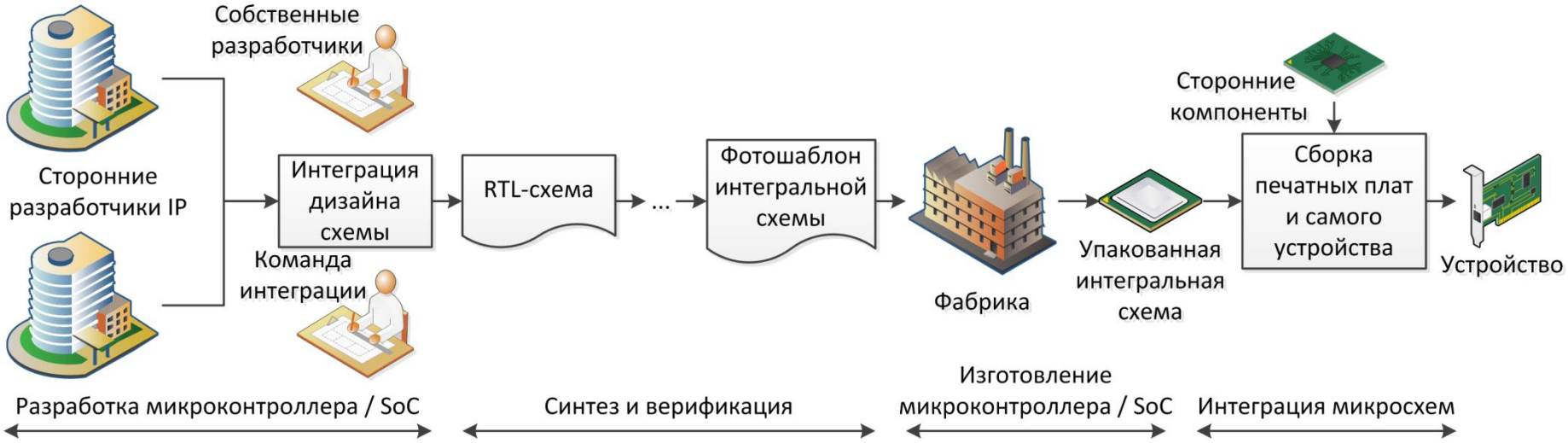
27 апреля

Разработка интегральных схем

```
module blinking (
    input CLOCK_50,
    output [1:0] LEDG
);

    /* reg */
    reg data1 = 1'b1;
    reg [31:0] counter;
    reg state;

    /* assign */
    assign LEDG[0] = state;
    assign LEDG[1] = data1;

    /* always */
    always @ (posedge CLOCK_50) begin
 counter <= counter + 1;
 state <= counter[26]; // <--- data to change
    end

endmodule
```

- В разработке сложных электрических цепей (интегральных схем) используются языки описания аппаратуры
  - VHDL
    - «На основе» языка ADA
  - Verilog
    - «На основе» языка Си
  - …

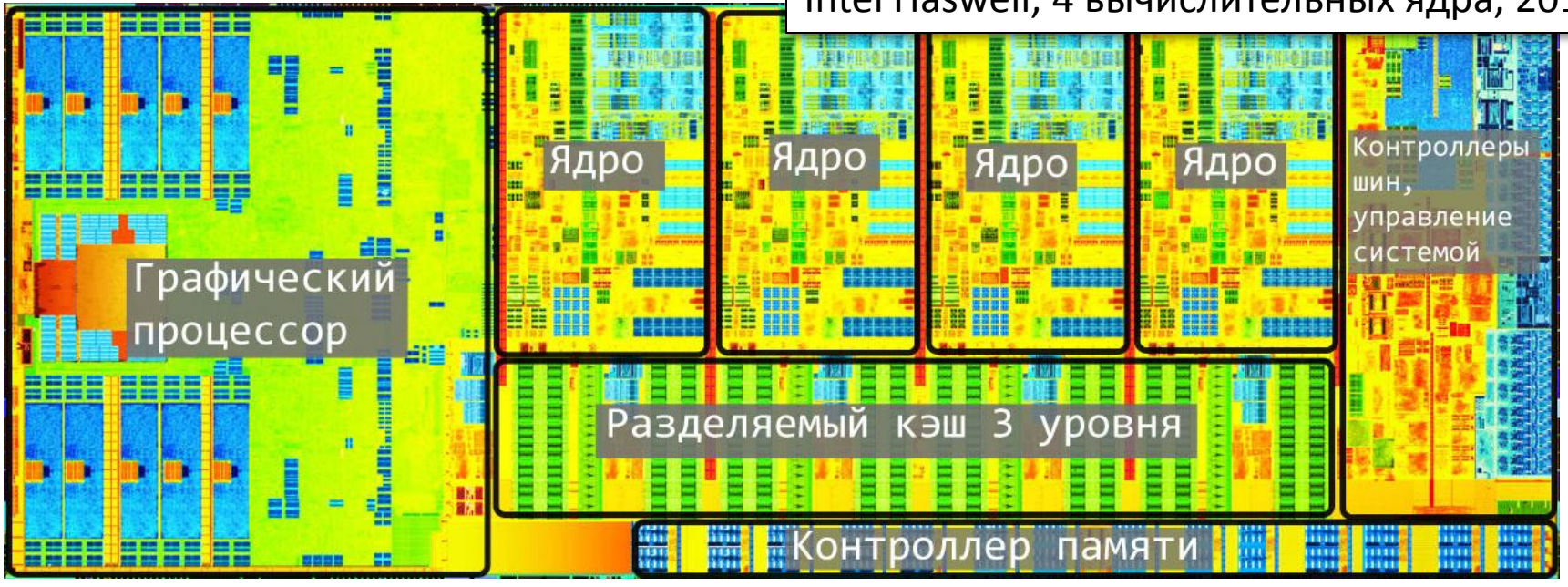
Этапы разработки и изготовления компонент ЭВМ



Закон Мура (Moore's law)

- Число транзисторов на кристалле будет удваиваться каждые 24 месяца
- Гипотеза выдвинута в 1965 году Гордоном Муром (один из основателей Intel)
- Ограничения
  - Атомарная природа вещества
  - Скорость света
- Негативная сторона – предельно быстрое устаревание вычислительной техники
- Открытый вопрос: область применимости

Intel Haswell, 4 вычислительных ядра, 2013

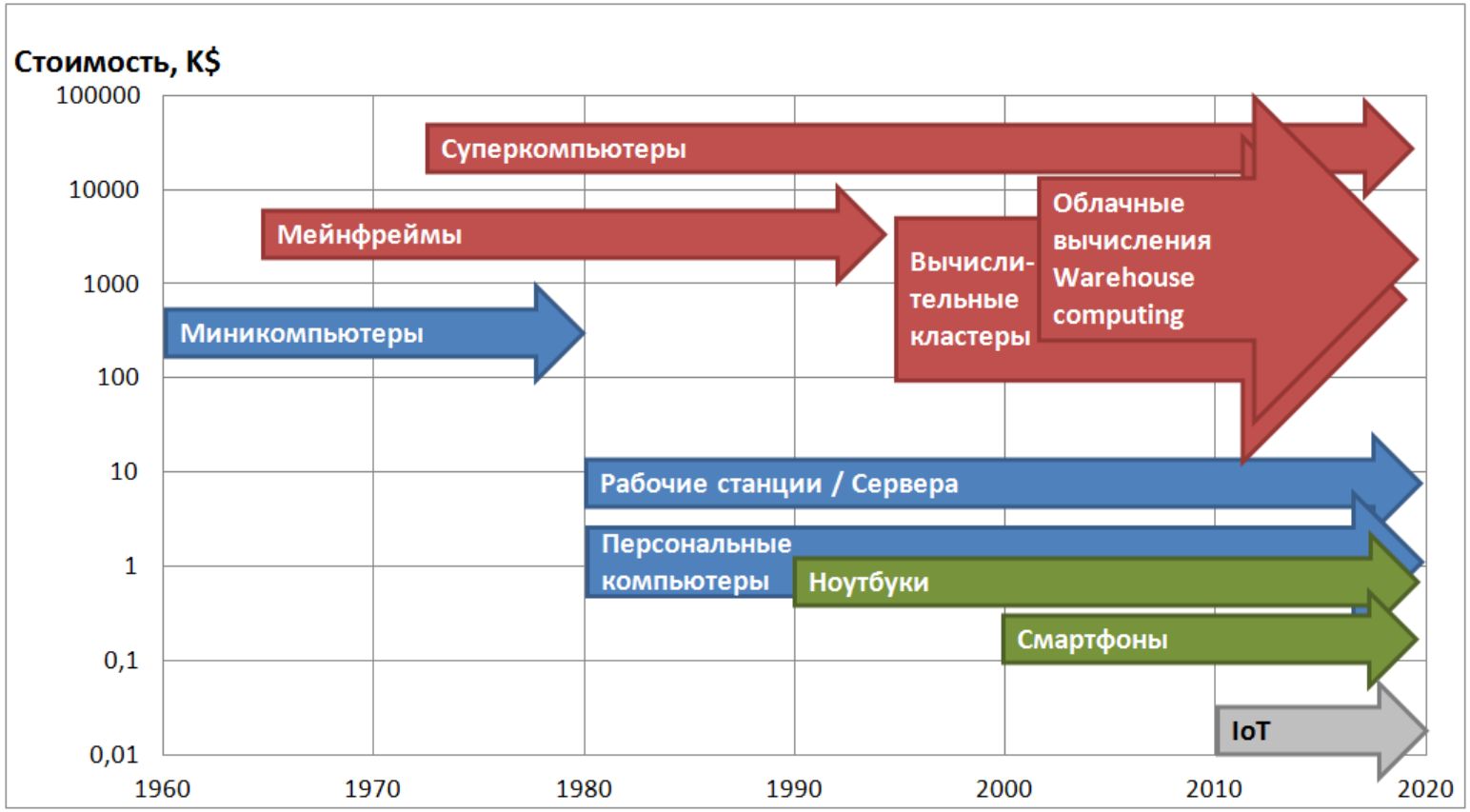


Закон Гроша (Grosch's law)

- Производительность компьютера увеличивается как квадрат стоимости
  - Емкость мирового рынка компьютеров – 5 машин
- Гипотеза выдвинута в 1965 году Хербом Грошем (второй ведущий ученый IBM, после Эккерта)
- 1997: закон полностью опровергнут
- Применимость к определенному классу машин
  - Рабочая станция
  - Майнфрейм
  - Суперкомпьютер
- Новые вычислительные/информационные ресурсы
  - Поисковые системы
  - Облачные вычисления

Закон Белла (Bell's law)

- Примерно каждое десятилетие появляется новый, более дешевый класс компьютеров, основанный на новой программной платформе, сетях и интерфейсах, в результате чего возникают новые отрасли индустрии и области применения компьютеров
- Сформулирован в 1972 Гордоном Беллом (разработчик компьютеров PDP в компании DEC)



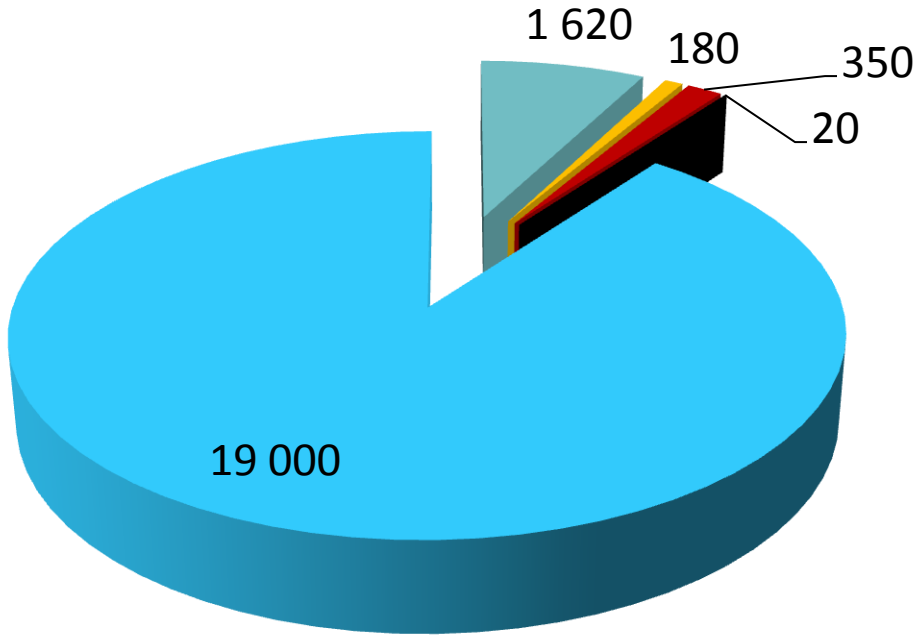
Различные классы компьютеров

RFID $0.05

Характе-ристика	Персональ-ные мобиль-ные устройства	Настольные компьютеры	Серверы	Кластеры / ЦОД	Встраива-емые системы
Цена всей системы	$100 - $1000	$300 - $2500	$5000 - $10000000	$100000 - $200000000	$10 - $100000
Цена процессора	$10 - $100	$50 - $500	$200 - $2000	$50 - $250	$0.01-$100
Критические требования	Стоимость, энергопо-требление , производи-тельность при работе с аудио/видео, быстрота реакции	Цена/произ-водитель-ность, энергопо-требление, производи-тельность графики	Пропускная способность, готовность, масштаби-руемость, энергопо-требление	Цена/произ-водитель-ность, пропускная способность, энергети-чекая пропорци-ональность	Цена, энергопо-требление , производи-тельность при решении прикладных задач

Различные классы компьютеров: кого больше?

Мировые продажи в 2010 году (млн. шт.)



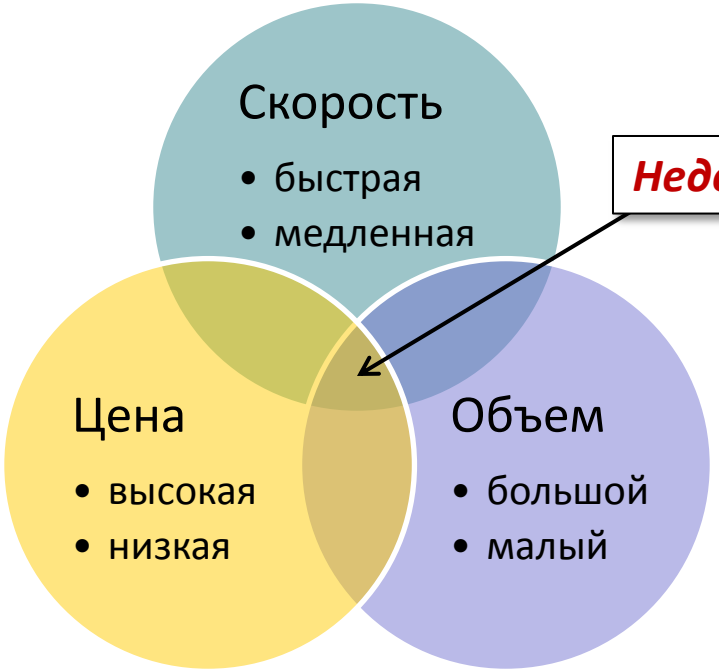
Процессоры архитектуры ARM:
отгружено с заводов 6.1 миллиарда шт.

Оперативная память (RAM)

- Основные свойства
  - RAM традиционно оформляется в виде отдельного чипа.
  - Единица хранения – клетка/ячейка (один бит на ячейку).
  - Оперативная память состоит из нескольких чипов RAM.
- Статическая память (SRAM)
  - Каждая ячейка хранит значение одного бита с помощью схемы из 4 или 6 транзисторов.
  - При наличии питания, сохраняет значение неограниченно долго.
  - Относительно устойчива к радиации, ЭМП
  - Быстрее и дороже чем DRAM.
- Динамическая память (DRAM)
  - Состоит из конденсатора и транзистора.
  - Сохраняемое значение должно обновляться каждые 10-100 мс.
  - Более чувствительная к воздействиям (ЭМП, радиация,…) чем SRAM.
  - Медленней и дешевле чем SRAM.

SRAM vs DRAM

Тип памяти	Транз. на 1 бит	Относ. время доступа	Устойчивая	Контроль	Относ. стоимость	Применение
SRAM	4 или 6	1×	Да	Нет	100×	Кэш
DRAM	1	10×	Нет	Да	1×	Основная оперативная память

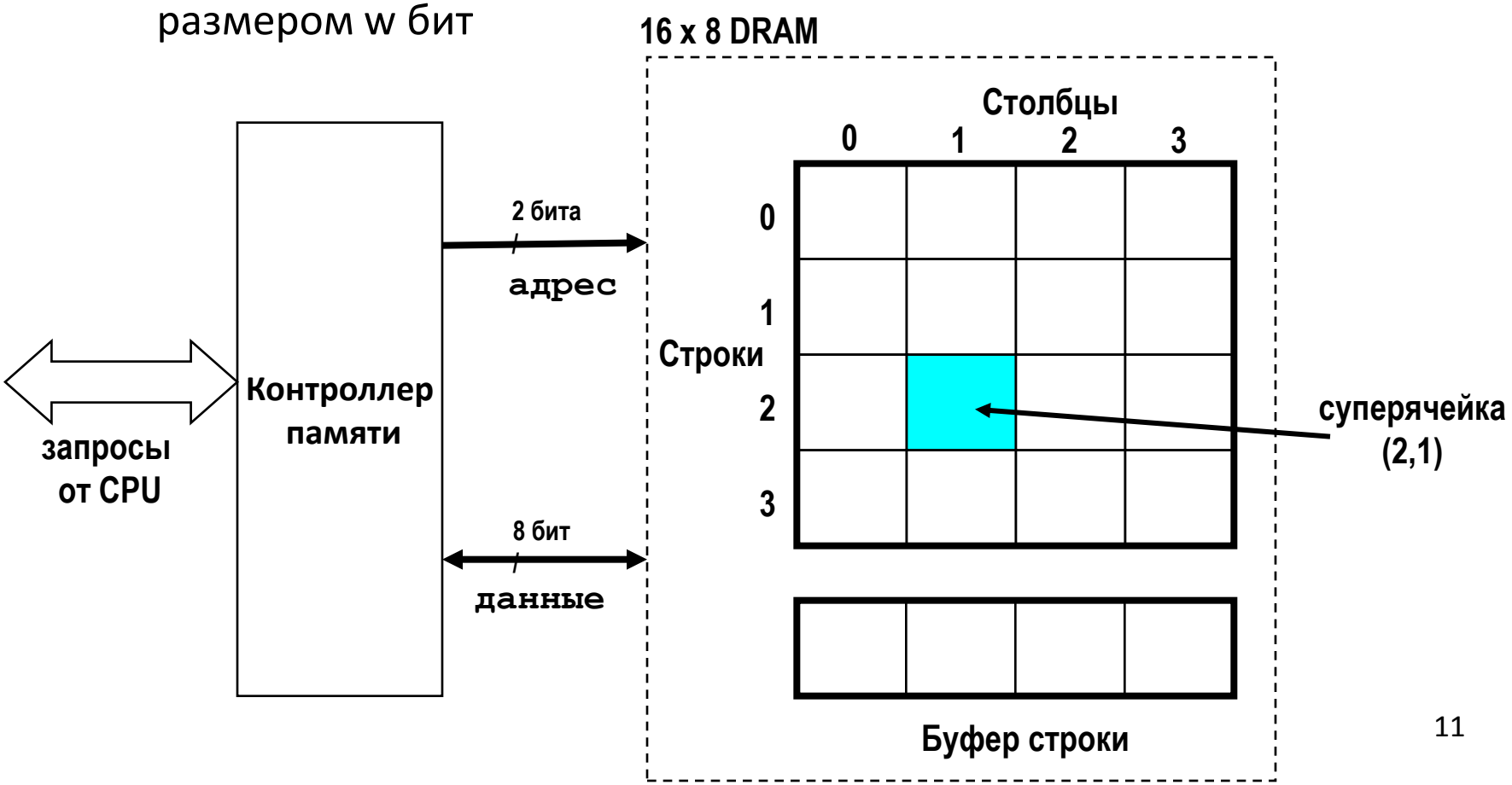


Недостижимо

Можно выбрать преимущества только двух характеристик в ущерб третьей

Типовая организация DRAM

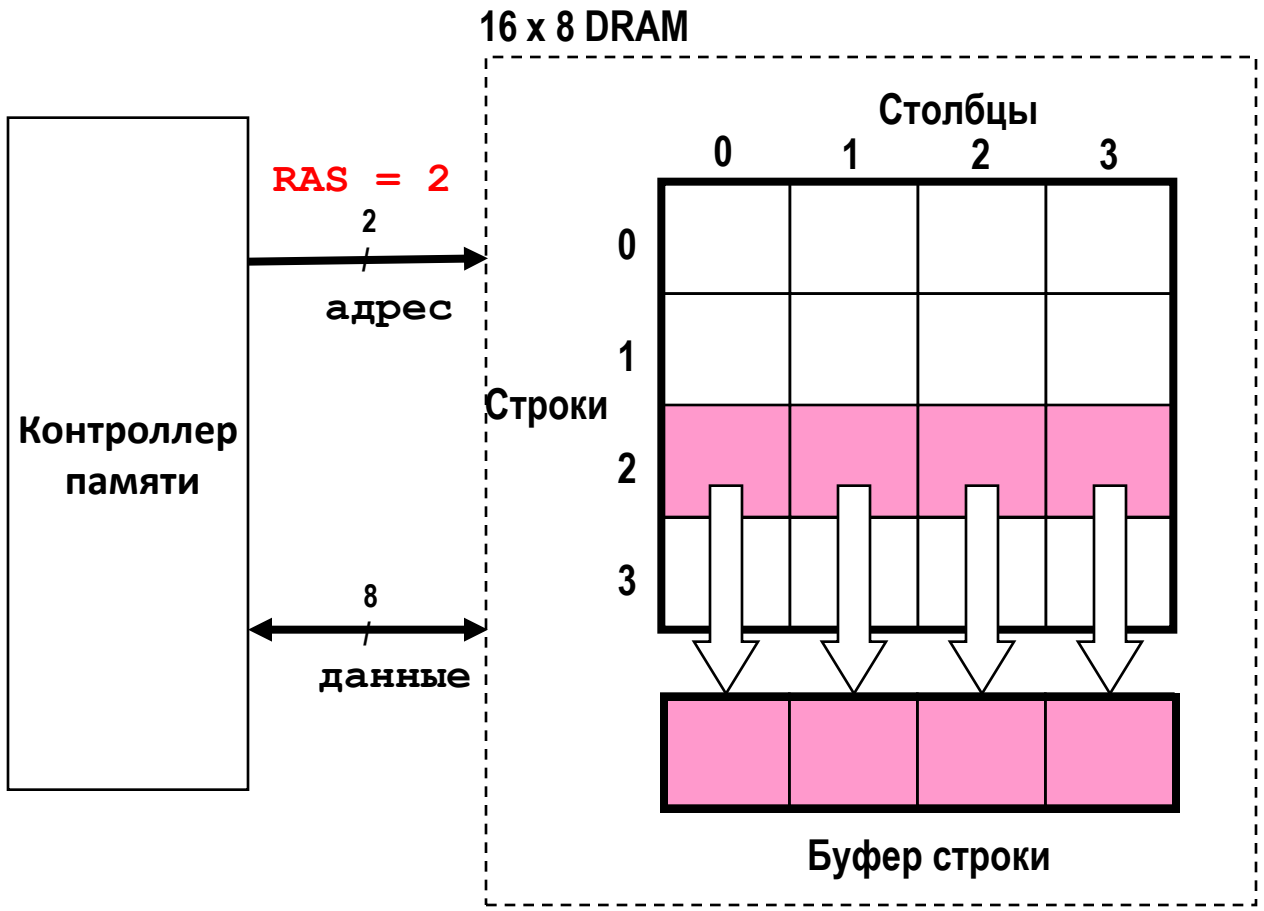
- d × w DRAM:
  - Общий объем данных dw бит организован как d суперячеек размером w бит



Чтение суперячейки DRAM (2,1)

Шаг 1(а): Строб адреса строки (RAS) указывает строку 2.

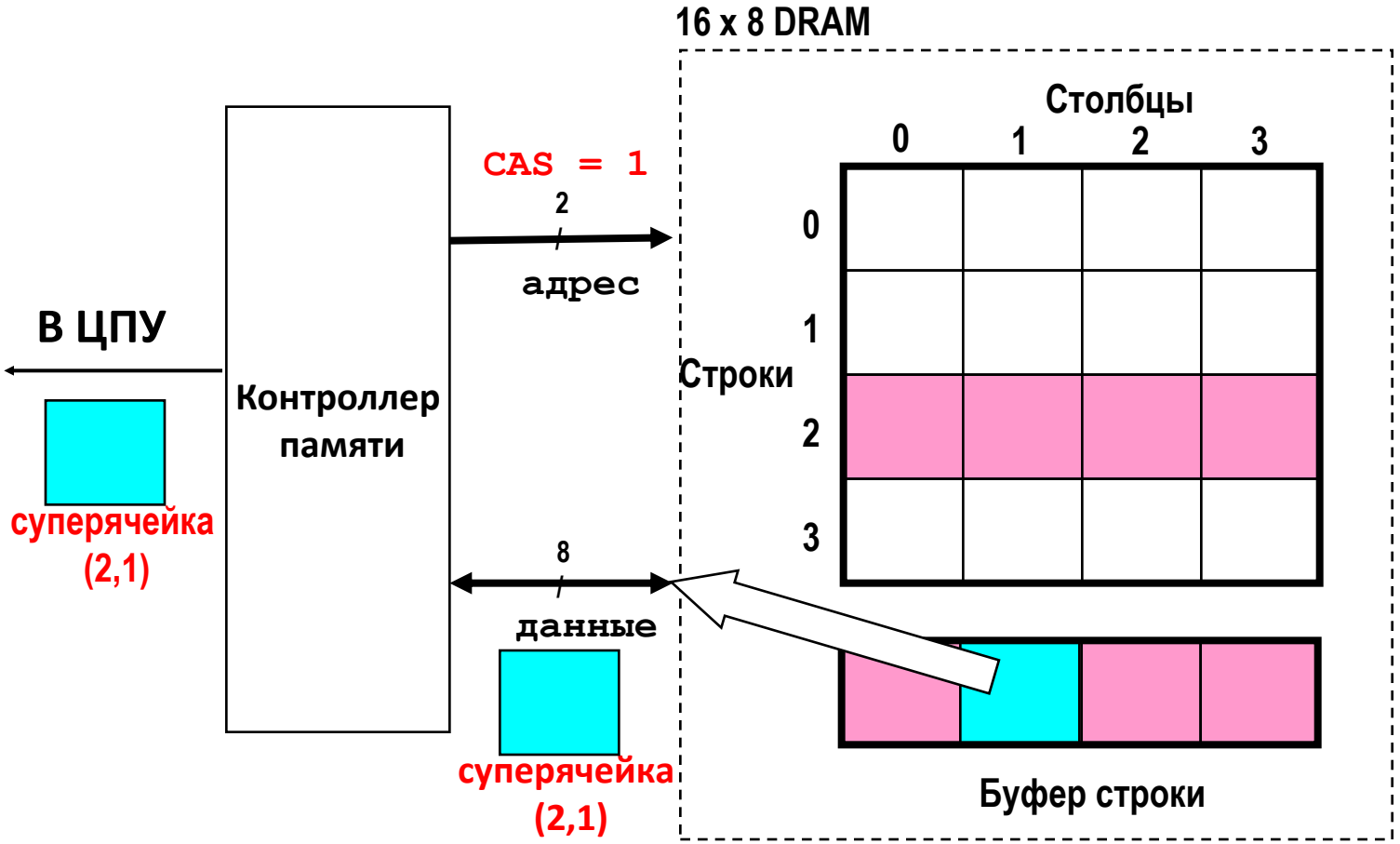
Шаг 1(б): Строка 2 копируется из DRAM в буфер строки.



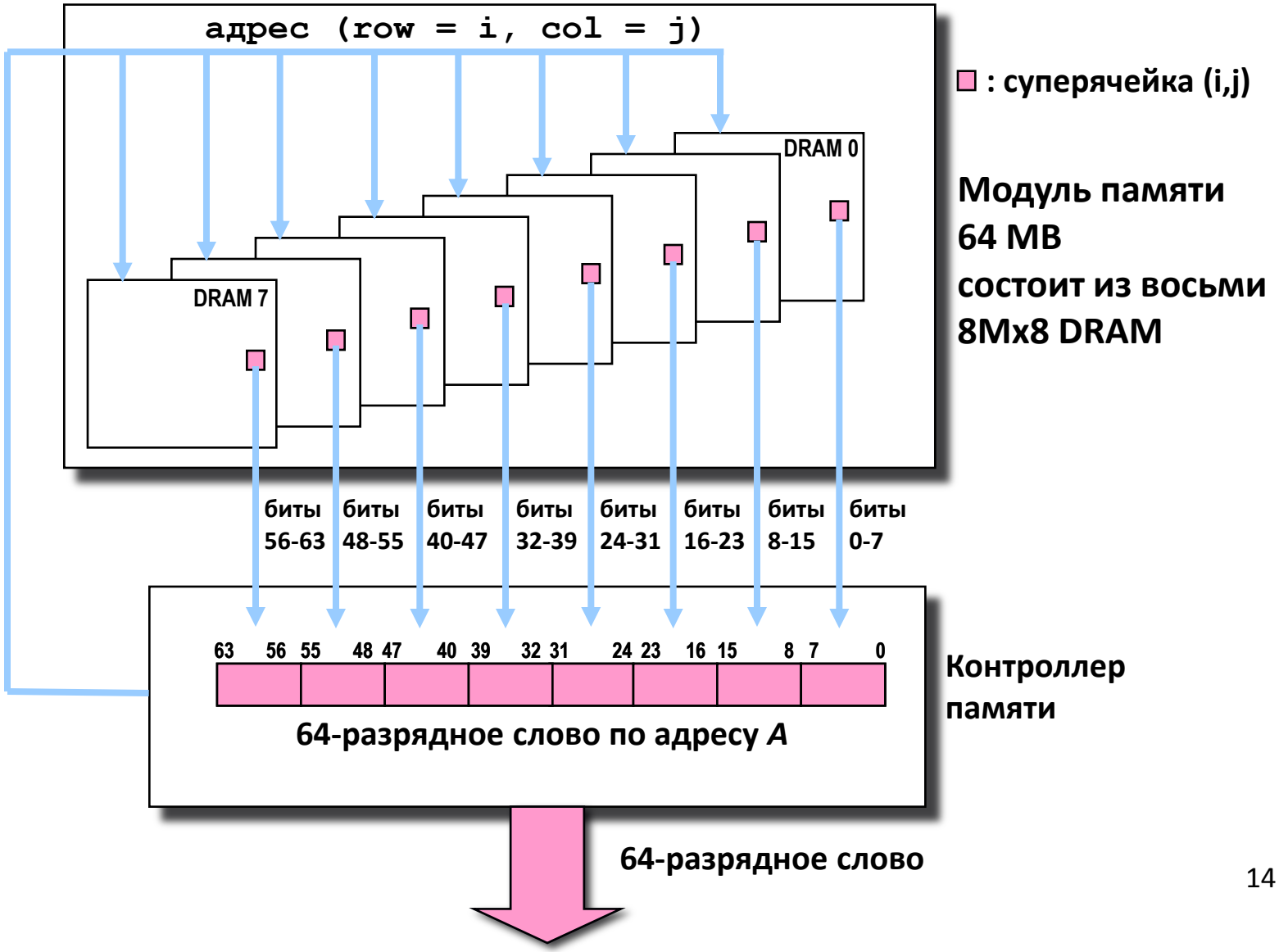
Чтение суперячейки DRAM (2,1)

Шаг 2(а): Строб адреса столбца (CAS) указывает столбец 1.

Шаг 2(б): Суперячейка (2,1) копируется из буфера на линии шины данных и далее в ЦПУ.



Расслоение памяти



: суперячейка (i,j)

Модуль памяти
64 MB
состоит из восьми
8Mx8 DRAM

Контроллер памяти

Развитие DRAM

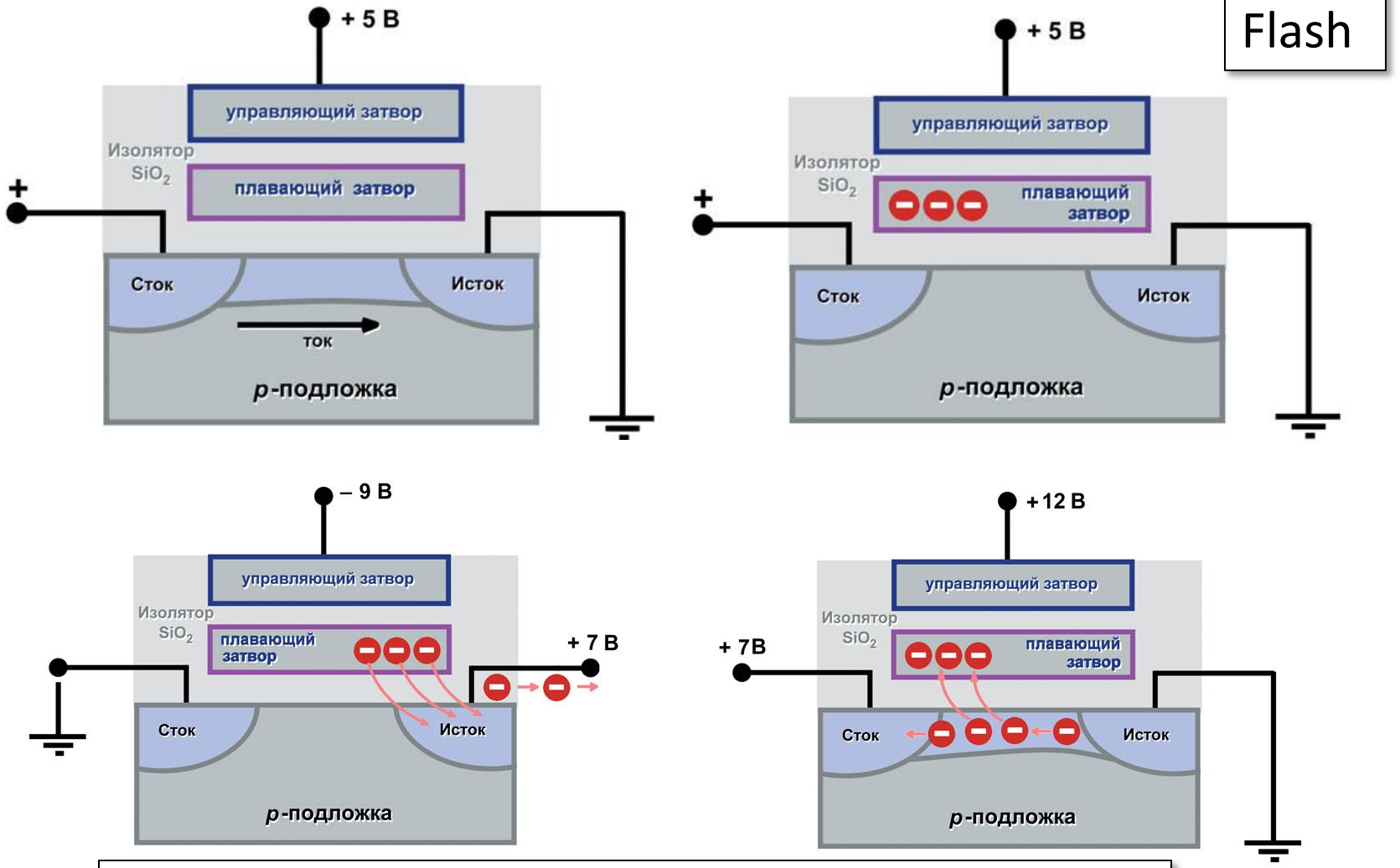
- Организация ячейки DRAM принципиально не менялась с момента изобретения в 1966 году.
  - Коммерческий выпуск начат Intel в 1970.
- Модули DRAM с улучшенным интерфейсом:
  - Синхронная DRAM (SDRAM)
    - Синхронизируется с системными часами
    - Позволяет повторно использовать адрес строки (т.е., RAS, CAS, CAS, CAS)
  - Синхронная DRAM с удвоенной частотой (DDR SDRAM)
    - Управляется фронтами – две посылки данных за один такт
    - В 2016 году, стандартная память для большинства серверов и настольных компьютеров
      - Intel Core (Skylake) поддерживает как минимум DDR3 SDRAM
      - Текущее поколение - DDR4 SDRAM, массовый выпуск с конца 2014 года

Энергонезависимая память

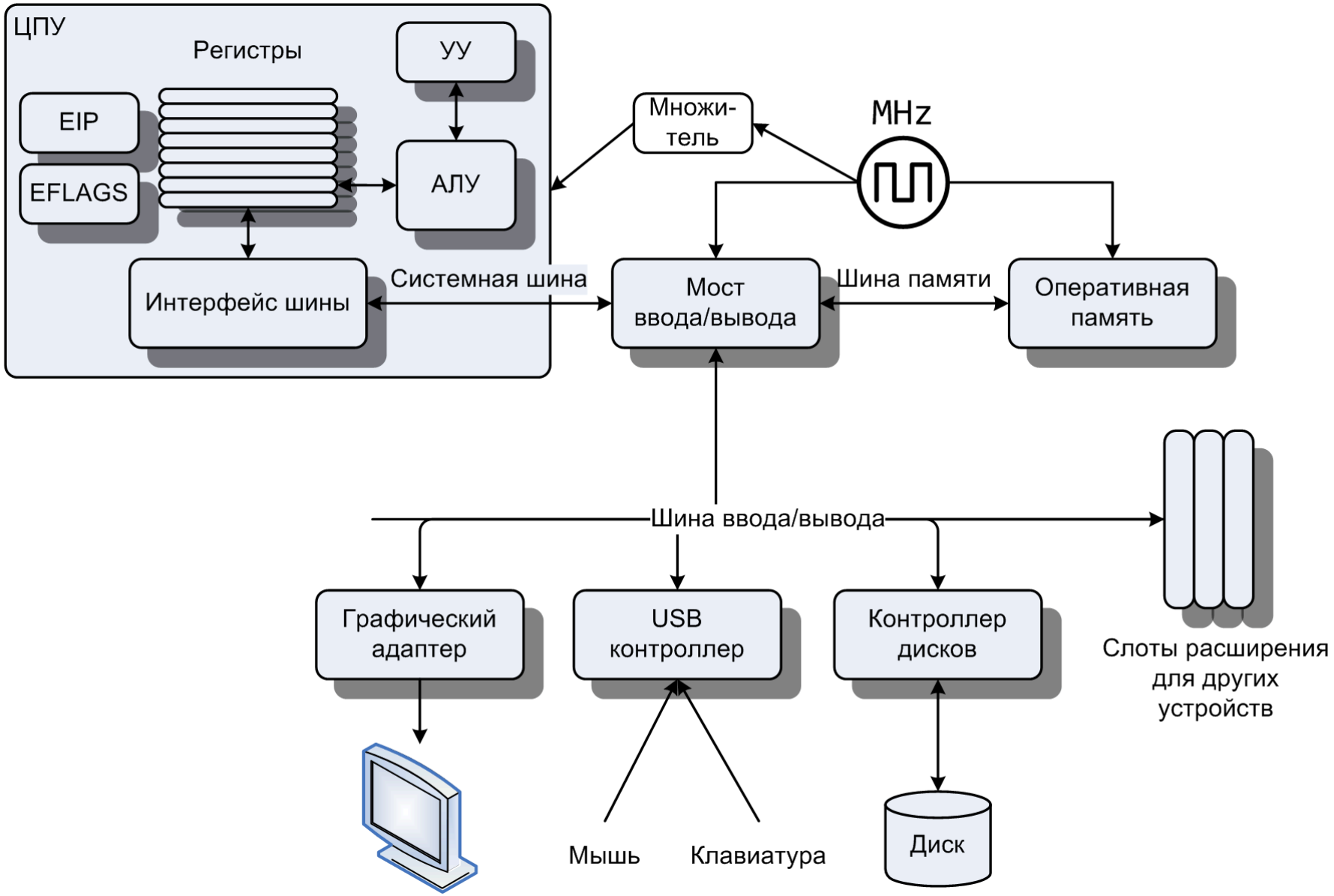
- DRAM и SRAM – энергозависимы
  - Информация теряется при отключении электропитания.
- Энергонезависимая память сохраняет свое состояние даже при отключении питания
  - ROM: программируется на этапе производства
  - Программируемая ROM (PROM): может быть запрограммирована пользователем один раз
  - Стираемая PROM (EPROM): может быть стерта (УФ, рентген)
  - Электрически стираемая PROM (EEPROM): стирание происходит через подачу электрического сигнала
  - Флеш-память: EEPROM с частичной возможностью стирания (по секторам)
    - Выдерживает порядка 100,000 циклов перезаписи.
- Сфера применения энергонезависимой памяти
  - Встраиваемые программы размещаются в ROM (BIOS, контроллеры дисков, сетевых и графических адаптеров, аппаратно-криптографические средства,…)
  - Твердотельные диски (заменяют обычные диски в переносных накопителях, смартфонах, плеерах, и т.д.)
  - Кеш в обычных дисковых накопителях.

Энергонезависимая память

Flash

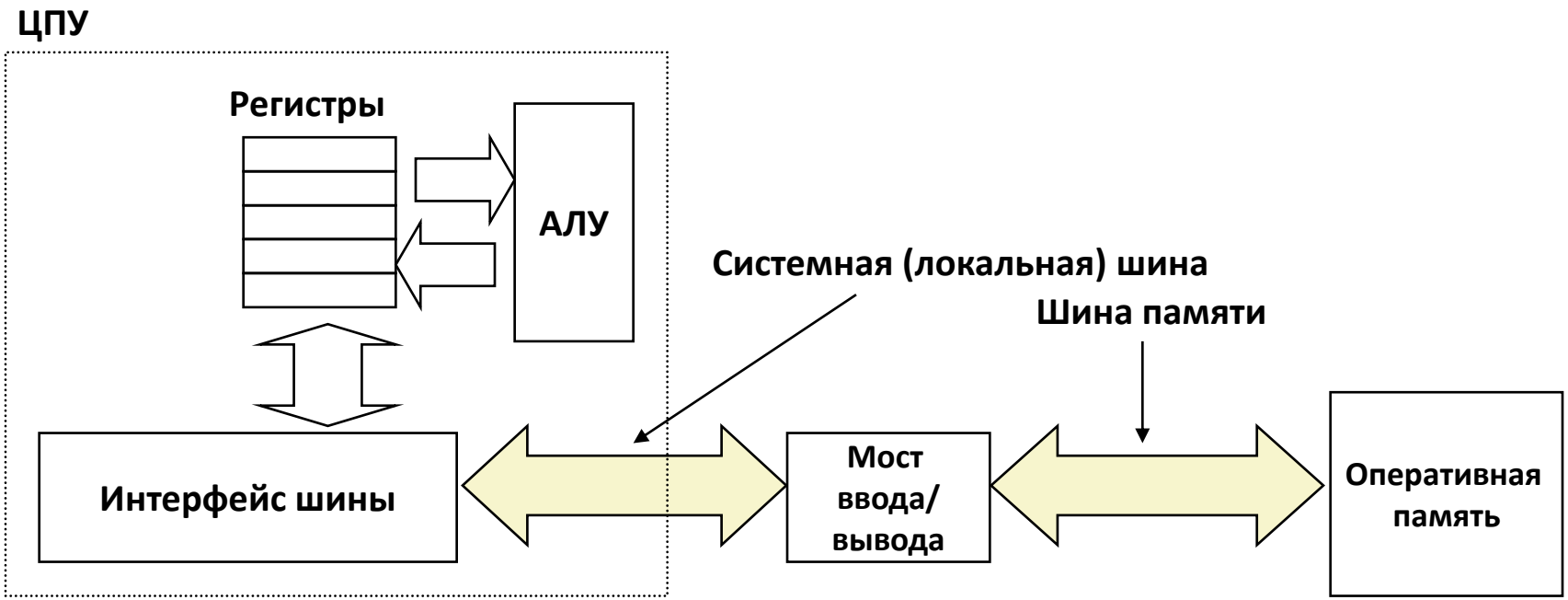


Для записи и стирания используется
квантовый эффект туннелирования Фаулера-Нордхейма



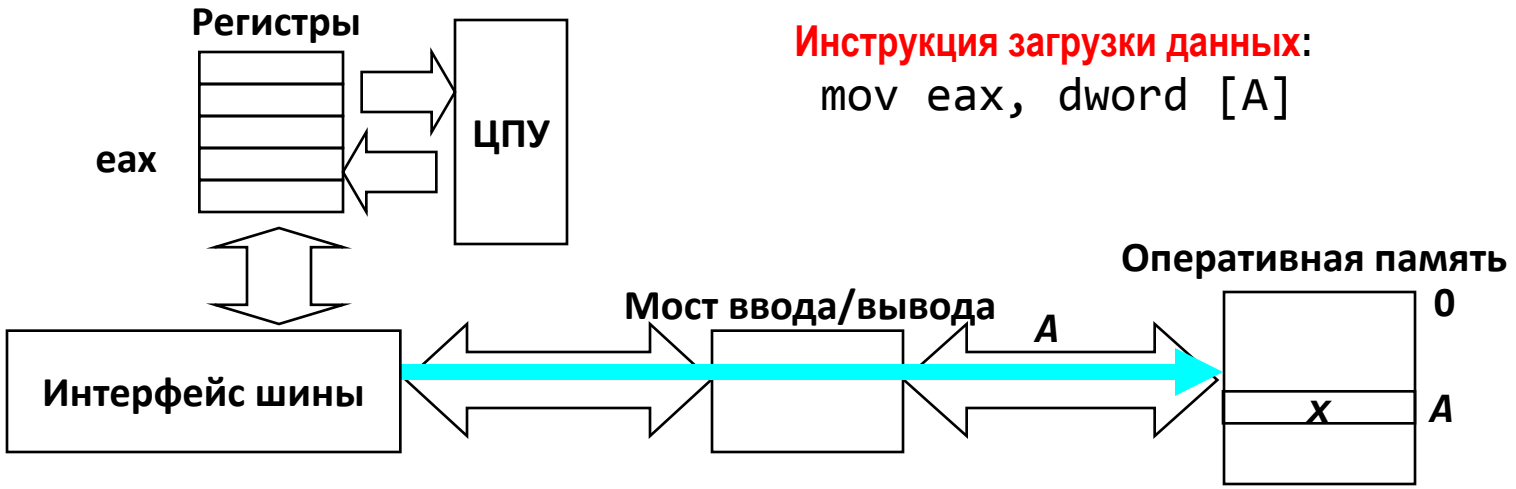
Типовое соединение ЦПУ и оперативной памяти

- Шина – набор проводов используемых для передачи данных, адресов, управляющих сигналов.
- Шины, как правило, используются несколькими устройствами.



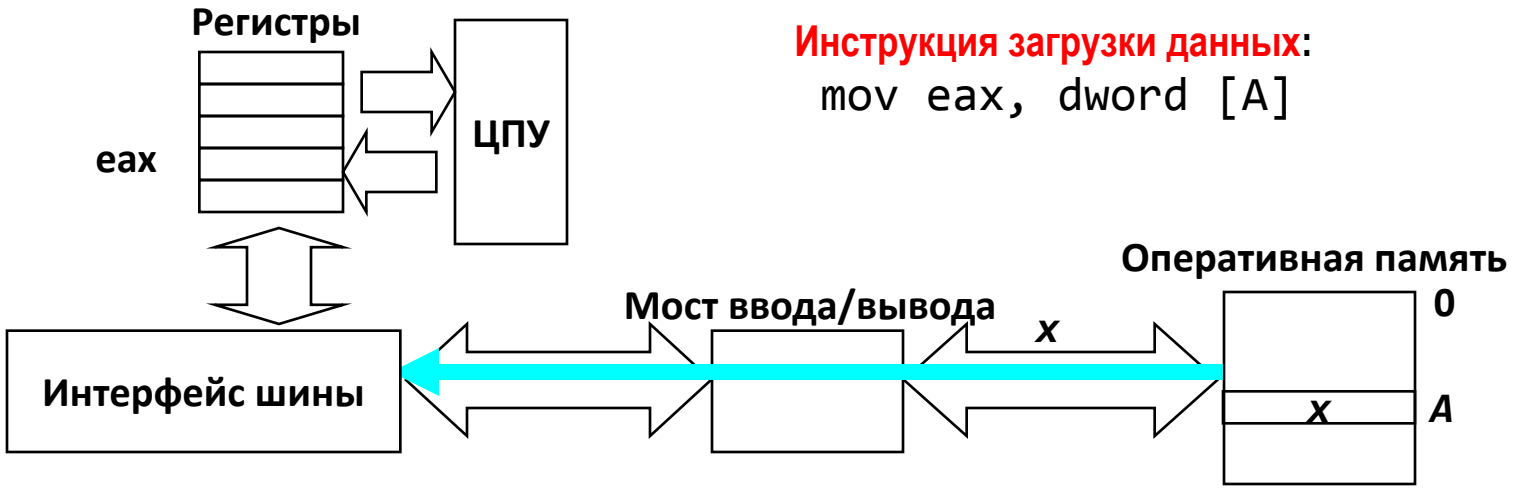
Чтение данных из памяти (1)

- ЦПУ передает адрес A интерфейсу шины памяти.



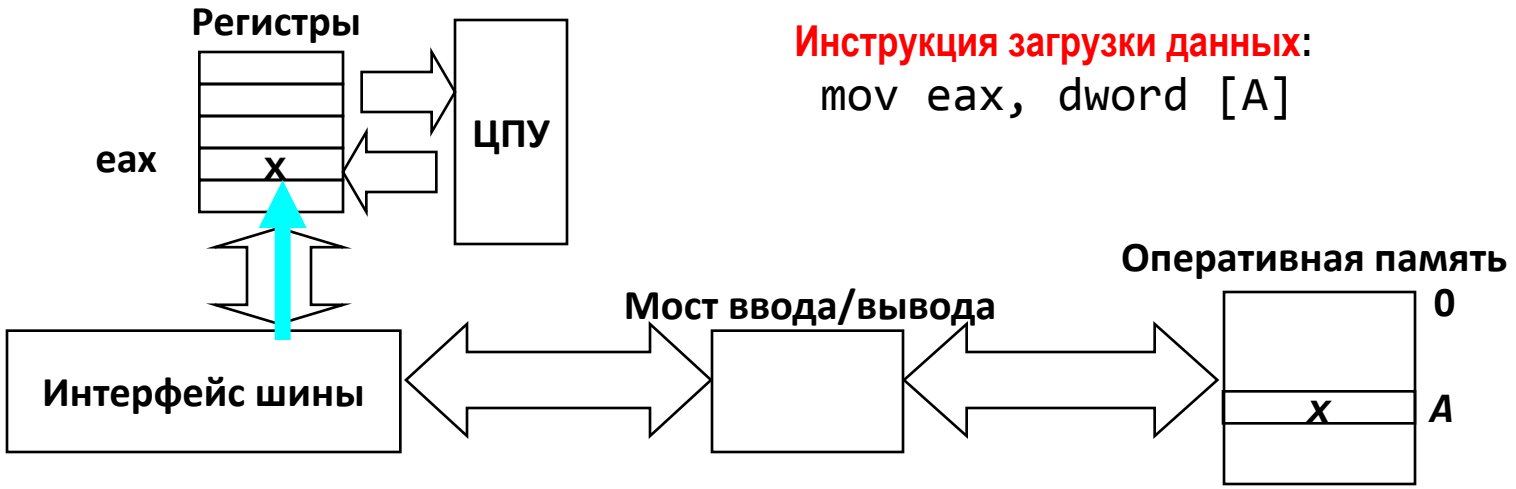
Чтение данных из памяти (2)

- Оперативная память получает запрос на выборку данных по адресу A из шины, осуществляет выборку значения x, и отправляет его назад, в шину.



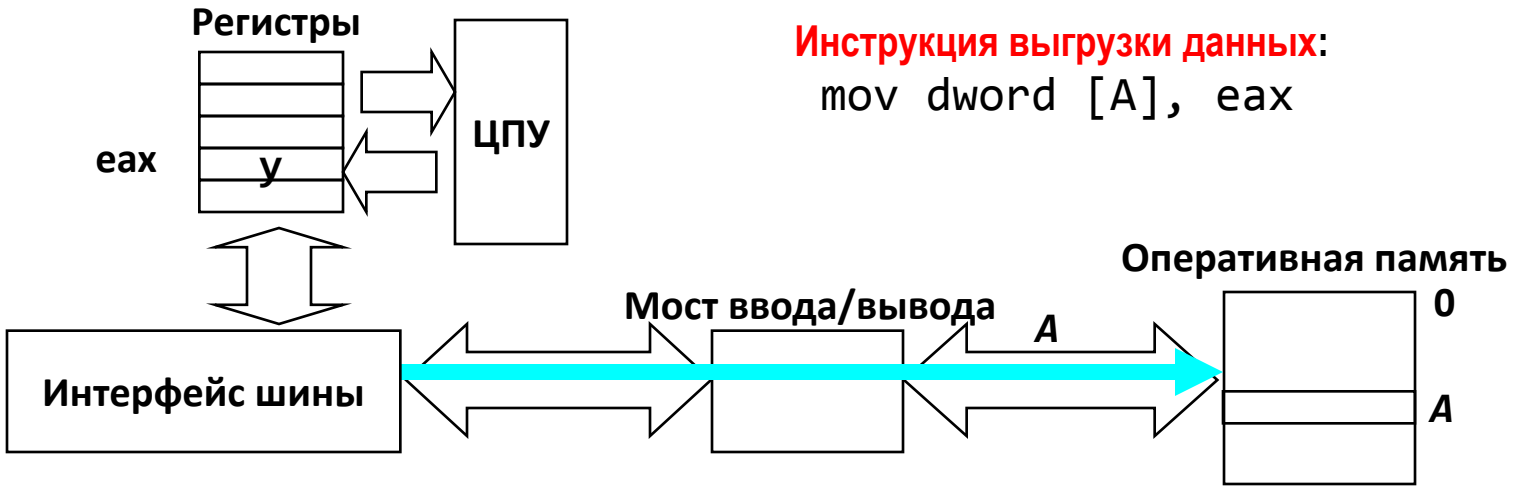
Чтение данных из памяти (3)

- ЦПУ считывает двойное слово x из шины и пересылает его в регистр eax.



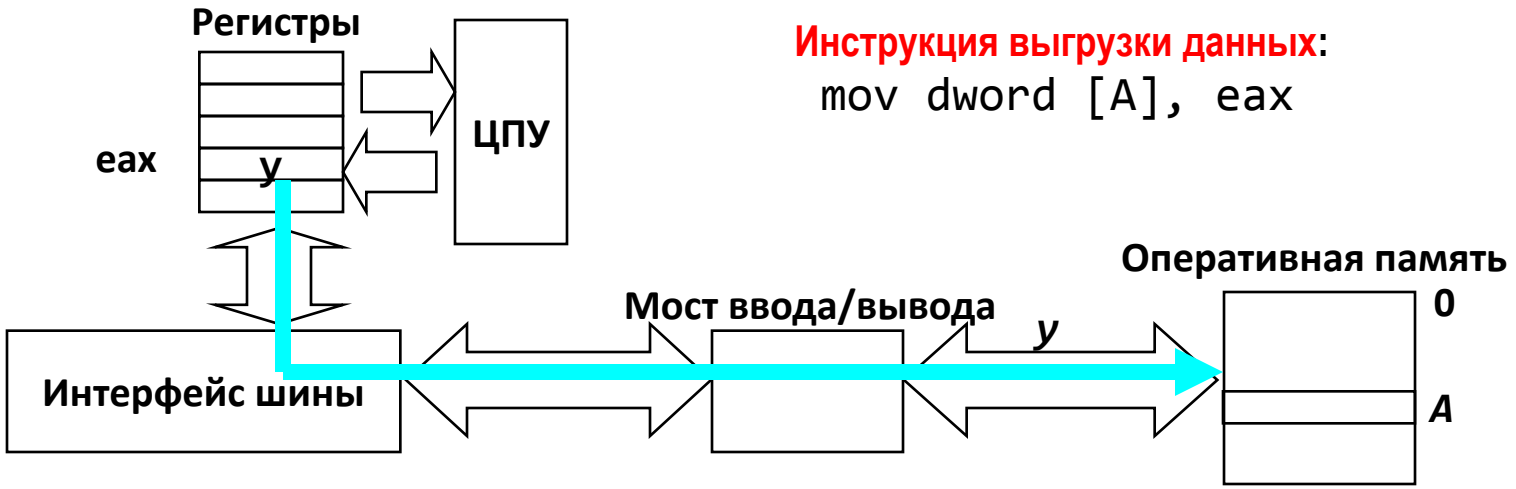
Запись данных в память (1)

- ЦПУ передает адрес A интерфейсу шины. Оперативная память считывает адрес и ждет посылки соответствующего значения.



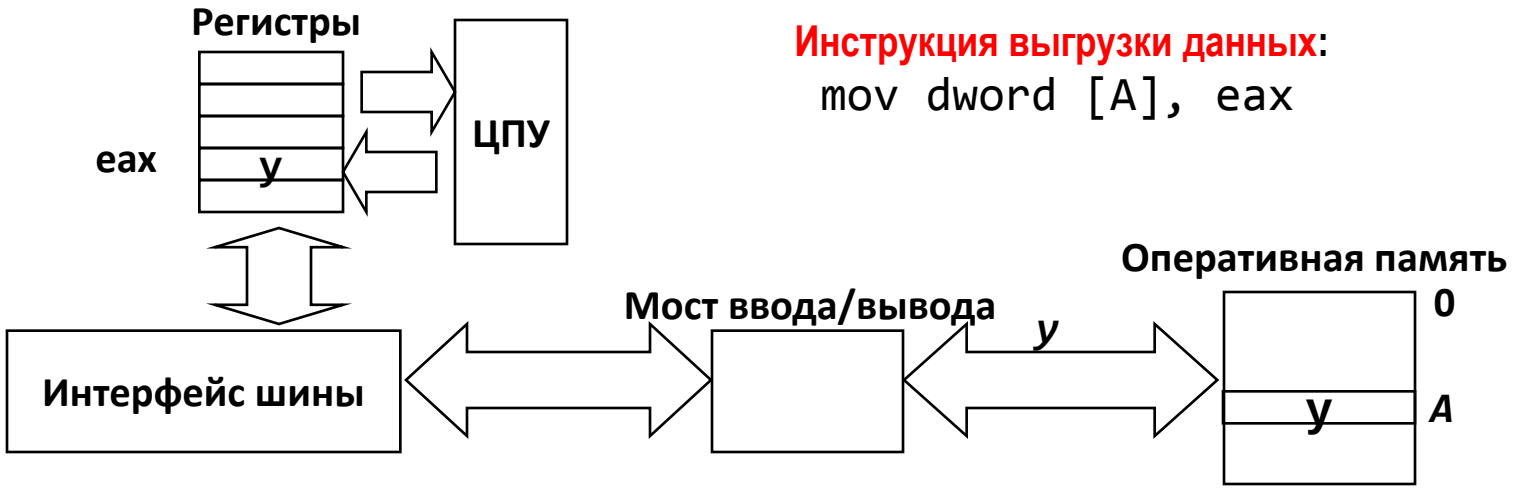
Запись данных в память (2)

- ЦПУ передает значение y интерфейсу шины.



Запись данных в память (3)

- Оперативная память получает двойное слово y из шины и сохраняет его по адресу A.



Port IO vs. Memory Mapped IO

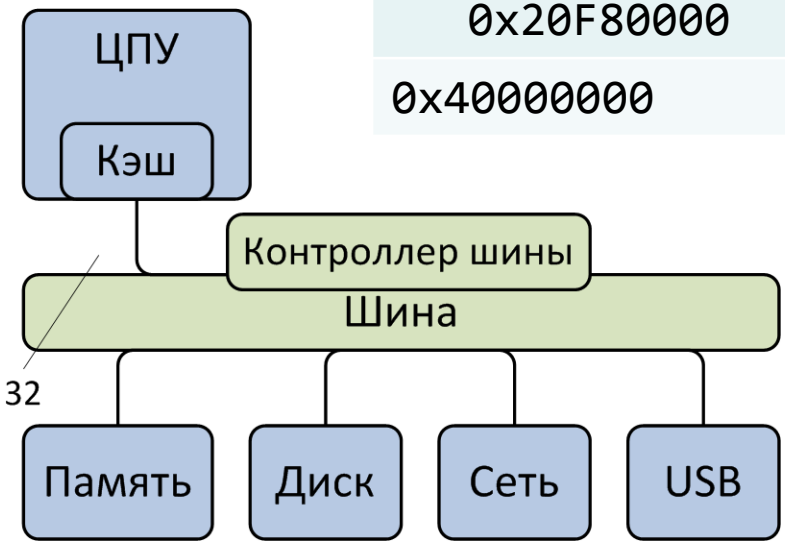
- Port IO: помимо пространства памяти вводится дополнительное пространство портов ввода/вывода
  - Работа с периферией: команды `in` и `out`
    - Все данные проходят через ЦПУ
  - Удобно при небольшом размере памяти

```
• IN EAX, DX ; AX, AL
    • EAX ← I/O[DX]
• OUT DX, EAX ; AX, AL
    • I/O[DX] ← EAX
```

- Memory Mapped IO: все управляющие регистры устройств отображаются на определенные адреса оперативной памяти
  - Требуется программировать контроллер памяти/северный мост
  - Перекрытая память не используется
  - Существенно более высокая производительность

Шина: точка зрения системного программиста

Базовый адрес	Размер	Описание
0x20000000	0x1000000	Flash
0x20000000	0x40000	Загрузчик
0x20040000	0xDC0000	Встроенное ПО
0x20A00000	0x100000	FFS
0x20F80000	0x40000	SECZONE
0x40000000	0x800000	RAM

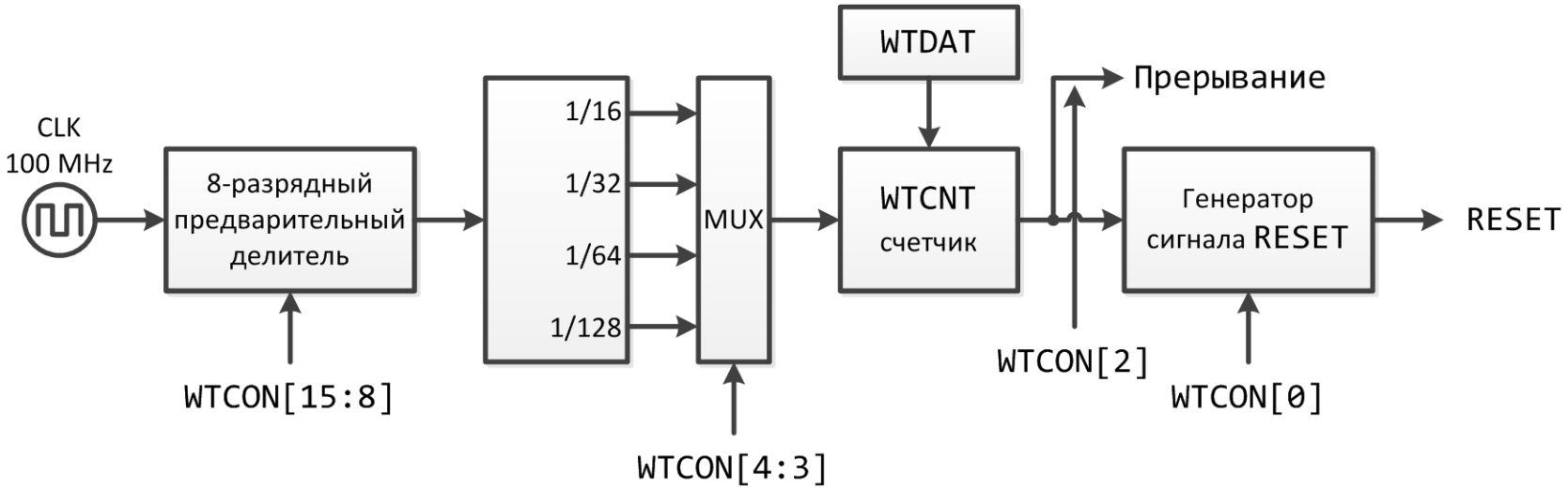


Шина – адресное пространство

```
int device_driver_transmit_data(device *dev,
                                phys_addr_t *buffer,
                                size_t buf_len)
   {
           offset_t offset = device->offset;
           uint32_t status;
           time_t time = 0;
           outl(offset + BUFFER_REGISTER, buffer);
           outl(offset + BUFLEN_REGISTER, buf_len);
           outl(offset + COMMAND_REGISTER, COMMAND_TRANSMIT);
           do {
                   wait(WAIT_INTERVAL);
                   time += WAIT_INTERVAL;
                   inl(&status, STATUS_REGISTER);
                   if ((status & ERROR_MASK) || (time >= TIMEOUT))
                           goto error;
           } while (!(status & COMPLETE_MASK));
           return 0;
           error:
           return -1;
   }
```

```
void outl(int* m_reg, int val) {
   *m_reg = val;
}
```

Пример простейшего периферийного устройства: WatchDog @ SoC Exynos4210



Базовый адрес `0x10060000`

Регистр	Смещение	Описание
WTCON	0x0	Управляющий регистр
WTDAT	0x4	Начальное значение счетчика
WTCNT	0x8	Счетчик таймера

Запуск устройства
```
WTCON[0] = 1
WTCON[5] = 1
```