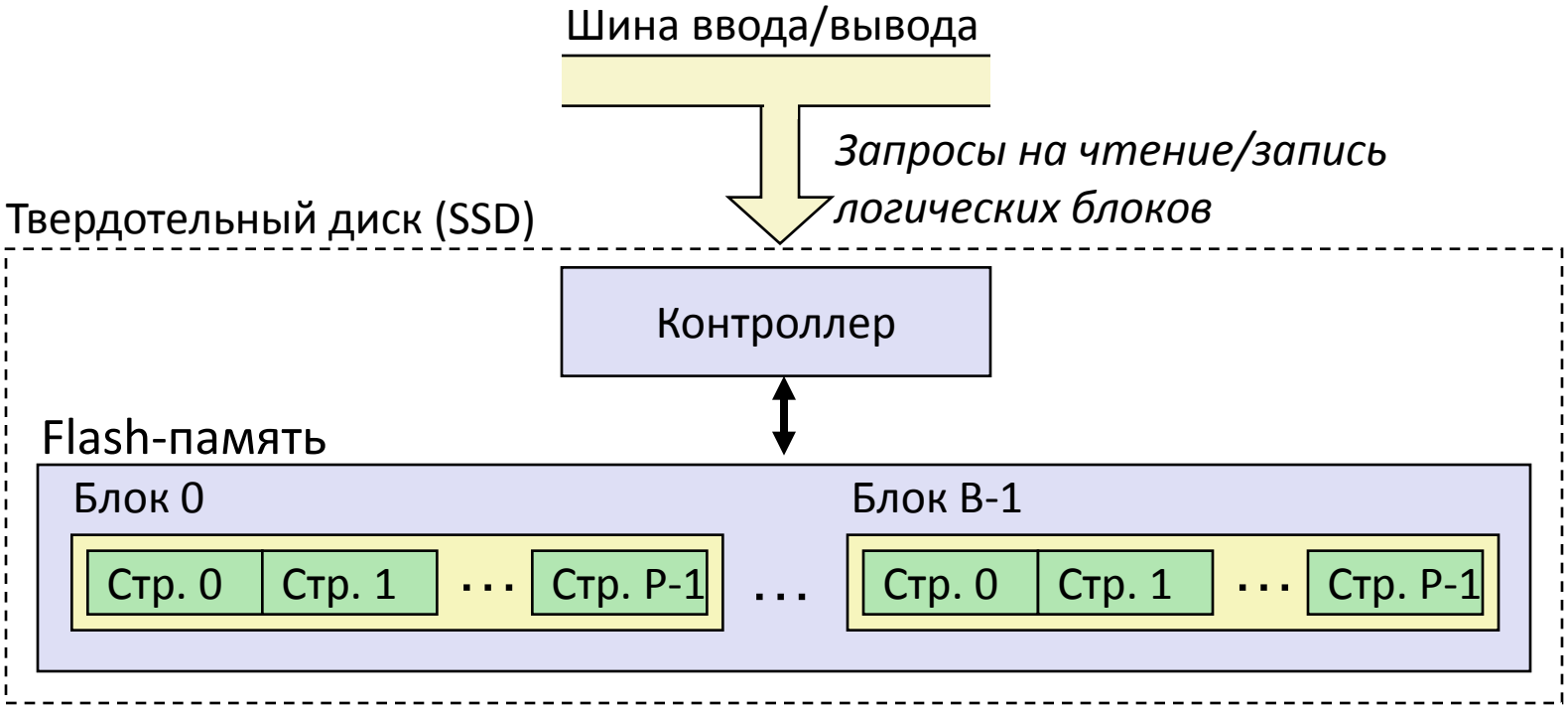
© 2018 МГУ/ВМК/СП

Лекция 0x16

25 апреля

© 2018 МГУ/ВМК/СП

Твердотельные диски (SSD)



- Размер
  - страницы: 512 – 4096 байт
  - блоки: 32 – 128 страниц
- Данные читаются/пишутся целыми страницами.
- Стирание данных – весь блок.
- Блок в некоторый момент вырабатывается
  - 100,000 перезаписей и более

© 2018 МГУ/ВМК/СП

Производительность SSD

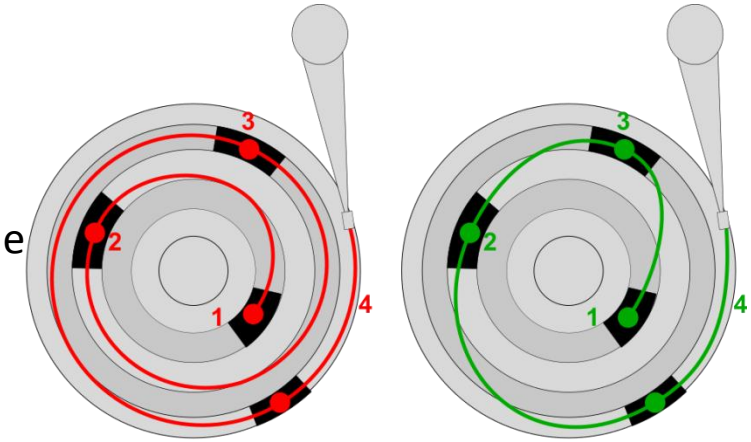
Последовательное чтение	250 МБ/с	Последовательная запись	170 MB/s
Произвольное чтение	140 МБ/с	Произвольная запись	14 MB/s
Время доступа	30 мкс	Время старта записи 300 мкс	

- Причины малой скорости произвольной записи
  - Высокая длительность стирания блока (≈1 мс.)
  - Запись одной страницы вызывает копирование всех остальных страниц, расположенных в данном блоке
    - Выделить (найти) новый блок и стереть его содержимое
    - Записать станицу в новый блок
    - Скопировать остальные страницы из исходного блока

© 2018 МГУ/ВМК/СП

SSD vs. обычные диски

- Преимущества
  - Нет движущихся частей → более быстрые, меньшее энергопотребление, устойчивы к внешним воздействиям
- Недостатки
  - Ограниченное количество перезаписей
    - Контроллер стремится смягчить износ, распределяя перезаписи
    - Пример: гарантируют возможность произвольной записи 1 петабайта (10^{15} байт) данных до момента выработки
  - Высокая стоимость относительно HDD
    - В 2010 году, на два порядка, в 2015 году, на один порядок
- Применение
  - Изначально: MP3 плееры, смартфоны, лэптопы
  - Закрепились на персоналках и серверах в качестве «быстрого» диска
- Подключение к шине PCI
  - AHCI, NCQ (Native Command Queuing)
  - NVMe, 2011 – непосредственное подключение SSD-диска к шине, как PCI-устройства



без поддержки NCQ NCQ

© 2018 МГУ/ВМК/СП

Тенденции в развитии запоминающих устройств

SRAM

Метрика	1980	1985	1990	1995	2000	2005	2010	*2010:1980*
$/МБ	19,200	2,900	320	256	100	75	60	*320*
t доступа (нс)	300	150	35	15	3	2	1.5	*200*

DRAM

Метрика	1980	1985	1990	1995	2000	2005	2010	*2010:1980*
$/МБ	8,000	880	100	30	1	0.1	0.06	*130,000*
t доступа (нс)	375	200	100	70	60	50	40	*9*
Размер (МБ)	0.064	0.256	4	16	64	2,000	8,000	*125,000*

HDD

Метрика	1980	1985	1990	1995	2000	2005	2010	*2010:1980*
$/МБ	500	100	8	0.30	0.01	0.005	0.0003	*1,600,000*
t доступа (мс)	87	75	28	10	8	*4*	*3*	*29*
Размер (МБ)	1	10	160	1,000	20,000	160,000	1,500,000	*1,500,000*

© 2018 МГУ/ВМК/СП

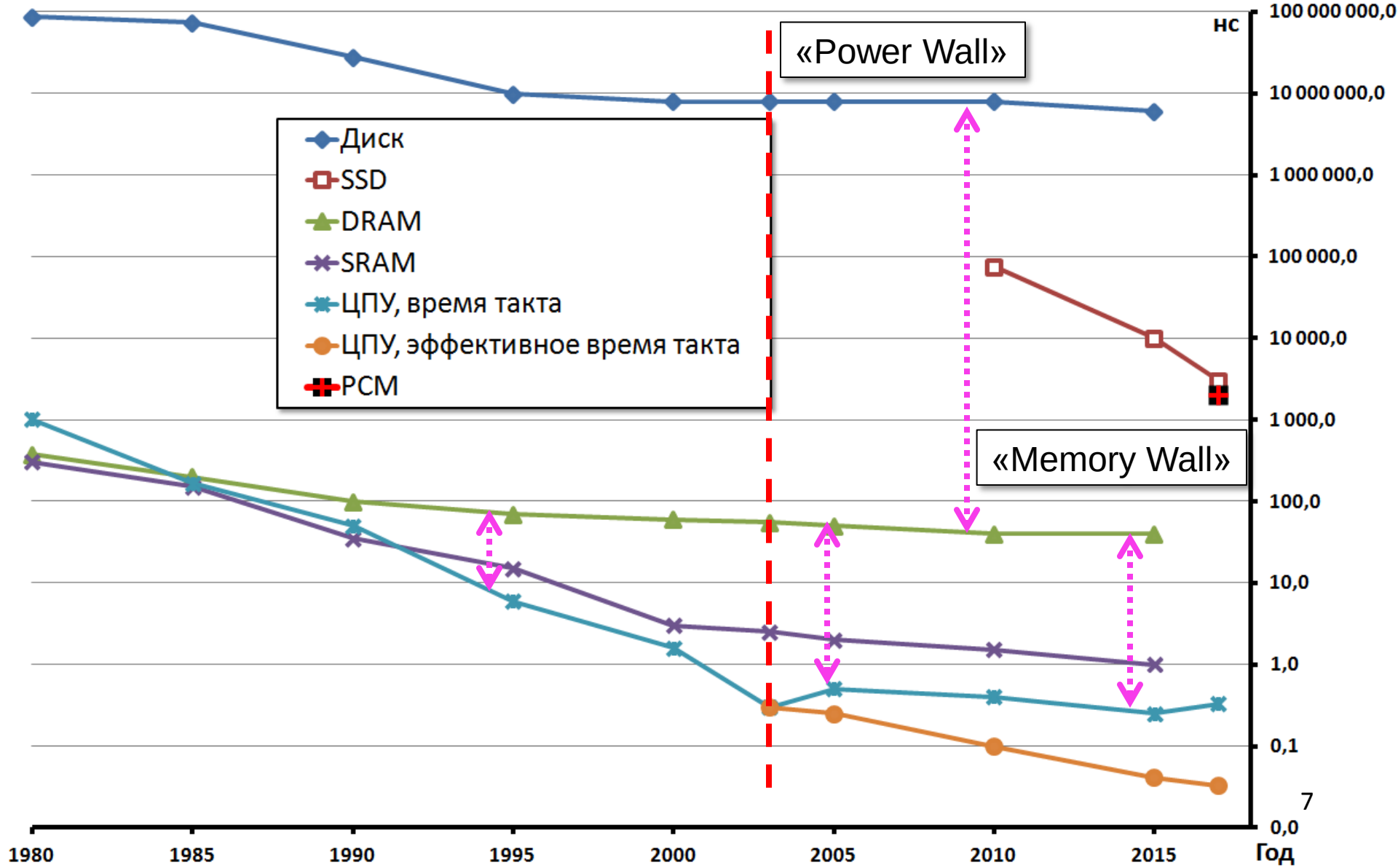
Частота ЦПУ

Разработчики аппаратуры столкнулись с “Power Wall”

	1980	1990	1995	2000	2003	2005	2010	2015	2015 ÷ 1980
ЦПУ	8086	80386	Pentium	P-III	P-4	Core 2	Core i7 Nehalem	Core i7 Haswell	
Частота (МГц)	1	20	150	600	3300	2000	2500	3000	3000
Длительность такта (нс.)	1000	50	6	1.6	0.3	0.5	0.4	0.25	3000
Количество ядер	1	1	1	1	1	2	4	8	8
Эффективная длительность такта (нс)	1000	50	6	1.6	0.3	0.25	0.1	0.04125	~24250

© 2018 МГУ/ВМК/СП

Разрыв между ЦПУ и памятью

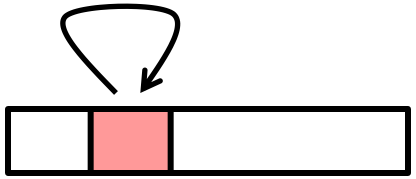


© 2018 МГУ/ВМК/СП

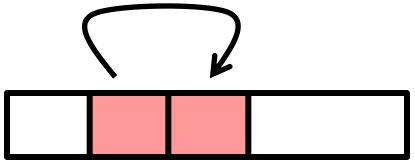
Локальность

- Основной принцип локальности: программа стремится использовать данные и инструкции с адресами близкими (либо точно такими же) к тем, которые использовались ранее.

- Временная локальность:
  - Повторные обращения



- Пространственная локальность:
  - В некоторый малый промежуток времени используются ячейки памяти с близкими адресами



© 2018 МГУ/ВМК/СП

Пример

```
sum = 0;
for (i = 0; i < n; i++)
    sum += a[i];
return sum;
```

- Выборка данных
  - Последовательные обращения к элементам массива. **Пространственная локальность**
  - Переменная sum используется на каждой итерации. **Временная локальность**
- Выборка инструкций
  - Последовательная выборка инструкции. **Пространственная локальность**
  - Повторное выполнение инструкций в цикле. **Временная локальность**

© 2018 МГУ/ВМК/СП

Оценка качества локальности

- Утверждение: способность беглым взглядом определить характер локальности кода является одним из необходимых навыков профессионального программиста.

- Вопрос: Достигается ли в функции `sum_array_rows` локальность обращений к массиву `a`?

```
int sum_array_rows(int a[M][N]) {
    int i, j, sum = 0;

    for (i = 0; i < M; i++)
        for (j = 0; j < N; j++)
            sum += a[i][j];
    return sum;
}
```

© 2018 МГУ/ВМК/СП

Еще один пример

- Вопрос: достигается ли в функции `sum_array_cols` локальность обращений к массиву `a`?

```
int sum_array_cols(int a[M][N]) {
    int i, j, sum = 0;

    for (j = 0; j < N; j++)
        for (i = 0; i < M; i++)
            sum += a[i][j];
    return sum;
}
```

© 2018 МГУ/ВМК/СП

Третий пример

- Вопрос: Как преобразовать гнездо циклов, что бы проход по 3-мерному массиву выполнялся с шагом 1 (и т.о. достичь пространственной локальности)?

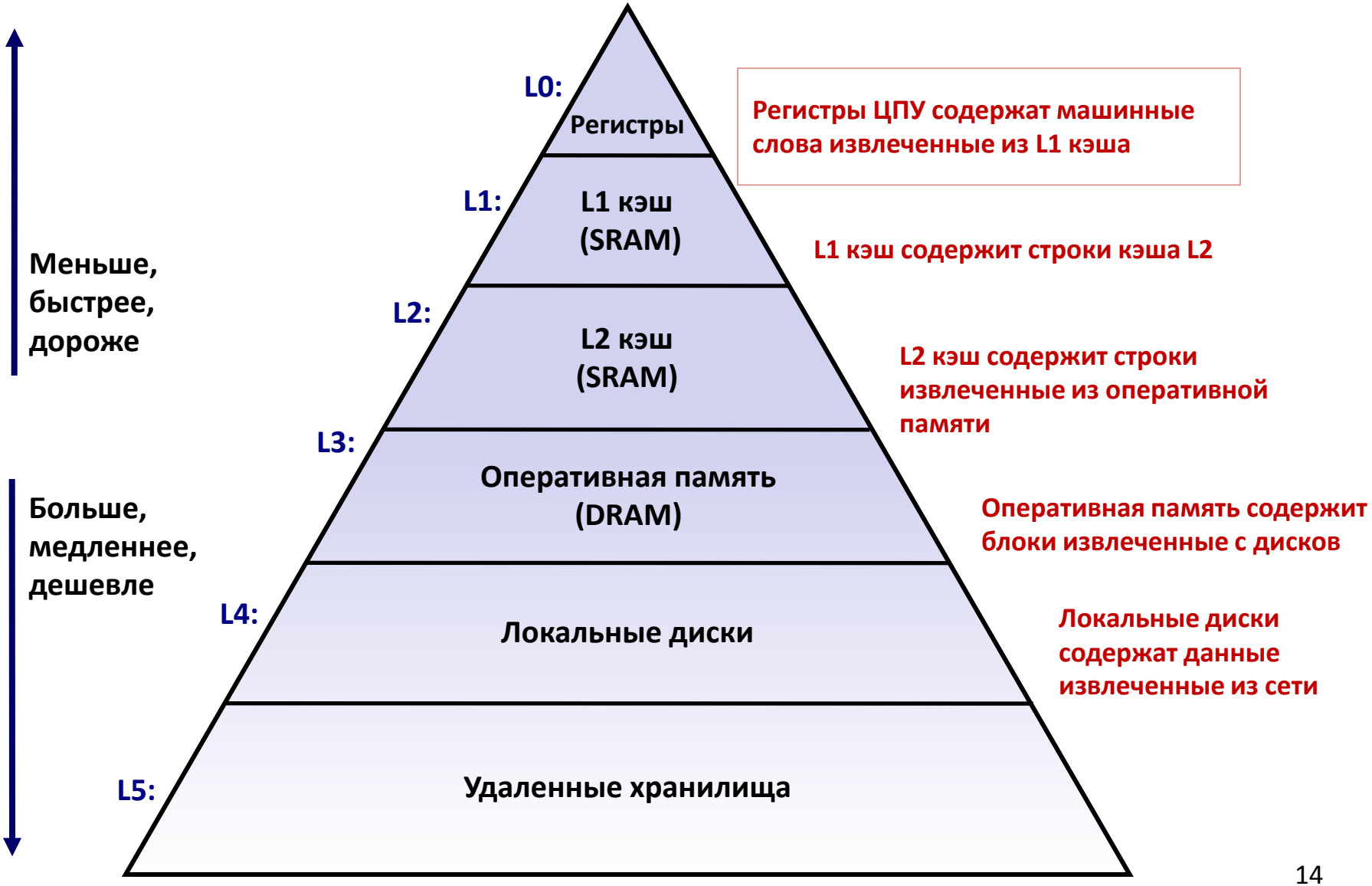
```
int sum_array_3d(int a[M][N][N]) {
    int i, j, k, sum = 0;

    for (i = 0; i < M; i++)
        for (j = 0; j < N; j++)
            for (k = 0; k < N; k++)
                sum += a[k][i][j];
    return sum;
}
```

© 2018 МГУ/ВМК/СП

Иерархия памяти

- Ряд фундаментальных свойств аппаратуры и ПО:
  - Более быстрые устройства хранения стоят дороже, имеют меньший объем и потребляют больше энергии.
  - Разрыв в скорости работы между ЦПУ и оперативной памятью увеличивается.
  - В хорошо написанных программах демонстрируется хорошая локальность.
- Данные свойства дополняют друг друга и …
- … выводят на идею иерархической организации памяти.

© 2018 МГУ/ВМК/СП

Иерархия памяти