

Операционная система Linux: Память, управление памятью



Артём
Поневин



Артем Поневин

Инженер

Luxoft



[Артем Поневин](#)

Предисловие

На этом занятии мы поговорим о:


- организации памяти в ПК;
- видах памяти;
- получении информации о памяти в Linux;
- возможностях настройки памяти в Linux.

По итогу занятия вы получите представление об устройстве оперативной памяти и научитесь получать информацию о текущем состоянии памяти в Linux.



План занятия

1. [Предисловие](#)
2. [Что такое память. Архитектура ПК.](#)
3. [Обзор утилит Linux для работы с CPU и NUMA\UMA архитектурой](#)
4. [Виртуальная память](#)
5. [Обзор утилит Linux для работы с ОЗУ](#)
6. [Cache, TLB, MMU](#)
7. [Обзор утилит Linux для работы с cache](#)
8. [SWAP](#)
9. [Обзор утилит Linux для работы с SWAP](#)
10. [Итоги](#)
11. [Домашнее задание](#)



Что такое память. Архитектура ПК.

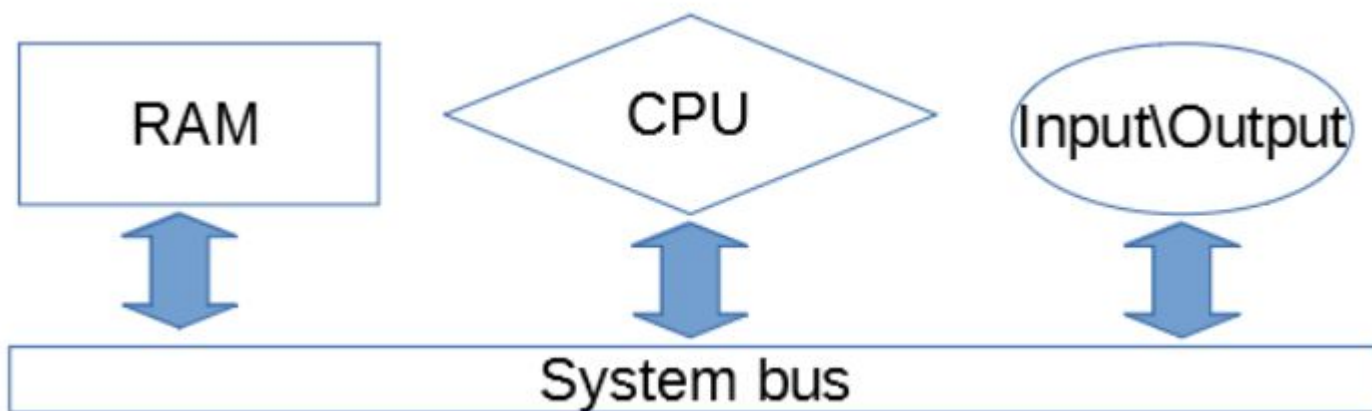
Память

А можно ли без нее?

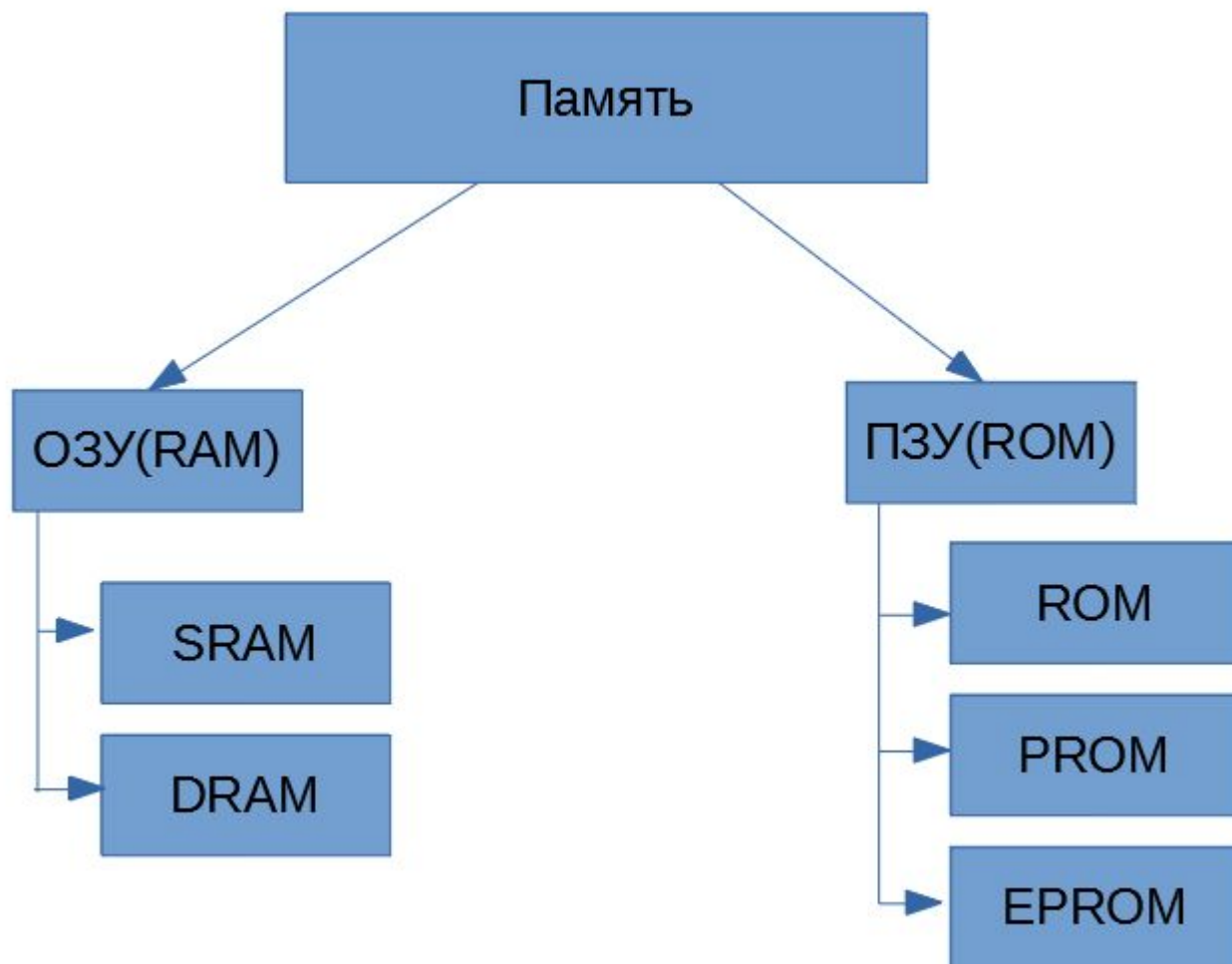


Минимальная необходимая архитектура ЭВМ

Схематическое изображение архитектуры ПК



Виды памяти



Виды памяти

- **RAM** (Random Access Memory, память с произвольным доступом) или **ОЗУ** (Оперативное Запоминающее Устройство) — энергозависимая память с произвольным доступом к ячейкам;
- **ROM** (Read Only Memory, память только для чтения) или **ПЗУ** (постоянное запоминающее устройство) — энергонезависимая память для хранения массива неизменяемых данных;
- **SRAM** (Static RAM) — статическая память с произвольным доступом;
- **DRAM** (Dynamic RAM) — динамическая память с произвольным доступом;
- **PROM, EPROM, EEPROM** — виды ROM.

Оперативное запоминающее устройство

RAM, ОЗУ — энергозависимое запоминающее устройство, в котором во время работы компьютера хранится исполняемый код и входные/выходные данные.



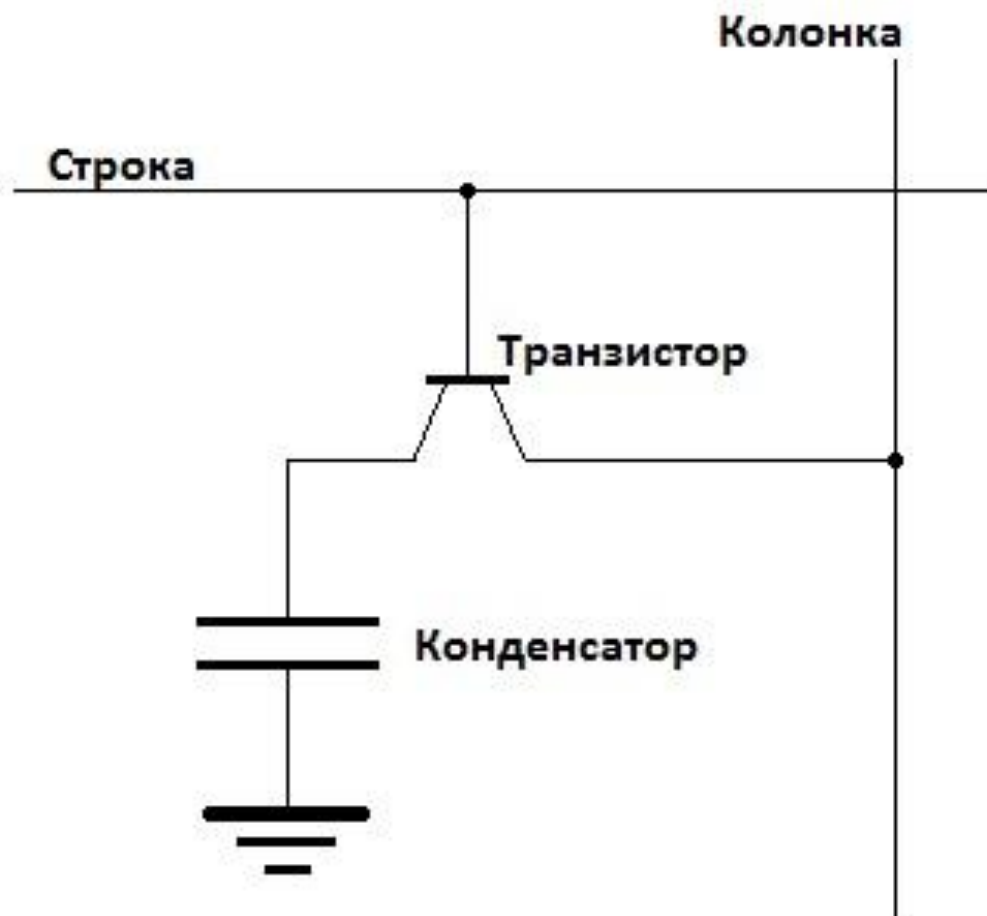


Энергозависимость памяти

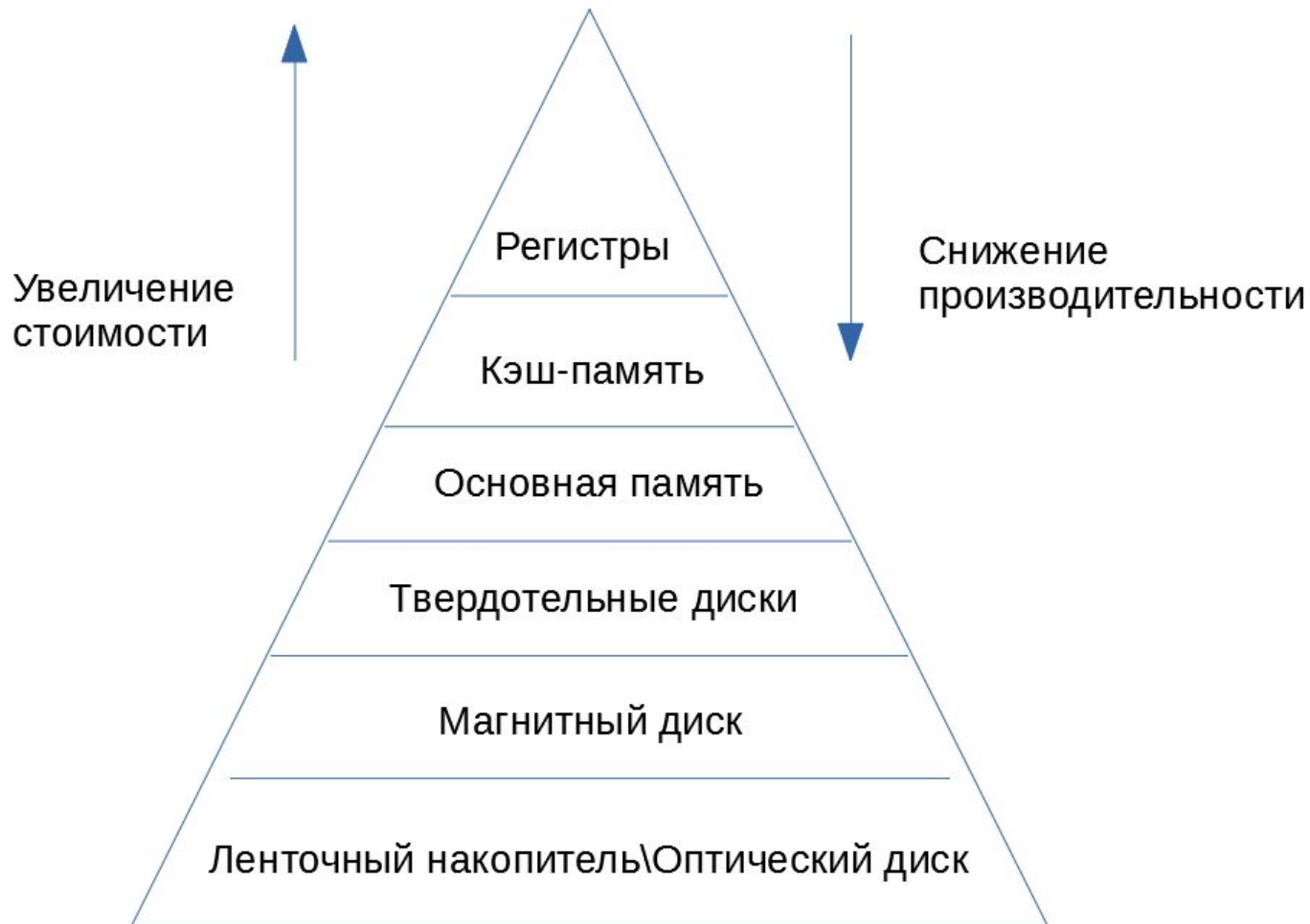
Оперативная память (в данном случае DRAM) энергозависима в силу того что одна ячейка памяти хранит один бит информации, при этом за состояние 1 принят **заряженный конденсатор**, за состояние 0 — разряженный.

При этом **скорость саморазряда** конденсатора велика и при отсутствии питания конденсатор переходит в состояние 0.

Схема ячейки DRAM



Иерархия памяти

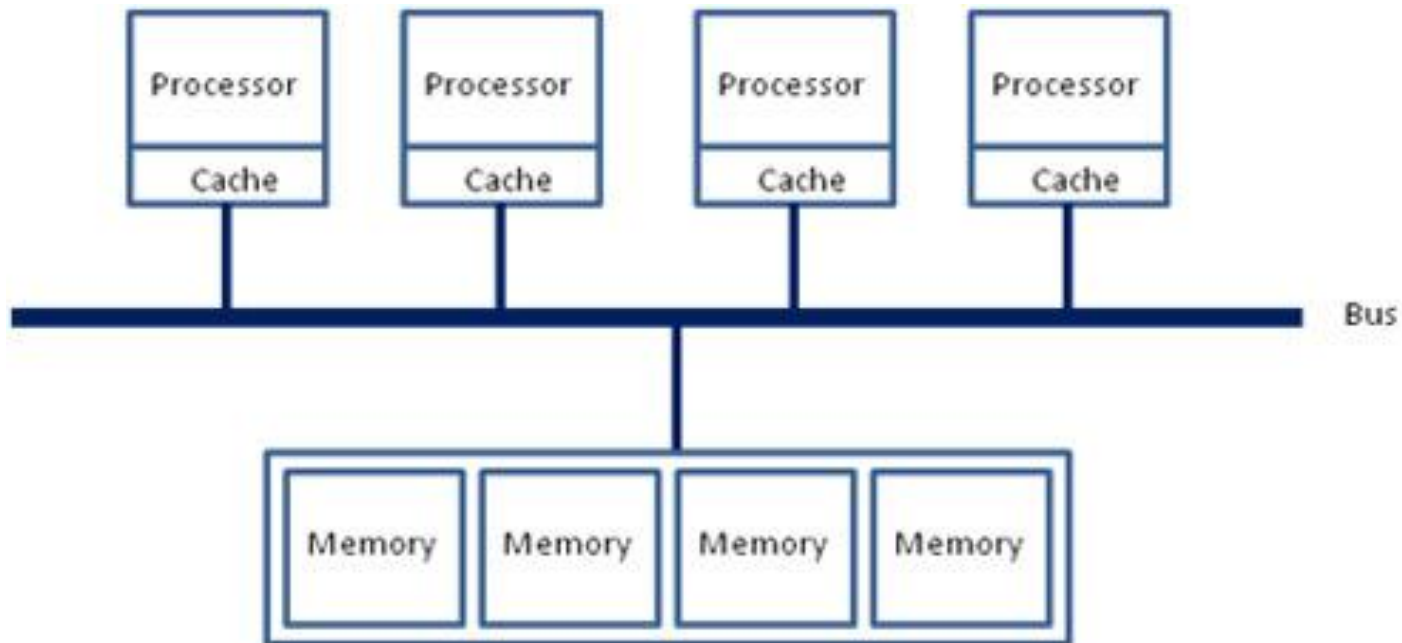


Архитектура UMA, NUMA

- **UMA** (Uniform Memory Access, равномерный доступ к памяти): все процессоры имеют доступ к совместно используемой памяти через общую шину;
- **NUMA** (Non-Uniform Memory Access, неравномерный доступ к памяти): для каждого процессора существует отдельный локальный модуль памяти, к которому он может обращаться напрямую.

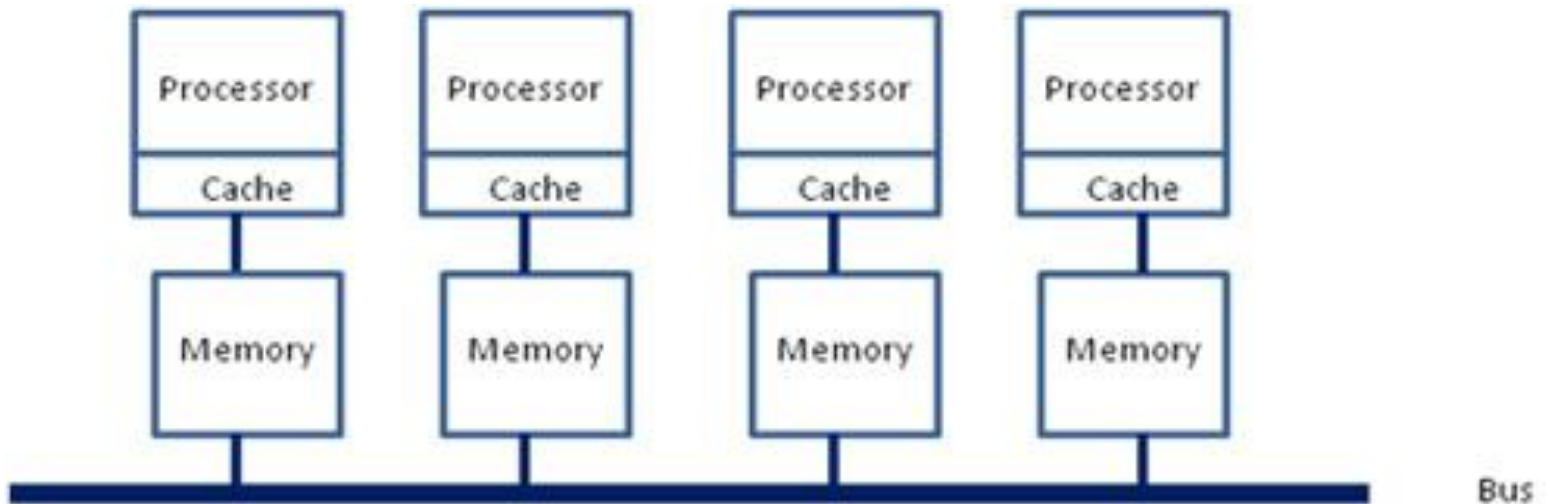
Архитектура UMA

- Процессор обращается к памяти через общую шину



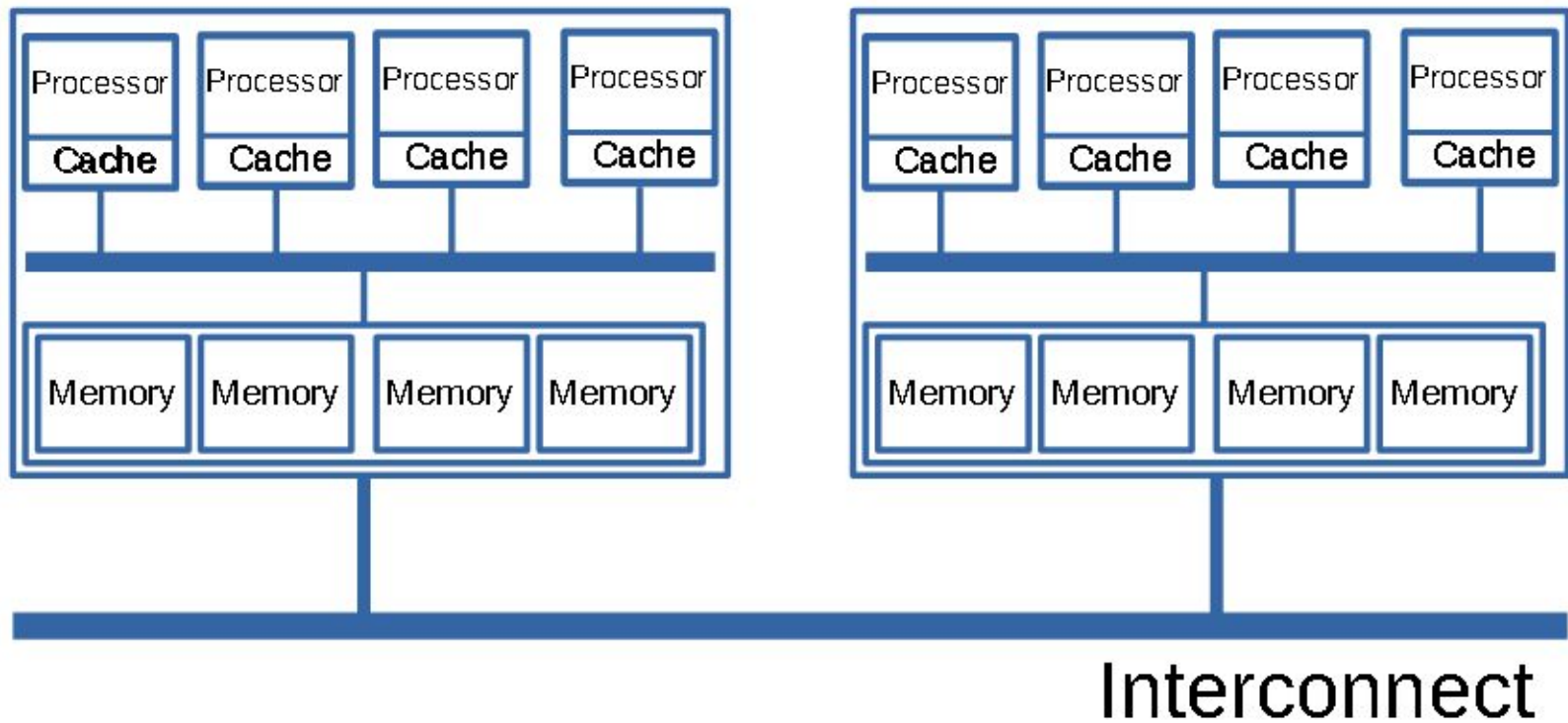
Архитектура NUMA


- Процессор имеет быстрый доступ к памяти, но только к “своей”.
- Писать в “чужую” память затратнее чем в архитектуре UMA.



Смешанная

➔ Многопроцессорные системы могут совмещать обе архитектуры





Обзор утилит Linux для работы с CPU и NUMA/UMA архитектурой

Команды Linux для работы с NUMA / UMA

- **lscpu** — позволяет отобразить информация о процессоре;
ls — list, cpu — processor.
- **numastat, numactl, numad** — позволяют получить информацию, настроить ПК под архитектурой NUMA;
- **dmidecode** — позволяет получить информацию о комплектующих ПК;
- **cat /proc/?** — позволяет в режиме чтения узнать большинство текущих настроек операционной системы.

Программа `lscpu`

`lscpu` — отображает информацию о процессоре, архитектуре, количестве ядер, модели процессора, поставщике процессора, тактовой частоте и т. д. Информация предоставляется в читаемом формате.

Пример:

```
user@user:~$ lscpu
```

Программы numastat, numactl и numad

numastat, numactl и numad — утилиты, которые отображают настройки и состояния NUMA-архитектуры. Также позволяют в ручном режиме запирать процесс на ноде.

Примеры:

```
user@user:~$ numactl --hardware
```

```
user@user:~$ numastat
```

```
user@user:~$ systemctl status numad
```

Программа dmidecode

dmidecode (DMI, Desktop Management Interface) — утилита, позволяющая программно получать информацию о комплектующих (“железе”). Например, можно получить информацию о BIOS, материнской плате, количеству и характеристикам модулей ОЗУ.

Примеры:

```
user@user:~$ dmidecode -t memory
```

```
user@user:~$ dmidecode -t baseboard
```

```
user@user:~$ man dmidecode
```

Специальная файловая система /proc

/proc/? — в каталог /proc монтируется файловая система, которая отображает информацию о процессах, состоянии и конфигурации ядра и системы.

Примеры:

```
user@user:~$ cat /proc/meminfo
```

```
user@user:~$ cat /proc/sys/vm/dirty_ratio
```

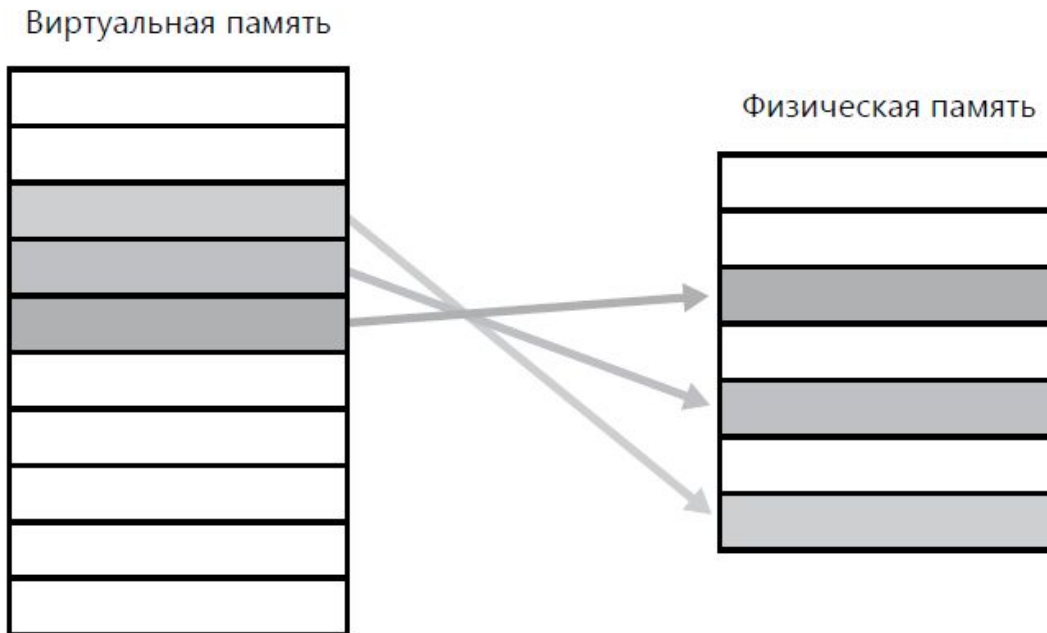
```
user@user:~$ cat /proc/{process_id}/maps
```



Виртуальная память

Виртуальная память

Абстракция, которая позволяет выполнять программы, требующие больше оперативной памяти, чем имеется в компьютере, путём автоматического перемещения частей программы между основной памятью и другими хранилищами.



Функции виртуальной памяти

- **изоляция процессов** — каждый процесс в системе видит свое пространство как единственное и непрерывное;
- **абстракция физической памяти** — физическая и виртуальная память могут быть разных размеров (виртуальная больше);
- **выделение адресов объектам, которые не загружены в память** — следствие абстракции физической памяти.

Организация виртуальной памяти

- **Страничная** — адресное пространство делится на *части одинакового фиксированного размера*, называемые виртуальными страницами. Адрес задается номером страницы в виртуальной памяти и смещением внутри страницы.
- **Сегментная** — адресное пространство делится на части, называемые сегментами, *размер* которых *определяется смысловым значением* содержащейся в них информации. Виртуальный адрес задается номером сегмента и смещением внутри сегмента.
- **Странично-сегментная** — различные сочетания страничной и сегментной памяти.

Сегментная организация памяти

- выделяемые сегменты могут иметь переменный размер;
- каждый сегмент — линейная последовательность адресов, начиная с 0;
- размер сегмента может меняться динамически. В таблице сегментов хранится длина сегментов;
- логический адрес — номер сегмента и смещение внутри сегмента;
- у процесса может быть доступ к нескольким сегментам.

Сегментная организация памяти

- Таблица сегментов (дескрипторов) похожа на таблицу страниц в страничной организации памяти;
- Сегментная организация памяти подвержена фрагментации.



Страничная организация памяти

- **логическая** память делится на страницы — смежные области одинаковой длины;
- **физическая** память делится на фреймы такого же размера;
- **связь** между логической и физической памятью процесса осуществляется с помощью **таблицы страниц**: системной структуры, выделяемой процессу для трансляции его логических адресов в физические.

Таблица страниц


Структура данных, используемая системой виртуальной памяти в операционной системе компьютера для хранения сопоставления между виртуальным адресом и физическим адресом.



Источник изображения: <http://komputernulja.ru>

Виды страничной виртуальной памяти

- **Собственная память** (private memory) — память, выделенная конкретному процессу;
- **Общая память** (shared memory) — память, предназначенная для взаимодействия процессов;
- **Анонимная память** (anonymous Memory) — память, выделенная непосредственно в ОЗУ, без сопоставленного файла. Позволяет процессам в своем адресном пространстве резервировать больше памяти, чем доступно в системе.



Обзор утилит Linux для работы с ОЗУ

Команды Linux для работы с памятью

- **top** — отображает потребление ресурсов процессами в реальном времени (и другую информацию);
- **free** — утилита для отображения состояния памяти;
- **cat /etc/proc/meminfo** — раздел файловой системы /proc с текущей информацией о состоянии памяти в операционной системе.

Программа top

top — позволяет в виде динамически обновляемой таблицы вывести перечень запущенных процессов и оценить какой объем ресурсов они потребляют.

htop — та же утилита с псевдографическим интерфейсом.

Примеры:

```
user@user:~$ top
```

```
user@user:~$ htop
```

Программа free

free — утилита, которая отображает объем свободной и использованной оперативной памяти ПК.

Примеры:

```
user@user:~$ free -h
```

```
user@user:~$ free -h -t
```

```
user@user:~$ free -h -s 5 -c 5
```

Специальная файловая система `/proc`

`/proc/meminfo` — в каталог `/proc` монтируется файловая система, которая отображает информацию о процессах, состоянии и конфигурации ядра и системы.

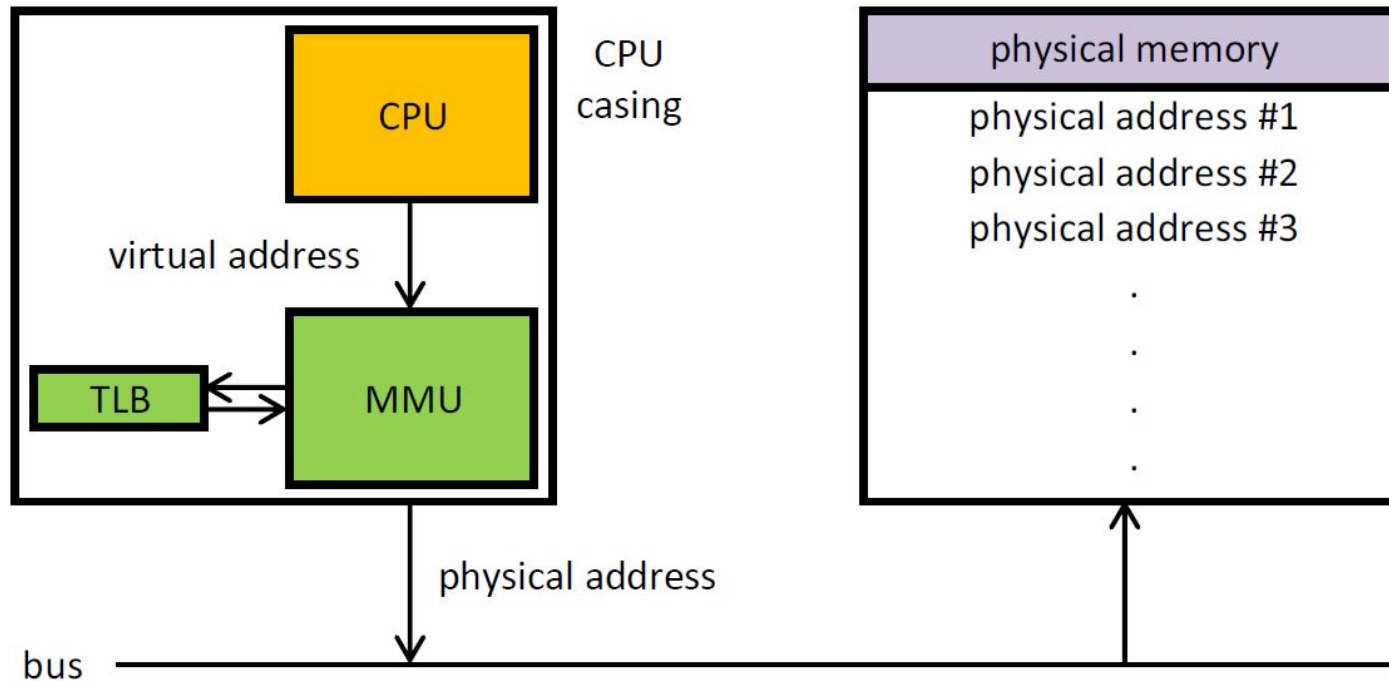
Примеры:

```
user@user:~$ cat /proc/meminfo | grep Mem
```

Cache, TLB, MMU

- **Cache** — промежуточный буфер с быстрым доступом для хранения информации, которая с большой вероятностью будет запрошена.
- **TLB** — это специальный кэш, который хранит соответствие страниц в виртуальной и физической памяти, которые с наибольшей вероятностью будут запрошены.
- **MMU** — компонент аппаратного обеспечения компьютера, отвечающий за управление доступом к памяти

Cache, TLB, MMU



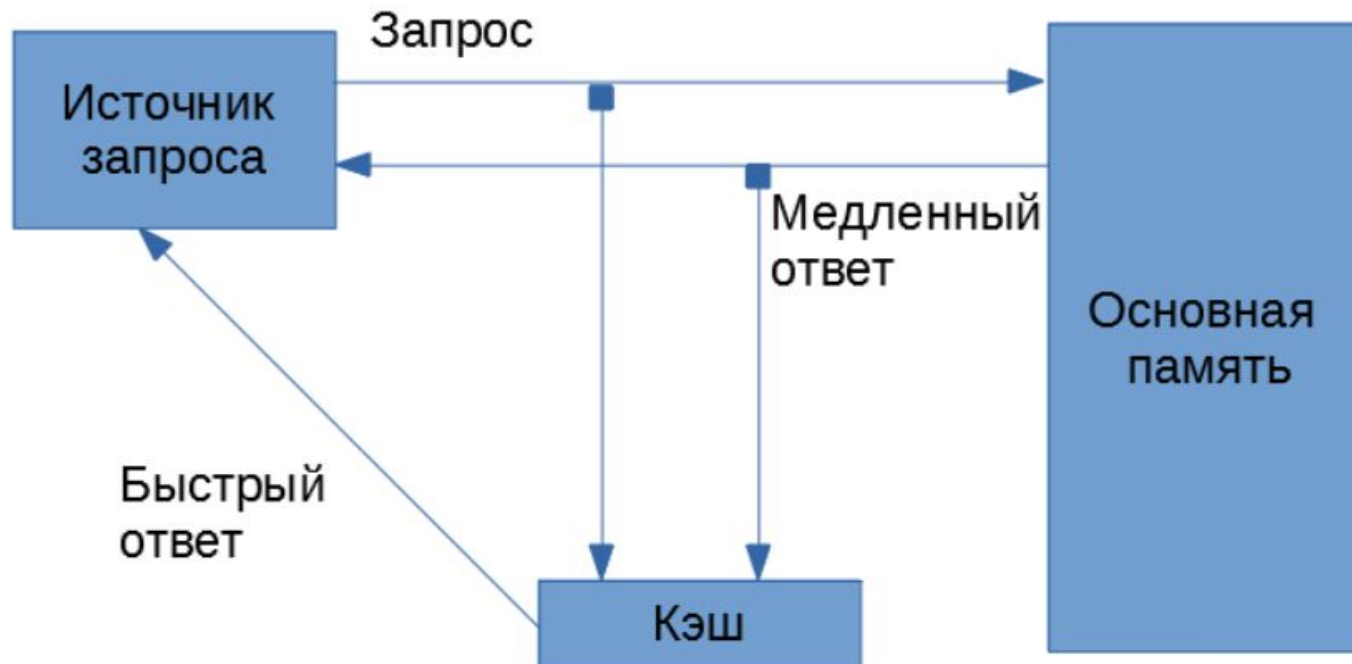
CPU: Central Processing Unit

MMU: Memory Management Unit

TLB: Translation lookaside buffer

Ускорение трансляции страниц

Кэш-память — сверхбыстрая память используемая процессором, для временного хранения данных, которые наиболее часто используются.



Что хранит кэш?

- компоненты исполняемых программ;
- данные, с которыми работают исполняемые программы;
- TLB.



Cache, TLB, MMU

Виды кэша

По алгоритмам отображения оперативной памяти в кэш:

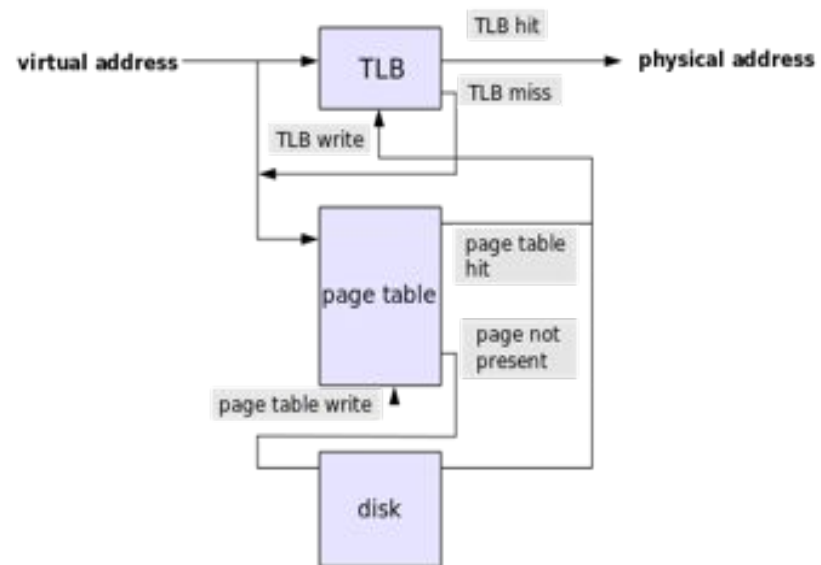
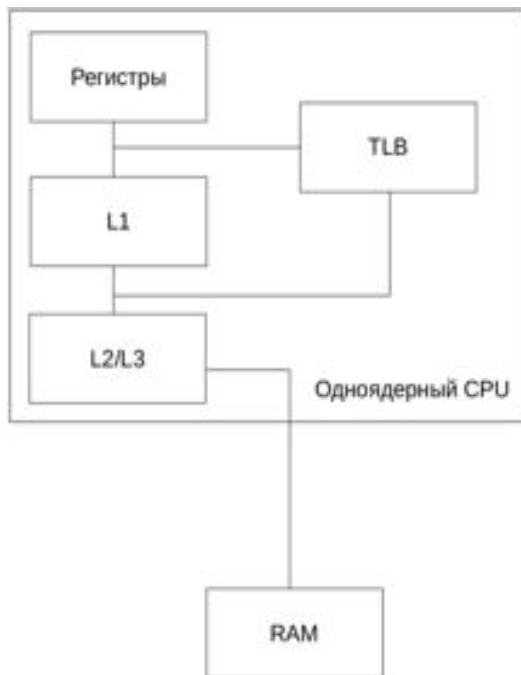
- полностью ассоциативный кэш;
- кэш прямого отображения;
- множественный ассоциативный кэш.

По архитектуре:

- Cache L1;
- Cache L2;
- Cache L3;
- Cache L4.

TLB, Translation lookaside buffer

TLB (translation lookaside buffer, буфер ассоциативной трансляции) — специализированный кэш центрального процессора, используемый для ускорения трансляции адреса виртуальной памяти в адрес физической памяти.

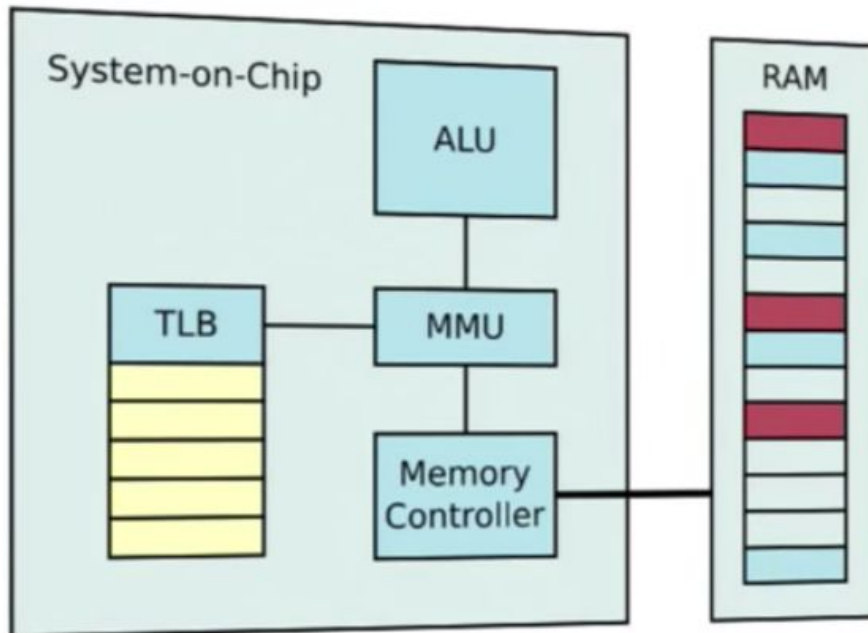


Свойства TLB

- специализированный аппаратный кэш процессора;
- TLB хранит несколько наиболее часто используемые записи таблицы страниц;
- TLB-кэш внутренне схож с LX-кэшами, только служит для трансляции виртуальных адресов и представляет собой таблицу из сетов, внутри которых записи: «виртуальный адрес ↔ физический адрес»;
- PML4 адресована только по физическим адресам, поэтому сама PML4 может кэшироваться только в L2-L3 кэшах.

MMU, Memory management unit

Memory management unit (MMU, модуль управления памятью) — компонент аппаратного обеспечения компьютера, отвечающий за управление доступом к памяти, запрашиваемым центральным процессором.





Обзор утилит Linux для работы с cache

Команды Linux для работы с кэш

sysctl -a | grep dirty — найти в настройках Linux параметры, которые используются при кэшировании (“грязные” страницы).

В файле `/etc/sysctl.conf`:

- **vm.dirty_background_ratio = 10** — процент от памяти, которая может быть использована для хранения кэша перед записью на диск в фоновом режиме.
- **vm.dirty_ratio = 15** — максимум памяти, которое может быть выделено под кэш до записи на диск. При достижении этого значения ввод\вывод блокируется до освобождения кэша.
- **vm.dirty_expire_centisecs = 3000** — время нахождения “грязных” страниц в кэше.

Команды Linux для работы с кэш

- **sync** — команда заставляет Linux записать все кэшированные данные на диск;
- **echo 1 > /proc/sys/vm/drop_caches** — очистка кэша PageCache;
- **echo 2 > /proc/sys/vm/drop_caches** — очистка inode и dentrie;
- **echo 3 > /proc/sys/vm/drop_caches** — очистка inode и dentrie и PageCache.



SWAP

SWAP

Механизм виртуальной памяти, при котором часть данных из оперативной памяти (ОЗУ) перемещается на хранение в ПЗУ.

Amount of RAM in the system	Recommended swap space	Recommended swap space if allowing for hibernation
<= 2GB	2 times the amount of RAM	3 times the amount of RAM
>2GB-8GB	Equal to the amount of RAM	2 times the amount of RAM
>8GB-64GB	At least 4GB	1.5 times amount of RAM
>64GB	At least 4GB	Hibernation not recommended

Источник изображения: <https://access.redhat.com>



Обзор утилит Linux для работы с SWAP

Программы **mkswap**, **swapon**, **swapoff**

- **swapon -s; grep Swap /proc/meminfo; free -h** — команды показывают использование файла подкачки;
- **mkswap /swapfile** — размечает файл /swapfile как файловую систему swap;
- **swapoff\swapon /swapfile** — отключение/подключение файла подкачки в систему;
- **cat /etc/fstab** — выводит список устройств для монтирования. Можно приписать swap раздел.

Команды для тонкой настройки SWAP

Посмотреть текущие/поменять до перезагрузки:

- `/proc/sys/vm/vfs_cache_pressure` — размер дискового кэша
- `/proc/sys/vm/swappiness` — процент переноса данных в swap

Для сохранения изменений после перезагрузки:

В файле `/etc/sysctl.conf`:

- `vm.vfs_cache_pressure = {ваше значение}`
- `vm.swappiness = {ваше значение}`



Итоги

Итоги

Сегодня мы рассмотрели ОЗУ в Linux:

- Архитектура ПК и топология процессор-память;
- Организация памяти;
- Файлы/разделы подкачки;
- Утилиты для работы с памятью.

Домашнее задание

Давайте посмотрим ваше [домашнее задание](#).

- Вопросы по домашней работе задавайте **в чате** мессенджера Slack.
- Задачи можно сдавать **по частям**.
- Зачёт по домашней работе проставляется после того, как **приняты все задачи**.

**Задавайте вопросы и
пишите отзыв о лекции!**

Артем Поневин