Образовательные проекты

mail.ru group

Системное программирование

Ядро. Устройство. Планировщики процессов.

Сергей Бронников

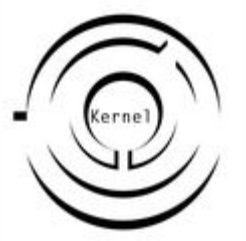
Содержание занятия

1. Операционная система
2. Прерывания
3. Загрузчик ОС
4. Ядро Linux
5. Планировщик ОС

Операционная система

ОС - диспетчер физических ресурсов

Ядро

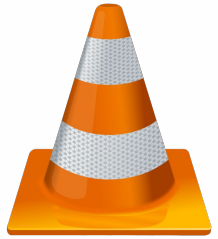


Драйверы

Системные вызовы

Браузер

Плеер



Терминал

Загрузчик

read()
write()
open()
close()
fork()

Компилятор

Почта

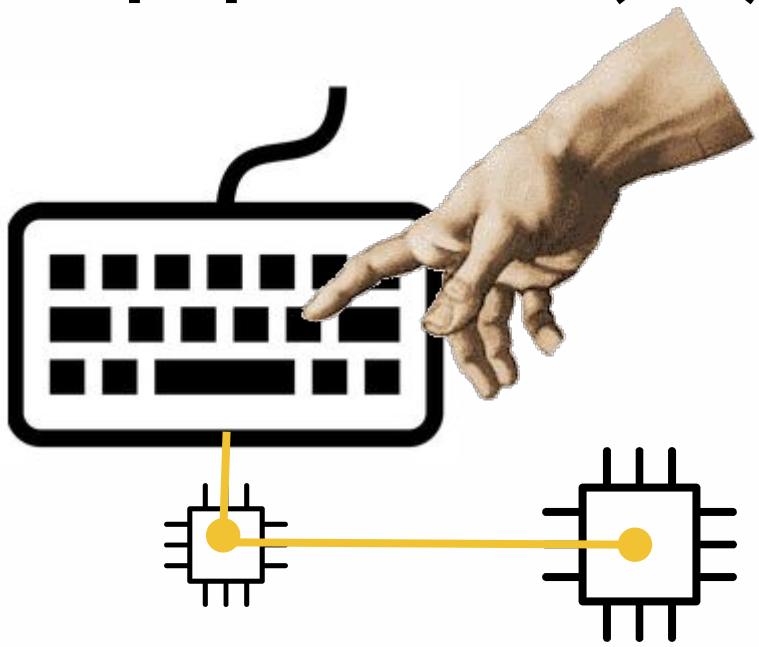
Это операционная система

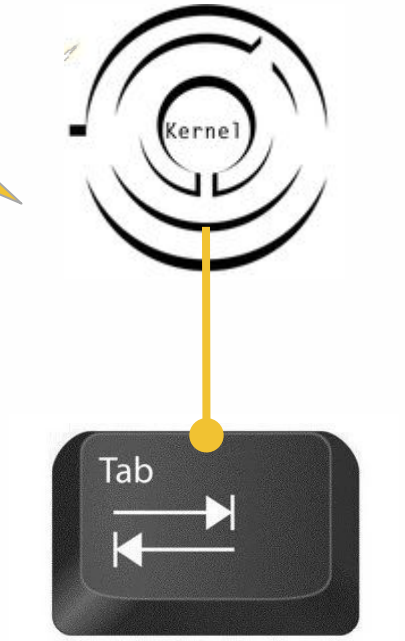
Пространство ядра ОС

Это **НЕ** операционная система

Пространство пользователя

Прерывания (1/2)





Обработчик знает номер
Вызван обработчик в
планирует
обработку

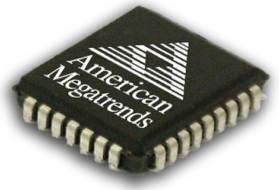
Прерывания (2/2)

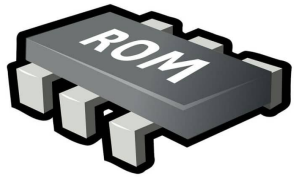
Загрузчик (1/3)

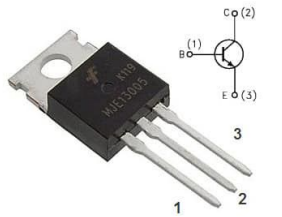
Кто запустил ядро?

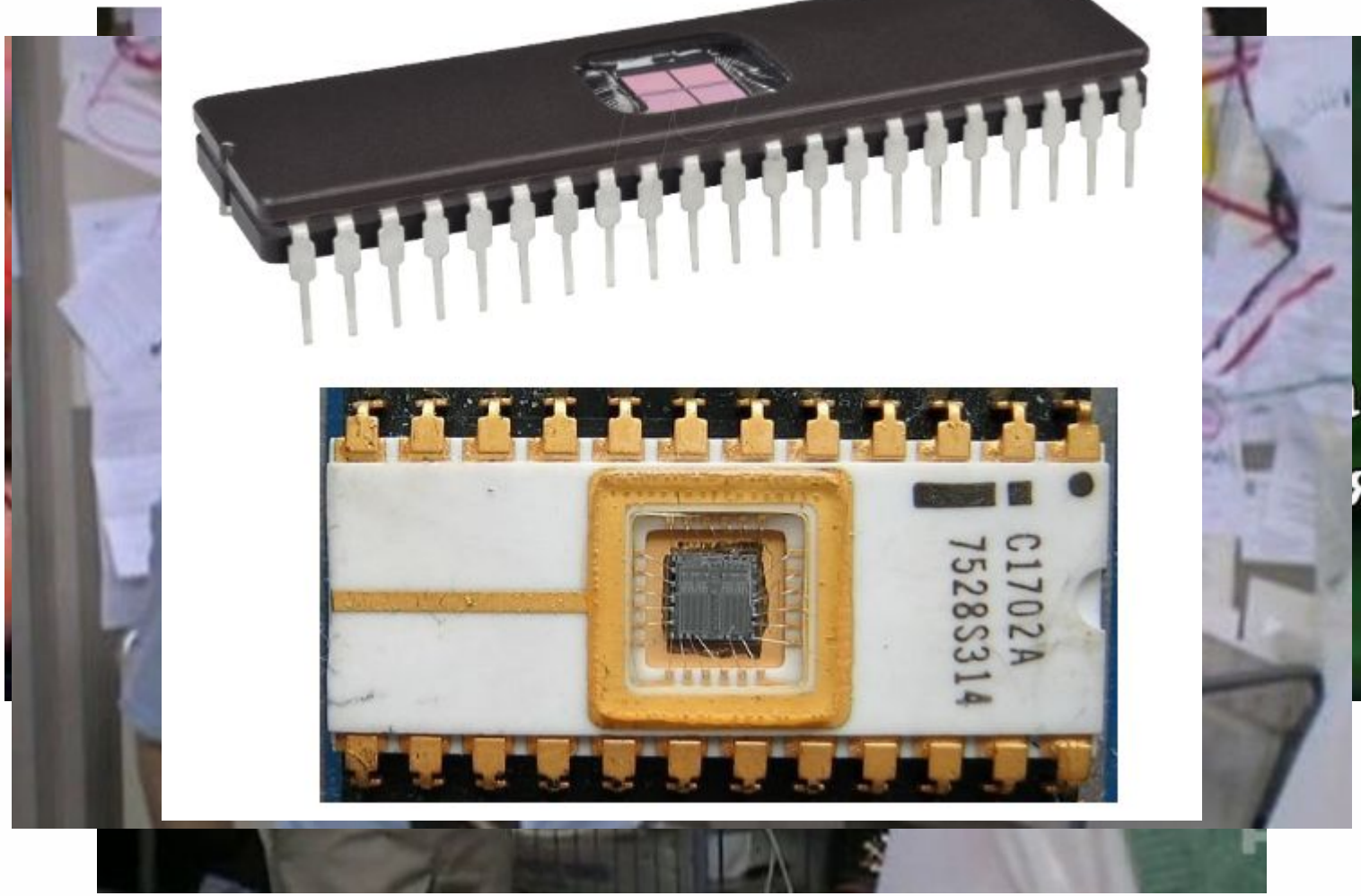
Загрузчик (2/3)

Загрузчик

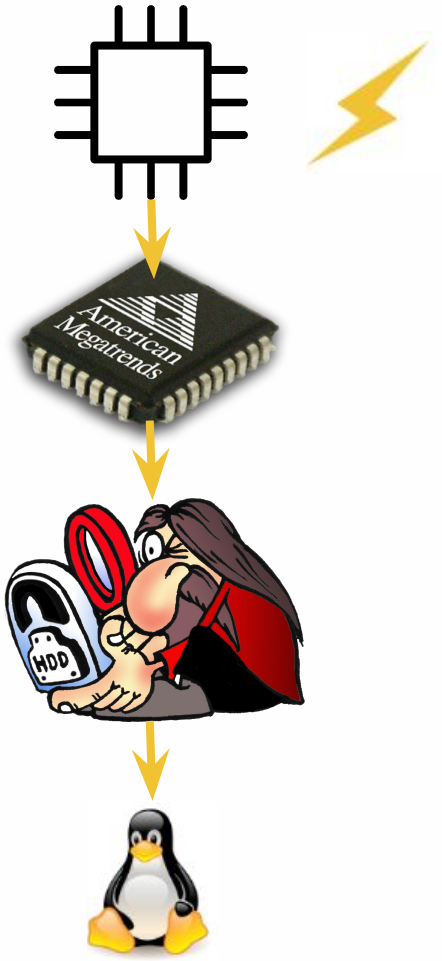


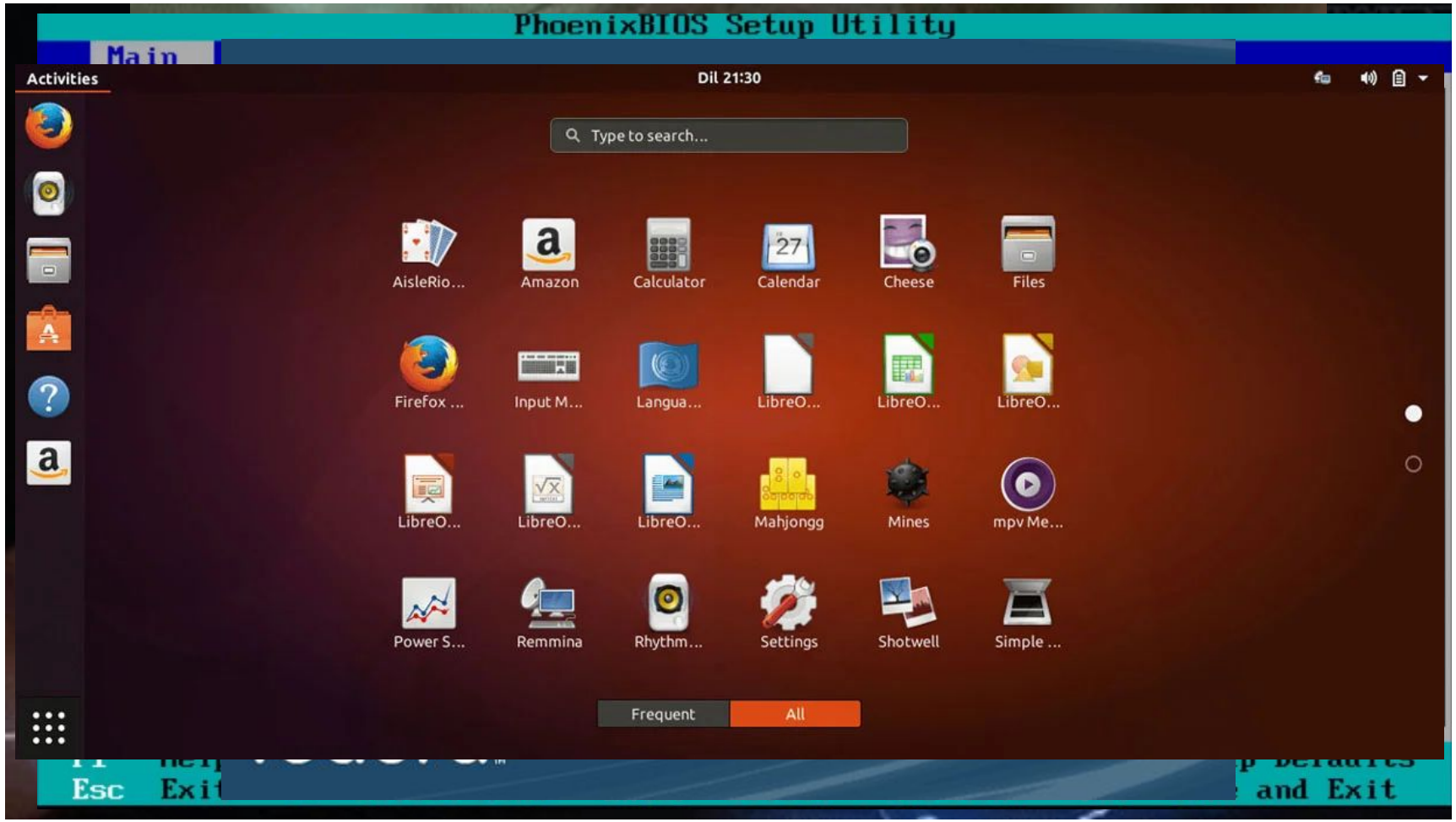






Загрузчик (3/3)





Загрузчик (3/3)

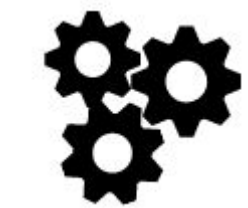
Вопрос (2 балла):

Почему BIOS сам не запускает ядро?

Он не знает, что такое файловая система, и не может найти ядро

Лампа на слайде означает вопрос за баллы

Ядро Linux



Процессы

Время

IPC

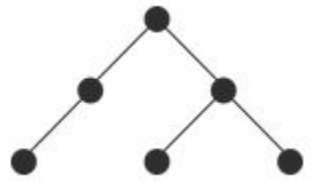
Пользователи

Аппаратура

Файловая система

Виртуализация

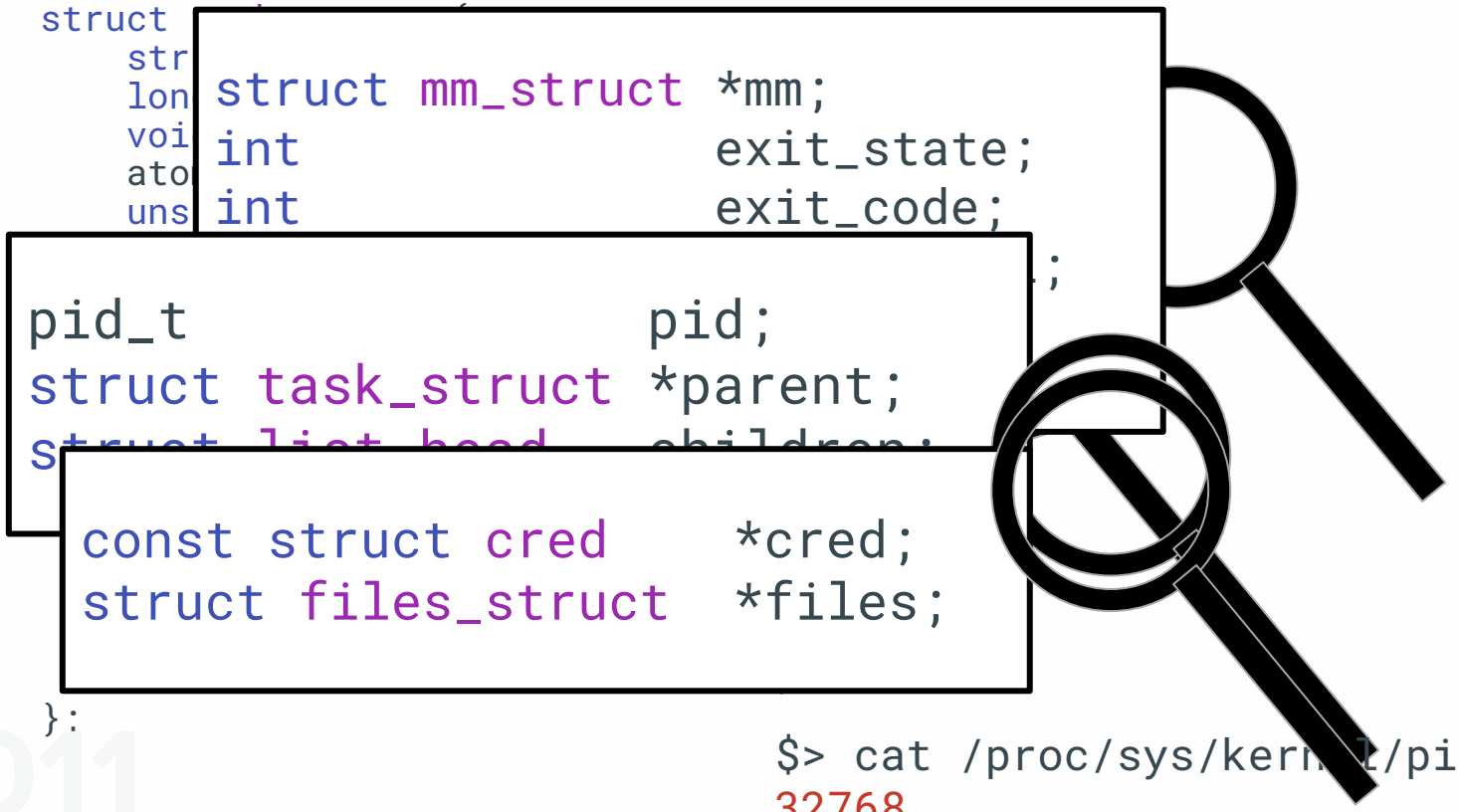
Сеть



Структуры данных

Ядро Linux. Процессы (1/3)

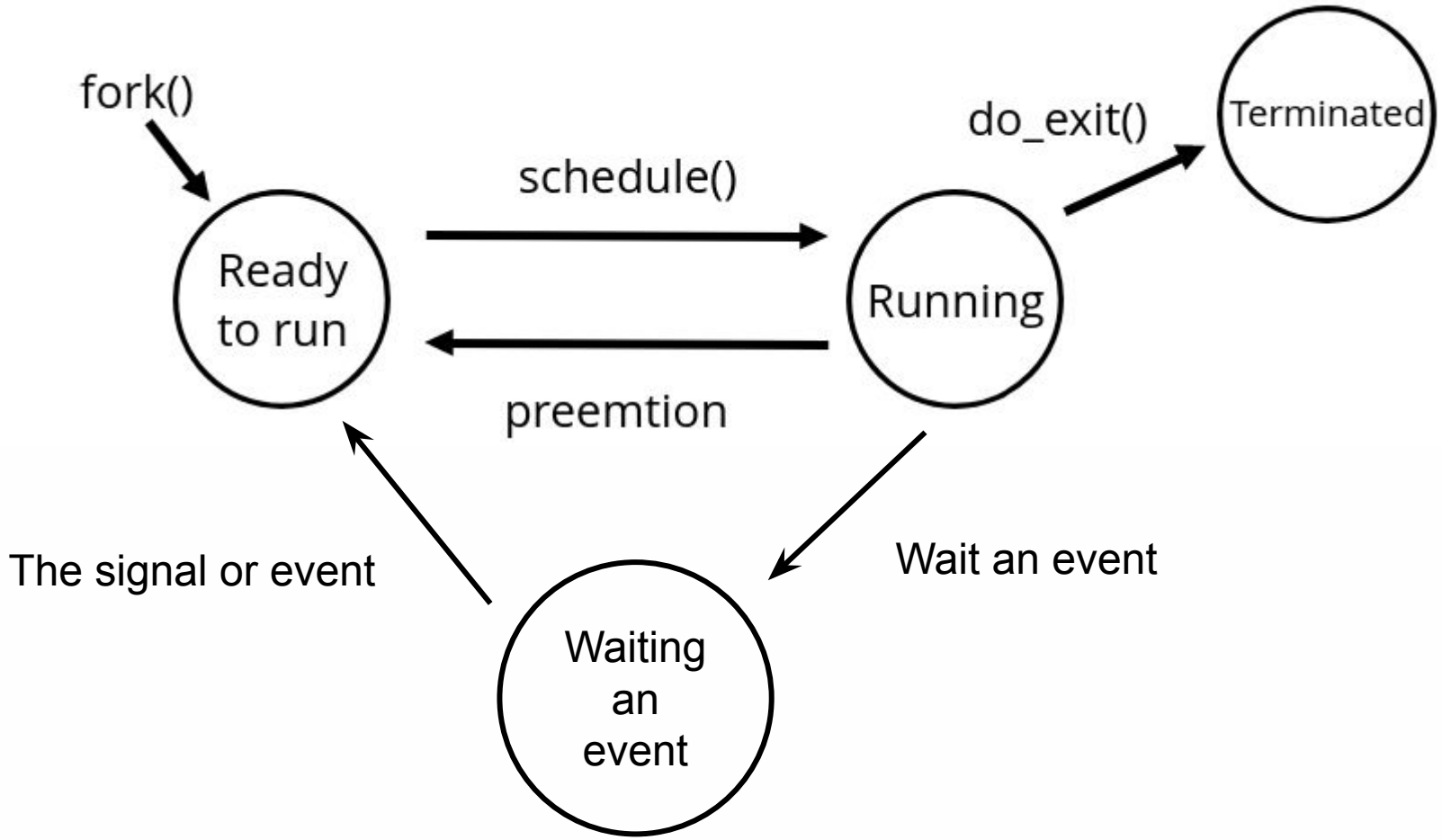
```
/**
 * 30.09.2018
 * #include/linux/sched.h
 * 618 lines.
 */
struct
    str
    lon
    voi
    ato
    uns
    struct mm_struct *mm;
    int              exit_state;
    int              exit_code;
                                  ;
    pid_t                pid;
    struct task_struct *parent;
    struct list_head    children;
    const struct cred    *cred;
    struct files_struct  *files;
};
```



```
pid_t
getpid(void);

pid_t
getppid(void);
```

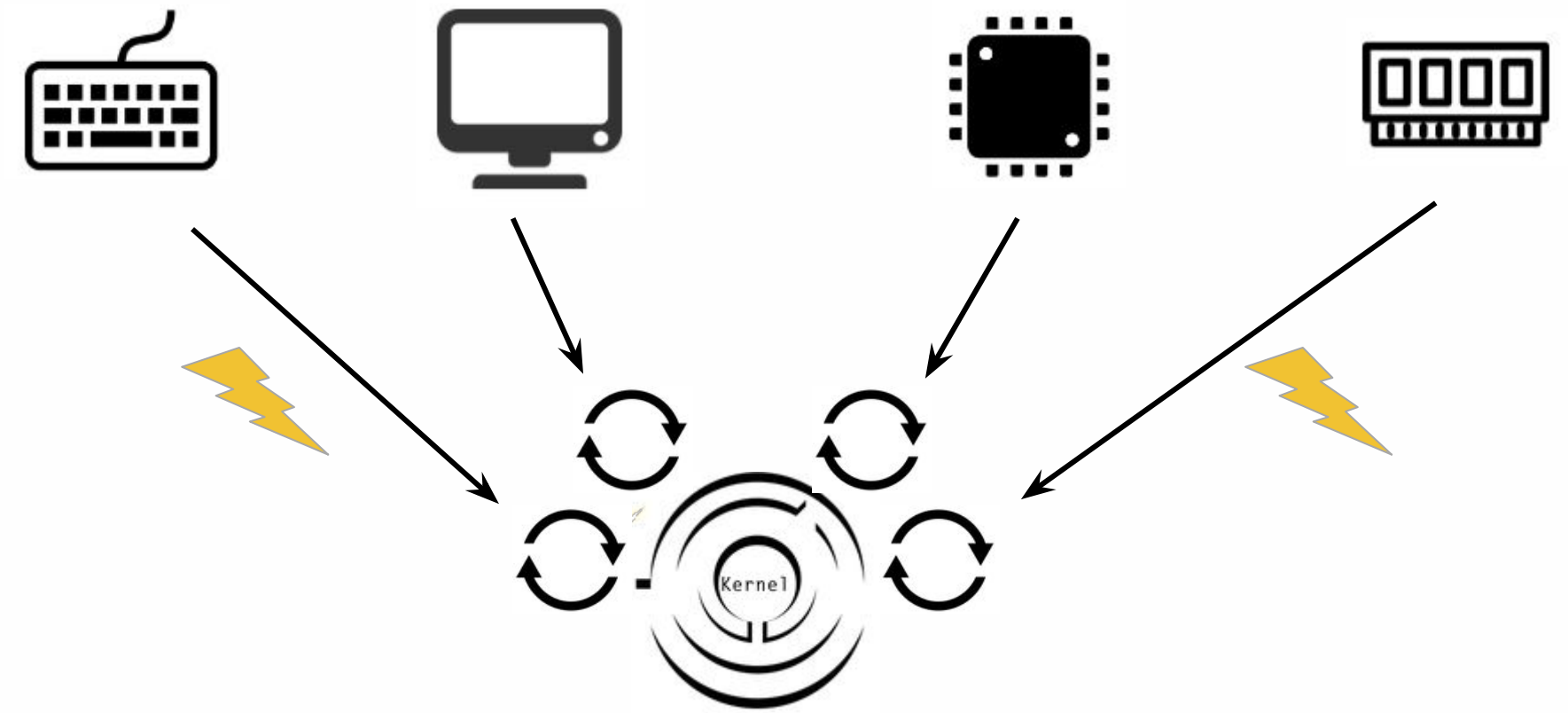
```
$> cat /proc/sys/kernel/pid_max
32768
```

Ядро Linux. Процессы (2/3)

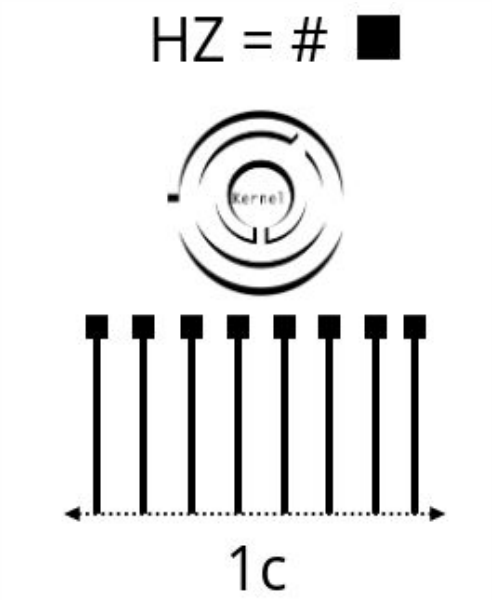


Ядро Linux. Процессы (3/3)

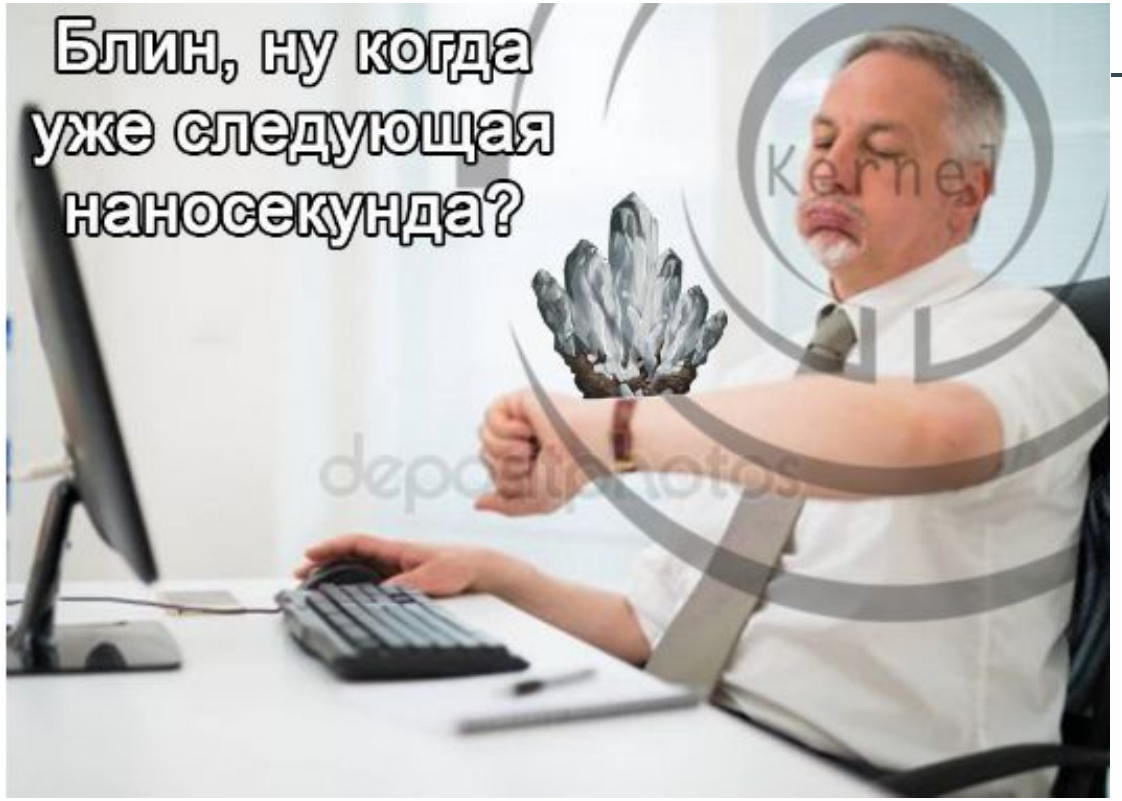
Ядро Linux. Аппаратное обеспечение.



Ядро Linux. Время (1/3)



Периодические и разовые задачи

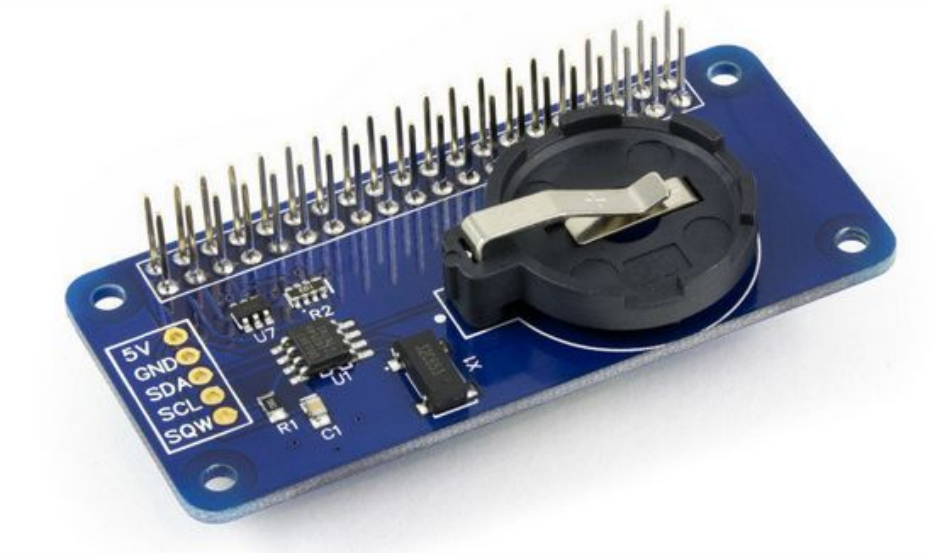


`-$(uname -r)`

Слишком высокий **HZ**

Ядро Linux. Время (2/3)

Real Time Clock

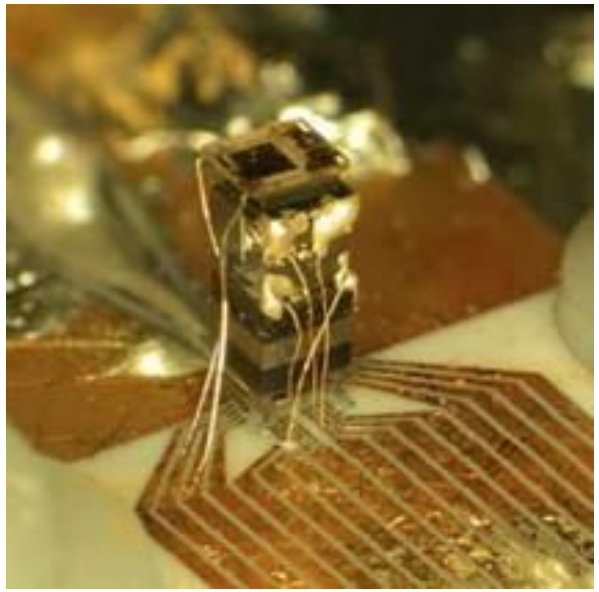


Detected 2400.131 MHz processor.
Calibrating delay loop... 4799.56 BogoMIPS

Ядро Linux. Время (3/3)

PPS (Pulse Per Second)

Источник сигнала

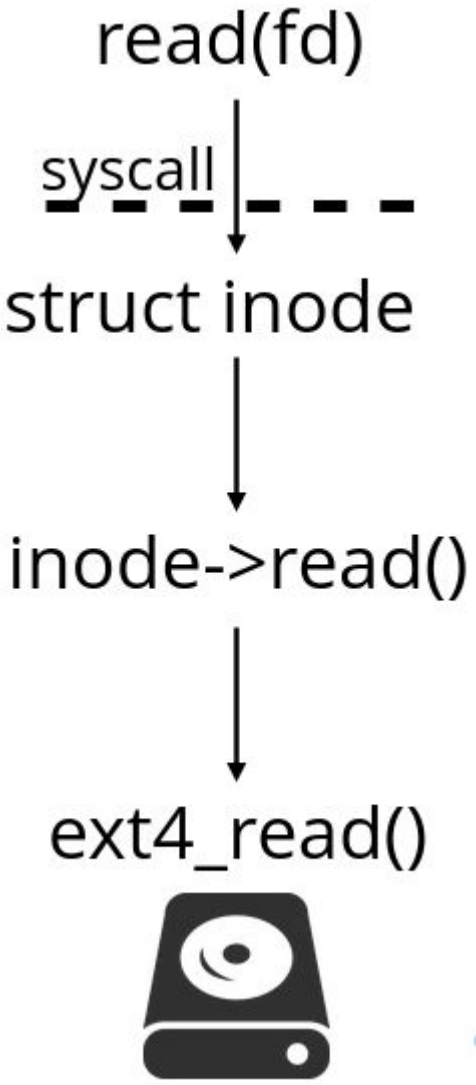


Атомные часы

Точность - пикосекунды (10^{-12})

Ядро Linux. Файловая система.

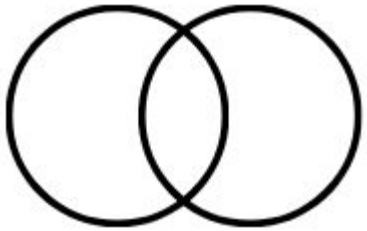
```
/**
 * 30.09.2018
 * 33 virtual functions, 149 lines.
 */
struct file_operations {
    loff_t (*llseek) (struct file *, loff_t, int);
    ssize_t (*read) (struct file *, char __user *, size_t, loff_t *);
    ssize_t (*write) (struct file *, const char __user *, size_t, loff_t *);
    int (*mmap) (struct file *, struct vm_area_struct *);
    int (*open) (struct inode *, struct file *);
    int (*flush) (struct file *, fl_owner_t id);
    int (*fsync) (struct file *, loff_t, loff_t, int datasync);
    int (*flock) (struct file *, int, struct file_lock *);
    long (*fallocate)(struct file *file, int mode, loff_t offset, loff_t len)
};

struct super_block {
    struct file_system_type          *s_type;
    const struct super_operations    *s_op;
    int                              s_count;
    struct list_head                 s_mounts;
};
```



Ядро Linux. IPC

Мьютексы, семафоры

Очереди сообщений

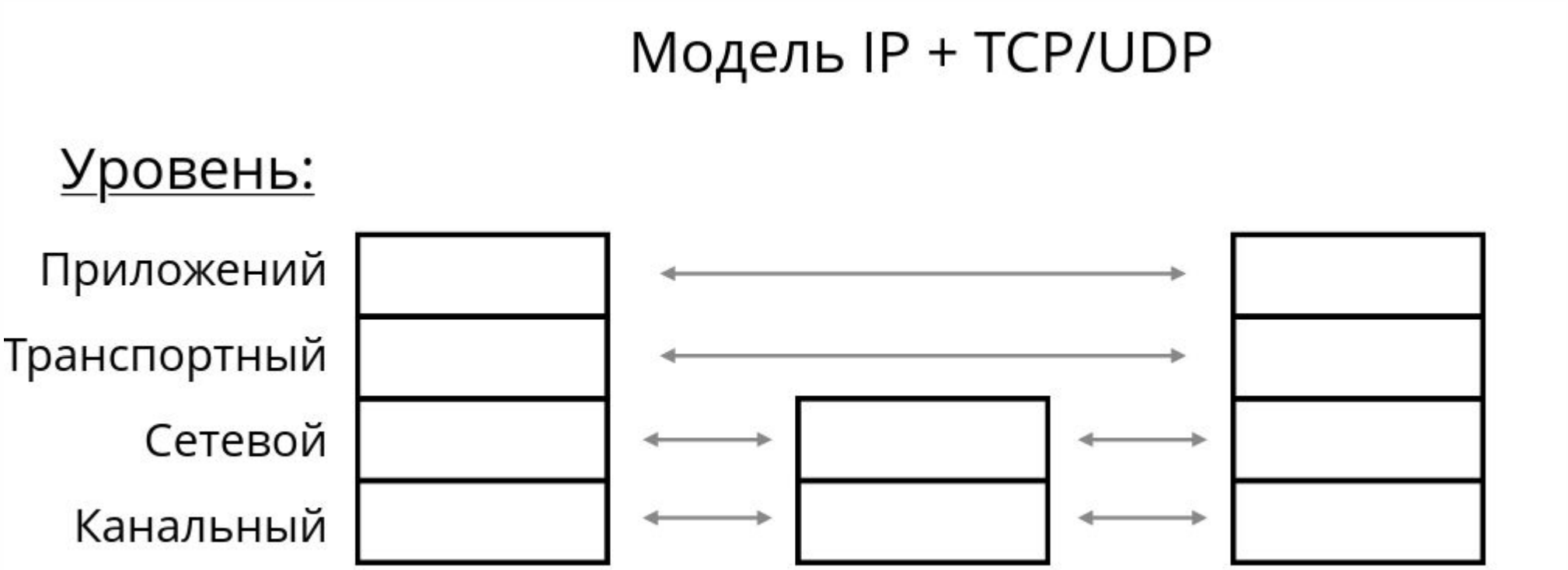


Разделяемая память

Доменные сокеты

Pipe

Ядро Linux. Сеть.



Ядро Linux. Пользователи.

```
/**
 * 30.09.2018.
 * 39 lines.
 */
struct cred {
    kuid_t          uid;            /* real UID of the task */
    kgid_t          gid;            /* real GID of the task */
    kuid_t          suid;           /* saved UID of the task */
    kgid_t          sgid;           /* saved GID of the task */
    kuid_t          euid;           /* effective UID of the task */
    kgid_t          egid;           /* effective GID of the task */
    struct user_struct *user;       /* real user ID subscription */
};
```

Ядро Linux. Структуры данных.

Список

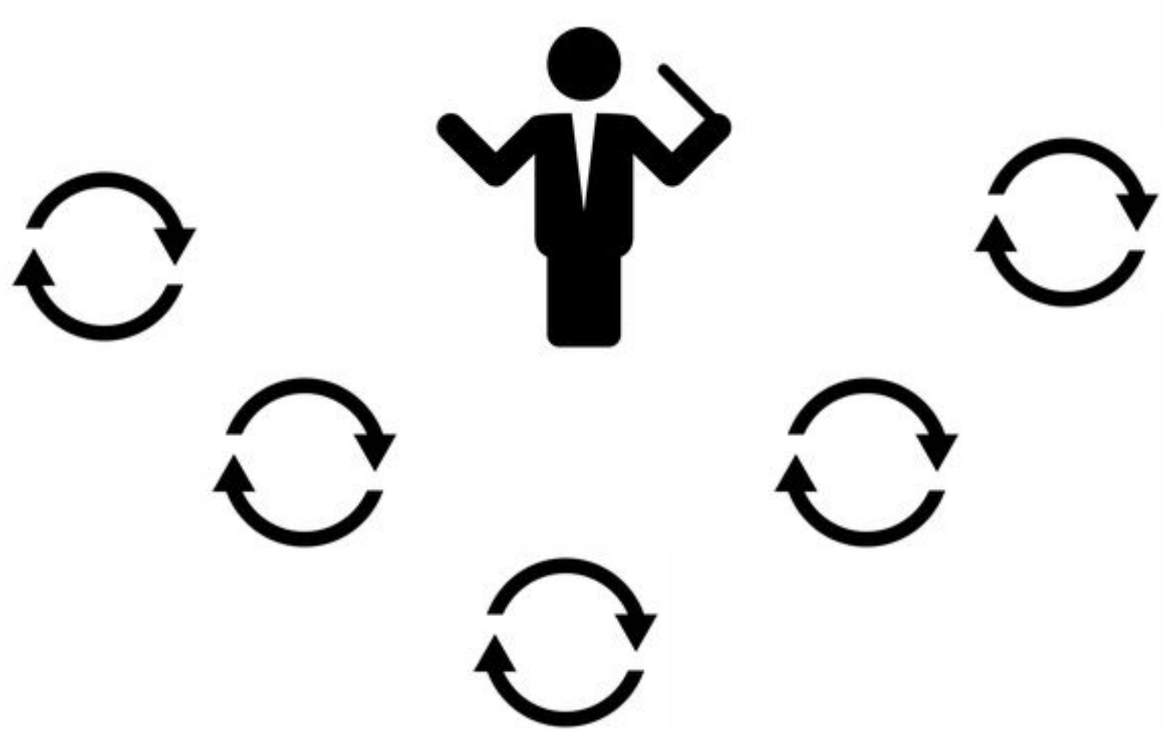
```
struct list_head {
    struct list_head *next, *prev;
};
```

Красно-черное дерево

```
struct rb_node {
    unsigned long  __rb_parent_color;
    struct rb_node *rb_right;
    struct rb_node *rb_left;
};
struct rb_root {
    struct rb_node *rb_node;
};
```

```
struct my_struct {
        struct rb_node node;
        int a;
        int b;
        const char *c;
};
struct my_struct ms1, ms2;
INIT_LIST_HEAD(&ms1);
rb_insert_color(&tree, &ms1);
list_add(&ms1, &ms2);
```

Ядро Linux. Планировщик (1/8)



Кооперативная многозадачность

voluntary yield

Вытесняющая многозадачность

mandatory preemtion

Ядро Linux. Планировщик (2/8)я

dump scheduler

O(1) scheduler

Completely Fair Scheduler

Обработка транзакций

Вычисления

Ядро Linux. Планировщик (3/8)

Вопрос (2 балла)**:**

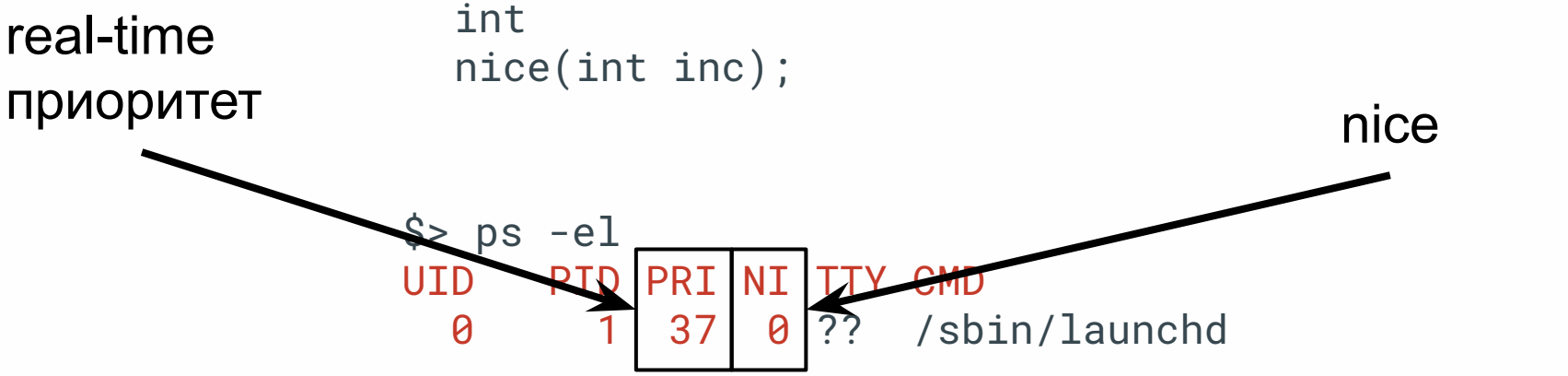
Какие есть два типа многозадачности?

Кооперативная и вытесняющая

Ядро Linux. Планировщик (4/8)

nice -- execute a utility with an altered scheduling priority
renice -- alter priority of running processes
chrt -- manipulate the real-time attributes of a process

```
int
getpriority(int which, id_t who);

int
setpriority(int which, id_t who, int prio);

int
nice(int inc);
```



Timeslice / quantum

запуск ⟵ · ⟶ прерывание

Ядро Linux. Планировщик (5/8)

Задача: конвертация видео

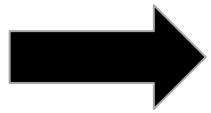
FFmpeg

Быстрее закончится

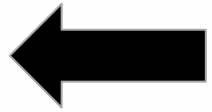
Потери кешей, позже кончится

Пример

Приоритет



Приоритет



Задача: шахматы с ИИ

Низкая интерактивность

Быстрая реакция

Ядро Linux. Планировщик (6/8)

Квант времени в Linux - относительная величина

Процесс может не использовать квант целиком

Ядро Linux. Планировщик (7/8)

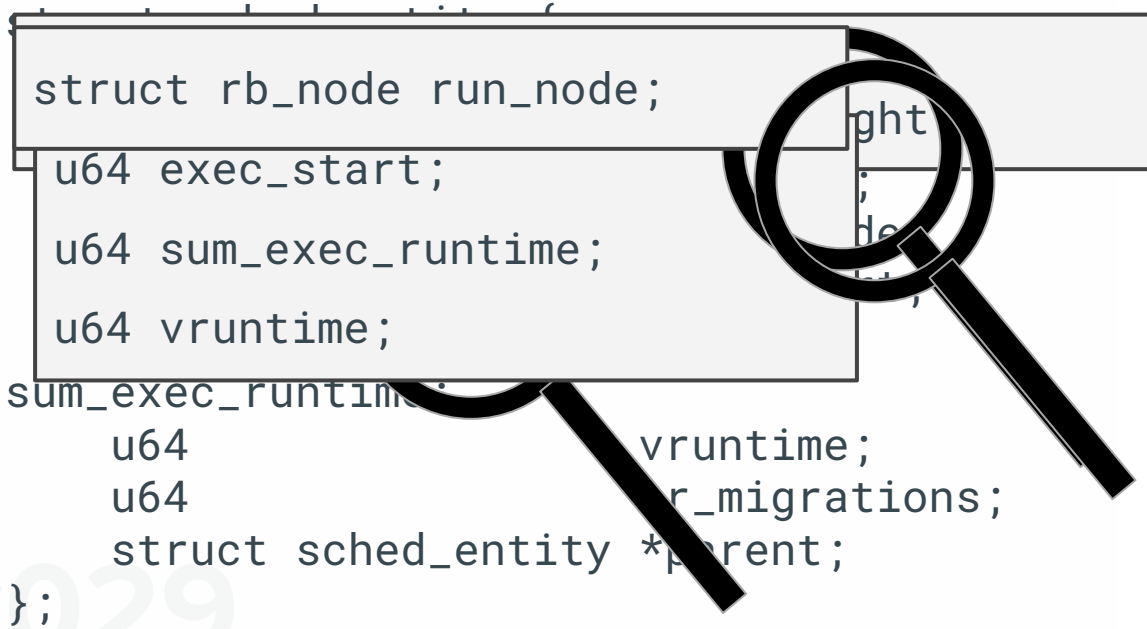
Идеальный планировщик

N - число процессов,

идеальное деление процессора - *1/N* -> 0, бесконечно частое переключение

Complete Fair Scheduler

```
/**
 * 30.09.2018.
 * 36 lines.
 */
struct rb_node run_node;
u64 exec_start;
u64 sum_exec_runtime;
u64 vruntime;
sum_exec_runtim
    u64                     vruntime;
    u64                     r_migrations;
    struct sched_entity *parent;
};
```



$$= \frac{1024}{1.25^{nice}}; P_j = \frac{W_j}{\sum_{i=1}^{n} W_i}$$

Пример:

$$nice1 = 1, nice2 = 0$$

$$weight1 = 820, weight2 = 1024$$

$$P_1 = 55$$

Ядро Linux. Планировщик (8/8)

Сортировка слиянием через корутины

Задание №1

#031

25

Баллов
за задание

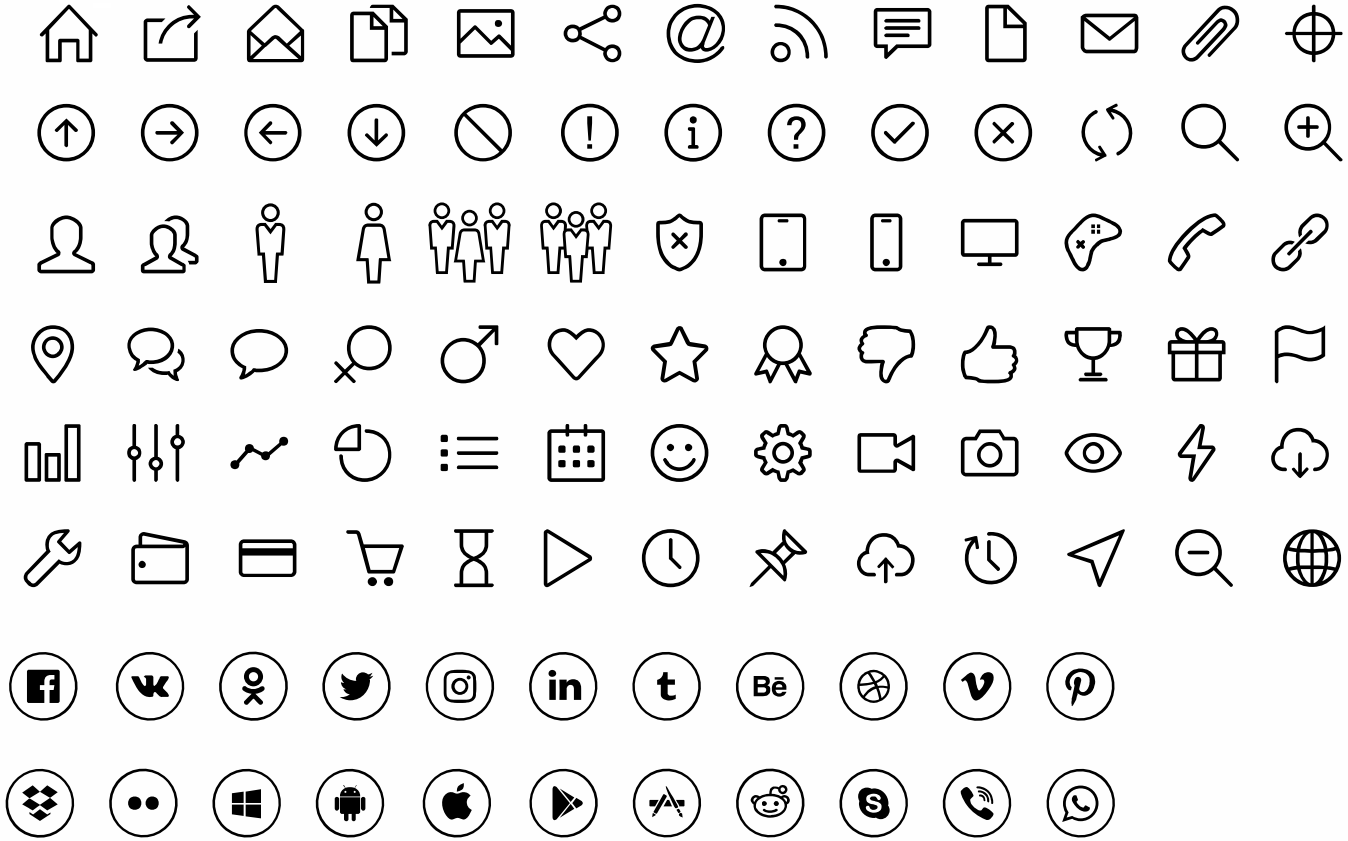
25.03.21

Срок
сдачи

- 15 баллов: после каждой выполненной строки выполняется переключение.
- 20 баллов: каждой из N корутин выдается по T / N миллисекунд, где T - target latency, подаваемое на вход. После каждой строки проверять, что у процесса кончилось время. Если да, то переключать.
- 25 баллов: тоже, что на 15, но чтение с диска должно быть асинхронным. См aio_read.

СПАСИБО
ЗА ВНИМАНИЕ

Иконки



Цвета

#FB2B38

#000000

#535353

#74777B