Архитектура операционной системы

Введение

Правила

- В начале каждой лекции 5-10 минутный БЛИЦ по материалам прошлой лекции
- Практика: stepik+review
- Вопросы!
  - нет вопросов = все понятно
- Регистрация и дальнейшая переписка:
  - class-os(a)osll.ru
    - ФИО
    - stepik-id
    - github login

Рейтинг

4.04+	Практика	C, Assembly, Tools, and Bootstrapping	1
8.04	БЛИЦ		0.4
11.04+	Практика	Memory Management	1
15.04	БЛИЦ		0.4
18.04+	Практика	User-Level Environments	1
22.04	БЛИЦ		0.4
29.04	БЛИЦ		0.4
2.05+	Практика	Самостоятельная работа (1780) простой сертификат	0.8
6.05	БЛИЦ		0.4
9.05+	Практика	Самостоятельная работа (1780) сертификат с отличием	1
13.05	БЛИЦ		0.4
16.05+	Практика	Preemptive Multitasking	1
20.05	БЛИЦ		0.4
23.05+	Практика	Spawn and Shell	1
27.05	БЛИЦ		0.4
		Итого	**10**

Экзамена нет

```
Перевод 5-балльную систему:

0, 1, 2, 3 -- 2 балла,
4, 5 -- 3 балла,
6, 7 -- 4 балла,
8, 9, 10 -- 5 баллов.
```

Вопросы?

О курсе

- Цели:
  - Понимание архитектуры ОС
  - Опыт программирования компонент ОС

- Пререквизиты:
  - Владение Linux на уровне пользователя
  - Навык написания и чтения кода на C
  - Знание архитектуры ЭВМ
  - Способность понимать технические тексты на английском

Основные источники

- MIT 6.828 Operating System Engineering
- xv6 OS sources
- K&R Язык программирования Си
- Intel X86 Arch manuals
- Linux man pages
- Linux sources
-

Что такое операционная система и зачем она нужна?

Предпосылки и генезис

- Загрузчики, мониторы, библиотеки
- Пакетный режим(аналогия конвейера): Процессор, Терминалы, IO
- Разделение времени, многозадачность: вытесняющая/невытесняющая
- Управление пользователями
- Реальное время
- Файловые системы и системы хранения
- Распределенные вычисления
- Мобильные системы
- ...

Истоки «Стандартизации»

- POSIX(UNIX)
  - Процесс и файл
  - Одна программа=одна функция + оболочка
  - Ядро / пространство пользователя
  - Hardware agnostic … язык!
  - Файл это все!

Ядро

- Монолитное
- Микрокернел
- Экзокернел
- Модульное

Абстракция для приложений

```
int fd = open("/var/log/syslog",1);

printf("Hello");

int pid = fork();
```

Системный вызов

```
mov eax, 4
mov ebx, 1
mov ecx, num
mov edx, 5
int 80h
```

ОС с точки зрения приложений

- Абстракция оборудования для удобства и переностимости
- Совместное использование процессоров группой приложений
- Изоляция ошибок приложений друг от друга
- Переносимость данных между приложениями

Основные абстракции ОС

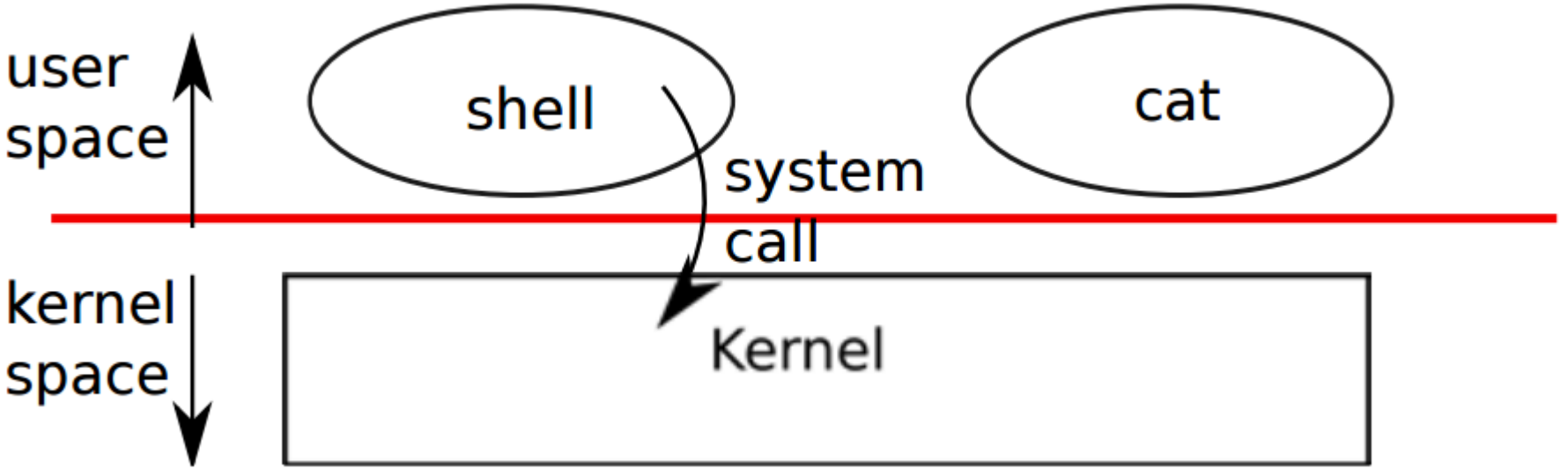
- Процесс (поток)
- Файл, файловая система
- Сокет
- Память [процесса]
- Устройство

Сервисы предоставляемые ОС

- Управление процессами
- Управление памятью
- Файлы, их содержимое, каталоги и директории
- Модель безопасности
- Межпроцессное взаимодействие
- Разное:
  - Терминалы, сети, таймеры, периферия

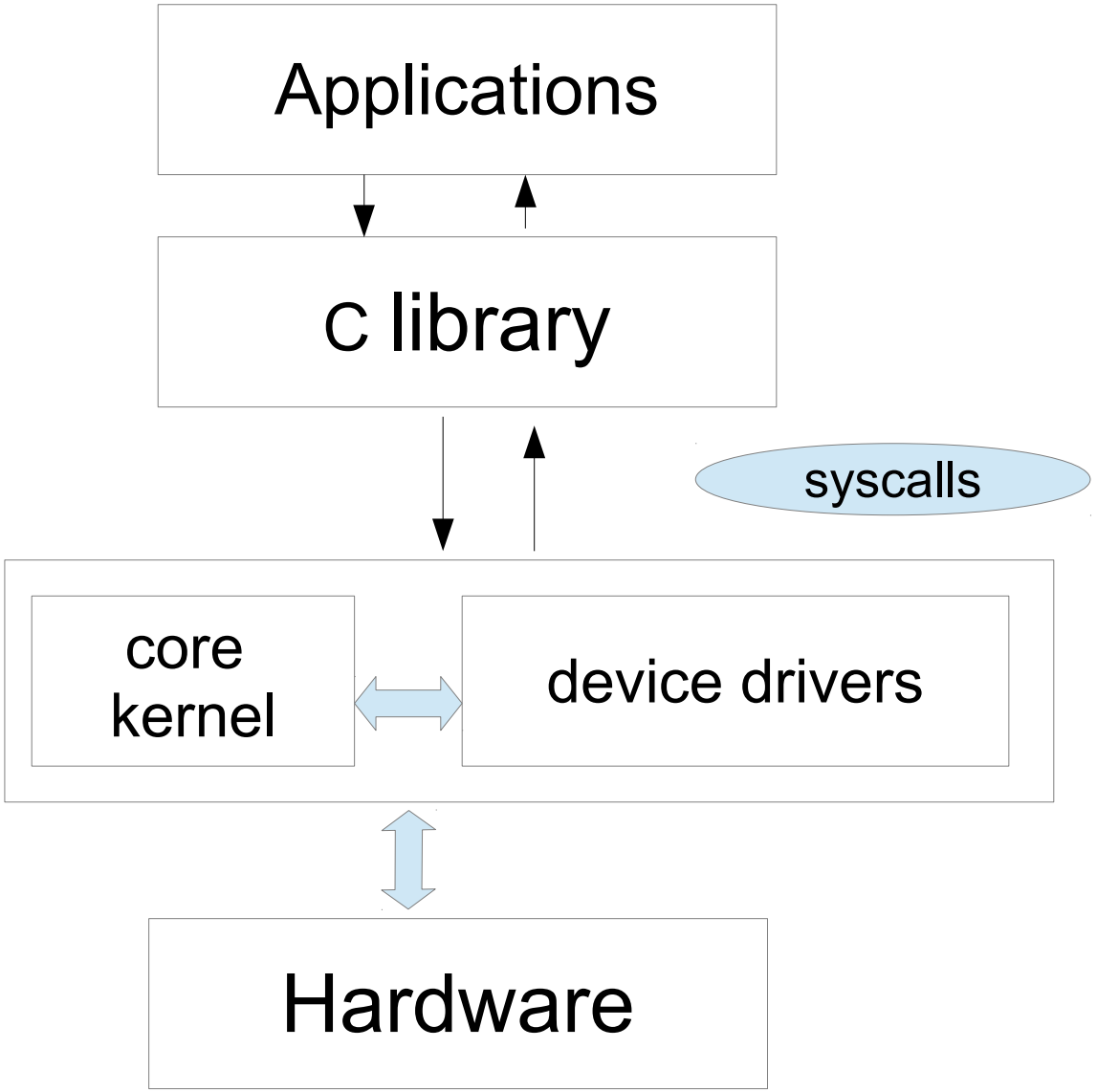
Что такое ОС?

- Библиотека управления оборудованием?
- Реализация абстракции вычислительной машины?

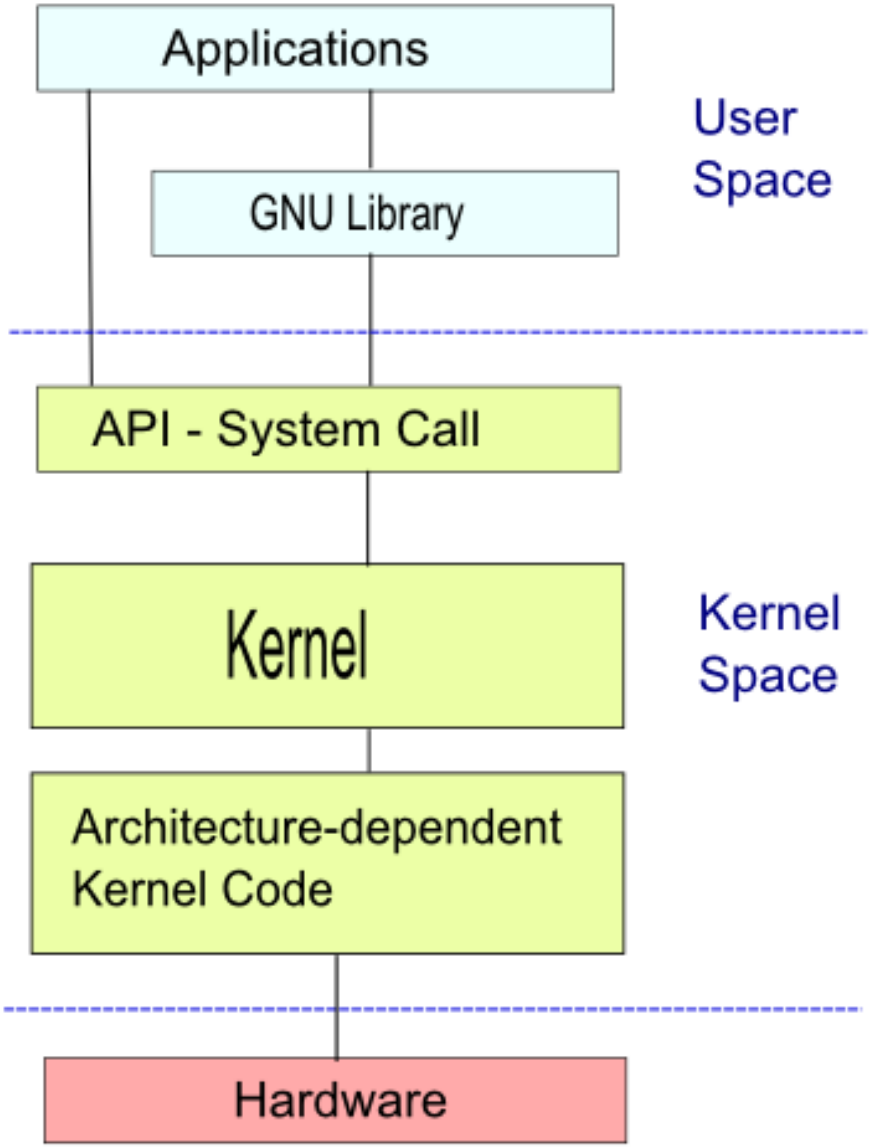


Аппаратура

Упрощенная архитектура Linux

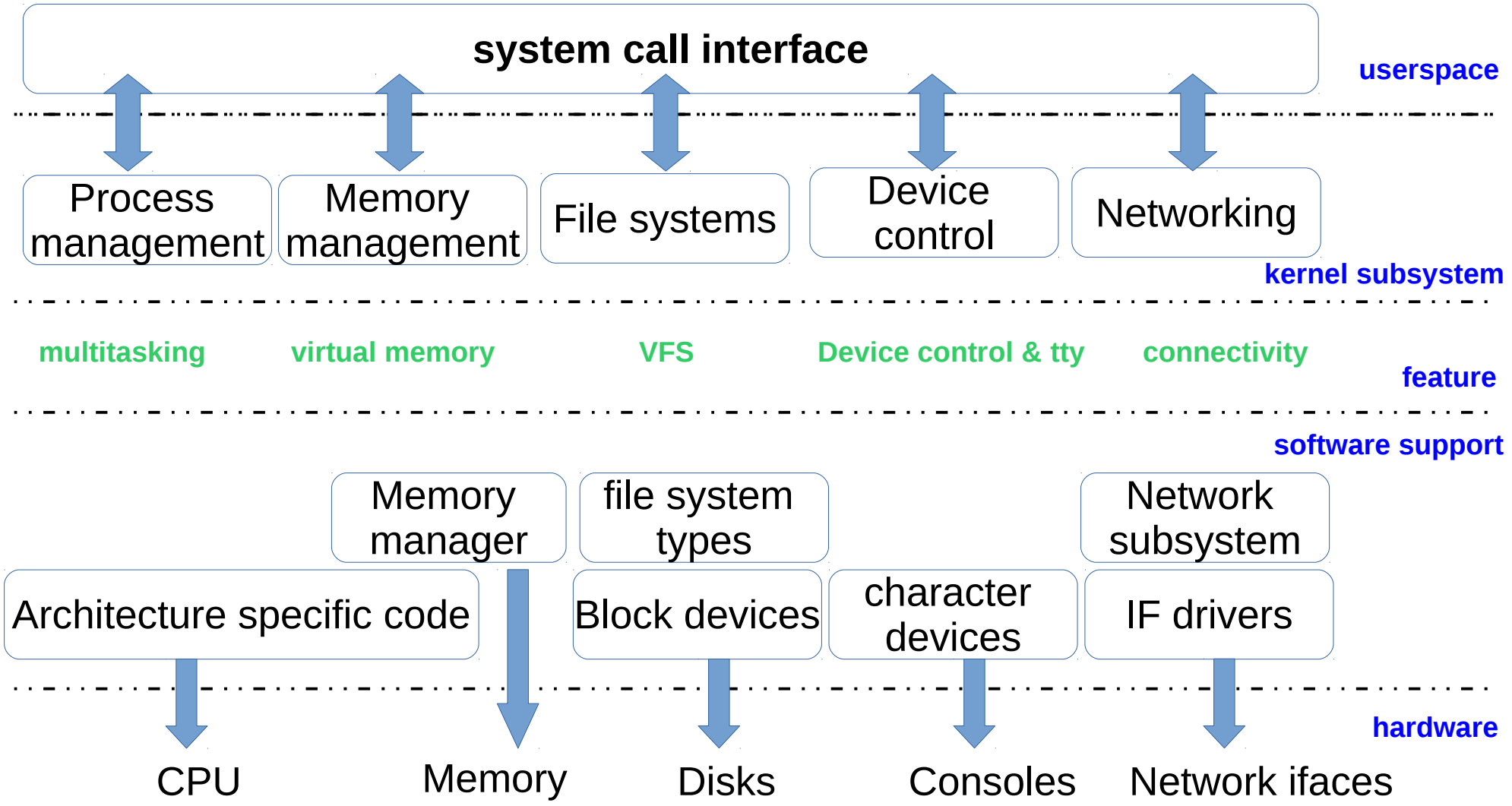


Ядро Linux (1/2)



- Kernel main parts:
  - scheduler
  - VMM
  - IPC

Ядро Linux (2/2)



Сколько системных вызовов нужно?

System call	Description
fork()	Create process
exit()	Terminate current process
wait()	Wait for a child process to exit
kill(pid)	Terminate process pid
getpid()	Return current process's id
sleep(n)	Sleep for n seconds
exec(filename, *argv)	Load a file and execute it
sbrk(n)	Grow process's memory by n bytes
open(filename, flags)	Open a file; flags indicate read/write
read(fd, buf, n)	Read n byes from an open file into buf
write(fd, buf, n)	Write n bytes to an open file
close(fd)	Release open file fd
dup(fd)	Duplicate fd
pipe(p)	Create a pipe and return fd's in p
chdir(dirname)	Change the current directory
mkdir(dirname)	Create a new directory
mknod(name, major, minor)	Create a device file
fstat(fd)	Return info about an open file
link(f1, f2)	Create another name (f2) for the file f1
unlink(filename)	Remove a file

Процессы и память

- Адресное пространство
- Память (инструкции, данные, стек)
- Состояние
- Разделение времени доступа к CPU

Ввод вывод и файловые дескрипторы

- FD – ключ (уникальное имя) управляемого ядром объекта, с которым приложение может выполнять операции ввода-вывода
- Объекты I/O
  - Файлы
  - Каналы
  - Устройства
  - Терминалы
  - Сокеты
  - ….

Пример: каналы

- Буфер в памяти
- 2 файловых дескриптора (концы FIFO)

Файловая система

- Правила именования бинарных объектов
- Правила хранения данных (блоками)
- Всевозможные атрибуты
  - Время доступа
  - Права
  - Режимы
  - Отображения на устройства

Как все это загружается и работает?