ИТМО ВТ
кафедра вычислительной техники

Системное программное обеспечение.

Программирование на языке Си

Дергачёв А. М.
Жмылёв С. А.
Афанасьев Д. Б.

Версия 12

ИТМО ВТ
кафедра вычислительной техники

Системное программное обеспечение:

- Языки СПО
- Системные вызовы
- Ввод-вывод
- Процессы и потоки

ИТМО ВТ
кафедра вычислительной техники

Программный интерфейс

Любая программа принимает аргументы и переменные окружения

Код возврата – число, отображающее корректность завершения

ИТМО ВТ
кафедра вычислительной техники

Структура программ

```
int main(int argc, char *argv[], char *envp[]) {

    /* … */

    return 0;
}
```

ИТМО ВТ
кафедра вычислительной техники

Структура программ

```
#include <stdlib.h>
int main(int argc, char *argv[], char *envp[]) {

    /* … */

    return EXIT_SUCCES;
}
```

ИТМО ВТ
кафедра вычислительной техники

Компиляция программ

```
# gcc -c program.c
# gcc -o program program.o

# gcc -o program program.c
```

ИТМО ВТ
кафедра вычислительной техники

Код возврата

```
# rm -f /etc/passwd 2<&-
# echo $?
1
# echo Hello, world!
Hello, world!
# echo $?
0
```

ИТМО ВТ
кафедра вычислительной техники

Популярная ошибка

Использование void main() недопустимо!

```
# cat void.c
void main(void) {}
# ./void
# echo $?
16
```

ИТМО ВТ
кафедра вычислительной техники

Роль системы

- Многозадачность
- Виртуализация памяти
- Управление устройствами
- Обработка прерываний
- Расширение набора операций, доступных программам.

ИТМО ВТ
кафедра вычислительной техники

Системные вызовы

- Обращение к функции ядра системы
- Использование аппаратного обеспечения через единый API

ИТМО ВТ
кафедра вычислительной техники

Системные вызовы[2]

- Имеют интерфейс libc
- Имеют привилегии ядра операционной системы

ИТМО ВТ
кафедра вычислительной техники

Обращение к системе

```
…
/* завершение программы с кодом 2 */
_exit(2);
…
```

ИТМО ВТ
кафедра вычислительной техники

Обращение к системе

```
.globl _start
_start:
pushq $2
movq $1, %rax
int $0x80
```

ИТМО ВТ
кафедра вычислительной техники

Файл

Что такое файл?

Everything is a file!

Кроме потоков и ядра

ИТМО ВТ
кафедра вычислительной техники

Дескриптор файла

- это целое неотрицательное число, абстрагирующее процессы от файлов, с которыми они работают.

ИТМО ВТ
кафедра вычислительной техники

Таблица дескрипторов открытых файлов

Номер дескриптора	Имя файла	Режим доступа
0	/dev/tty	O_RDWR | O_LARGEFILE
1	/dev/tty	
2	/dev/tty	
3	/etc/passwd	O_RDONLY
4	/dev/mtdblock3	O_RDWR
...		
255		

ИТМО ВТ
кафедра вычислительной техники

Стандартные потоки ввода-вывода

```
# grep FILENO /usr/include/unistd.h
#define STDIN_FILENO    0
#define STDOUT_FILENO   1
#define STDERR_FILENO   2
```

ИТМО ВТ
кафедра вычислительной техники

open(2)

```
int open(
    const char *path,          /* путь к файлу */
    int oflag,                 /* режим доступа */
    /* mode_t mode */          /* права доступа */
);
```

Функция возвращает номер дескриптора или код ошибки

ИТМО ВТ
кафедра вычислительной техники

Основные режимы доступа

O_RDONLY — только для чтения

O_WRONLY — только для записи

O_RDWR — для записи/чтения

O_CREAT — создать, если не существует

O_APPEND — запись с конца

O_TRUNC — запись с начала

ИТМО ВТ
кафедра вычислительной техники

lseek(2)

```
off_t lseek(
    int fildes,    /* номер открытого файла */
    off_t offset,        /* смещение позиции */
    int whence                    /* действие */
);
```

Функция возвращает полученное смещение в байтах или код ошибки

ИТМО ВТ
кафедра вычислительной техники

read(2)

```
ssize_t read(
        int fildes,    /* номер открытого файла */
        void *buf,                  /* буфер чтения */
        size_t nbyte          /* количество байт */
);
```

Функция возвращает количество прочитанных байт или код ошибки

ИТМО ВТ
кафедра вычислительной техники

write(2)

```
ssize_t write(
        int fildes,             /* номер дескриптора */
        const void *buf,        /* буфер записи */
        size_t nbyte            /* количество байт */
);
```

Функция возвращает количество записанных байт или код ошибки

ИТМО ВТ
кафедра вычислительной техники

close(2)

```
int close(
        int fildes          /* номер дескриптора */
);
```

Функция возвращает 0
или код ошибки

ИТМО ВТ
кафедра вычислительной техники

dup(2) и dup2(2)

```
int dup(
    int fildes     /* номер открытого файла */
);
int dup2(int fildes, int fildes2);
```

Функция возвращает номер нового дескриптора или код ошибки

ИТМО ВТ
кафедра вычислительной техники

stat(2)

```
int stat(
    const char *restrict path, /* путь к файлу*/
    struct stat *restrict buf          /* результат */
);
```

Функция возвращает 0
или код ошибки

ИТМО ВТ
кафедра вычислительной техники

Ошибки в системных вызовах

Код возврата системного вызова:

< 0 – ошибка в ходе выполнения вызова

= 0 – успешное выполнение

> 0 – результат успешного выполнения

ИТМО ВТ
кафедра вычислительной техники

Стандартизация ошибок

- Унификация ошибочных кодов
- Переменная errno
- Функция perror(3)
- Функция strerror(3)

ИТМО ВТ
кафедра вычислительной техники

Пример ошибки

```
if (read(7, buf, 1) < 0) {
    fprintf(stderr, "%d ", errno);
    perror("read");
    _exit(1);
}
```

```
9 read: Bad file number
```

ИТМО ВТ
кафедра вычислительной техники

Заголовочные файлы

- unistd.h — объявления UNIX
- stdio.h — стандартный ввод/вывод
- fcntl.h — операции с файлами
- sys/types.h — системные типы
- sys/stat.h — системные статусы

ИТМО ВТ
кафедра вычислительной техники

Пример чтения/записи

```
#include <unistd.h>
int main(int argc, char *argv[]) {
   int bytes;
   char buf[256];
   while((bytes = read(STDIN_FILENO, buf, sizeof(buf))) > 0) {
      if (write(STDERR_FILENO, buf, bytes) < 0) {
         return 1;
      }
   }
   return bytes;
}
```

ИТМО ВТ
кафедра вычислительной техники

Makefile

```
PROJS=main
CC=gcc
CFLAGS=-m64

all: $(PROJS)
	@echo Done!

$(PROJS):
	$(CC) $(CFLAGS) -o $@ $(@:=.c)
```

ИТМО ВТ
кафедра вычислительной техники

Утилита make

```
# make
gcc -m64 -o main main.c
Done!
# ./main
Hello, world!
```

ИТМО ВТ
кафедра вычислительной техники

Ввод-вывод со смещением

```
ssize_t pread(int fildes, void *buf, size_t
nbyte, off_t offset);

ssize_t pwrite(int fildes, const void *buf,
size_t nbyte, off_t offset);
```

Функции возвращают количество байт или код ошибки

ИТМО ВТ
кафедра вычислительной техники

Рассеянный ввод-вывод

```
ssize_t readv(
int fildes,                    /* дескриптор файла */
const struct iovec *iov, /* массив структур */
int iovcnt          /* размер массива структур */
);
```

Функция возвращает количество введённых байт или код ошибки

ИТМО ВТ
кафедра вычислительной техники

Структура iovec (I/O vector)

```
#include <sys/uio.h>

typedef struct iovec {
        void    *iov_base;      /* адрес начала */
        size_t  iov_len;        /* длина сегмента */
} iovec_t;
```

ИТМО ВТ
кафедра вычислительной техники

Сброс кэшей

```
void sync(
        void        /* не принимает аргументов */
);
```

Код возврата функции не информативен

ИТМО ВТ
кафедра вычислительной техники

Работа с файловыми дескрипторами

```
int fcntl(
    int fildes,                    /* дескриптор */
    int cmd,                       /* команда */
    ...      /* переменное число аргументов */
);
```

Код возврата функции интерпретируется в зависимости от команды

ИТМО ВТ
кафедра вычислительной техники

Команды fcntl(2)

F_DUPFD / F_DUP2FD — аналог dup/dup2

F_FREESP — освободить место

F_GETFD / F_SETFD — флаг close on exec

F_GETFL / F_SETFL — флаги доступа

F_GETLK / F_SETLK — блокировка файла

F_GETLKW / F_SETLKW

F_RDLCK / F_WRLCK / F_UNLCK

ИТМО ВТ
кафедра вычислительной техники

Проверка доступа

```
int access(const char *path, int amode);
```

R_OK – чтение

W_OK – запись

X_OK – исполнение

F_OK – существование

ИТМО ВТ
кафедра вычислительной техники

Изменение прав доступа

```
int chmod(const char *path, mode_t mode);
int fchmod(int fildes, mode_t mode);
```

S_ISUID	04000
S_IRWXU	00700
(S_ISUID | S_URWXU)	04700

ИТМО ВТ
кафедра вычислительной техники

Изменение владельца

```
int chown(const char *path, uid_t owner,
    gid_t group);
int fchown(int fildes, uid_t owner,
    gid_t group);
```

Функции возвращают 0

либо код ошибки

ИТМО ВТ
кафедра вычислительной техники

Маска создания файла

```
mode_t umask(
    mode_t cmask        /* значение маски */
);
```

Возвращает предыдущее значение маски

Как получить текущее значение?

ИТМО ВТ
кафедра вычислительной техники

Работа с ссылками

```
int link(
    const char *existing,       /* путь к файлу */
    const char *new             /* путь к ссылке */
);
int unlink(const char *path);
```

Функции возвращают 0

либо код ошибки

ИТМО ВТ
кафедра вычислительной техники

Символьные ссылки

```
int symlink(const char *name1,
   const char *name2);


ssize_t readlink(
   const char *restrict path,          /* ссылка */
   char  *restrict buf,                 /* буфер */
   size_t bufsiz                /* размер буфера */
);
```

ИТМО ВТ
кафедра вычислительной техники

Работа с каталогами

```
int mkdir(
   const char *path,        /* путь к каталогу */
   mode_t mode              /* режим доступа */
);
int rmdir(const char *path);
```

Функции возвращают 0

либо код ошибки

ИТМО ВТ
кафедра вычислительной техники

Рабочая директория

```
int chdir(const char *path);
int fchdir(int fildes);
```

getcwd(3) возвращает указатель на буфер либо -1 и имеет прототип:

```
char *getcwd(char *buf, size_t size);
```

ИТМО ВТ
кафедра вычислительной техники

Чтение каталогов

```
DIR *opendir(const char *dirname);

struct dirent *readdir(DIR *dirp);

void rewinddir(DIR *dirp);

int closedir(DIR *dirp);
```

ИТМО ВТ
кафедра вычислительной техники

Структура dirent

```
typedef struct dirent {
   ino_t d_ino;          /* номер индексного
                                  дескриптора */
   off_t d_off;        /* смещение от начала */
   unsigned short reclen;  /* длина записи */
   char d_name[];              /* имя файла */
} dirent_t;
```

ИТМО ВТ
кафедра вычислительной техники

Файлы устройств

```
int mknod(
   const char *path,          /* путь к файлу */
   mode_t mode,   /* режим доступа и тип */
   dev_t dev                  /* устройство */
);
```

Функция возвращает 0

или код возврата

ИТМО ВТ
кафедра вычислительной техники

Модель памяти

0xFFFFFFFF

0xC0000000

0x00000000

ИТМО ВТ
кафедра вычислительной техники

Выделение памяти

Расширение сегмента данных:

```
int brk(void *endds);
void *sbrk(intptr_t incr);
```

Выделение новых сегментов из анонимной памяти:

```
void *mmap(void *addr, size_t len, int prot,
    int flags, int fildes, off_t off);
```

ИТМО ВТ
кафедра вычислительной техники

Утилита pmap(1)

```
helios$ pmap $$
08043000      20K   rw---    [ stack ]
08050000     552K   r-x--    /usr/bin/bash
080E9000      76K   rwx--    /usr/bin/bash
080FC000     300K   rwx--    [ heap ]
FEB20000      64K   rwx--    [ anon ]
FEB40000      56K   r-x--    /lib/module.so
```

ИТМО ВТ
кафедра вычислительной техники

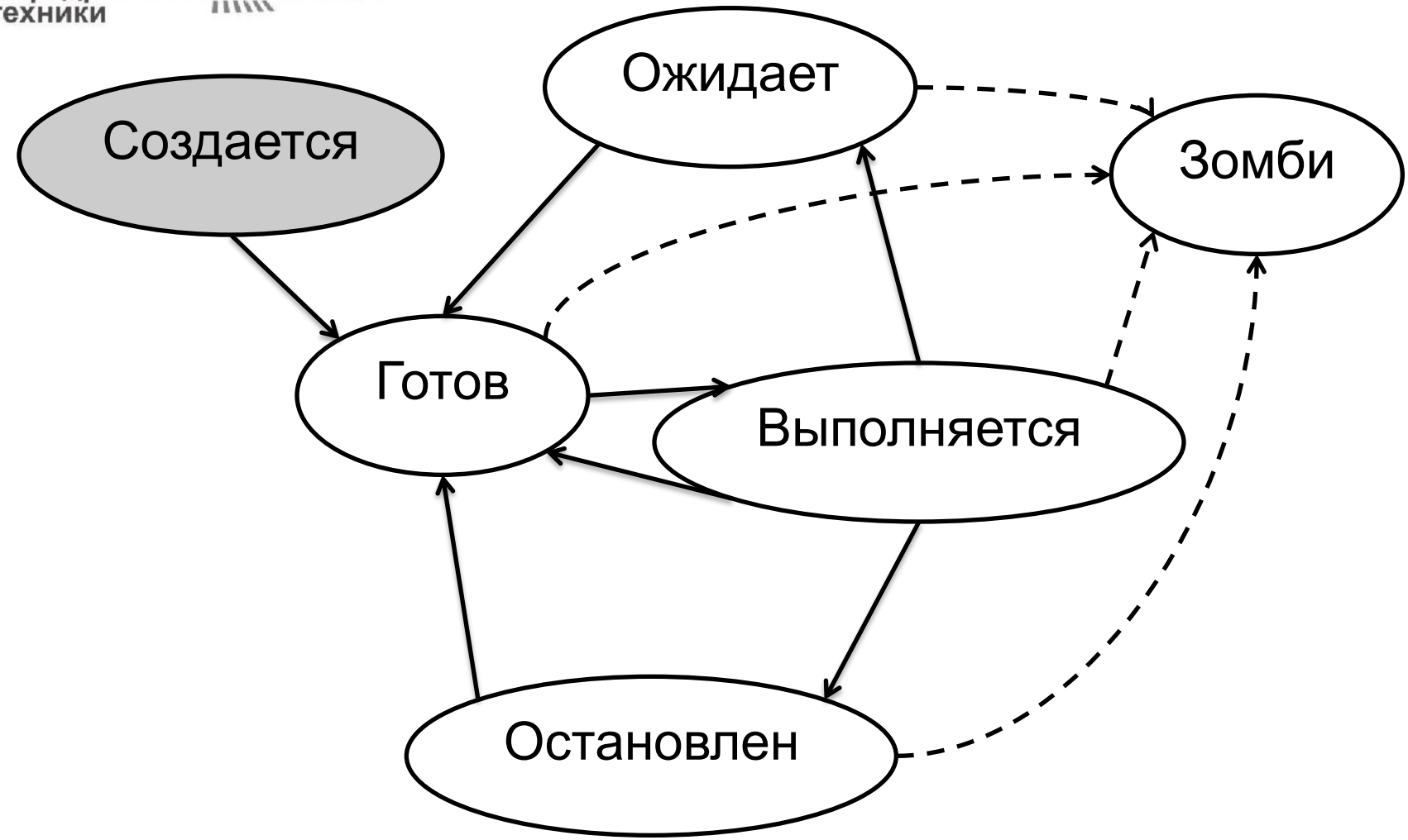
Процесс

Процесс — это совокупность программы и метаинформации, описывающей её выполнение ©KorG

Выполняются параллельно и формально независимы друг от друга

ИТМО ВТ
кафедра вычислительной техники

Состояния процесса



ИТМО ВТ
кафедра вычислительной техники

Жизненный цикл

Любой процесс, кроме init, создаётся в ходе выполнения fork(2) или vfork(2)

Любой процесс умирает тогда и только тогда, когда переходит в состояние «Зомби» и родительский процесс выполняет один из вызовов: wait(2), waitpid(2), waitid(2)

ИТМО ВТ
кафедра вычислительной техники

fork(2)

```
pid_t fork(void);
```

Функция возвращает 0 дочернему процессу, идентификатор дочернего процесса – родительскому или код ошибки

ИТМО ВТ
кафедра вычислительной техники

Семейство exec

- execl – список (list) аргументов
- execlp – поиск по PATH
- execle – список аргументов и указатель на environment
- execv – указатель на аргументы (argv)
- execvp – поиск по PATH
- execve – argv и envp

ИТМО ВТ
кафедра вычислительной техники

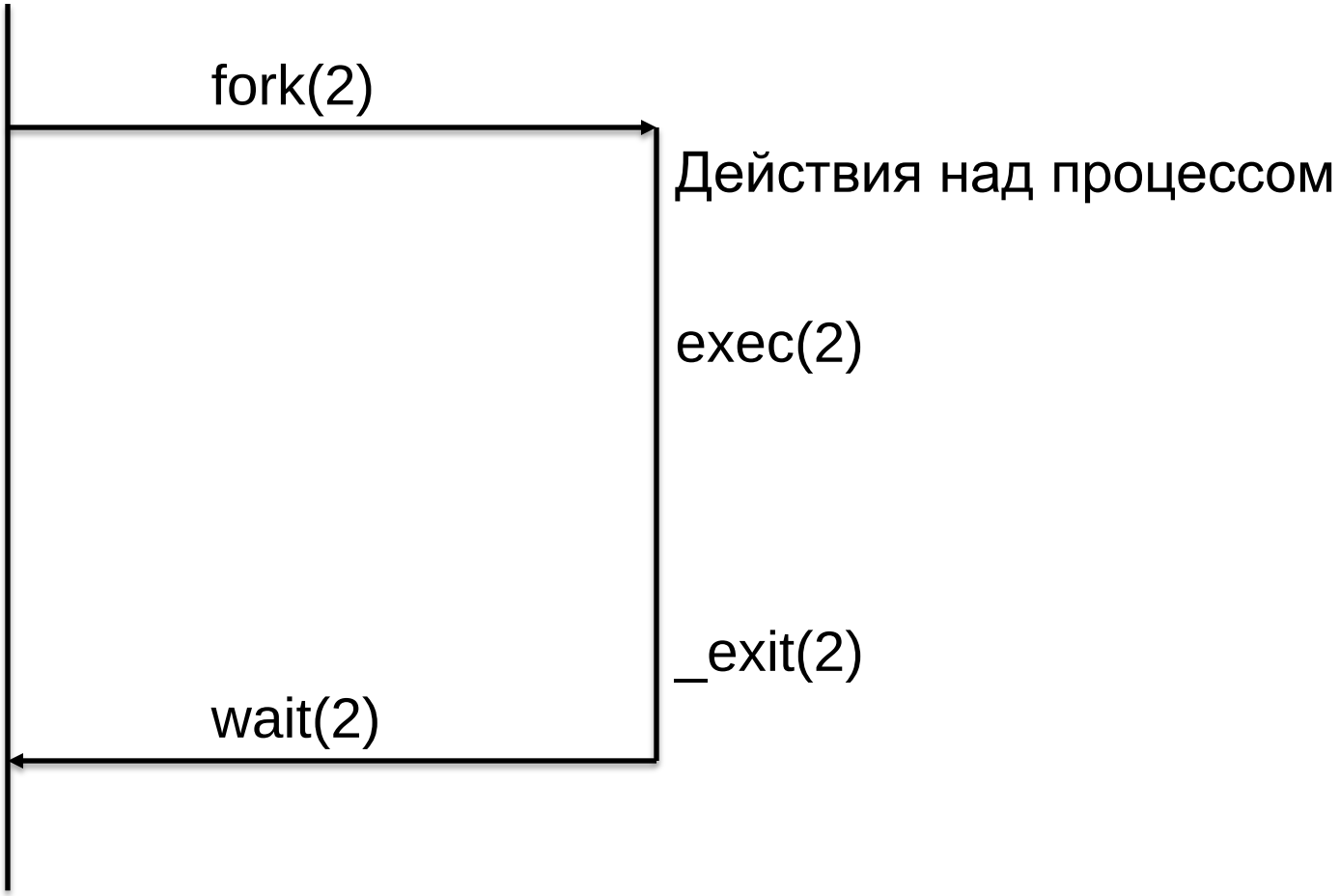
execle(2)

```
int execle(
    const char *path,          /* путь к файлу */
    const char *arg0,  /* нулевой аргумент */
    /* ... */,
    (char *)0,                          /* NULL */
    char *const envp[]          /* окружение */
);
```

Функция возвращает код ошибки

ИТМО ВТ
кафедра вычислительной техники

Модель fork/exec



ИТМО ВТ
кафедра вычислительной техники

Поток (thread)

- наименьший набор инструкций, которым может быть выделена квота процессорного времени

Имеют единое адресное пространство

Наиболее используемый стандарт – POSIX. Реализуются семейством функций с префиксом «pthread_»

ИТМО ВТ
кафедра вычислительной техники

Легковесный процесс

- абстракция на уровне ядра ОС, для описания процессов, разделяющих единое адресное пространство и системные ресурсы

В некоторых системах реализуют параллелизм потоков

ИТМО ВТ
кафедра вычислительной техники

Межпроцессное взаимодействие

- System V semaphores
- pipe
- socket
- file
- signal
- message queue
- System V shared memory

ИТМО ВТ
кафедра вычислительной техники

Межпоточное взаимодействие

- Любое межпроцессное взаимодействие
- mutex
- rwlock
- volatile переменные
- Общее адресное пространство

ИТМО ВТ
кафедра вычислительной техники

Мьютекс

– это простейший объект синхронизации, имеющий два состояния: «заблокирован» и «свободен»

ИТМО ВТ
кафедра вычислительной техники

Семафор

– это объект синхронизации, имеющий множество состояний: 0(заблокирован), 1, 2, …

Значение семафора отражает количество свободных мест

ИТМО ВТ
кафедра вычислительной техники

Сигналы

– это механизм ОС, для уведомления процесса о некотором событии.

Стандартные реакции на сигнал:

- SIG_IGN — игнорировать
- SIG_QUIT — завершиться
- SIG_ERR — дамп памяти
- SIG_HOLD — останов процесса

ИТМО ВТ
кафедра вычислительной техники

Очереди сообщений

– это механизм передачи некоторого объёма данных между процессами, основанный на сигналах

ИТМО ВТ
кафедра вычислительной техники

Каналы pipe

- Именованные
  - Имя: путь на ФС
- Неименованные
  - Имя: номер дескриптора файла

Данные не хранятся нигде!

ИТМО ВТ
кафедра вычислительной техники

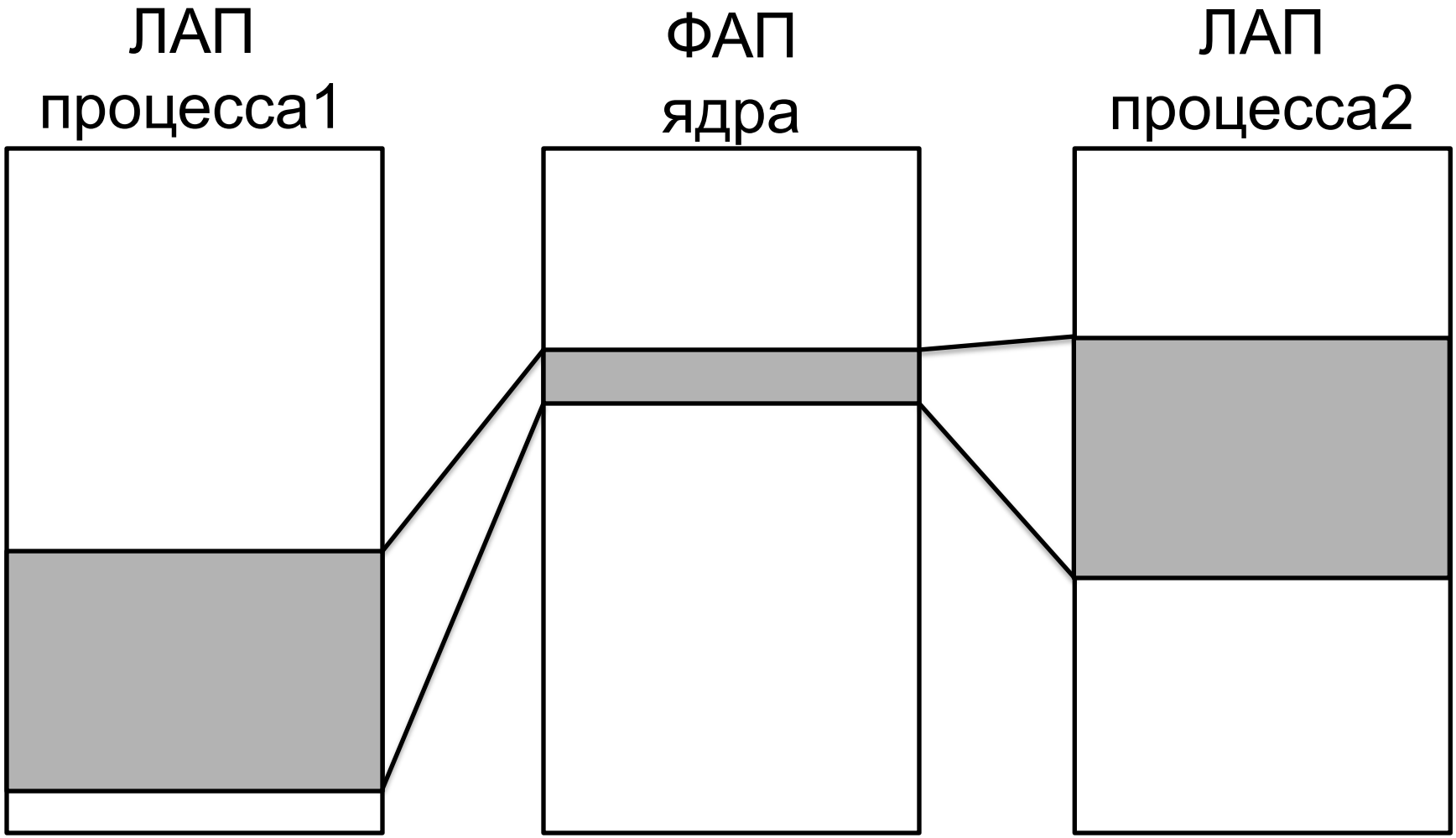
Разделяемая память

– механизм для работы с общей для нескольких процессов памятью

Сегмент разделяемой памяти хранится до его принудительного удаления

ИТМО ВТ
кафедра вычислительной техники

Общий сегмент



ИТМО ВТ
кафедра вычислительной техники

Сокеты

Различаются по семействам адресов:

- UNIX domain
- Сетевые
  - BSD сокеты
- Прочие

ИТМО ВТ
кафедра вычислительной техники

UNIX domain socket

Адрес: путь к файлу с типом socket

Данные: двусторонний pipe

ИТМО ВТ
кафедра вычислительной техники

BSD socket

Адрес: IP адрес + номер порта

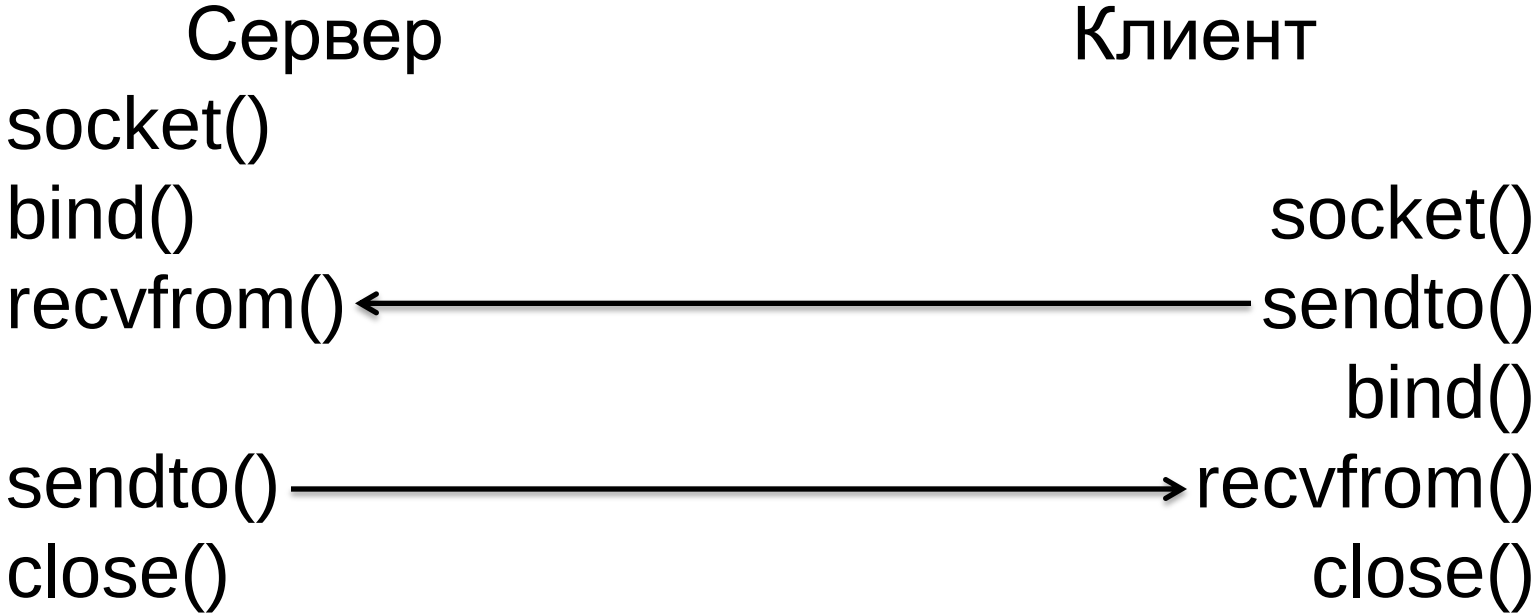
Где хранятся данные?

ИТМО ВТ
кафедра вычислительной техники

Обмен по TCP

кафедра вычислительной техники

Обмен по UDP



ИТМО ВТ
кафедра вычислительной техники

socket(2)

```
int socket(
        int domain,      /* семейство адресов */
        int type,                    /* тип */
        int protocol               /* протокол */
);
```

Функция возвращает номер дескриптора или код ошибки

ИТМО ВТ
кафедра вычислительной техники

bind(2)

```
int bind(
        int s,      /* номер дескриптора сокета*/
        const struct sockaddr *name,    /* имя */
        int namelen         /* длина поля имени */
);
```

Функция возвращает 0
или код ошибки

ИТМО ВТ
кафедра вычислительной техники

listen(2)

```
int listen(
        int s,      /* номер дескриптора сокета */
        int backlog                 /* длина очереди */
);
```

Функция возвращает 0
или код ошибки

ИТМО ВТ
кафедра вычислительной техники

accept(2)

```
int  accept(
    int  s,        /* номер дескриптора сокета */
    struct sockaddr *addr, /* адрес клиента */
    socklen_t *addrlen     /* размер адреса */
);
```

Функция возвращает номер дескриптора или код ошибки

ИТМО ВТ
кафедра вычислительной техники

connect(2)

```
int connect(
        int s,   /* номер дескриптора сокета */
        const struct sockaddr *name,  /* имя */
        int namelen      /* длина поля имени */
);
```

Функция возвращает 0
или код ошибки

ИТМО ВТ
кафедра вычислительной техники

Механизм STREAMS

Альтернатива сокетам в System V UNIX

ИТМО ВТ
кафедра вычислительной техники

Полезные функции

- isatty(3C)
- gethostbyname(3NSL)
  gethostbyaddr(3NSL)
- htons(3SOCKET)
  htonl(3SOCKET)
  ntohs(3SOCKET)
  ntohl(3SOCKET)
- usleep(3C)

ИТМО ВТ
кафедра вычислительной техники

Удачи на экзамене!