

**Московский Физико-Технический Институт
Физтех-школа Аэрокосмических технологий
Институт Аэромеханики и Летательной Техники**



ФАКТ

**Архитектура Компьютера
и Операционные Системы.
Часть 2. Основы операционных систем**

Лекция 9: Файловая система Unix. ч2

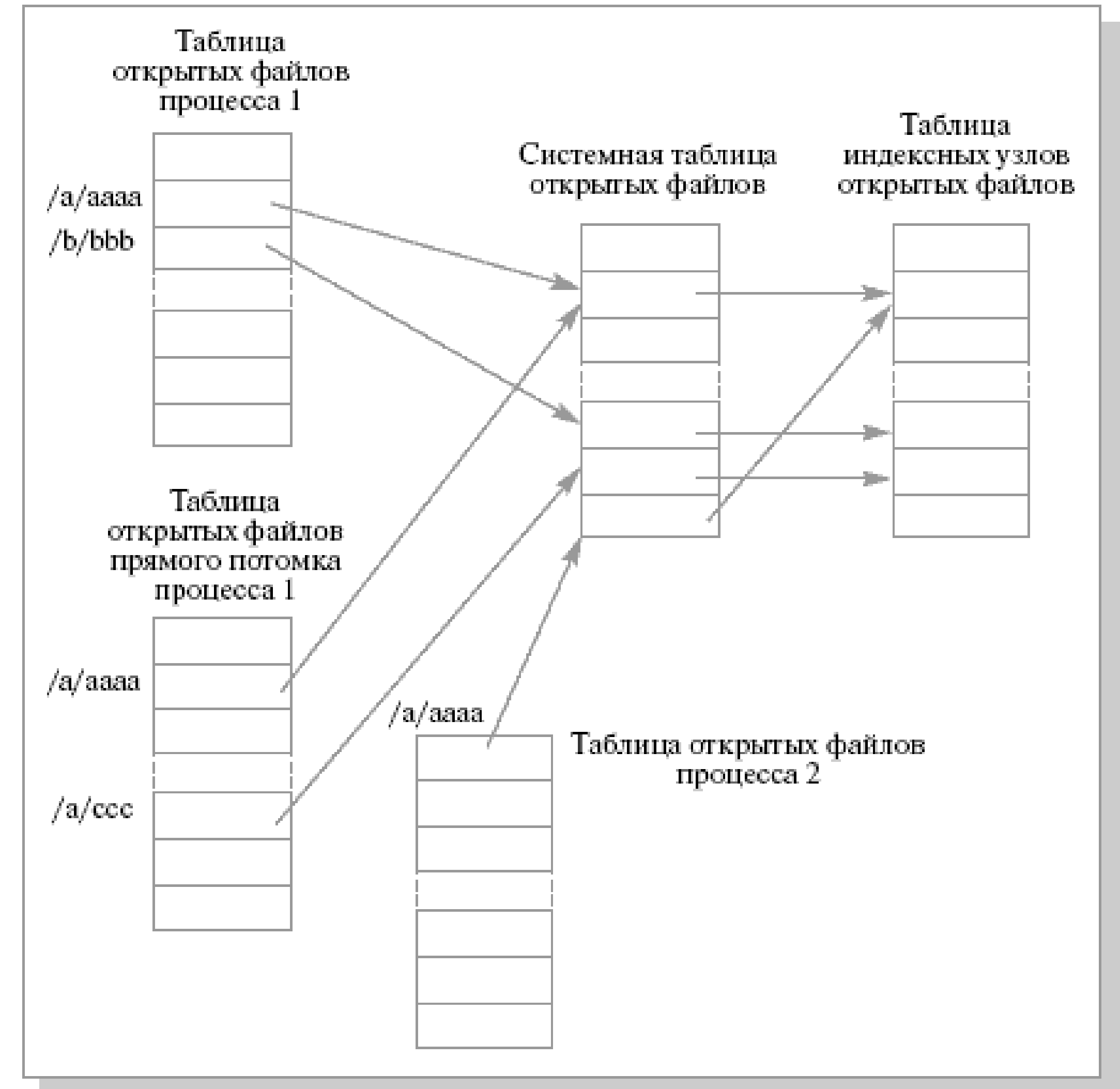
**Новиков Андрей Валерьевич
д.ф.-м.н.**

Жуковский

Данные об открытых файлах в ОС

Информация о файле, необходимая процессу:

- ❑ данные, специфичные для этого процесса;
- ❑ данные, общие для близко родственных процессов, совместно использующих файл, например, указатель текущей позиции ;
- ❑ данные, являющиеся общими для всех процессов, использующих файл, – атрибуты и расположение файла



Системные вызовы для выполнения операций над файлами

- ❑ Создание: `creat()`
- ❑ Открытие/заккрытие: `open()` / `close()`
- ❑ Чтение / запись: `read()` / `write()`
- ❑ Изменение размера: `ftruncate()`
- ❑ Изменение текущей позиции: `lseek()`

Создание файла

```
#include <fcntl.h>

int creat(
    char *path,
    int mode
);
```

Создание пустого файла. Эквивалентно вызову `open(path, O_CREAT|O_WRONLY|O_TRUNC, mode)`. Если файла с указанным именем не существовало, он будет создан и открыт только для записи. Если файл уже существовал, то он открывается также только для записи, при этом его длина уменьшается до 0, другие атрибуты сохраняются.

- ❑ **path** – путь к файлу
- ❑ **mode** – права доступа
- ❑ Возвращаемое значение:
 - `fd >= 0` – файловый дескриптор
 - `-1` – ошибка, расшифровка в **errno**

Изменение размера файла

```
#include <sys/types.h>
#include <unistd.h>

int ftruncate(
    int fd,
    size_t length
);
```

Изменение длины открытого *регулярного (обычного)* файла

- ❑ **fd** – файловый дескриптор открытого файла
- ❑ **length** – нужная длина файла в байтах
 - если меньше текущей длины – файл обрезается, информация теряется;
 - если больше текущей длины – файл дополняется нулевыми байтами до нужного размера
- ❑ Возвращаемое значение:
 - 0 – успех
 - 1 – ошибка, расшифровка в **errno**

Изменение текущей позиции

```
#include <sys/types.h>
#include <unistd.h>

off_t lseek(
    int fd,
    off_t offset,
    int whence
);
```

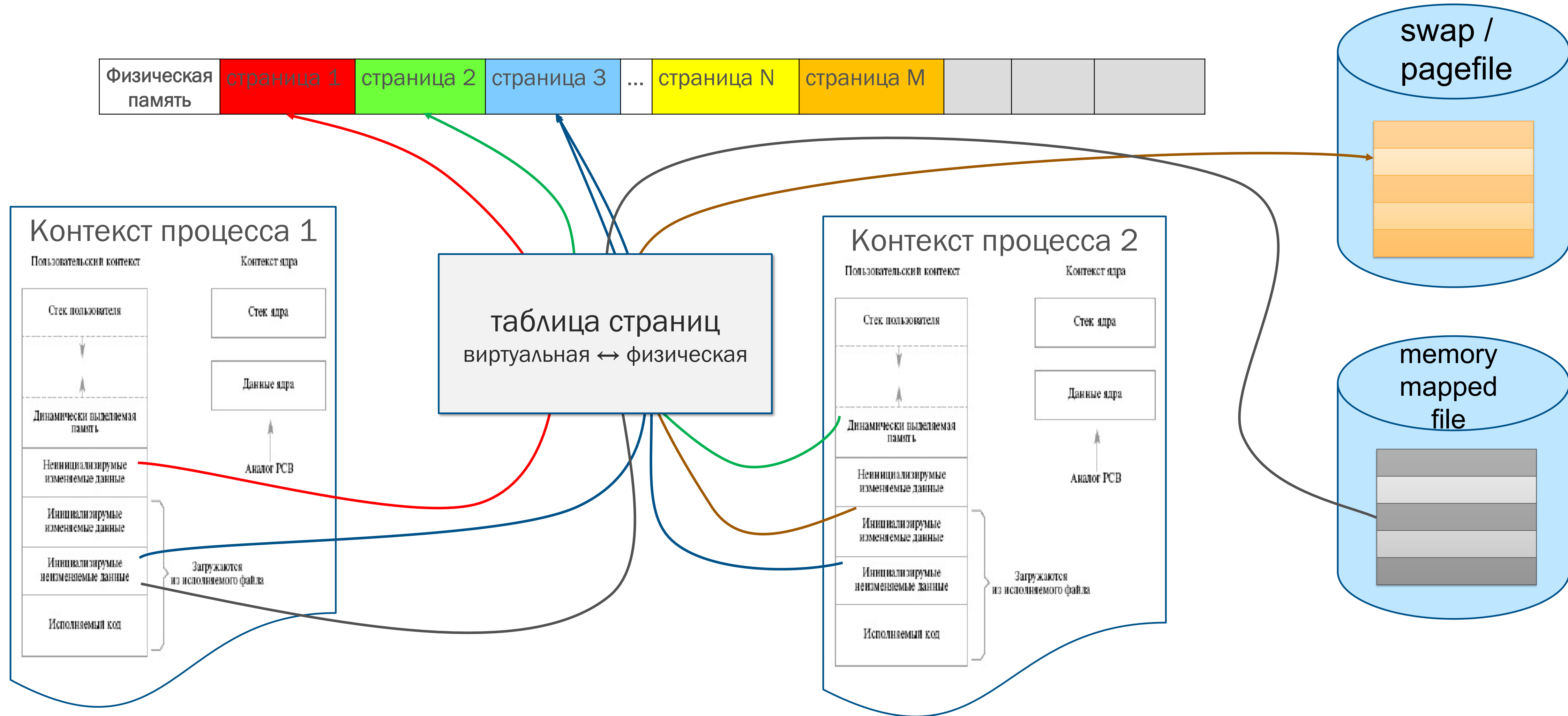
Изменяет положения указателя текущей позиции в открытом *регулярном* файле.

- ❑ **fd** – файловый дескриптор
- ❑ **offset** – сдвиг в байтах
- ❑ **whence** – откуда считается сдвиг
 - SEEK_SET – от начала
 - SEEK_CUR – от текущей позиции
 - SEEK_END – от конца
- ❑ Возвращаемое значение:
 - ≥ 0 – новая текущая позиция относительно начала файла
 - 1 – ошибка, расшифровка в **errno**

Файлы отображаемые в память

- ❑ memory mapped files
- ❑ Содержимое файла отображается непосредственно в адресное пространство процесса
- ❑ Работа с файлом как с массивом в памяти.
- ❑ Используются механизмы виртуальной памяти и страничной организации памяти

Виртуальная память



Работа с memory-mapped файлами

- ❑ 1. Открыть файл с помощью обычного системного вызова ``open()``. При этом файл отображается в «дисковое пространство».
- ❑ 2. Отобразить файл из «дискового пространства» в «адресное пространство процесса» с помощью системного вызова ``mmap()``. Возможно отобразить целиком или частично.
- ❑ 2а. После `mmap()` файл можно закрыть, выполнив системный вызов ``close()``, так как необходимая информация о расположении файла на диске уже сохранена в других структурах данных при вызове `mmap()`.
- ❑ 3. После окончания работы с содержимым файла выполнить ``munmap()``.

Отображение файла в память

```
#include <sys/types.h>
#include <sys/mman.h>

void* mmap(
    void* start, size_t length,
    int prot, int flags,
    int fd, off_t offset
);
```

Отображения предварительно открытого файла в адресное пространство вычислительной системы.

- ❑ **fd** – файловый дескриптор предварительно открытого файла, который отображаем
- ❑ **offset** – смещение в байтах от начала файла, начиная с которого отображать содержимое
- ❑ **length** – размер отображаемой части файла в байтах; length может превышать длину файла, но потом при доступе к отсутствующим данным возникнет сигнал SIGBUS
- ❑ **start** – начало области адресного пространства, в которую будет отображен файл, = NULL, если автоматический режим

Отображение файла в память

```
#include <sys/types.h>
#include <sys/mman.h>

void* mmap(
    void* start, size_t length,
    int prot, int flags,
    int fd, off_t offset
);
```

- ❑ **prot** – protection, разрешенные операции над областью памяти, в которую будет отображен файл
 - PROT_READ - разрешено чтение
 - PROT_WRITE - разрешена запись
 - PROT_EXEC – разрешено исполнение
 - PROT_NONE – запрещено всё
 - объединение с помощью "побитовое или" – " | "
- ❑ Значение параметра prot не может быть шире, чем операции над файлом, заявленные при его открытии в параметре flags системного вызова open() .
Например, нельзя открыть файл только для чтения, а при его отображении в память использовать значение prot = PROT_READ | PROT_WRITE.

Отображение файла в память

```
#include <sys/types.h>
#include <sys/mman.h>

void* mmap(
    void* start, size_t length,
    int prot, int flags,
    int fd, off_t offset
);
```

- ❑ **flags** – способ отображения файла в адресное пространство
 - MAP_SHARED – все изменения в памяти доступны другим процессам и сохраняются в файл
 - MAP_PRIVATE – все изменения в памяти доступны только локально, в файл не сохраняются
 - MAP_* - множество других флагов
- ❑ Возвращаемое значение:
 - указатель на начало памяти, куда отображён файл
 - MAP_FAILED (== void*(-1)) – ошибка, расшифровка в `errno`

Заккрытие отображения файла

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

int munmap(
    void *start,
    size_t length
);
```

Прекращает отображение файла в память и сохраняет изменения в файл (если выставлялись атрибуты PROT_WRITE, MAP_SHARED)

- ❑ **start** – адрес начала памяти отображающей файл, который возвращён mmap()
- ❑ **length** – длина области памяти, ДОЛЖНА совпадать со значением в mmap().
- ❑ Возвращаемое значение:
 - 0 – успех
 - 1 – ошибка, расшифровка в **errno**

Литература

1. В.Е. Карпов, К.А. Коньков - Основы операционных систем // М.: Национальный открытый университет «ИНТУИТ», 2016
2. В.Е. Карпов, К.А. Коньков - Основы операционных систем. Практикум // М.: Национальный открытый университет «ИНТУИТ», 2016
<https://intuit.ru/studies/courses/2249/52/info>
3. Google 😊

