Системный вызов mmap

Прототип системного вызова

```
#include <sys/types.h>
#include <unistd.h>
#include <sys/mman.h>
void *mmap (void *start, size_t length, int prot, int flags, int fd, off_t offset);
```

Описание системного вызова

Системный вызов **mmap** служит для отображения предварительно открытого файла (например, с помощью системного вызоваа open()) в адресное пространство вычислительной системы. После его выполнения файл может быть закрыт (например, системным вызовом close()), что никак не повлияет на дальнейшую работу с отображенным файлом.

Настоящее описание не является полным описанием системного вызова, а предназначено только для использования в рамках данного курса. Для получения полной информации обращайтесь к UNIX Manual.

Параметр **fd** является файловым дескриптором для файла, который мы хотим отобразить в адресное пространство (т.е. значением, которое вернул системный вызов open()).

Ненулевое значение параметра **addr** может использоваться только очень квалифицированными системными программистами, поэтому мы на наших занятиях будем асегда полагать его равным значению **NULL**, позволяя операционной системе самой выбрать начало области адресного пространства, в которую будет отображен файл.

В память будет отображаться часть файла, начиная с позиции внутри его, заданной значением параметра **offset** - смещение от начала файла в байтах, и длиной равной значению параметра **length** (естественно, тоже в байтах). Значение параметра **length** можно указать и существенно большим, чем реальная длина от позиции **offset** до конца существующего файла. На поведении системного вызова это никак не отразится, но в дальнейшем при попытке доступа к ячейкам памяти, лежащим вне границ реального файла, возникнет сигнал **SIGBUS** (реакция на него по умолчанию - прекращение процесса с образованием core файла).

Параметр **flags** определяет способ отображения файла в адресное пространство. В рамках нашего курса мы будем использовать только два его возможных значения: **MAP_SHARED** и **MAP_PRIVATE**. Если в качестве его значения выбрано **MAP_SHARED**, то полученное отображение файла впоследствии будет использоваться и другими процессами, вызвавшими **mmap** для этого файла с аналогичными значениями параметров, а все изменения, сделанные в отображенном файле, будут сохранены во вторичной

памяти. Если в качестве значения параметра **flags** указано **MAP_PRIVATE**, то процесс получает отображение файла в свое монопольное распоряжение, но все изменения в нем не могут быть занесены во вторичную память (т.е., проще говоря, не сохранятся).

Параметр **prot** определяет разрешенные операции над областью памяти, в которую будет отображен файл. В качестве его значения мы будем использовать значения **PROT_READ** (разрешено чтение), **PROT_WRITE** (разрешена запись) или их комбинацию через операцию "побитовое или" - "|". Необходимо отметить две существенные особенности системного вызова, связанные с этим параметром:

1. 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**.
2. В результате ошибки в операционной системе Linux при работе на 486-х и 586-х процессорах попытка записать в отображение файла, открытое только для записи, более 32-х байт одновременно приводит к ошибке (возникает сигнал о нарушении защиты памяти).

При нормальном завершении системный вызов возвращает начальный адрес области памяти, в которую отображен файл (или его часть), при возникновении ошибки - специальное значение **MAP_FAILED** .