Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

Лабораторная работа №3 по курсу

«Операционные системы»

Группа: М8О-209БВ-24

Студент: Крюков Д.М.

Преподаватель: Миронов Е.С.

Оценка: _____

Дата: 10.12.25

Москва, 2025

Постановка задачи

Вариант 13.

Составить и отладить программу на языке Си, осуществляющую работу с процессами и взаимодействие между ними в одной из двух операционных систем. В результате работы программа (основной процесс) должен создать для решение задачи один или несколько дочерних процессов. Взаимодействие между процессами осуществляется через системные сигналы/события и/или через отображаемые файлы (memory-mapped files).

Родительский процесс создает два дочерних процесса. Child1 переводит строки в нижний регистр. Child2 превращает все пробельные символы в символ «_».

Общий метод и алгоритм решения

Использованные системные вызовы:

- open, close — открытие / закрытие файлов;
- ftruncate — установка размера файла (для mmap);
- mmap, munmap — создание/удаление отображения файла в адресное пространство;
- fork, execl — создание дочерних процессов (в коде используется fork + execl для запуска бинарников child1/child2);
- kill — отправка сигнала процессу;
- signal — установка обработчиков сигналов;
- pause — ожидание сигнала;
- wait/waitpid — ожидание завершения дочерних процессов.

Описание работы алгоритма.

Работа программы начинается с того, что родительский процесс создаёт три файла (shared1.dat, shared2.dat, shared3.dat), задаёт им размер и отображает их в память через mmap. Эти файлы используются как общие области памяти: первый — для передачи строки от родителя к child1, второй — от child1 к child2, третий — от child2 обратно к родителю. После подготовки памяти родитель создаёт два дочерних процесса с помощью fork и запускает их программы через execl. Каждый дочерний процесс получает доступ только к тем отображённым файлам, которые нужны ему для чтения и записи.

При вводе строки пользователем родитель записывает её в shared1.dat и отправляет сигнал SIGUSR1 первому дочернему процессу, тем самым инициируя конвейер обработки. Child1, получив сигнал, считывает строку из общей памяти, переводит все символы в нижний регистр и записывает результат в shared2.dat, после чего уведомляет родителя сигналом SIGUSR2. Родитель, получив этот сигнал, понимает, что первая стадия обработки завершена, и теперь пересылает сигнал SIGUSR1 второму дочернему процессу. Child2, аналогично, получает сигнал, считывает данные из shared2.dat, заменяет пробелы и табуляции на символ подчёркивания, записывает итоговую строку в shared3.dat и снова уведомляет родителя сигналом SIGUSR2. Родитель, получив сигнал во второй раз, устанавливает флаг готовности и выводит итоговую строку пользователю. Далее цикл повторяется, пока пользователь не введёт exit, после чего родитель посылает обоим дочерним процессам SIGTERM, освобождает отображённые области памяти и завершает работу.

Синхронизация процессов основана исключительно на пользовательских сигналах SIGUSR1 и SIGUSR2, что соответствует требованиям варианта. Поскольку сигналов всего два, родитель использует внутренний флаг состояния (stage), позволяющий определить, от какого дочернего

процесса пришёл сигнал и какая операция должна выполняться дальше. Переменные, которые изменяются в обработчиках сигналов (ready_flag, stage и stop_flag в дочерних процессах), объявлены как volatile sig_atomic_t, поскольку они должны корректно меняться в асинхронном режиме. Тип sig_atomic_t гарантирует атомарность операций чтения и записи, а volatile предотвращает оптимизации компилятора, которые могли бы привести к тому, что основная программа не заметила бы изменения, выполненного обработчиком сигнала.

Таким образом, программа демонстрирует работу межпроцессного взаимодействия через общую память, а также синхронизацию на основе сигналов, где каждый процесс выполняет свою часть обработки строго после получения уведомления. Эта связка mmap + сигналы обеспечивает предсказуемый и линейный обмен данными между несколькими процессами.

Код программы

parent.c

```
#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <signal.h>

#include <sys/mman.h>

#include <string.h>

#include <fcntl.h>

#include <sys/wait.h>


static char* to_child1 = NULL;

static char* from_child2 = NULL;

static const size_t SH = 4096;

static volatile sig_atomic_t ready_flag = 0;

static volatile sig_atomic_t stage = 0;

static pid_t c1 = 0, c2 = 0;


void sig_handler(int s) {

    if (s == SIGUSR2) {

        if (stage == 1) {

            stage = 2;
```

```
            kill(c2, SIGUSR1);
        } else if (stage == 2) {
            ready_flag = 1;
            stage = 0;
        }
    }
}

int main() {
    int f1 = open("shared1.dat", O_RDWR | O_CREAT | O_TRUNC, 0666);
    int f2 = open("shared2.dat", O_RDWR | O_CREAT | O_TRUNC, 0666);
    int f3 = open("shared3.dat", O_RDWR | O_CREAT | O_TRUNC, 0666);

    if (f1 < 0 || f2 < 0 || f3 < 0) { perror("open"); return 1; }

    if (ftruncate(f1, SH) < 0 || ftruncate(f2, SH) < 0 || ftruncate(f3, SH) < 0) {
        perror("ftruncate"); return 1;
    }
    to_child1 = mmap(NULL, SH, PROT_WRITE | PROT_READ, MAP_SHARED, f1, 0);
    from_child2 = mmap(NULL, SH, PROT_WRITE | PROT_READ, MAP_SHARED, f3, 0);
    close(f1); close(f2); close(f3);

    c1 = fork();
    if (c1 == 0) execl("./child1", "./child1", NULL);
    if (c1 < 0) { perror("ошибка форка c1"); return 1; }

    c2 = fork();
    if (c2 == 0) execl("./child2", "./child2", NULL);
    if (c2 < 0) { perror("ошибка форка c2"); kill(c1, SIGTERM); return 1; }
```

```
    signal(SIGUSR2, sig_handler);
    while (1) {
        char input[SH];
        fflush(stdout);

        if (!fgets(input, SH, stdin)){
            break;
        }
        if (strcmp(input, "exit\n") == 0){
            break;
        }
        memset(to_child1, 0, SH);
        memcpy(to_child1, input, strlen(input) + 1);
        stage = 1;
        ready_flag = 0;
        kill(c1, SIGUSR1);
        while (!ready_flag) pause();

        printf("%s\n", from_child2);
    }
    kill(c1, SIGTERM);
    kill(c2, SIGTERM);

    wait(NULL);
    wait(NULL);

    if (to_child1 && to_child1 != MAP_FAILED) munmap(to_child1, SH);
    if (from_child2 && from_child2 != MAP_FAILED) munmap(from_child2, SH);
    return 0;
}
```

child1.c

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <signal.h>
#include <fcntl.h>
#include <string.h>
#include <sys/mman.h>

static char* indata = NULL;
static char* outdata = NULL;
static const size_t SH = 4096;
static volatile sig_atomic_t stop_flag = 0;

void sig_handler(int s) {
    if (s == SIGTERM) {
        stop_flag = 1;
        return;
    }
    if (s == SIGUSR1) {
        char buf[SH];
        memcpy(buf, indata, SH);
        size_t len = strnlen(buf, SH);
        for (size_t i = 0; i < len; i++) {
            if (buf[i] >= 'A' && buf[i] <= 'Z')
                buf[i] = buf[i] + ('a' - 'A');
        }
        memset(outdata, 0, SH);
        memcpy(outdata, buf, len + 1);
        kill(getppid(), SIGUSR2);
```

```
    }
}

int main() {
    int f1 = open("shared1.dat", O_RDONLY);
    if (f1 < 0) { perror("ошибка shared1.dat"); return 1; }
    int f2 = open("shared2.dat", O_RDWR);
    if (f2 < 0) { perror("ошибка shared2.dat"); close(f1); return 1; }

    indata = mmap(NULL, SH, PROT_READ, MAP_SHARED, f1, 0);
    outdata = mmap(NULL, SH, PROT_WRITE | PROT_READ, MAP_SHARED, f2, 0);

    close(f1);
    close(f2);

    signal(SIGUSR1, sig_handler);
    signal(SIGTERM, sig_handler);

    while (!stop_flag) pause();

    if (indata && indata != MAP_FAILED) munmap(indata, SH);
    if (outdata && outdata != MAP_FAILED) munmap(outdata, SH);
    return 0;
}
```

child2.c

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <signal.h>
#include <fcntl.h>
```

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

static char* indata = NULL;
static char* outdata = NULL;
static const size_t SH = 4096;
static volatile sig_atomic_t stop_flag = 0;

void sig_handler(int s) {
    if (s == SIGTERM) {
        stop_flag = 1;
        return;
    }
    if (s == SIGUSR1) {
        char buf[SH];
        memcpy(buf, indata, SH);
        size_t len = strnlen(buf, SH);
        for (size_t i = 0; i < len; i++) {
            if (buf[i] == ' ' || buf[i] == '\t')
                buf[i] = '_';
        }
        memset(outdata, 0, SH);
        memcpy(outdata, buf, len + 1);
        kill(getppid(), SIGUSR2);
    }
}

int main() {
    int f1 = open("shared2.dat", O_RDONLY);
    if (f1 < 0) { perror("ошибка shared2.dat"); return 1; }
```

```
    int f2 = open("shared3.dat", O_RDWR);
    if (f2 < 0) { perror("ошибка shared3.dat"); close(f1); return 1; }

    indata = mmap(NULL, SH, PROT_READ, MAP_SHARED, f1, 0);
    outdata = mmap(NULL, SH, PROT_WRITE | PROT_READ, MAP_SHARED, f2, 0);

    close(f1);
    close(f2);

    signal(SIGUSR1, sig_handler);
    signal(SIGTERM, sig_handler);

    while (!stop_flag) pause();

    if (indata && indata != MAP_FAILED) munmap(indata, SH);
    if (outdata && outdata != MAP_FAILED) munmap(outdata, SH);
    return 0;
}
```

Протокол работы программы

./parent

HeLLo WoRLD

hello_world

BYE BYE

bye_bye

exit

Strace:

```
execve("./parent", ["./parent"], 0x7fff9e3dddb8 /* 29 vars */) = 0

brk(NULL)                               = 0x37e01000

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7cf35fd2a000

access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3

fstat(3, {st_mode=S_IFREG|0644, st_size=29003, ...}) = 0

mmap(NULL, 29003, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7cf35fd22000

close(3)                                = 0

openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0p\236\2\0\0\0\0\0"..., 832) = 832

pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 840, 64) = 840

fstat(3, {st_mode=S_IFREG|0755, st_size=2003408, ...}) = 0

pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 840, 64) = 840

mmap(NULL, 2055800, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7cf35fb2c000

mmap(0x7cf35fb54000, 1462272, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x28000) = 0x7cf35fb54000

mmap(0x7cf35fcb9000, 352256, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x18d000) =
0x7cf35fcb9000

mmap(0x7cf35fd0f000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x1e2000) = 0x7cf35fd0f000

mmap(0x7cf35fd15000, 52856, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0)
= 0x7cf35fd15000

close(3)                                = 0

mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7cf35fb29000

arch_prctl(ARCH_SET_FS, 0x7cf35fb29740) = 0
```

```
set_tid_address(0x7cf35fb29a10)         = 10185

set_robust_list(0x7cf35fb29a20, 24)     = 0

rseq(0x7cf35fb29680, 0x20, 0, 0x53053053) = 0

mprotect(0x7cf35fd0f000, 16384, PROT_READ) = 0

mprotect(0x403000, 4096, PROT_READ)     = 0

mprotct(0x7cf35fd60000, 8192, PROT_READ) = 0

prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0

munmap(0x7cf35fd22000, 29003)           = 0

openat(AT_FDCWD, "shared1.dat", O_RDWR|O_CREAT|O_TRUNC, 0666) = 3

openat(AT_FDCWD, "shared2.dat", O_RDWR|O_CREAT|O_TRUNC, 0666) = 4

openat(AT_FDCWD, "shared3.dat", O_RDWR|O_CREAT|O_TRUNC, 0666) = 5

ftruncate(3, 4096)                      = 0

ftruncate(4, 4096)                      = 0

ftruncate(5, 4096)                      = 0

mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_SHARED, 3, 0) = 0x7cf35fd29000

mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_SHARED, 5, 0) = 0x7cf35fd28000

close(3)                                = 0

close(4)                                = 0

close(5)                                = 0

rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0

clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLDstrace: Process
10186 attached

, child_tidptr=0x7cf35fb29a10) = 10186

[pid 10186] set_robust_list(0x7cf35fb29a20, 24 <unfinished ...>

[pid 10185] rt_sigprocmask(SIG_SETMASK, [],  <unfinished ...>

[pid 10186] <... set_robust_list resumed>) = 0

[pid 10185] <... rt_sigprocmask resumed>NULL, 8) = 0

[pid 10185] rt_sigprocmask(SIG_BLOCK, ~[],  <unfinished ...>

[pid 10186] rt_sigprocmask(SIG_SETMASK, [],  <unfinished ...>

[pid 10185] <... rt_sigprocmask resumed>[], 8) = 0

[pid  10185]  clone(child_stack=NULL,  flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD
<unfinished ...>

[pid 10186] <... rt_sigprocmask resumed>NULL, 8) = 0
```

[pid 10186] execve("./child1", ["./child1"], 0x7fff97a9b618 /* 29 vars */strace: Process 10187 attached

 <unfinished ...>

[pid 10185] <... clone resumed>, child_tidptr=0x7cf35fb29a10) = 10187

[pid 10185] rt_sigprocmask(SIG_SETMASK, [], <unfinished ...>

[pid 10187] set_robust_list(0x7cf35fb29a20, 24 <unfinished ...>

[pid 10185] <... rt_sigprocmask resumed>NULL, 8) = 0

[pid 10187] <... set_robust_list resumed>) = 0

[pid 10185] rt_sigaction(SIGUSR2, {sa_handler=0x401246, sa_mask=[USR2], sa_flags=SA_RESTORER|SA_RESTART, sa_restorer=0x7cf35fb6bdf0}, <unfinished ...>

[pid 10187] rt_sigprocmask(SIG_SETMASK, [], <unfinished ...>

[pid 10185] <... rt_sigaction resumed>{sa_handler=SIG_DFL, sa_mask=[], sa_flags=0}, 8) = 0

[pid 10187] <... rt_sigprocmask resumed>NULL, 8) = 0

[pid 10185] fstat(0, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}) = 0

[pid 10187] execve("./child2", ["./child2"], 0x7fff97a9b618 /* 29 vars */ <unfinished ...>

[pid 10185] getrandom("\x2a\x4f\x18\x93\x11\x3d\x8e\x6a", 8, GRND_NONBLOCK) = 8

[pid 10185] brk(NULL)   = 0x37e01000

[pid 10185] brk(0x37e22000)   = 0x37e22000

[pid 10185] read(0, <unfinished ...>

[pid 10187] <... execve resumed>)   = 0

[pid 10186] <... execve resumed>)   = 0

[pid 10187] brk(NULL <unfinished ...>

[pid 10186] brk(NULL <unfinished ...>

[pid 10187] <... brk resumed>)   = 0xd622000

[pid 10186] <... brk resumed>)   = 0x3f20000

[pid 10187] mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x769e1a643000

[pid 10186] mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 10187] access("/etc/ld.so.preload", R_OK <unfinished ...>

[pid 10186] <... mmap resumed>)   = 0x74d13a92a000

[pid 10187] <... access resumed>)   = -1 ENOENT (No such file or directory)

[pid 10186] access("/etc/ld.so.preload", R_OK <unfinished ...>

[pid 10187] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC <unfinished ...>

[pid 10186] <... access resumed>)   = -1 ENOENT (No such file or directory)

```
[pid 10187] <... openat resumed>)       = 3

[pid 10187] fstat(3,  <unfinished ...>

[pid 10186] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC <unfinished ...>

[pid 10187] <... fstat resumed>{st_mode=S_IFREG|0644, st_size=29003, ...}) = 0

[pid 10186] <... openat resumed>)       = 3

[pid 10187] mmap(NULL, 29003, PROT_READ, MAP_PRIVATE, 3, 0 <unfinished ...>

[pid 10186] fstat(3,  <unfinished ...>

[pid 10187] <... mmap resumed>)         = 0x769e1a63b000

[pid 10186] <... fstat resumed>{st_mode=S_IFREG|0644, st_size=29003, ...}) = 0

[pid 10187] close(3 <unfinished ...>

[pid 10186] mmap(NULL, 29003, PROT_READ, MAP_PRIVATE, 3, 0 <unfinished ...>

[pid 10187] <... close resumed>)        = 0

[pid 10186] <... mmap resumed>)         = 0x74d13a922000

[pid   10187]   openat(AT_FDCWD,   "/lib/x86_64-linux-gnu/libc.so.6",   O_RDONLY|O_CLOEXEC
<unfinished ...>

[pid 10186] close(3 <unfinished ...>

[pid 10187] <... openat resumed>)       = 3

[pid 10186] <... close resumed>)        = 0

[pid 10187] read(3,  <unfinished ...>

[pid   10186]   openat(AT_FDCWD,   "/lib/x86_64-linux-gnu/libc.so.6",   O_RDONLY|O_CLOEXEC
<unfinished ...>

[pid                        10187]                        <...                        read
resumed>"\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0p\236\2\0\0\0\0\0"..., 832) = 832

[pid 10186] <... openat resumed>)       = 3

[pid 10187] pread64(3,  <unfinished ...>

[pid 10186] read(3,  <unfinished ...>

[pid                       10187]                       <...                       pread64
resumed>"\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 840, 64) = 840

[pid                        10186]                        <...                        read
resumed>"\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0p\236\2\0\0\0\0\0"..., 832) = 832

[pid 10187] fstat(3,  <unfinished ...>

[pid 10186] pread64(3,  <unfinished ...>

[pid 10187] <... fstat resumed>{st_mode=S_IFREG|0755, st_size=2003408, ...}) = 0

[pid                       10186]                       <...                       pread64
resumed>"\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 840, 64) = 840
```

```
[pid 10187] pread64(3,  <unfinished ...>

[pid 10186] fstat(3,  <unfinished ...>

[pid                    10187]                    <...                    pread64
resumed>"\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 840, 64) = 840

[pid 10186] <... fstat resumed>{st_mode=S_IFREG|0755, st_size=2003408, ...}) = 0

[pid 10187] mmap(NULL, 2055800, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0 <unfinished ...>

[pid 10186] pread64(3,  <unfinished ...>

[pid 10187] <... mmap resumed>)         = 0x769e1a445000

[pid                    10186]                    <...                    pread64
resumed>"\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 840, 64) = 840

[pid        10187]        mmap(0x769e1a46d000,        1462272,        PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x769e1a46d000

[pid 10186] mmap(NULL, 2055800, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0 <unfinished ...>

[pid 10187] mmap(0x769e1a5d2000, 352256, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x18d000 <unfinished ...>

[pid 10186] <... mmap resumed>)         = 0x74d13a72c000

[pid 10187] <... mmap resumed>)         = 0x769e1a5d2000

[pid        10186]        mmap(0x74d13a754000,        1462272,        PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000 <unfinished ...>

[pid        10187]        mmap(0x769e1a628000,        24576,        PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e2000 <unfinished ...>

[pid 10186] <... mmap resumed>)         = 0x74d13a754000

[pid 10187] <... mmap resumed>)         = 0x769e1a628000

[pid 10186] mmap(0x74d13a8b9000, 352256, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x18d000 <unfinished ...>

[pid        10187]        mmap(0x769e1a62e000,        52856,        PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 10186] <... mmap resumed>)         = 0x74d13a8b9000

[pid 10187] <... mmap resumed>)         = 0x769e1a62e000

[pid        10186]        mmap(0x74d13a90f000,        24576,        PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e2000 <unfinished ...>

[pid 10187] close(3 <unfinished ...>

[pid 10186] <... mmap resumed>)         = 0x74d13a90f000

[pid 10187] <... close resumed>)        = 0

[pid        10186]        mmap(0x74d13a915000,        52856,        PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0 <unfinished ...>
```

[pid 10187] mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 10186] <... mmap resumed>) = 0x74d13a915000

[pid 10187] <... mmap resumed>) = 0x769e1a442000

[pid 10186] close(3 <unfinished ...>

[pid 10187] arch_prctl(ARCH_SET_FS, 0x769e1a442740 <unfinished ...>

[pid 10186] <... close resumed>) = 0

[pid 10187] <... arch_prctl resumed>) = 0

[pid 10186] mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 10187] set_tid_address(0x769e1a442a10 <unfinished ...>

[pid 10186] <... mmap resumed>) = 0x74d13a729000

[pid 10187] <... set_tid_address resumed>) = 10187

[pid 10186] arch_prctl(ARCH_SET_FS, 0x74d13a729740 <unfinished ...>

[pid 10187] set_robust_list(0x769e1a442a20, 24 <unfinished ...>

[pid 10186] <... arch_prctl resumed>) = 0

[pid 10187] <... set_robust_list resumed>) = 0

[pid 10186] set_tid_address(0x74d13a729a10 <unfinished ...>

[pid 10187] rseq(0x769e1a442680, 0x20, 0, 0x53053053 <unfinished ...>

[pid 10186] <... set_tid_address resumed>) = 10186

[pid 10187] <... rseq resumed>) = 0

[pid 10186] set_robust_list(0x74d13a729a20, 24) = 0

[pid 10187] mprotect(0x769e1a628000, 16384, PROT_READ <unfinished ...>

[pid 10186] rseq(0x74d13a729680, 0x20, 0, 0x53053053 <unfinished ...>

[pid 10187] <... mprotect resumed>) = 0

[pid 10186] <... rseq resumed>) = 0

[pid 10187] mprotect(0x403000, 4096, PROT_READ) = 0

[pid 10186] mprotect(0x74d13a90f000, 16384, PROT_READ <unfinished ...>

[pid 10187] mprotect(0x769e1a679000, 8192, PROT_READ <unfinished ...>

[pid 10186] <... mprotect resumed>) = 0

[pid 10187] <... mprotect resumed>) = 0

[pid 10186] mprotect(0x403000, 4096, PROT_READ <unfinished ...>

[pid 10187] prlimit64(0, RLIMIT_STACK, NULL, <unfinished ...>

[pid 10186] <... mprotect resumed>) = 0

```
[pid 10187] <... prlimit64 resumed>{rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
[pid 10186] mprotect(0x74d13a960000, 8192, PROT_READ <unfinished ...>
[pid 10187] munmap(0x769e1a63b000, 29003 <unfinished ...>
[pid 10186] <... mprotect resumed>)     = 0
[pid 10187] <... munmap resumed>)       = 0
[pid 10186] prlimit64(0, RLIMIT_STACK, NULL,  <unfinished ...>
[pid 10187] openat(AT_FDCWD, "shared2.dat", O_RDONLY <unfinished ...>
[pid 10186] <... prlimit64 resumed>{rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
[pid 10186] munmap(0x74d13a922000, 29003) = 0
[pid 10186] openat(AT_FDCWD, "shared1.dat", O_RDONLY <unfinished ...>
[pid 10187] <... openat resumed>)       = 3
[pid 10186] <... openat resumed>)       = 3
[pid 10187] openat(AT_FDCWD, "shared3.dat", O_RDWR <unfinished ...>
[pid 10186] openat(AT_FDCWD, "shared2.dat", O_RDWR <unfinished ...>
[pid 10187] <... openat resumed>)       = 4
[pid 10186] <... openat resumed>)       = 4
[pid 10187] mmap(NULL, 4096, PROT_READ, MAP_SHARED, 3, 0 <unfinished ...>
[pid 10186] mmap(NULL, 4096, PROT_READ, MAP_SHARED, 3, 0 <unfinished ...>
[pid 10187] <... mmap resumed>)         = 0x769e1a642000
[pid 10186] <... mmap resumed>)         = 0x74d13a929000
[pid 10187] mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_SHARED, 4, 0 <unfinished ...>
[pid 10186] mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_SHARED, 4, 0 <unfinished ...>
[pid 10187] <... mmap resumed>)         = 0x769e1a641000
[pid 10186] <... mmap resumed>)         = 0x74d13a928000
[pid 10187] close(3 <unfinished ...>
[pid 10186] close(3 <unfinished ...>
[pid 10187] <... close resumed>)        = 0
[pid 10186] <... close resumed>)        = 0
[pid 10187] close(4 <unfinished ...>
[pid 10186] close(4 <unfinished ...>
[pid 10187] <... close resumed>)        = 0
[pid 10186] <... close resumed>)        = 0
```

```
[pid      10187]      rt_sigaction(SIGUSR1,      {sa_handler=0x4011d6,      sa_mask=[USR1],
sa_flags=SA_RESTORER|SA_RESTART, sa_restorer=0x769e1a484df0},  <unfinished ...>

[pid      10186]      rt_sigaction(SIGUSR1,      {sa_handler=0x4011d6,      sa_mask=[USR1],
sa_flags=SA_RESTORER|SA_RESTART, sa_restorer=0x74d13a76bdf0},  <unfinished ...>

[pid 10187] <... rt_sigaction resumed>{sa_handler=SIG_DFL, sa_mask=[], sa_flags=0}, 8) = 0

[pid 10186] <... rt_sigaction resumed>{sa_handler=SIG_DFL, sa_mask=[], sa_flags=0}, 8) = 0

[pid      10187]      rt_sigaction(SIGTERM,      {sa_handler=0x4011d6,      sa_mask=[TERM],
sa_flags=SA_RESTORER|SA_RESTART, sa_restorer=0x769e1a484df0},  <unfinished ...>

[pid      10186]      rt_sigaction(SIGTERM,      {sa_handler=0x4011d6,      sa_mask=[TERM],
sa_flags=SA_RESTORER|SA_RESTART, sa_restorer=0x74d13a76bdf0},  <unfinished ...>

[pid 10187] <... rt_sigaction resumed>{sa_handler=SIG_DFL, sa_mask=[], sa_flags=0}, 8) = 0

[pid 10186] <... rt_sigaction resumed>{sa_handler=SIG_DFL, sa_mask=[], sa_flags=0}, 8) = 0

[pid 10187] pause( <unfinished ...>

[pid 10186] pause(HeLLo WoRLD

 <unfinished ...>

[pid 10185] <... read resumed>"HeLLo WoRLD\n", 1024) = 12

[pid 10185] kill(10186, SIGUSR1)        = 0

[pid 10186] <... pause resumed>)        = ? ERESTARTNOHAND (To be restarted if no handler)

[pid 10185] pause( <unfinished ...>

[pid 10186] --- SIGUSR1 {si_signo=SIGUSR1, si_code=SI_USER, si_pid=10185, si_uid=0} ---

[pid 10186] getppid()                   = 10185

[pid 10186] kill(10185, SIGUSR2 <unfinished ...>

[pid 10185] <... pause resumed>)        = ? ERESTARTNOHAND (To be restarted if no handler)

[pid 10186] <... kill resumed>)         = 0

[pid 10185] --- SIGUSR2 {si_signo=SIGUSR2, si_code=SI_USER, si_pid=10186, si_uid=0} ---

[pid 10186] rt_sigreturn({mask=[]} <unfinished ...>

[pid 10185] kill(10187, SIGUSR1 <unfinished ...>

[pid 10186] <... rt_sigreturn resumed>) = -1 EINTR (Interrupted system call)

[pid 10185] <... kill resumed>)         = 0

[pid 10187] <... pause resumed>)        = ? ERESTARTNOHAND (To be restarted if no handler)

[pid 10185] rt_sigreturn({mask=[]} <unfinished ...>

[pid 10186] pause( <unfinished ...>

[pid 10185] <... rt_sigreturn resumed>) = -1 EINTR (Interrupted system call)

[pid 10187] --- SIGUSR1 {si_signo=SIGUSR1, si_code=SI_USER, si_pid=10185, si_uid=0} ---

[pid 10185] pause( <unfinished ...>
```

```
[pid 10187] getppid()                   = 10185
[pid 10187] kill(10185, SIGUSR2 <unfinished ...>
[pid 10185] <... pause resumed>)        = ? ERESTARTNOHAND (To be restarted if no handler)
[pid 10187] <... kill resumed>)         = 0
[pid 10185] --- SIGUSR2 {si_signo=SIGUSR2, si_code=SI_USER, si_pid=10187, si_uid=0} ---
[pid 10187] rt_sigreturn({mask=[]} <unfinished ...>
[pid 10185] rt_sigreturn({mask=[]} <unfinished ...>
[pid 10187] <... rt_sigreturn resumed>) = -1 EINTR (Interrupted system call)
[pid 10185] <... rt_sigreturn resumed>) = -1 EINTR (Interrupted system call)
[pid 10187] pause( <unfinished ...>
[pid 10185] fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}) = 0
[pid 10185] write(1, "hello_world\n", 12hello_world
) = 12
[pid 10185] write(1, "\n", 1
)           = 1
[pid 10185] read(0, BYE BYE
"BYE BYE\n", 1024)  = 8
[pid 10185] kill(10186, SIGUSR1)        = 0
[pid 10186] <... pause resumed>)        = ? ERESTARTNOHAND (To be restarted if no handler)
[pid 10185] pause( <unfinished ...>
[pid 10186] --- SIGUSR1 {si_signo=SIGUSR1, si_code=SI_USER, si_pid=10185, si_uid=0} ---
[pid 10186] getppid()                   = 10185
[pid 10186] kill(10185, SIGUSR2)        = 0
[pid 10185] <... pause resumed>)        = ? ERESTARTNOHAND (To be restarted if no handler)
[pid 10186] rt_sigreturn({mask=[]} <unfinished ...>
[pid 10185] --- SIGUSR2 {si_signo=SIGUSR2, si_code=SI_USER, si_pid=10186, si_uid=0} ---
[pid 10186] <... rt_sigreturn resumed>) = -1 EINTR (Interrupted system call)
[pid 10185] kill(10187, SIGUSR1 <unfinished ...>
[pid 10186] pause( <unfinished ...>
[pid 10185] <... kill resumed>)         = 0
[pid 10187] <... pause resumed>)        = ? ERESTARTNOHAND (To be restarted if no handler)
[pid 10185] rt_sigreturn({mask=[]} <unfinished ...>
[pid 10187] --- SIGUSR1 {si_signo=SIGUSR1, si_code=SI_USER, si_pid=10185, si_uid=0} ---
```

```
[pid 10185] <... rt_sigreturn resumed>) = -1 EINTR (Interrupted system call)
[pid 10187] getppid( <unfinished ...>
[pid 10185] pause( <unfinished ...>
[pid 10187] <... getppid resumed>)      = 10185
[pid 10187] kill(10185, SIGUSR2 <unfinished ...>
[pid 10185] <... pause resumed>)        = ? ERESTARTNOHAND (To be restarted if no handler)
[pid 10187] <... kill resumed>)         = 0
[pid 10185] --- SIGUSR2 {si_signo=SIGUSR2, si_code=SI_USER, si_pid=10187, si_uid=0} ---
[pid 10187] rt_sigreturn({mask=[]} <unfinished ...>
[pid 10185] rt_sigreturn({mask=[]} <unfinished ...>
[pid 10187] <... rt_sigreturn resumed>) = -1 EINTR (Interrupted system call)
[pid 10185] <... rt_sigreturn resumed>) = -1 EINTR (Interrupted system call)
[pid 10187] pause( <unfinished ...>
[pid 10185] write(1, "bye_bye\n", 8bye_bye
)    = 8
[pid 10185] write(1, "\n", 1
)           = 1
[pid 10185] read(0, exit
"exit\n", 1024)     = 5
[pid 10185] kill(10186, SIGTERM)        = 0
[pid 10186] <... pause resumed>)        = ? ERESTARTNOHAND (To be restarted if no handler)
[pid 10185] kill(10187, SIGTERM <unfinished ...>
[pid 10186] --- SIGTERM {si_signo=SIGTERM, si_code=SI_USER, si_pid=10185, si_uid=0} ---
[pid 10185] <... kill resumed>)         = 0
[pid 10187] <... pause resumed>)        = ? ERESTARTNOHAND (To be restarted if no handler)
[pid 10185] wait4(-1,  <unfinished ...>
[pid 10186] rt_sigreturn({mask=[]} <unfinished ...>
[pid 10187] --- SIGTERM {si_signo=SIGTERM, si_code=SI_USER, si_pid=10185, si_uid=0} ---
[pid 10186] <... rt_sigreturn resumed>) = -1 EINTR (Interrupted system call)
[pid 10187] rt_sigreturn({mask=[]} <unfinished ...>
[pid 10186] munmap(0x74d13a929000, 4096 <unfinished ...>
[pid 10187] <... rt_sigreturn resumed>) = -1 EINTR (Interrupted system call)
[pid 10187] munmap(0x769e1a642000, 4096 <unfinished ...>
```

```
[pid 10186] <... munmap resumed>)       = 0
[pid 10187] <... munmap resumed>)       = 0
[pid 10186] munmap(0x74d13a928000, 4096 <unfinished ...>
[pid 10187] munmap(0x769e1a641000, 4096) = 0
[pid 10187] exit_group(0)               = ?
[pid 10186] <... munmap resumed>)       = 0
[pid 10186] exit_group(0)               = ?
[pid 10187] +++ exited with 0 +++
[pid 10185] <... wait4 resumed>NULL, 0, NULL) = 10187
[pid 10186] +++ exited with 0 +++
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=10187, si_uid=0, si_status=0, si_utime=0, si_stime=0} ---
wait4(-1, NULL, 0, NULL)                = 10186
munmap(0x7cf35fd29000, 4096)            = 0
munmap(0x7cf35fd28000, 4096)            = 0
exit_group(0)                           = ?
+++ exited with 0 +++
```

Вывод

В ходе лабораторной работы освоена технология Memory-Mapped Files для организации межпроцессного взаимодействия. Разработана многопроцессная программа с конвейерной обработкой данных, где процессы обмениваются информацией через отображаемые файлы с использованием mmap() и синхронизируются сигналами. Реализация подтверждает эффективность технологии File Mapping для передачи данных между процессами без создания лишних копий.