

# Отчёт по лабораторной работе №14

---

Грузинова Елизавета Константиновна

## Именованные каналы

---

Приобретение практических навыков работы с именованными каналами.

Изучите приведённые в тексте программы `server.c` и `client.c`. Взяв данные примеры за образец, напишите аналогичные программы, внося следующие изменения:

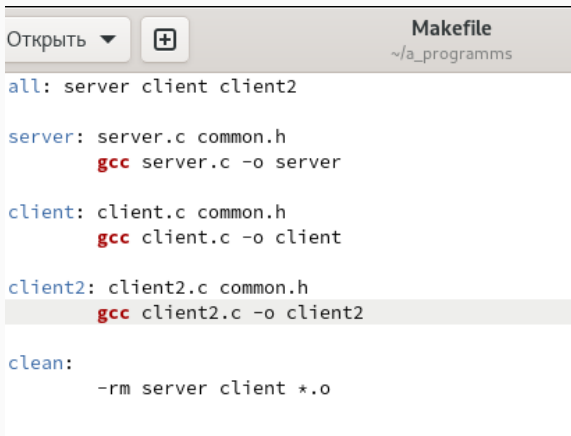
1. Работает не 1 клиент, а несколько (например, два).
2. Клиенты передают текущее время с некоторой периодичностью (например, раз в пять секунд). Используйте функцию `sleep()` для приостановки работы клиента.

3. Сервер работает не бесконечно, а прекращает работу через некоторое время (например, 30 сек). Используйте функцию `clock()` для определения времени работы сервера. Что будет в случае, если сервер завершит работу, не закрыв канал?

Для передачи данных между неродственными процессами можно использовать механизм именованных каналов (named pipes). Данные передаются по принципу FIFO (First In First Out) (первым записан — первым прочитан), поэтому они называются также FIFO pipes или просто FIFO. Именованные каналы отличаются от неименованных наличием идентификатора канала, который представлен как специальный файл (соответственно имя именованного канала — это имя файла). Поскольку файл находится на локальной файловой системе, данное IPC используется внутри одной системы.

Изучите приведённые в тексте программы `server.c` и `client.c`. Взяв данные примеры за образец, напишите аналогичные программы, внося следующие изменения:

1. Работает не 1 клиент, а несколько (например, два). Также привести необходимые для работы тексты будущих командных файлов. (рис. 1, 2, 3, 4)



The image shows a screenshot of a text editor window titled "Makefile" with the path "~/a\_programms" displayed below the title. The editor has a menu bar with "Открыть" (Open) and a "+" icon. The content of the Makefile is as follows:

```
all: server client client2

server: server.c common.h
    gcc server.c -o server

client: client.c common.h
    gcc client.c -o client

client2: client2.c common.h
    gcc client2.c -o client2

clean:
    -rm server client *.o
```

Figure 1: Текст Makefile



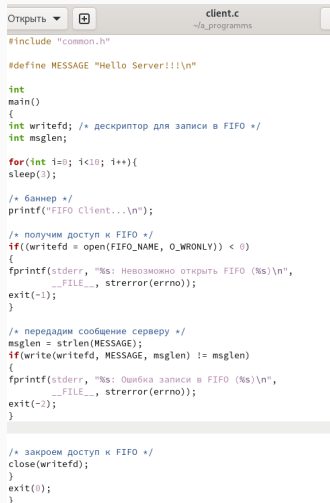
```
#ifndef __COMMON_H__
#define __COMMON_H__

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>
#include <time.h>
#define FIFO_NAME "/tmp/fifo"
#define MAX_BUFF 80

#endif /* __COMMON_H__ */
```

Figure 2: Текст common.h

# Выполнение лабораторной работы



```
client.c
~/a_programms

#include "common.h"

#define MESSAGE "Hello Server!!!\n"

int
main()
{
    int writefd; /* дескриптор для записи в FIFO */
    int msglen;

    for(int i=0; i<10; i++){
        sleep(3);

        /* баннер */
        printf("FIFO Client...\n");

        /* получим доступ к FIFO */
        if((writefd = open(FIFO_NAME, O_WRONLY)) < 0)
        {
            fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
                __FILE__, strerror(errno));
            exit(-1);
        }

        /* передадим сообщение серверу */
        msglen = strlen(MESSAGE);
        if(write(writefd, MESSAGE, msglen) != msglen)
        {
            fprintf(stderr, "%s: Ошибка записи в FIFO (%s)\n",
                __FILE__, strerror(errno));
            exit(-2);
        }

        /* закроем доступ к FIFO */
        close(writefd);
    }
    exit(0);
}
```

Figure 3: Текст будущей программы client

```
[ekgruzinova@fedora a_programms]$ ./client  
FIFO Client...  
FIFO Client...
```

Figure 4: Работа программы client при запущенном сервере

2. Клиенты передают текущее время с некоторой периодичностью (например, раз в пять секунд). Используйте функцию `sleep()` для приостановки работы клиента. (рис. 5, 6)

# Выполнение лабораторной работы

```
#include "common.h"
#include <time.h>
#define MESSAGE "Hello Server!!!\n"

int
main()
{
    int writefd; /* дескриптор для записи в FIFO */
    int msglen;
    long int ttime;

    for(int i=0; i<15; i++){
        ttime=time(NULL);
        printf(ctime(&ttime));

        /* баннер */
        printf("FIFO Client...\n");

        /* получим доступ к FIFO */
        if((writefd = open(FIFO_NAME, O_WRONLY)) < 0)
        {
            fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
                __FILE__, strerror(errno));
            exit(-1);
        }

        /* передадим сообщение серверу */
        msglen = strlen(MESSAGE);
        if(write(writefd, MESSAGE, msglen) != msglen)
        {
            fprintf(stderr, "%s: Ошибка записи в FIFO (%s)\n",
                __FILE__, strerror(errno));
            exit(-2);
        }
        sleep(1);
    }
    /* закроем доступ к FIFO */
    close(writefd);

    exit(0);
}
```

Figure 5: Текст будущей программы client2

```
[ekgruzinova@fedora a_programms]$ ./client2  
Sat Jun  4 21:14:10 2022  
FIFO Client...  
Sat Jun  4 21:14:15 2022  
FIFO Client...  
Sat Jun  4 21:14:20 2022  
FIFO Client...  
Sat Jun  4 21:14:25 2022  
FIFO Client...  
Sat Jun  4 21:14:30 2022
```

Figure 6: Работа программы client2 при запущенном сервере

3. Сервер работает не бесконечно, а прекращает работу через некоторое время (например, 30 сек). Используйте функцию `clock()` для определения времени работы сервера. (рис. 7, 8, 9)

# Выполнение лабораторной работы

```
#include "common.h"

int
main()
{
    int readfd; /* дескриптор для чтения из FIFO */
    int n;
    char buff[MAX_BUFF]; /* буфер для чтения данных из FIFO */

    /* баннер */
    printf("FIFO Server...\n");

    /* создаем файл FIFO с открытыми для всех
     * правами доступа на чтение и запись
     */
    if(mknod(FIFO_NAME, S_IFIFO | 0666, 0) < 0)
    {
        fprintf(stderr, "%s: Невозможно создать FIFO (%s)\n",
            __FILE__, strerror(errno));
        exit(-1);
    }

    /* откроем FIFO на чтение */
    if((readfd = open(FIFO_NAME, O_RDONLY)) < 0)
    {
        fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
            __FILE__, strerror(errno));
        exit(-2);
    }
    clock_t now_time(NULL), start_time(NULL);
    while(now_start < 0)
    /* читаем данные из FIFO и выводим на экран */
    while((n = read(readfd, buff, MAX_BUFF)) > 0)
    {
        if(write(1, buff, n) != n)
        {
            fprintf(stderr, "%s: Ошибка вывода (%s)\n",
                __FILE__, strerror(errno));
            exit(-3);
        }
    }
    now_time(NULL);
    printf("Время работы сервера завершено спустя %ld - секунд\n", (now_start));
}
```

Figure 7: Изменения в файле server.c(1)



```
close(readfd); /* закроем FIFO */

/* удалим FIFO из системы */
if(unlink(FIFO_NAME) < 0)
{
    fprintf(stderr, "%s: Невозможно удалить FIFO (%s)\n",
        __FILE__, strerror(errno));
    exit(-4);
}
exit(0);
}
```

Figure 8: Изменения в файле server.c(1)

```
[ekgruzinova@fedora a_programms]$ ./server
FIFO Server...
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Время работы сервера завершено спустя 30 - секунд
```

Figure 9: Работа программы server

Если сервер завершит работу, не закрыв канал, то файл FIFO не исчезнет, вследствие чего его нельзя будет запустить во второй раз, что приведет к недееспособности сервера.

## Выводы

---

В процессе выполнения лабораторной работы я приобрела практические навыки работы с именованными каналами.

Спасибо за внимание!