

Отчет по лабораторной работе №14

Шмаков Максим¹

2022, 4 июня , Москва

¹RUDN University, Moscow, Russian Federation

Цель работы

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?

Выполнение лабораторной работы

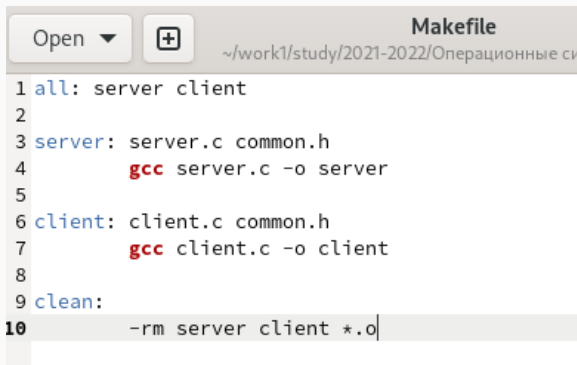
Создаю необходимые файлы (рис. [-@fig:001])

```
[mpshmakov@fedora lab14]$ touch common.h server.c client.c Makefile
```

Рис. 1: рис. 1

Далее изменяю коды этих программ.

Makefile оставил таким, каким он был. (рис. [-@fig:002])



```
Makefile
~/work1/study/2021-2022/Операционные системы

1 all: server client
2
3 server: server.c common.h
4     gcc server.c -o server
5
6 client: client.c common.h
7     gcc client.c -o client
8
9 clean:
10    -rm server client *.o
```

Рис. 2: рис. 2

В файл common.h добавил unistd.h и time.h. Они нужны для корректной работы других файлов. (рис. [-@fig:003])



```
1 /*
2  * common.h - заголовочный файл со стандартными определениями
3  */
4
5 #ifndef __COMMON_H__
6 #define __COMMON_H__
7
8 #include <stdio.h>
9 #include <stdlib.h>
10 #include <string.h>
11 #include <errno.h>
12 #include <sys/types.h>
13 #include <sys/stat.h>
14 #include <fcntl.h>
15 #include <time.h>
16 #include <unistd.h>
17
18 #define FIFO_NAME "/tmp/fifo"
19 #define MAX_BUFF 80
20
21 #endif /* __COMMON_H__ */
```

Рис. 3: рис. 3

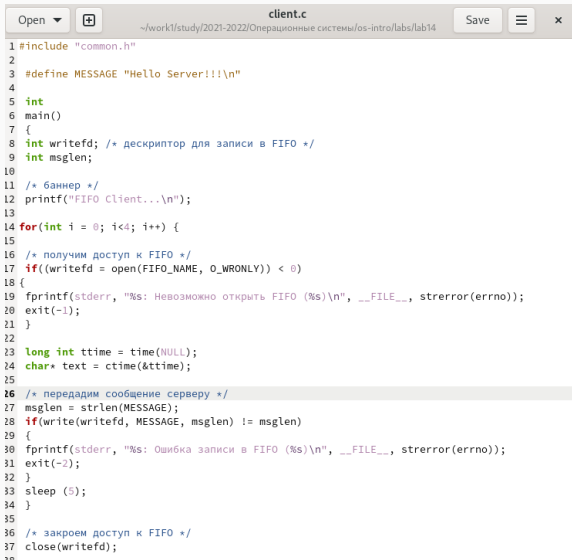
В файл `server.c` добавил цикл `while`, который следит, чтобы работа сервера завершилась через 30 секунд. (рис. [-@fig:004])



```
server.c
~/work1/study/2021-2022/Операционные системы/os-intro/labs/lab14
Save [Menu] X

11 printf("FIFO Server...\n");
12
13 /* создаем файл FIFO с открытыми для всех
14  * правами доступа на чтение и запись
15  */
16 if(mknod(FIFO_NAME, S_IFIFO | 0666, 0) < 0)
17 {
18     fprintf(stderr, "Невозможно создать FIFO (%s)\n",
19             __FILE__, strerror(errno));
20     exit(-1);
21 }
22
23 /* откроем FIFO на чтение */
24 if((readfd = open(FIFO_NAME, O_RDONLY)) < 0)
25 {
26     fprintf(stderr, "Невозможно открыть FIFO (%s)\n",
27             __FILE__, strerror(errno));
28     exit(-2);
29 }
30 /* читаем данные из FIFO и выводим на экран */
31 clock_t start = time(NULL);
32
33 while(time(NULL)-start < 30) {
34     while((n = read(readfd, buff, MAX_BUFF)) > 0)
35     {
36         if(write(1, buff, n) != n)
37         {
38             fprintf(stderr, "Ошибка вывода (%s)\n",
39                     __FILE__, strerror(errno));
40             exit(-3);
41         }
42     }
43 }
44
45 close(readfd); /* закроем FIFO */
46
47 /* удалим FIFO из системы */
48 if(unlink(FIFO_NAME) < 0)
49 {
50     fprintf(stderr, "Невозможно удалить FIFO (%s)\n",
51             __FILE__, strerror(errno));
52     exit(-4);
53 }
```

В client.c добавил цикл for, который отвечает за число сообщений (4 сообщения). Также написал команду sleep 5, которая приостанавливает работу клиента на 5 секунд. (рис. [-@fig:005])



```
client.c
~/work1/study/2021-2022/Операционные системы/os-intro/labs/lab14

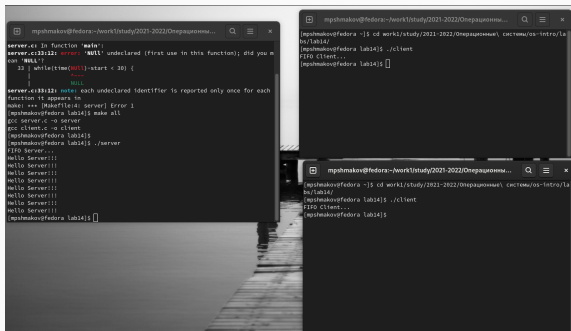
1 #include "common.h"
2
3 #define MESSAGE "Hello Server!!!\n"
4
5 int
6 main()
7 {
8     int writefd; /* дескриптор для записи в FIFO */
9     int msglen;
10
11     /* баннер */
12     printf("FIFO Client...\n");
13
14     for(int i = 0; i < 4; i++) {
15
16         /* получим доступ к FIFO */
17         if((writefd = open(FIFO_NAME, O_WRONLY)) < 0)
18         {
19             fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n", __FILE__, strerror(errno));
20             exit(-1);
21         }
22
23         long int ttime = time(NULL);
24         char* text = ctime(&ttime);
25
26         /* передадим сообщение серверу */
27         msglen = strlen(MESSAGE);
28         if(write(writefd, MESSAGE, msglen) != msglen)
29         {
30             fprintf(stderr, "%s: Ошибка записи в FIFO (%s)\n", __FILE__, strerror(errno));
31             exit(-2);
32         }
33         sleep (5);
34     }
35
36     /* закроем доступ к FIFO */
37     close(writefd);
38 }
```

```
[mpshmakov@fedora lab14]$ make all  
gcc server.c -o server  
gcc client.c -o client  
[mpshmakov@fedora lab14]$
```

Рис. 6: рис. 6

Далее я проверил работу кода.

Открыл 3 консоли. В первой прописал ./server, а на других 2ух ./client. Каждый клиент вывел по 4 сообщения и через 30 секунд сервер прекратил работу. (рис. [-@fig:007])



The image shows three terminal windows from a Fedora system. The leftmost window shows the execution of a C program named 'server.c'. It contains a 'main' function that declares a 'NULL' variable, which causes a compiler warning. The program then enters a loop that prints 'Hello Server!!' four times and then exits. The middle window shows the execution of the 'server' program, which prints 'Hello Server!!' four times. The rightmost window shows the execution of the 'client' program, which prints 'Hello Client!!' four times. The background of the terminal windows is a dark, abstract image.

```
mpshmakov@fedora:~/work1/study/2021-2022/Операционные...  
server.c: In function 'main':  
server.c:32:32: error: 'NULL' undeclared (first use in this function); did you mean 'NULL'?  
    32 | while(time(NULL)-start < 30) {  
        |                        ^~~~~  
server.c:33:32: note: each undeclared identifier is reported only once for each function it appears in  
    33 |     make() == (Makefile:4: server) Error 1  
    34 | }  
    35 |  
    36 | gcc server.c -o server  
    37 | gcc client.c -o client  
    38 | mpsmakov@fedora lab14$ ./server  
Hello Server!!  
Hello Server!!  
Hello Server!!  
Hello Server!!  
Hello Server!!  
mpsmakov@fedora lab14$  
  
mpsmakov@fedora:~/work1/study/2021-2022/Операционные...  
lab14/ lab14/  
mpsmakov@fedora lab14$ ./client  
Hello Client!!  
Hello Client!!  
Hello Client!!  
Hello Client!!  
mpsmakov@fedora lab14$  
  
mpsmakov@fedora:~/work1/study/2021-2022/Операционные...  
lab14/ lab14/  
mpsmakov@fedora lab14$ ./client  
Hello Client!!  
Hello Client!!  
Hello Client!!  
Hello Client!!  
mpsmakov@fedora lab14$
```

Рис. 7: рис. 7

Выводы

В ходе работы я научился работать с именованными каналами