

# Именованные каналы

## Лабораторная работа №14

---

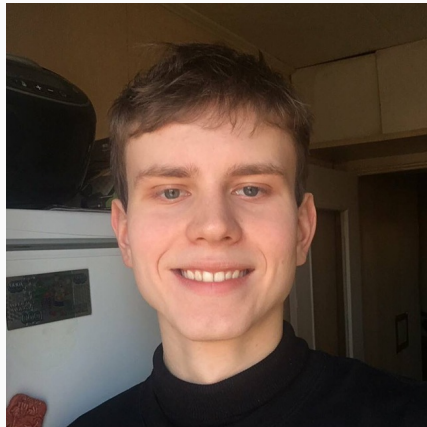
Татур С. А.

Российский университет дружбы народов, Москва, Россия

## Информация

---

- Татур Стефан Андреевич
- студент 1 курса, группа НММбд-03-22
- Российский университет дружбы народов



## Вводная часть

---

- Приобретение практических навыков работы с именованными каналами.

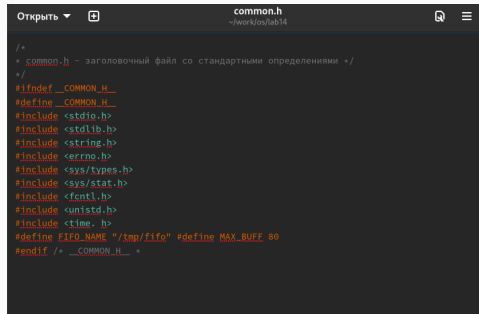
Изучить приведённые в тексте программы `server.c` и `client.c`. Взяв данные примеры за образец, написать аналогичные программы, внося следующие изменения: 1. Работает не 1 клиент, а несколько (например, два). 2. Клиенты передают текущее время с некоторой периодичностью (например, раз в пять секунд). 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 №14

---

# Внесение изменений в программы

```
lab14]$ touch common.h
lab14]$ touch server.c
lab14]$ touch client.c
lab14]$ touch Makefile
```

A screenshot of a code editor window titled 'common.h' with a subtitle '~/.work/os/lab14'. The editor shows the content of the 'common.h' header file. The code includes standard C headers and defines some constants. The text is as follows:

```
/*
 * common.h - заголовочный файл со стандартными определениями */
*/

#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__ */
```



# Внесение изменений в программы

```
Открыть ▾ + server.c ~\work\os\lab14
/*
 * server.c - реализация сервера
 * чтобы запустить пример, необходимо:
 * 1. запустить программу server на одной консоли;
 * 2. запустить программу client на другой консоли. */
#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-An");
    /* 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);
    }
    /* открываем FIFO на чтение */
    if((readfd = open(FIFO_NAME, O_RDONLY)) < 0)
    {
        fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n", __FILE__, strerror(errno));
        exit(-2);
    }
    /* начало отсчёта времени */
    clock_t start = time(NULL);
    /* цикл работает пока с момента начала отсчёта времени прошло меньше 30 секунд */
    while (time(NULL) - start < 30)
```

```
Открыть ▾ + client.c ~\work\os\lab14
/*
 * client.c - реализация клиента
 * чтобы запустить пример, необходимо:
 * 1. запустить программу server на одной консоли;
 * 2. запустить программу client на другой консоли.
 */
#include "common.h"
#define MESSAGE "Hello Server!\n"
int main()
{
    int writefd;
    /* дескриптор для записи в FIFO */
    int msglen;
    /* баннер */
    printf("FIFO client-An");
    /* цикл, отвечающий за отправку сообщения о текущем времени */
    for(int i=0; i<4; i++)
    {
        /* 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);
            break;
        }
        /* текущее время */
        long int time=time(NULL);
        char* text=time(&time);
        /* передадим сообщение серверу */
        msglen = strlen(MESSAGE);
```

# Внесение изменений в программы

```
• Makefile
~/work/os/lab14

all: server client

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

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

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

```
[macos@osx1014 ~]$ make all
gcc server.c -o server
gcc client.c -o client
```

```
Hello Server!!!  
Hello Server!!!  
Hello Server!!!  
Hello Server!!!  
Hello Server!!!  
Hello Server!!!  
Hello Server!!!  
Hello Server!!!  
FIFO Server...An[r
```

```
server.c: Невозможно создать FIFO (File exists)
```

1. 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.
2. Чтобы создать неименованный канал из командной строки нужно использовать символ |, служащий для объединения двух и более процессов: процесс\_1 | процесс\_2 | процесс\_3...
3. Чтобы создать именованный канал из командной строки нужно использовать либо команду «mknod», либо команду «mkfifo».

6. При чтении меньшего числа байтов, чем находится в канале или FIFO, возвращается требуемое число байтов, остаток сохраняется для последующих чтений. При чтении большего числа байтов, чем находится в канале или FIFO, возвращается доступное число байтов. Процесс, читающий из канала, должен соответствующим образом обработать ситуацию, когда прочитано меньше, чем заказано.
7. Запись числа байтов, меньшего емкости канала или FIFO, гарантированно атомарно. Это означает, что в случае, когда несколько процессов одновременно записывают в канал, порции данных от этих процессов не перемешиваются. При записи большего числа байтов, чем это позволяет канал или

## Результаты

---

В ходе выполнения были приобретены навыки работы с именованными каналами.