Отчёт по лабораторной работе №15

Серегин Денис Алексеевич

Содержание

List of Tables

List of Figures

1 Цель работы

Приобрести практические навыки работы с именованными каналами.

2 Задания

Изучите приведённые в тексте программы server.c и client.c. Взяв данные примеры за образец, напишите аналогичные программы, внеся 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?

3 Выполнение лабораторной работы

1. Я изучил приведённые в тексте программы server.c и client.c., а также common.h

   (рис 1. 3.1)

```
#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>

#define FIFO_NAME "/tmp/fifo"
#define MAX_BUFF 80

#endif
```

Figure 3.1: Файл common.h

   (рис 2. 3.2)

```c
#include "common.h"
int
main()
{
        int readfd;
        int n;
        char buff[MAX_BUFF];
        printf("FIFO Server...\n");

        if(mknod(FIFO_NAME, S_IFIFO | 0666, 0) < 0)
        {
                fprintf(stderr, "%s: Невозможно создать FIFO (%s)\n
                        __FILE__, strerror(errno));
                exit(-1);
        }

        if((readfd = open(FIFO_NAME, O_RDONLY)) < 0)
        {
                fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n
                        __FILE__, strerror(errno));
                exit(-2);
        }

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

Figure 3.2: Файл server.c

(рис 3. 3.3)

```c
#include "common.h"
#define MESSAGE "Hello Server!!!\n"

int
main()
{
        int writefd;
        int msglen;

        printf("FIFO Client...\n");

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

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

        close(writefd);
        exit(0);
}
```

Figure 3.3: Файл client.c

2. Взяв за образец примеры, написал аналогичные программы, внеся следующие изменения:

3. Работает не 1 клиент, а несколько (например, два).

4. Клиенты передают текущее время с некоторой периодичностью (например, раз в пять секунд). Используйте функцию sleep() для приостановки работы клиента.

5. Сервер работает не бесконечно, а прекращает работу через некоторое время (например, 30 сек).

   (рис 4. 3.4)

```c
#include "common.h"
#define MESSAGE "Hello Server!!!\n"

int
main()
{
        int writefd;
        int msglen;
        int count;
        long long int t;
        char message[10];

        for (count=0; count<=5; ++count)
        {
        sleep(5);
        t=(long long int) time(0);
        sprintf(message, "%lli", t);
        if((writefd = open(FIFO_NAME, O_WRONLY)) < 0)
        {
        fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
                __FILE__, strerror(errno));
        exit(-1);
        }

        msglen = strlen(MESSAGE);
        if(write(writefd, MESSAGE, msglen) != msglen)
        {
        fprintf(stderr, "%s: Ошибка записи в FIFO (%s)\n",
                __FILE__, strerror(errno));
        exit(-2);
        }
}
        close(writefd);
        exit(0);
}
```

Figure 3.4: Новый файл client.c

(рис 5. 3.5)

```c
#include "common.h"

int
main()
{
        int readfd;
        int n;
        char buff[MAX_BUFF];
        printf("FIFO Server...\n");

        if(mknod(FIFO_NAME, S_IFIFO | 0666, 0) < 0)
        {
                fprintf(stderr, "%s: Невозможно создать FIFO (%s)\n",
                        __FILE__, strerror(errno));
                exit(-1);
        }

        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<30)
        {
                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("server timeout, %li - seconds passed\n",(now-start));
        close(readfd);

        if(unlink(FIFO_NAME) < 0)
        {
```

Figure 3.5: Новый файл server.c (1)

(рис 6. 3.6)

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

Figure 3.6: Новый файл server.c (2)

(рис 7. 3.7)

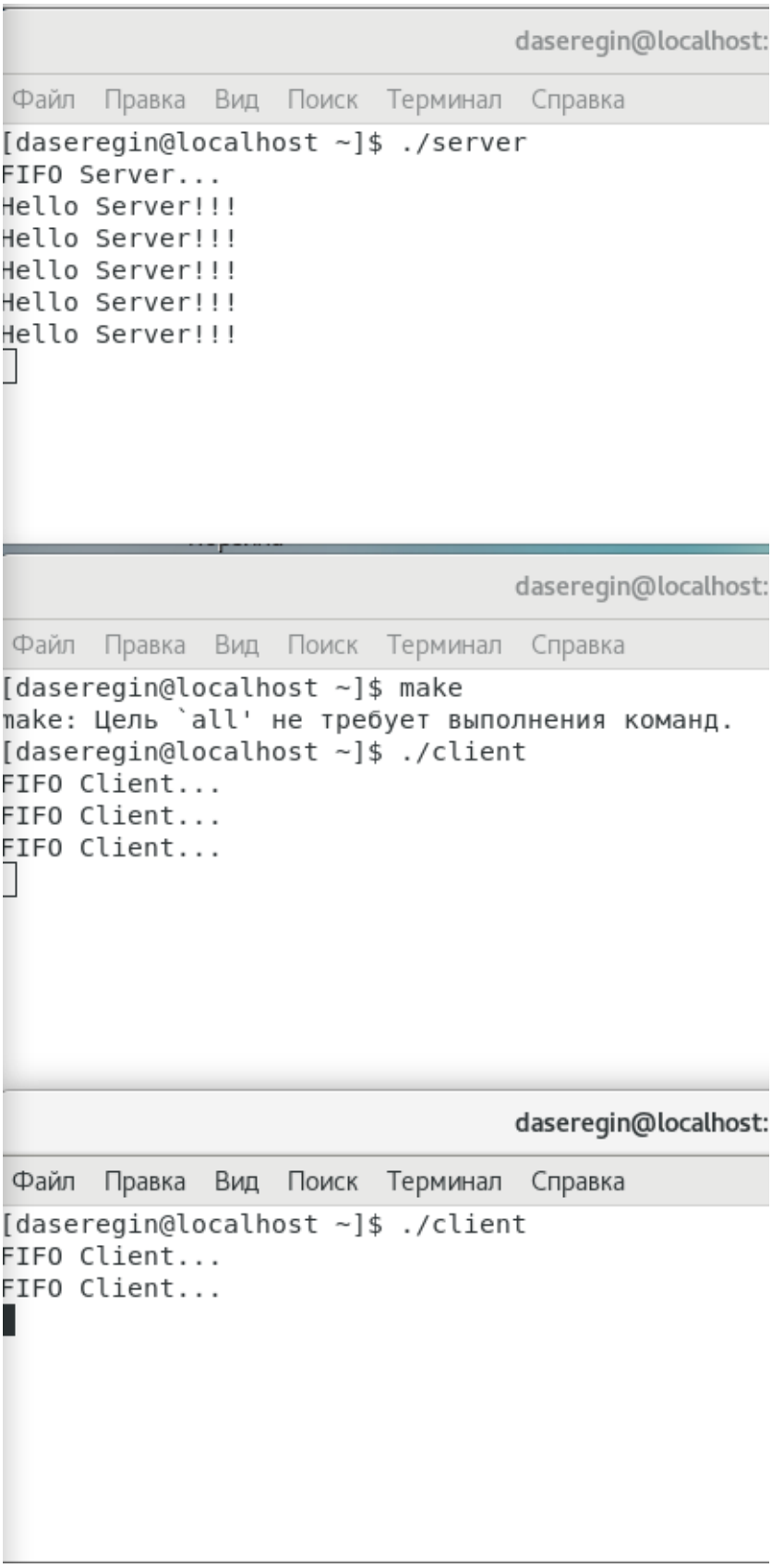

Figure 3.7: Работа программы с несколькими клиентами

4 Вывод

Я смог приобрести практические навыки работы с именованными каналами.

5 Использованные ресурсы

https://www.opennet.ru/docs/RUS/linux_parallel/node17.html