

Презентация к лабораторной работе №14

Шмырин Михаил Сергеевич

Российский Университет Дружбы Народов

Цель работы

Цель данной лабораторной работы - приобрести практические навыки работы с именованными каналами.

Выполнение лабораторной работы

1. Для выполнения лабораторной работы создал четыре файл с помощью команды `touch` и откроем их в `emacs` для редактирования (рис. 1)

```
midhs@msshnihrin ~/os-intro/labs $ cd lab14
midhs@msshnihrin ~/os-intro/labs/lab14 $ touch common.h server.c client.c Makefile
midhs@msshnihrin ~/os-intro/labs/lab14 $ ls
client.c  common.h  Makefile  presentation  report  server.c
midhs@msshnihrin ~/os-intro/labs/lab14 $
```

Рис. 1: Создание файлов

- Изменим код программ, предоставленных в тексте задания лабораторной работы. В файл `common.h` добавил стандартные заголовочные файлы `unistd.h` и `time.h`, которые необходимы для работы других файлов (рис. 2). Этот файл предназначен для заголовочных файлов, чтобы не прописывать их в других программах каждый раз

```
/*
 * 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__ */
```

Рис. 2: `common.h`

Затем в файл server.c добавим цикл while для контроля за временем работы сервера (рис. 3, 4), причем время от начал работы сервера до настоящего не должно превышать 30 секунд

```
1/*
2 * server.c - реализация сервера
3 *
4 * чтобы запустить пример, необходимо:
5 * 1. запустить программу server на одной консоли;
6 * 2. запустить программу client на другой консоли.
7 */
8
9#include "common.h"
10
11int
12main()
13{
14    int readfd; /* дескриптор для чтения из FIFO */
15    int n;
16    char buff[MAX_BUFF]; /* буфер для чтения данных из FIFO */
17
18    /* баннер */
19    printf("FIFO Server...\n");
20
21    /* создаем файл FIFO с открытыми для всех
22     * правами доступа на чтение и запись
23     */
24    if(mknod(FIFO_NAME, S_IFIFO | 0666, 0) < 0)
25    {
26        fprintf(stderr, "%s: Невозможно создать FIFO (%s)\n",
27                __FILE__, strerror(errno));
28        exit(-1);
29    }
30}
```

Рис. 3: server.c

```
/* 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 start = time(NULL);

/* читаем данные из FIFO и выводим на экран */
while(time(NULL)-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);
        }
    }
    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);
}
```

Рис. 4: server.c

В файл client.c добавим цикл for который будет отвечать за количество сообщений о текущем времени (4 сообщения), и команду sleep(5) для остановки работы клиента через 5 секунд. Также я изменил выводимое сообщение на текущее время (рис. 5, 6)

```
/*
 * 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...\n");
    for (int i=0; i<4; i++)
    {
        if((writefd = open(FIFO_NAME, O_WRONLY)) < 0)
        {
            printf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
                __FILE__, strerror(errno));
            exit(-1);
        }
        long int Time = time(NULL);
        char* text = ctime(&Time);
```

Рис. 5: client.c

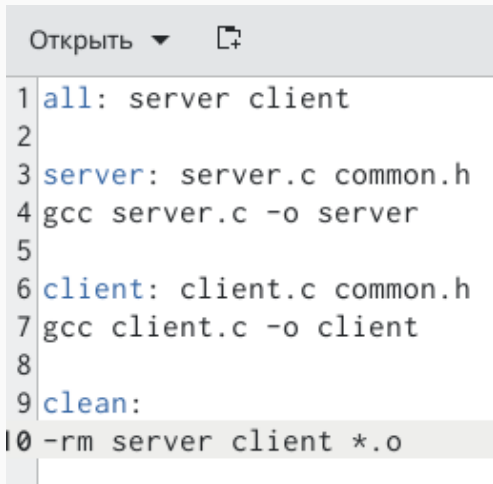
```
/* передадим сообщение серверу */
msglen = strlen(MESSAGE);
if(write(writefd, MESSAGE, msglen) != msglen)
{
    fprintf(stderr, "%s: Ошибка записи в FIFO (%s)\n",
        __FILE__, strerror(errno));
    exit(-2);
}
sleep (5);
}

/* закроем доступ к FIFO */
close(writefd);

exit(0);
}
```

Рис. 6: client.c

Makefile оставил без изменений (рис. 7)




```
Открыть ▼ 
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. 7: Makefile

- Используя команду `make all` (рис. 8), скомпилировал необходимые для работы файлы

```
mtlhs@msshmlhrin ~/os-intro/labs/lab14 $ make all
gcc client.c -o client
mtlhs@msshmlhrin ~/os-intro/labs/lab14 $ ls
client client.c common.h Makefile Makefile~ presentation report server server.c
mtlhs@msshmlhrin ~/os-intro/labs/lab14 $
```

Рис. 8: Компиляция файлов

4. Проверим работу написанного кода

Открыл три терминала, в первом окне запустил программу `./server`, во втором и третьем `./client`. В результате каждый терминал-клиент вывел по четыре сообщения о текущем времени. Спустя 30 секунд работы сервера был прекращена (рис. 9 , 10 , 11)

```
midhs@msshmthrin ~/os-intro/labs/lab14 $ cd ~/os-intro/labs/lab14
midhs@msshmthrin ~/os-intro/labs/lab14 $ ./server
FIFO Server...
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
midhs@msshmthrin ~/os-intro/labs/lab14 $
```

Рис. 9: Работа сервера

```
midhs@msshmthrln ~/os-intro/labs/lab14 $ cd ~/os-intro/labs/lab14
midhs@msshmthrln ~/os-intro/labs/lab14 $ ./client
FIFO Client...
midhs@msshmthrln ~/os-intro/labs/lab14 $ █
```

Рис. 10: Работа сервера

```
midhs@msshmthrln ~/os-intro/labs/lab14 $ cd ~/os-intro/labs/lab14
midhs@msshmthrln ~/os-intro/labs/lab14 $ ./client
FIFO Client...
midhs@msshmthrln ~/os-intro/labs/lab14 $ █
```

Рис. 11: Работа сервера

Если клиент завершит свою работу, не закрыв канал, то при повторном запуске сервера появится ошибка “Невозможно закрыть FIFO”, так как уже существует один канал.

Выводы

Я приобрел практические навыки по работе с именованными каналами.