Лабораторная работа №14

Именованные каналы

Панченко Денис Дмитриевич

Содержание

Список иллюстраций

1 Цель работы

Приобретение практических навыков работы с именованными каналами.

2 Выполнение лабораторной работы

Создадим файлы: common.h, server.c, client.c, Makefile (рис. 2.1).

```
[ddpanchenko@ddpanchenko os]$ cd lab14
[ddpanchenko@ddpanchenko lab14]$ touch common.h server.c client.c Makefile
```

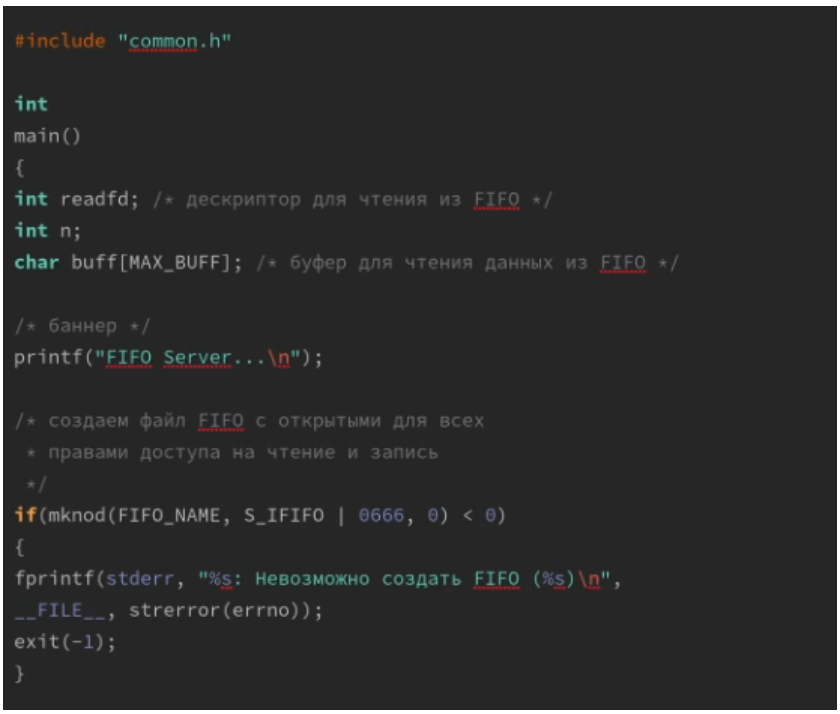
Рис. 2.1: Создание файлов

Файл common.h (рис. 2.2).

```
/*
* 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>

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

Рис. 2.2: common.h

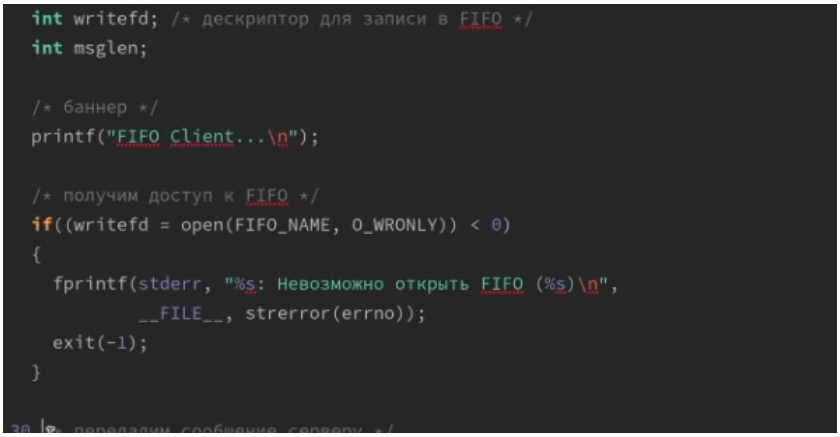
Файл server.c (рис. 2.3).



```c
#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);
}
```

Рис. 2.3: server.c

Файл client.c (рис. 2.4).



```c
  int writefd; /* дескриптор для записи в FIFO */
  int msglen;

  /* 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);
  }

30 /* передадим сообщение серверу */
```

Рис. 2.4: client.c

Makefile (рис. 2.5).

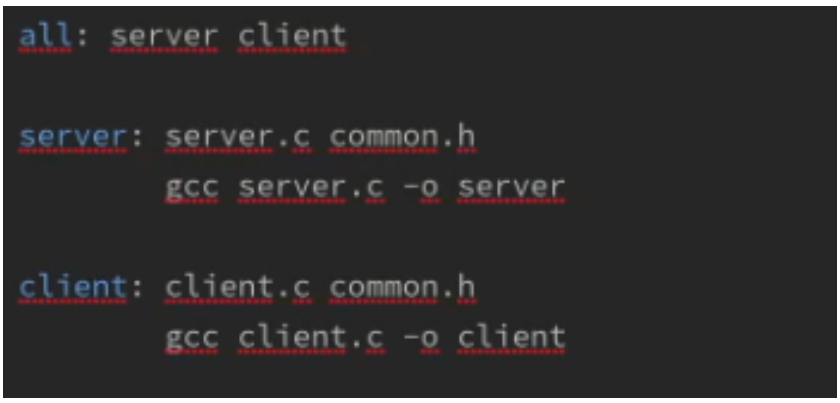


Рис. 2.5: Makefile

Запуск server.c (рис. 2.6).

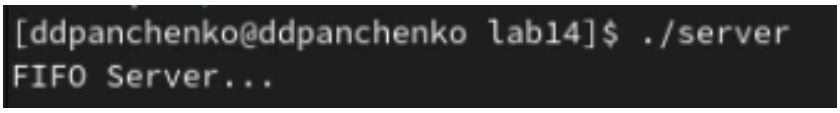


Рис. 2.6: Запуск программы

Запуск client.c (рис. 2.7).

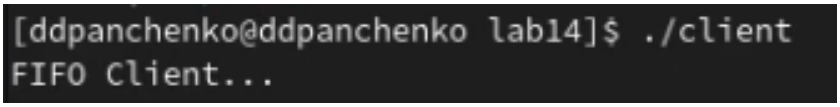


Рис. 2.7: Запуск программы

3 Вывод

Я приобрел практические навыки работы с именованными каналами.

4 Контрольные вопросы

1. Именованные каналы (FIFO) - это файлы, находящиеся в файловой системе, которые используются для обмена данными между процессами. Неименованные каналы - это временные файлы, создаваемые с помощью системного вызова pipe, которые используются для передачи данных между процессами, запущенными в рамках одного компьютера.

2. Нет, создание неименованного канала из командной строки невозможно.

3. Да, создание именованного канала из командной строки возможно с помощью утилиты mkfifo.

4. Функция pipe создает неименованный канал и возвращает два файловых дескриптора, один для чтения и один для записи в канал.

5. Функция mkfifo создает именованный канал с заданным именем и правами доступа и возвращает 0 в случае успешного создания и -1 в случае ошибки.

6. При чтении из FIFO меньшего числа байтов, чем находится в канале, процесс будет заблокирован до тех пор, пока не появятся новые данные в канале. При чтении большего числа байтов, процесс получит только те данные, которые есть в канале на данный момент.

7. При записи в FIFO меньшего числа байтов, чем позволяет буфер, данные будут записаны в канал, но процесс записи не будет завершен до тех пор, пока не будет записано достаточное количество данных, чтобы заполнить

буфер. При записи большего числа байтов, данные будут записаны в канал, а оставшиеся данные будут ожидать записи в буфере.

8. Два или более процессов могут читать или записывать в канал, но при этом может возникнуть проблема “гонки” (race condition), когда два процесса пытаются одновременно читать или записывать в канал. Для решения этой проблемы необходимо использовать синхронизацию процессов.

9. Функция write используется для записи данных в файл или файлоподобное устройство. Возвращает количество записанных байтов или -1 в случае ошибки. Число 1 в вызове функции в программе server.c (строка 42) означает, что записывается 1 байт.

10. Функция strerror возвращает строку с описанием ошибки, соответствующей заданному коду errno. Она принимает один аргумент - код ошибки errno. Например, strerror(errno) вернет строку, описывающую ошибку, которая произошла в программе.