

Лабораторная работа №14

Именованные каналы

Рогожина Н.А.

14 апреля 2023

Российский университет дружбы народов, Москва, Россия

Информация

- Рогожина Надежда Александровна
- Студентка 1го курса, НКАбд-02-22
- Компьютерные и информационные науки
- Российский университет дружбы народов
- Github

Вводная часть

- Приобретение практических навыков работы с именованными каналами.

Содержание лабораторной работы

Одним из видов взаимодействия между процессами в операционных системах является обмен сообщениями. Под сообщением понимается последовательность байтов, передаваемая от одного процесса другому. В операционных системах типа UNIX есть 3 вида межпроцессорных взаимодействий:

- общесистемные (именованные каналы, сигналы),
- System V Interface Definition (SVID – разделяемая память, очередь сообщений)
- BSD (сокеты)

Для передачи данных между неродственными процессами можно использовать механизм именованных каналов (named pipes). Данные передаются по принципу FIFO (First In First Out) (первым записан — первым прочитан), поэтому они называются также FIFO pipes или просто FIFO. 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.

Задание

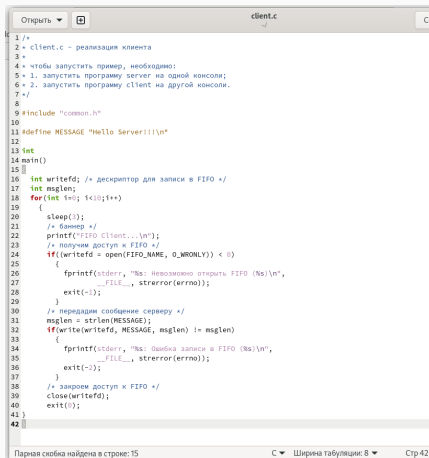
Задачи:

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.

Выполнение

Создание файла для первого клиента

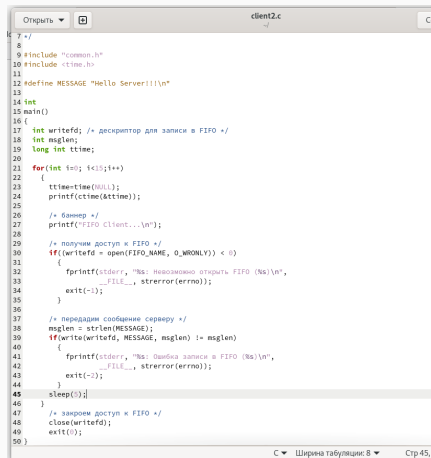


```
1 /*
2 * client.c - реализация клиента
3 *
4 * чтобы запустить пример, необходимо:
5 * 1. запустить программу server на одной консоли;
6 * 2. запустить программу client на другой консоли.
7 */
8
9 #include "common.h"
10
11 #define MESSAGE "Hello Server!!!\n"
12
13 int
14 main()
15 {
16     int writefd; /* дескриптор для записи в FIFO */
17     int msglen;
18     for(int i=0; i<10;i++)
19     {
20         sleep(3);
21         /* баннер */
22         printf("FIFO Client...\n");
23         /* получим доступ к FIFO */
24         if((writefd = open(FIFO_NAME, O_WRONLY)) < 0)
25         {
26             fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
27                     __FILE__, strerror(errno));
28             exit(-1);
29         }
30         /* передадим сообщение серверу */
31         msglen = strlen(MESSAGE);
32         if(write(writefd, MESSAGE, msglen) != msglen)
33         {
34             fprintf(stderr, "%s: Ошибка записи в FIFO (%s)\n",
35                     __FILE__, strerror(errno));
36             exit(-1);
37         }
38         /* закроем доступ к FIFO */
39         close(writefd);
40         exit(0);
41     }
42 }
```

Парная скобка найдена в строке: 15 С Ширина табуляции: 8 Стр 42

Рис. 1: client.c

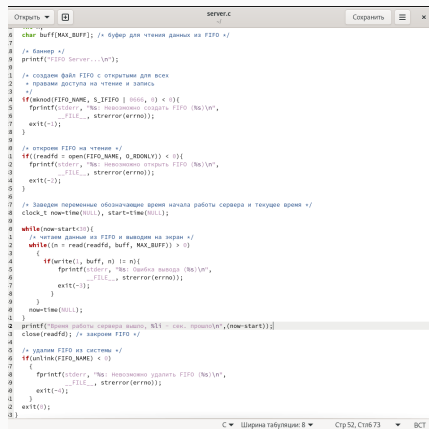
Создание файла для второго клиента



```
1 7 */
2 8
3 9 #include "common.h"
4 10 #include <time.h>
5 11
6 12 #define MESSAGE "Hello Server!!!\n"
7 13
8 14 int
9 15 main()
10 16 {
11 17     int writefd; /* дескриптор для записи в FIFO */
12 18     int msglen;
13 19     long int ttime;
14 20
15 21     for(int i=0; i<15;i++)
16 22     {
17 23         ttime=time(NULL);
18 24         printf("ctime(&ttime));
19 25
20 26         /* баннер */
21 27         printf("FIFO Client...\n");
22 28
23 29         /* получим доступ к FIFO */
24 30         if((writefd = open(FIFO_NAME, O_WRONLY)) < 0)
25 31         {
26 32             fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
27 33                 __FILE__, strerror(errno));
28 34             exit(-1);
29 35         }
30 36
31 37         /* передадим сообщение серверу */
32 38         msglen = strlen(MESSAGE);
33 39         if(write(writefd, MESSAGE, msglen) != msglen)
34 40         {
35 41             fprintf(stderr, "%s: Ошибка записи в FIFO (%s)\n",
36 42                 __FILE__, strerror(errno));
37 43             exit(-2);
38 44         }
39 45         sleep(5);
40 46     }
41 47     /* закроем доступ к FIFO */
42 48     close(writefd);
43 49     exit(0);
44 50 }
```

Рис. 2: client2.c

Создание файла сервера



```
server.c
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
1  #include <stdio.h>
2  #include <unistd.h>
3  #include <stdlib.h>
4  #include <sys/stat.h>
5  #include <sys/types.h>
6  #include <fcntl.h>
7  #include <time.h>
8  #include <errno.h>
9
10 char buff[MAX_BUF]; /* буфер для чтения данных из FIFO */
11
12 /* Баннер */
13 printf("FIFO Server...\n");
14
15 /* создаем файл FIFO с открытыми для всех
16  * правами доступа на чтение и запись
17  */
18 if(mknod(FIFO_NAME, S_IFIFO | 0666, 0) < 0){
19     fprintf(stderr, "%s: невозможно создать FIFO (%s)\n",
20             __FILE__, strerror(errno));
21     exit(-1);
22 }
23
24 /* открываем FIFO на чтение */
25 if((readfd = open(FIFO_NAME, O_RDONLY)) < 0){
26     fprintf(stderr, "%s: невозможно открыть FIFO (%s)\n",
27             __FILE__, strerror(errno));
28     exit(-2);
29 }
30
31 /* Задаем переменные обозначающие время начала работы сервера и текущее время */
32 clock_t now_time(NULL), start_time(NULL);
33
34 while(now-start<10){
35     /* читаем данные из FIFO и выводим на экран */
36     while((n = read(readfd, buff, MAX_BUF)) > 0)
37     {
38         if(write(1, buff, n) != n){
39             fprintf(stderr, "%s: ошибка вывода (%s)\n",
40                     __FILE__, strerror(errno));
41             exit(-3);
42         }
43     }
44     now_time(NULL);
45 }
46 printf("Время работы сервера вышло, %li - сек. прошло\n", (now-start));
47 close(readfd); /* закрываем FIFO */
48
49 /* удалим FIFO из системы */
50 if(unlink(FIFO_NAME) < 0)
51 {
52     fprintf(stderr, "%s: невозможно удалить FIFO (%s)\n",
53             __FILE__, strerror(errno));
54     exit(-4);
55 }
56 exit(0);
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Рис. 3: server.c

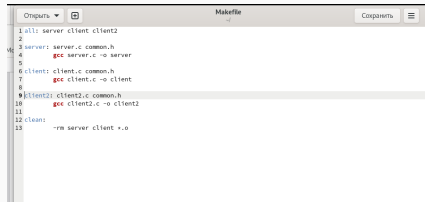
Создание заголовочного файла со стандартными определениями



The image shows a code editor window titled 'common.h'. The editor contains the following C code:

```
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
17 #define FIFO_NAME "/tmp/fifo"
18 #define MAX_BUFF 80
19
20 #endif /* __COMMON_H__ */
```

Рис. 4: common.h

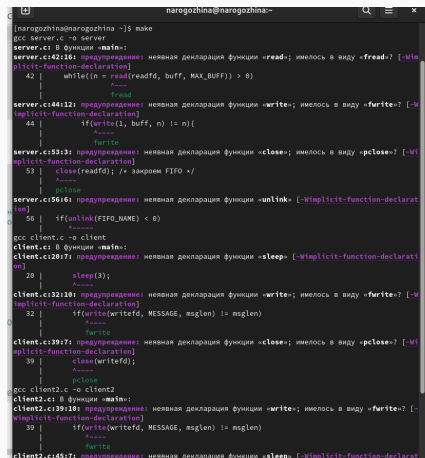


The screenshot shows a text editor window titled "Makefile" with a menu bar containing "Открыть", "Сохранить", and a list icon. The editor contains the following Makefile content:

```
1 all: server client client2
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 client2: client2.c common.h
10    gcc client2.c -o client2
11
12 clean:
13     -rm server client *.o
```

Рис. 5: Makefile

Запуск make



```
narogozhina@narogozhina:~$ make
gcc server.c -o server
server.c: В функции «main»:
server.c:42:16: предупреждение: неявная декларация функции «read»; имелось в виду «fread»? [-Wimplicit-function-declaration]
   42 |     while((n = read(readfd, buff, MAX_BUFF)) > 0)
      |                ^~~~~
      |                fread
server.c:44:12: предупреждение: неявная декларация функции «write»; имелось в виду «fwrite»? [-Wimplicit-function-declaration]
   44 |         if(write(1, buff, n) != n){
      |            ^~~~~
      |            fwrite
server.c:53:3: предупреждение: неявная декларация функции «close»; имелось в виду «pclose»? [-Wimplicit-function-declaration]
   53 |     close(readfd); /* закроем FIFO */
      |     ^~~~~
      |     pclose
server.c:56:16: предупреждение: неявная декларация функции «unlink» [-Wimplicit-function-declaration]
   56 |     if(unlink(FIFO_NAME) < 0)
      |        ^~~~~~
      |        unlink
gcc client.c -o client
client.c: В функции «main»:
client.c:20:7: предупреждение: неявная декларация функции «sleep» [-Wimplicit-function-declaration]
   20 |     sleep(3);
      |     ^~~~~
      |     sleep
client.c:32:10: предупреждение: неявная декларация функции «write»; имелось в виду «fwrite»? [-Wimplicit-function-declaration]
   32 |     if(write(writefd, MESSAGE, msglen) != msglen)
      |        ^~~~~
      |        fwrite
client.c:39:7: предупреждение: неявная декларация функции «close»; имелось в виду «pclose»? [-Wimplicit-function-declaration]
   39 |     close(writefd);
      |     ^~~~~
      |     pclose
gcc client2.c -o client2
client2.c: В функции «main»:
client2.c:39:10: предупреждение: неявная декларация функции «write»; имелось в виду «fwrite»? [-Wimplicit-function-declaration]
   39 |     if(write(writefd, MESSAGE, msglen) != msglen)
      |        ^~~~~
      |        fwrite
client2.c:45:7: предупреждение: неявная декларация функции «sleep» [-Wimplicit-function-declaration]
```

Рис. 6: make

Запуск сервера и клиентов

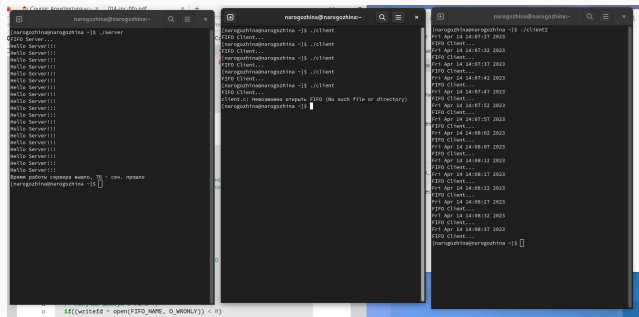


Рис. 7: Запуск сервера с клиентами

Выводы

В ходе лабораторной работы мы приобрели практические навыки работы с именованными каналами.