

Именованные каналы

Лебедева Ольга Андреевна

¹RUDN University, Moscow, Russian Federation

Приобретение практических навыков работы с именованными каналами.

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: общеюниксные (именованные каналы, сигналы), System V Interface Definition (SVID — разделяемая память, очередь сообщений, семафоры) и 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 :

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.

Сначала мы создали необходимые файлы, записав код через редактор emacs.(рис. 1)

A terminal window with a dark background and light green text. It shows a series of commands being executed in a shell. The user is 'oalebedeva' on host 'dk2n22'. The commands are: 'mkdir lab14', 'cd lab14', and then five 'emacs' commands to create files: 'common.h', 'server.c', 'client.c', 'Makefile', and 'client.c'. The last line is partially cut off with a '^' character.

```
oalebedeva@dk2n22 ~ $ mkdir lab14
oalebedeva@dk2n22 ~ $ cd lab14
oalebedeva@dk2n22 ~/lab14 $ emacs common.h
oalebedeva@dk2n22 ~/lab14 $ emacs server.c
oalebedeva@dk2n22 ~/lab14 $ emacs client.c
oalebedeva@dk2n22 ~/lab14 $ emacs Makefile
oalebedeva@dk2n22 ~/lab14 $ emacs client.c
^Coalebedeva@dk2n22 ~/lab14 $ emacs client1.c
```

Figure 1: Создание файлов

Далее, мы написали код в каждый из файлов. (рис. 2) (рис. 3) (рис. 4) (рис. 5) (рис. 6)

```
1 /*
2 * common.h - заголовочный файл со стандартными определениями
3 */
4 #ifndef __COMMON_H__
5 #define __COMMON_H__
6 #include <stdio.h>
7 #include <stdlib.h>
8 #include <string.h>
9 #include <errno.h>
10 #include <sys/types.h>
11 #include <sys/stat.h>
12 #include <fcntl.h>
13 #define FIFO_NAME "/tmp/fifo"
14 #define MAX_BUFF 80
15 #endif /* __COMMON_H__ */
```

Figure 2: Код файла common.h

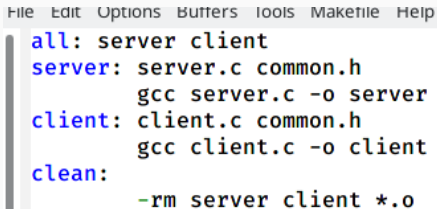
Ход работы

```
1 #include "common.h"
2 int
3 main()
4 {
5     int readfd; /* дескриптор для чтения из FIFO */
6     int n;
7     char buff[MAX_BUFF]; /* буфер для чтения данных из FIFO */
8     /* баннер */
9     printf("FIFO Server...\n");
10
11     if(mknod(FIFO_NAME, S_IFIFO | 0666, 0) < 0)
12     {
13         fprintf(stderr, "%s: Невозможно создать FIFO (%s)\n",
14             __FILE__, strerror(errno));
15         exit(-1);
16     }
17     /* откроем FIFO на чтение */
18     if((readfd = open(FIFO_NAME, O_RDONLY)) < 0)
19     {
20         fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
21             __FILE__, strerror(errno));
22         exit(-2);
23     }
24     clock_t now=time(NULL), start=time(NULL);
25     while(now-start<30)
26     {
27         while((n = read(readfd, buff, MAX_BUFF)) > 0)
28         {
29             if(write(1, buff, n) !=n)
30             {
31                 fprintf(stderr, "%s: Ошибка вывода (%s)\n",
32                     __FILE__, strerror(errno));
33             }
34         }
35         now=time(NULL);
36     }
37     printf("server timeout, %li - second passed\n", (now-start));
38     close(readfd); /* закроем FIFO */
39     /* удалим FIFO из системы */
40     if(unlink(FIFO_NAME) < 0)
41     {
42         fprintf(stderr, "%s: Невозможно удалить FIFO (%s)\n",
43             __FILE__, strerror(errno));
44         exit(-4);
45     }
46     exit(0);
47 }
```



```
1 #include "common.h"
2 #define MESSAGE "Hello Server!!!\n"
3 int
4 main()
5 {
6     int msg, len, i;
7     long int t;
8     for(i=0; i<20;i++)
9     {
10         sleep(3);
11         t = time(NULL);
12         printf("FIFO Client...\n");
13
14         if((msg = open(FIFO_NAME,O_WRONLY))<0)
15         {
16             fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
17                 __FILE__, strerror(errno));
18             exit(-1);
19         }
20         len = strlen(MESSAGE);
21
22         if(write(msg,MESSAGE, len) != len)
23         {
24             fprintf(stderr, "%s: Ошибка в записи в FIFO (%s)\n",
25                 __FILE__, strerror(errno));
26             exit(-2);
27         }
28         close(msg);
29     }
30     exit(0);
31 }
32
```

Figure 4: Код файла client.c



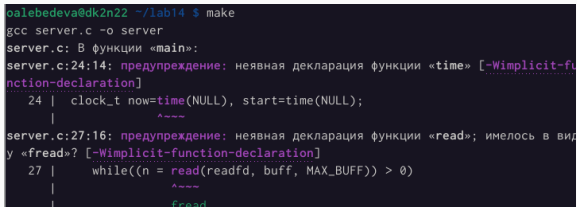
```
File Edit Options Buffers Tools Makefile Help
all: server client
server: server.c common.h
    gcc server.c -o server
client: client.c common.h
    gcc client.c -o client
clean:
    -rm server client *.o
```

Figure 5: Код Makefile

Ход работы

```
1 #include "common.h"
2 #define MESSAGE "Hello Server!!!\n"
3 int
4 main()
5 {
6     int msg, len, l;
7     long long int t;
8     char message[10];
9     for(count = 0; count<-5;++count)
10     {
11         sleep(5);
12         t = (long long int) time(0);
13         sprintf(message, "%lli", t);
14
15         if((writefd = open(FIFO_NAME, O_WRONLY)) < 0)
16         {
17             fprintf(stderr, "%s: Невозможно открыть FIFO(%s)\n",
18                 __FILE__, strerror(errno));
19             exit(-1);
20         }
21
22         msglen = strlen(MESSAGE);
23         if(write(writefd, MESSAGE, msglen) != msglen)
24         {
25             fprintf(stderr, "%s: Ошибка записи в FIFO (%s)\n",
26                 __FILE__, strerror(errno));
27             exit(-2);
28         }
29         /* закроем доступ к FIFO */
30         close(writefd);
31         exit(0);
32     }
```

Запустили Makefile. (рис. 7)



```
oalebedeva@dk2n22 ~/lab14 $ make
gcc server.c -o server
server.c: В функции «main»:
server.c:24:14: предупреждение: неявная декларация функции «time» [-Wimplicit-fu
nction-declaration]
   24 |   clock_t now=time(NULL), start=time(NULL);
      |               ^~~~
server.c:27:16: предупреждение: неявная декларация функции «read»; имелось в вид
е «fread»? [-Wimplicit-function-declaration]
   27 |   while((n = read(readfd, buff, MAX_BUFF)) > 0)
      |               ^~~~
      |               fread
```

Figure 7: Запуск Makefile

Открыли второй терминал. В одном из них запустили server, в другом - client. (рис. 8)

```

kali@kali:~/c22-2-rub14$ cd lab14
kali@kali:~/c22-2-rub14/lab14$ ./client
FIFO Client...
FIFO Client...
FIFO Client...
FIFO Client...
FIFO Client...
FIFO Client...
FIFO Client...
FIFO Client...
FIFO Client...
FIFO Client...
client.c:22:4: предупреждение: неточная декларация функции «write», имелась в виду «fwrite» [-Wimplicit-function-declaration]
  22 | if(write(msg,MESSAGE, len) != len)
     |     ^~~~~
client.c:26:13: предупреждение: неточная декларация функции «close», имелась в виду «pclose» [-Wimplicit-function-declaration]
   26 |         close(msg);
      |         ^~~~~
kali@kali:~/c22-2-rub14/lab14$ ./server
FIFO Server...
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
server Listen, 30 - second passed
kali@kali:~/c22-2-rub14/lab14$

```

Figure 8: Работа программы

Приобрели практические навыки работы с именованными каналами.