Отчет по лабораторной работа №14

Именованные каналы

Лушин Артем Андреевич

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Приобретение практических навыков работы с именованными каналами.

2 Выполнение лабораторной работы

1) Я создал каталог 14 и в нем создал файлы: client.c, client2.c,common.h,server.c. Взял пример с файла лабораторной работы и написал измененную программму, которая выполняет все нужные критерии.

Рис. 2.1: Код в файле client.c

Рис. 2.2: Код в файле client2.c

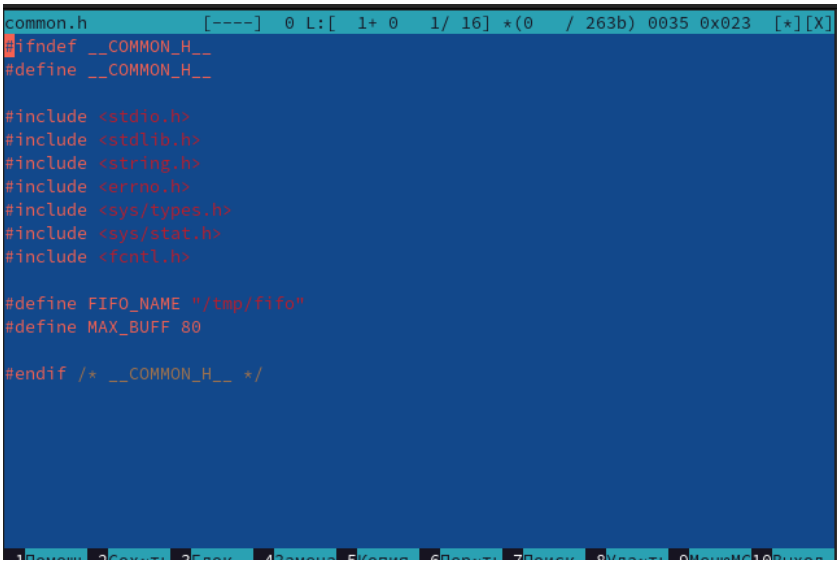

Рис. 2.3: Код в файле common.h

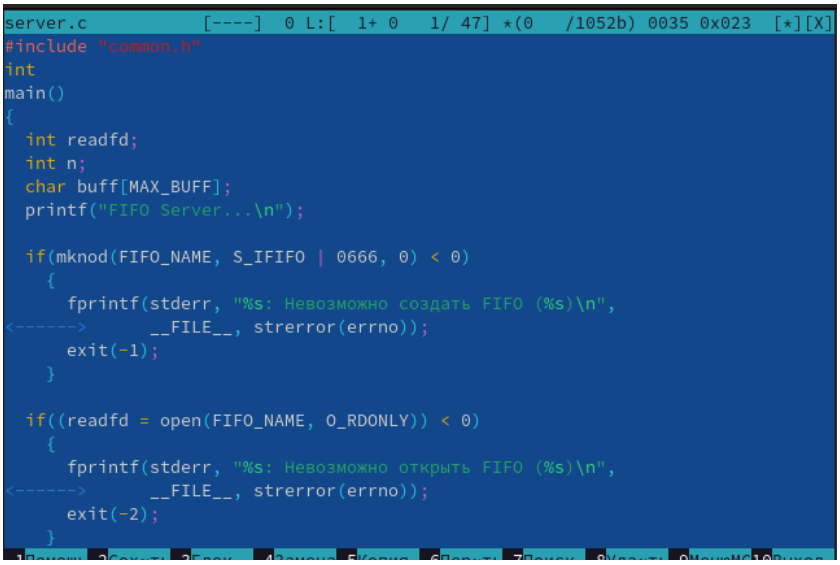

Рис. 2.4: Код в файле server.c

2) Создал файл Makefile для запуска файлов. Код взял в файле для лабораторной и немного изменил его.

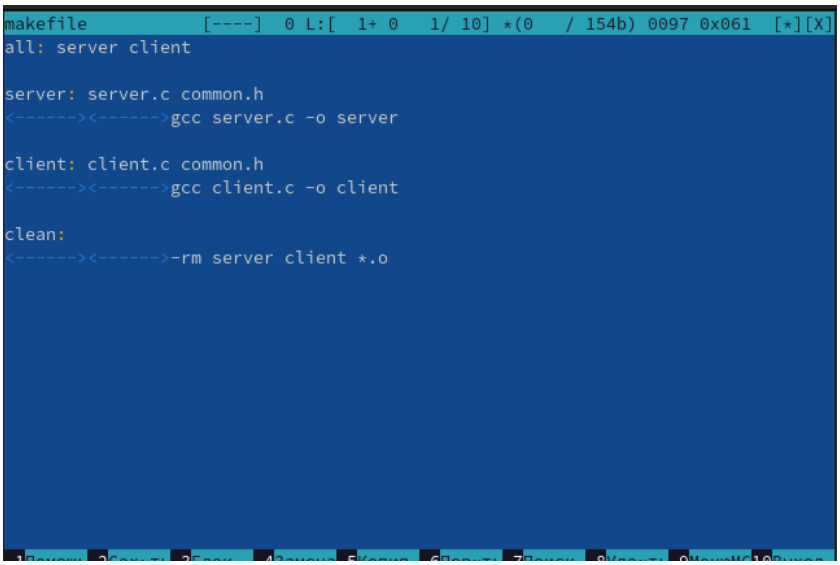

Рис. 2.5: Код в файле Makefile

3) Прописал команду make для создания программ. Создалось две нужные программы.

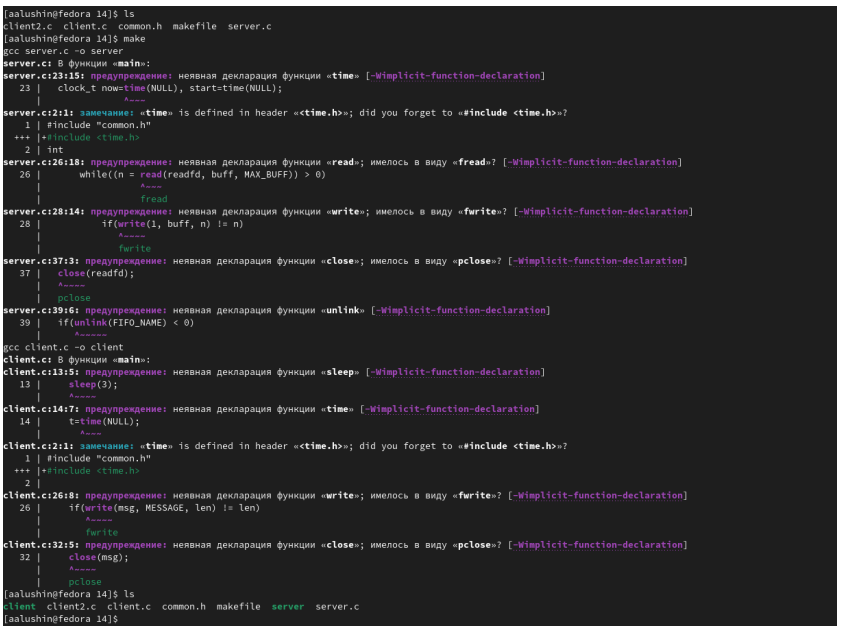

Рис. 2.6: Создание программ

4) Запустил с двух консолей программы client и server. Запуск производится с двух консолей, чтобы имитировать двух пользователей. Если хотя бы один из пользователей не будет доступен, то программа не будет работать.

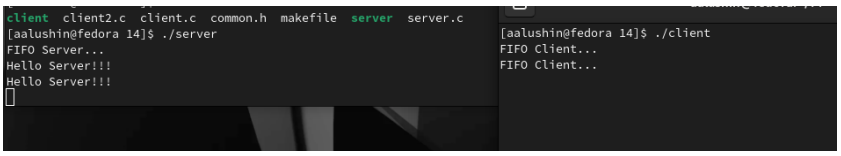

Рис. 2.7: Запуск программы

3 Контрольные вопросы

1. В чем ключевое отличие именованных каналов от неименованных?

Именованные каналы отличаются от неименованных наличием идентификатора канала, который представлен как специальный файл (соответственно имя именованного канала — это имя файла).

2. Возможно ли создание неименованного канала из командной строки?

Создание неименованного канала из командной строки возможно командой pipe.

3. Возможно ли создание именованного канала из командной строки?

Создание именованного канала из командной строки возможно с помощью mkfifo.

4. Опишите функцию языка C, создающую неименованный канал.

Функция языка C, создающая неименованный канал: int read(int pipe_fd, void area, int cnt); int write(int pipe_fd, void area, int cnt); Первый аргумент этих вызовов - дескриптор канала, второй - указатель на область памяти, с которой происходит обмен, третий - количество байт. Оба вызова возвращают число переданных байт (или -1 - при ошибке).

5. Опишите функцию языка C, создающую именованный канал.

Функция языка C, создающая именованный канал: int mkfifo (const char *pathname, mode_t mode); Первый параметр — имя файла, идентифицирующего канал, второй параметр маска прав доступа к файлу. Вызов функции mkfifo() создаёт файл канала (с именем, заданным макросом FIFO_NAME): mkfifo(FIFO_NAME, 0600);

6. Что будет в случае прочтения из fifo меньшего числа байтов, чем находится в канале? Большего числа байтов?

При чтении меньшего числа байтов, возвращается требуемое число байтов, остаток сохраняется для следующих чтений. При чтении большего числа байтов, возвращается доступное число байтов 7. Запись числа байтов, меньшего емкости канала или FIFO, гарантированно атомарно. Это означает, что в случае, когда несколько процессов одновременно записывают в канал, порции данных от этих процессов не перемешиваются. При записи большего числа байтов, чем это позволяет канал или FIFO, вызов write(2) блокируется до освобождения требуемого места. При этом атомарность операции не гарантируется. Если процесс пытается записать данные в канал, не открытый ни одним процессом на чтение, процессу генерируется сигнал SIGPIPE, а вызов write(2) возвращает 0 с установкой ошибки (errno=EP1PE) (если процесс не установил обработки сигнала SIGPIPE, производится обработка по умолчанию – процесс завершается).

7. Аналогично, что будет в случае записи в fifo меньшего числа байтов, чем позволяет буфер? Большего числа байтов?

Запись числа байтов, меньшего емкости канала или FIFO, гарантированно атомарно. Это означает, что в случае, когда несколько процессов одновременно записывают в канал, порции данных от этих процессов не перемешиваются. При записи большего числа байтов, чем это позволяет канал или FIFO, вызов write(2) блокируется до освобождения требуемого места. При этом атомарность операции не гарантируется. Если процесс пытается записать данные в канал, не

открытый ни одним процессом на чтение, процессу генерируется сигнал SIGPIPE, а вызов write(2) возвращает 0 с установкой ошибки (errno=EP1PE) (если процесс не установил обработки сигнала SIGPIPE, производится обработка по умолчанию – процесс завершается).

8. Могут ли два и более процессов читать или записывать в канал?

Два и более процессов могут читать и записывать в канал.

9. Опишите функцию write (тип возвращаемого значения, аргументы и логику работы). Что означает 1 (единица) в вызове этой функции в программе server.c (строка 42)?

Функция write записывает length байтов из буфера buffer в файл, определенный дескриптором файла fd. Эта операция чисто ‘двоичная’ и без буферизации. При единице возвращает действительное число байтов. Функция write возвращает число действительно записанных в файл байтов или -1 при ошибке, устанавливая при этом errno.

10. Опишите функцию strerror

Строковая функция strerror - функция языков C/C++, транслирующая код ошибки, который обычно хранится в глобальной переменной errno, в сообщение об ошибке, понятном человеку.

4 Выводы

Я приобрел практические навыки работы с именованными файлами.

Список литературы