Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Операционные системы и системное программирование

ОТЧЕТ
по лабораторной работе №7
на тему
“Блокировки чтения/записи”

Выполнил: студент группы 350501
М.А. Василюк

Проверил: старший преподаватель каф. ЭВМ
Л.П. Поденок

Минск 2025

СОДЕРЖАНИЕ

1 ЗАДАНИЕ ЛАБОРАТОРНОЙ РАБОТЫ

1.1 Цель работы

Программа в режиме конкурентного доступа читает из и пишет в файл, содержащий записи фиксированного формата.

1.2 Исходные данные к работе

Очередь сообщений представляет собой классическую структуру – кольцевой буфер, содержащий указатели на сообщения, и пара указателей на голову и хвост.

Файл должен содержать не менее 10 записей. Создается и наполняется с помощью любых средств.

Программа должна выполнять следующие операции:

1) `LST` – Отображение содержимого файла с последовательной нумерацией записей

2) `GET Rec_No` – получение записи с порядковым номером `Rec_No`;

3) Модификацию полей записи

4) `PUT` – сохранение последней прочитанной и модифицированной записи по месту. Интерфейс с пользователем на «вкус» студента.

Для отладки и тестирования используется не менее двух экземпляров программы.

Для блокирования записей используется `fcntl`() и исключительные блокировки на основе файловых описаний (`F_OFD_*`).

2 ВЫПОЛНЕНИЕ РАБОТЫ

2.1 Описание алгоритма выполнения работы

Программа представляет собой многопоточное приложение на C, разработанное с использованием `POSIX Threads`, которое решает классическую задачу взаимодействия "производитель-потребитель" посредством разделяемого кольцевого буфера, реализуя для синхронизации два подхода: на основе `POSIX` семафоров и мьютекса, и на основе мьютекса и `POSIX` условных переменных; главный поток обеспечивает динамическое управление потоками-производителями (генерируют сообщения со случайным содержимым и контрольной суммой) и потоками-потребителями (извлекают и проверяют сообщения), а также позволяет контролировать состояние буфера, потоков и размер очереди через пользовательский ввод, гарантируя корректное завершение работы и освобождение ресурсов.

2.2 Описание основных функций

Функция do_list()

Функция `do_list` отвечает за отображение содержимого файла записей на стандартный вывод. Сначала она устанавливает указатель чтения файла в начало. Затем выводит заголовок таблицы с номерами, именами устройств, инвентарными номерами и местоположениями. Далее, в цикле, она последовательно читает записи из файла. Для каждой успешно прочитанной записи она выводит ее порядковый номер и содержимое полей в форматированном виде. Цикл чтения продолжается до тех пор, пока не будет достигнут конец файла или произойдет ошибка чтения.

Функция do_get()

Функция `do_get` реализует получение записи из файла для последующего редактирования. Она запрашивает у пользователя номер записи, которую необходимо отредактировать. После проверки корректности введенного номера, функция вычисляет смещение нужной записи в файле и перемещает файловый указатель на эту позицию. Затем она считывает запись из файла в буфер `orig_rec` и копирует ее в рабочую область `work_rec`. Также запоминается номер прочитанной записи (`last_rec_no`) и устанавливается флаг has_work, сигнализирующий о наличии отредактированной записи. Далее функция отображает текущее содержимое записи и предлагает пользователю ввести новые значения для полей "Device `Name`", "`Location`" и "`Inventory Number`" с возможностью оставить поле без изменений, введя пустую строку. Введенные значения сохраняются в буфере `work_rec`.

Функция main()

Функция `main` является точкой входа в программу. Она проверяет количество аргументов командной строки: программа ожидает один аргумент – имя файла с данными. В случае некорректного количества аргументов выводится сообщение об использовании. Затем функция пытается открыть указанный файл в режиме чтения и записи. При ошибке открытия файла выводится сообщение об ошибке. Далее определяется размер файла и вычисляется количество записей на основе размера записи `record_t`. Если количество записей меньше 10, выводится предупреждение. Затем программа выводит приветственное сообщение и информацию о загруженном количестве записей. После этого программа входит в бесконечный цикл, в котором пользователю предлагается список команд: 'l' (отобразить список записей), 'g' (получить запись для редактирования), 'p' (сохранить изменения) и 'q' (выйти). В зависимости от введенной команды вызывается соответствующая функция (`do_list, do_get, do_put`). Команда 'q' приводит к выходу из цикла, закрытию файла и завершению программы. Неизвестные команды игнорируются.

3 РЕЗУЛЬТАТЫ РАБОТЫ ПРОГРАММЫ

```
Commands list:
l - show records list
g - get record and edit
p - put and save changes
q - exit
Select: l

#   Device Name                    Inv#     Location
 0  Device013                           1  Location676
 1  re                                  4  wr
 2  Device080                           3  Location441
 3  Device011                           4  Location082
 4  Device008                           5  Location135
 5  Device022                           6  Location656
 6  Device013                           7  Location484
 7  Device032                           8  Location998
 8  Device054                           1  Location530
 9  Device003                           2  Location694
10  Device073                           3  Location822
```

Рисунок 1 – вывод листа

```
cConsumer was created 0 (semaphores)
Commands list:
l - show records list
g - get record and edit
p - put and save changes
q - exit
Select: g
Enter record number to edit (0..14): 3

Current record #3:
  Device: Device011
  Location: Location082
  Inventory #: 4

New device name (1-79 chars, empty to keep): me
New location (1-79 chars, empty to keep): me
New inventory number (positive integer, empty to keep): 2
```

Рисунок 2 – пример изменения

4 ВЫВОД

В ходе выполнения лабораторной работы была разработана программа, демонстрирующая принципы конкурентного доступа к совместно используемому файлу посредством потоков `POSIX` и механизма блокировок чтения-записи, реализованного с использованием системного вызова `fcntl`. Программа предоставляет интерактивный интерфейс, управляемый главным потоком через стандартный ввод, позволяя пользователю просматривать содержимое файла записей фиксированного формата, получать отдельные записи для модификации и сохранять внесенные изменения обратно в файл. Для обеспечения корректной работы в условиях конкурентного доступа, применен алгоритм блокировки записей, использующий эксклюзивные блокировки на основе файловых дескрипторов (`F_OFD_*`), что предотвращает коллизии при одновременном чтении и записи данных несколькими экземплярами программы. При попытке модификации записи, программа осуществляет блокировку соответствующей области файла, предварительно проверяя ее текущий статус. В случае обнаружения изменений, внесенных другим процессом после чтения записи, программа сообщает о конфликте и предлагает повторить операцию. Разработанное решение наглядно иллюстрирует применение механизма блокировок `fcntl` для управления конкурентным доступом к файлу, обеспечивая целостность данных при одновременной работе нескольких процессов. В процессе выполнения операций программа выводит в стандартный поток вывода информацию о выполняемых действиях, такую как чтение, блокировка, запись и обнаружение конфликтов.