

Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №7
за 4 семестр
По дисциплине: «ОСиСП»
Тема: «СЕМАФОРЫ»

Выполнил:
студент 2-ого курса
Шевчук А. В.
группы ПО-3
Проверила:
Давидюк Ю. И.

Вариант 27 (7)

Задание для выполнения

Ознакомиться с руководством, теоретическими сведениями и лекционным материалом по использованию и функционированию средств синхронизации - семафоров Дейкстры, и их реализацией в Linux - System V IPC семафоры с POSIX-семафоры.

Написать две (или более) программы, которые, работая параллельно заиклено, обмениваются информацией согласно варианту. Передачу и получение информации каждым из процессов сопровождать выводом на экран информации типа "процесс такой-то передал/получил такую-то информацию". Синхронизацию работы процессов реализовать с помощью семафоров. Учтите, что при организации совместного доступа к разделяемому (например, файлу) вам понадобится применять, например, мьютексы.

Для наглядности запускаете свои процессы в разных окнах терминала. Запустите программы в нескольких экземплярах (одну первую и две/три вторых, две первых и две вторых...).

Первый процесс пишет в файл строку вида «pid - текущее время», каждый раз открывая и закрывая файл, а второй процесс эти строки читает и выводит, дополняя своим pid.

Ход работы

Код программы program7_1.c:

```
#include <fcntl.h>
#include <sys/stat.h>
#include <semaphore.h>
#include <stdio.h>
#include <time.h>
#include <sys/types.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <pthread.h>
int main(){
    long int ttime; // Переменная для сохранения текущего времени
    int i, j, ValuePID, fd, sizeRead;
    char *Value;
    char string[255];
    char sem_name1[]="mytex";
    char sem_name2[]="empty";
    char sem_name3[]="full";
    for(j=0; j<10; j++){
        sem_t *s1 = sem_open(sem_name1, O_CREAT, 0666, 1);
        sem_t *s2 = sem_open(sem_name2, O_CREAT, 0666, 1);
        sem_t *s3 = sem_open(sem_name3, O_CREAT, 0666, 0);
        sem_wait(s2);
        sem_wait(s1);
```

```

(void)umask(0);
if((fd=open("BUF", O_WRONLY | O_CREAT, 0666)) < 0){
    printf("Can't open file BUF\n");
    return(-1);
}
ValuePID=getpid();
/*Подготовка номера PID к выводу*/
int razryad=1;
i=ValuePID;
while ((i = i/10) != 0){
    razryad++;
}
Value = (char*) malloc(razryad+1);
for (i=razryad-1; i>=0; i--){
    Value[i]=(ValuePID % 10)+'0';
    ValuePID = ValuePID/10;
}
Value[razryad]='-';
if (write(fd, Value, strlen(Value))<= 0){
    printf("Can't write.\n");
    return(-1);
}
ttime = time (NULL); // Считываем текущее время
// С помощью функции ctime преобразуем считанное время в локальное, а затем в строку
if (write(fd, ctime (&ttime), strlen(ctime (&ttime)))<= 0){
    printf("Can't write.\n");
    return(-1);
}
printf("Prog1 отправила информацию в файл BUF.\n");
if(close(fd) < 0){
    printf("Can't close file BUF\n");
}
sem_post(s1);
sem_post(s3);
sem_close(s1);
sem_close(s2);
sem_close(s3);
}
return 0;
}

```

Код программы program7_2.c:

```

#include <fcntl.h>
#include <sys/stat.h>
#include <semaphore.h>
#include <stdio.h>
#include <sys/types.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <pthread.h>
int main(){
    int i, fd;
    char string[255];
    char sem_name1[]="mytex";
    char sem_name2[]="empty";
    char sem_name3[]="full";
    while (1){
        sem_t *s1 = sem_open(sem_name1, O_CREAT, 0666);
        sem_t *s2 = sem_open(sem_name2, O_CREAT, 0666);
        sem_t *s3 = sem_open(sem_name3, O_CREAT, 0666);
        sem_wait(s3);
        sem_wait(s1);
    }
}

```

```

if((fd=open("BUF", O_RDONLY)) < 0){
    printf("Can\'t open file BUF\n");
    return(-1);
}
if (read(fd, string, 255)<= 0){
    printf("Can\'t read.\n");
    return(-1);
}
printf("Prog2 получила информацию из файла BUF.\n");
if(close(fd) < 0){
    printf("Can\'t close file BUF\n");
}
for(i=0; i<strlen(string); i++){
    if (string[i]!='\n') printf("%c", string[i]);
}
printf("-%d\n", getpid());
sleep(5);
sem_post(s1);
sem_post(s2);
sem_close(s1);
sem_close(s2);
sem_close(s3);
}
return 0;
}

```

Результаты тестирования с одним окном для первой программы (цикл повторяющийся всего 10 раз) и двумя для второй (в которой использован бесконечный цикл):

1) начало тестирования:

The image shows two terminal windows side-by-side. The left window, titled 'andrey@andrey-VirtualBox: ~', contains the following commands and output:

```

andrey@andrey-VirtualBox:~$ gcc program7_1.c -pthread -o pr
og1
andrey@andrey-VirtualBox:~$ gcc program7_2.c -pthread -o pr
og2
andrey@andrey-VirtualBox:~$ ./prog1
Prog1 отправила информацию в файл BUF.

```

The right window, also titled 'andrey@andrey-VirtualBox: ~', shows the prompt 'andrey@andrey-VirtualBox:~\$' and is currently empty.

2) В процессе тестирования (нижнее окно было запущено последним) сообщения от второй программы поочерёдно появлялись в своих окнах консоли.

