Московский авиационный институт
Национальный исследовательский университет

Операционные системы
Кафедра 806

Лабораторная работа №2

Студент: Минибаев Айдар
Группа: М8О-301Б-18
Преподаватель: Миронов Е.С.
Дата: .
Подпись: .

Москва 2020

1. Постановка задачи

Составить и отладить программу на языке Си, осуществляющую работу с процессами и взаимодействие между ними в одной из двух операционных систем. В результате работы программа (основной процесс) должен создать для решение задачи один или несколько дочерних процессов. Взаимодействие между процессами осуществляется через системные сигналы/события и/или каналы (pipe).

Необходимо обрабатывать системные ошибки, которые могут возникнуть в результате работы.

Написание собственного простого целочисленного калькулятора с операциями "+","-". В дочернем процессе должны происходить вычисления выражений. В родительском процессе ввод/вывод.

2. Метод решения

1. Используя системный вызов pipe создать 2 канала, по которым будут обмениваться данными два процесса.
2. Используя системный вызов fork создать дочерний процесс.
3. В родительском процессе считывать данные со стандартного потока в цикле, пока данные вводятся.
4. Когда в родительском процессе считались данные, необходимо записать их в канал с помощью системного вызова write. Затем родительский процесс считывает результат из второго канала. Но пока дочерний процесс не запишет данные во второй канал, родительский процесс будет заблокирован.
5. Пока родительский процесс на записал данные в канал. Дочерний процесс блокируется. И как только родительский процесс записал данные в первый канал дочерний процесс считывает их, производит вычисления и записывает результат во второй канал.
6. Как только дочерний процесс запишет результат вычислений во второй канал родительский процесс получит результат из второго канала и выведет их в стандартный поток. И затем снова будет ждать ввода со стандартного потока.

3. Программная реализация

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

void write_in_pipe(int *lhs, int *rhs, char *op, int fd) {
   write(fd, lhs, sizeof(int));
   write(fd, op, sizeof(char));
   write(fd, rhs, sizeof(int));
}

int main() {
   int fd1[2], fd2[2];
```

```
    int parent;

    if(pipe(fd1) == -1 || pipe(fd2) == -1) {
        printf("Can\'t create pipe\n");
        exit(0);
    }

    parent = fork();
    if(parent == -1) {
        printf("Can\'t fork child\n");
        exit(0);

    } else if(parent > 0) { // parent process
        close(fd1[0]);
        close(fd2[1]);
        int res, lhs, rhs;
        char op;
        while(scanf("%d %c %d", &lhs, &op, &rhs) == 3) {
            if(op != '+' &&  op != '-') {
                printf("Unknow operation\n");
                continue;
            }
            write_in_pipe(&lhs, &rhs, &op, fd1[1]);
            read(fd2[0], &res, sizeof(int));
            printf("%d\n", res);
        }
        close(fd1[1]);
        close(fd2[0]);

    } else { // child process
        close(fd1[1]);
        close(fd2[0]);
        int res, lhs, rhs;
        char op;
        while(read(fd1[0], &lhs, sizeof(int))) {
            read(fd1[0], &op, sizeof(char));
            read(fd1[0], &rhs, sizeof(int));
            res = (op == '+')
                ? lhs + rhs
                : lhs - rhs;
            write(fd2[1], &res, sizeof(int));
        }
        close(fd1[0]);
        close(fd2[1]);
    }
    return 0;
}
```

4. Тестирование

```
$ make
gcc lab2.c -o lab2.out -Wall -Wextra -Werror -pedantic
$ ./lab2.out
-1 + 1
```

```
0
1234 - 234
1000
10000000 +
2345678
12345678
1024 + 1024
2048
```

5. Выводы

Межпроцессорное взаимодействие можно осуществлять с помощью канала. В СИ канал создается с помощью системного вызова pipe. На мой взгляд, такой способ общения процессов очень удобен, так как при данном подходе не приходится сталкиваться с гонками, так как при использовании блокирующих системных вызовов read и write процессы блокируются, если им нечего считывать или буфер для записи полный. Так же одним из плюсов такого подхода к межпроцессорному взаимодействию является то, что каналом могут пользоваться только родственные процессы, так как канал находится в пределах ядра.