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

Лабораторная работа №6
По дисциплине: «ОСиСП»

Выполнил:
Студент 2 курса
Группы ПО-3
Кабачук Д.С.
Проверила:
Давидюк Ю. И.

Брест 2020

Средства межпроцессного взаимодействия

Вариант 11

Цель работы:

Изучить работу с средствами межпроцессного взаимодействия в Linux.

Задание:

Ознакомиться с руководством, теоретическими сведениями и лекционным материалом по использованию и функционированию средств взаимодействия.

Написать программу, которая порождает дочерний процесс, и общается с ним через средства взаимодействия согласно варианту, передавая и получая информацию согласно варианту. Передачу и получение информации каждым из процессов сопровождать выводом на экран информации типа "процесс такой-то передал/получил такую-то информацию". Сообщение вводит пользователь через терминал. Дочерние процессы начинают операции после получения сигнала SIGUSR1 от родительского процесса.

После отработки дочерний процесс должен возвращать результат родительскому процессу!

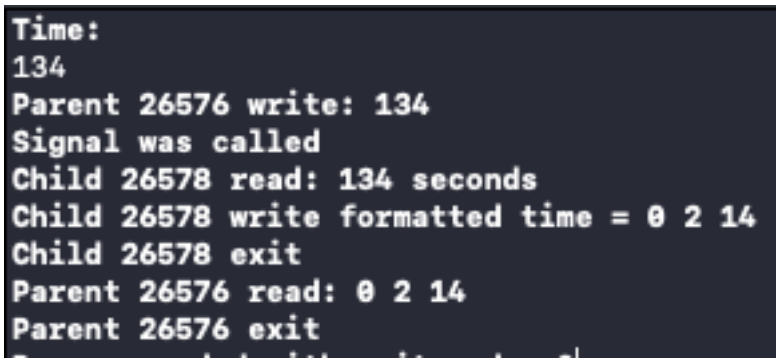
11	Программные каналы	Родитель передает время в секундах, потомок возвращает форматированное значение времени.

Код:

```
#include <unistd.h>
#include <string.h>
#include <stdio.h>
#include <sys/types.h>
#include <stdlib.h>
#include <signal.h>

int fd1[2], fd12[2];

void signalCalled() {
    printf("Signal was called\n");
    char time[100];
    int seconds;

    close(fd1[1]);

    read(fd1[0], &seconds, sizeof(seconds));
    printf("Child %d read: %d seconds\n", getpid(), seconds);

    close(fd1[0]);
    close(fd12[0]);

    int h = seconds / 3600;
    int m = (seconds / 60) % 60;
    int s = seconds % 60;
```

```
    sprintf(time, "%d %d %d", h, m, s);

    printf("Child %d write formatted time = %s\n", getpid(), time);
    write(fd12[1], &time, sizeof(time));

    close(fd12[1]);

    printf("Child %d exit\n", getpid());
}

int main() {
    int res;
    int seconds;

    if (pipe(fd1) < 0 || pipe(fd12) < 0) {
        printf("Can't create pipe\n");
        return -1;
    }

    res = fork();
    if (res < 0) {
        printf("Can't fork child\n");
        return -1;
    }
    else if (res > 0) {
        char num2[100];
        printf("Time: \n");
        scanf("%d", &seconds);

        close(fd1[0]);
        printf("Parent %d write: ", getpid());

        printf("%d\n", seconds);
        write(fd1[1], &seconds, sizeof(seconds));
        close(fd1[1]);

        kill(res, SIGUSR1);
        wait(NULL);

        close(fd12[1]);
        read(fd12[0], num2, strlen(num2));
        close(fd12[0]);

        printf("Parent %d read: ", getpid());
        printf("%s\n", num2);
        printf("Parent %d exit\n", getpid());
    } else {
        (void)signal(SIGUSR1, signalCalled);
        pause();
    }
}
```

Тестирование:



Вывод: изучил работу с средствами межпроцессного взаимодействия в ОС Linux.