

**实 验 报 告**

**（ 2019 / 2020 学年 第 1 学期）**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 课程名称 | 操作系统B | | | | | |
| 实验名称 | 进程通信 | | | | | |
| 实验时间 | 2020 | 年 | 04 | 月 | 8 | 日 |
| 指导单位 | 计算机学院计算机科学与技术系 | | | | | |
| 指导教师 | 段卫华 | | | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 学生姓名 | 刘炎 | 班级学号 | B8031801 |
| 学院(系) | 计软网安学院 | 专 业 | 软件工程嵌入式培养 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **实验名称** | 进程通信 | | | **指导教师** | 段卫华 |
| **实验类型** | 验证 | **实验学时** | 2 | **实验时间** | 2020.04.8 |
| 1. **实验目的和要求**   实验目的  1. 熟练使用 Linux 的 C 语言开发环境  2. 掌握 Linux 操作系统下的并发进程间同步  3. 掌握 Linux 操作系统下的进程间通信  实验内容  1. 了解常见的消息通信方式:信号机制、消息队列机制、共享内存机制和管道机制。  2. 掌握消息队列机制中常用的系统调用有:建立一个消息队列 msgget;向消息队列发送  消息 msgsnd;从消息队列接收消息 msgrcv;取或送消息队列控制信息 msgctl。  3. 掌握管道机制中常用的系统调用:建立管道文件 pipe;写操作 write,读操作 read。  4. 了解信号机制中常用的系统调用。  5. 了解共享内存机制中常用的系统调用。 | | | | | |
| 二、**实验环境(实验设备)**  Ubuntu系统 | | | | | |
| **三、实验原理及内容**  **1. 高级通信之共享内存**  **（1）机制：**  **进程间通过对共享内存中数据的读写，来进通信。系统中通信速度最高的一种机制。**  **（2）函数调用：**  **①shmget（）创建、获得一个共享存储区。**  **格式：shmid = shmget(key, size, flag)**  **key:共享存储区的名字, size: 大小单位字节, flag:用户设置的标准**  **返回一个共享存储区的标识符id。**  **②shmat（）共享存储区附接到进程虚拟电子空间.**  **当把shmget（）返回的id值给它，实现了附接后，进程就可以像用自己的空间一样去使用这个共享存储区。**  **返回共享存储区所附接到的进程的虚地址。**  **③ shmdt（）用完共享存储区后，把它从进程虚地址中断开。**  **调用成功返回0，否则-1.**  **④shmctl（）共享存储区进行控制，对其状态进行修改。**  **(3）发送进程**  **代码：**  **/\*共享内存的发送程序 sndshm.c,先运行发送程序,再运行接收程序\*/**  **#include <stdio.h>**  **#include <stdlib.h>**  **#include <string.h>**  **#include<unistd.h>**  **#include <sys/types.h>**  **#include <sys/shm.h>**  **int main()**  **{**  **int shmid;**  **char \*viraddr;**  **char buffer[BUFSIZ];**  **/\*共享内存的内部标识\*/**  **/\*定义附接到共享内存的虚拟地址\*/**  **/\*创建共享内存\*/**  **shmid = shmget(1234, BUFSIZ, 0666|IPC\_CREAT);**  **/\*附接到进程的虚拟地址空间\*/**  **viraddr = (char \*)shmat(shmid, 0, 0);**  **/\*循环输入信息,直到输入 end 结束\*/**  **while(1)**  **{**  **puts("Enter some text:");**  **fgets(buffer, BUFSIZ, stdin);**  **strcat(viraddr, buffer);**  **/\*追加到共享内存\*/**  **if(strncmp(buffer, "end", 3) ==0)**  **break;**  **}**  **shmdt(viraddr);**  **/\*断开链接\*/**  **return 0;**  **}**  **(2)接收进程**  **代码:**  **/\*共享内存的接收进程程序 rcvshm.c\*/**  **#include <stdio.h>**  **#include <stdlib.h>**  **#include <string.h>**  **#include <unistd.h>**  **#include <sys/types.h>**  **#include <sys/shm.h>**  **int main()**  **{**  **int shmid;**  **char \*viraddr;**  **/\*获取共享内存\*/**  **shmid = shmget(1234, BUFSIZ, 0666|IPC\_CREAT);**  **/\*附接到进程的虚拟地址空间\*/**  **viraddr = (char \*)shmat(shmid, 0, 0);**  **/\*打印信息内容\*/**  **printf("your message is :\n %s", viraddr);**  **/\*断开链接\*/**  **shmdt(viraddr);**  **/\*撤销共享内存\*/**  **shmctl(shmid,**  **IPC\_RMID, 0);**  **return 0;**  **}**  **结果：** | | | | | |
| **2.高级通信之消息队列的通信方式**  **（1）机制：进程收到多个消息时，可以排成消息队列。发送进程通过发送的高级原语向指定消息队列里发送信息。接收进程通过指针找到队列，读取相关消息。**  **（2）函数调用**  **①msgget（）创建一个消息，获得消息的描述符。**  **②msgsnd（）向指定消息队列里发送信息，并将消息链接到队列尾部。**  **③msgrcv（）从指定消息队列中接收消息。**  **④读取消息队列的消息并进行修改。**  **（3）发送进程**  **代码:**  **/\*发送消息进程 sndfile.c\*/**  **#include <stdio.h>**  **#include <stdlib.h>**  **#include <string.h>**  **#include <unistd.h>**  **#include <sys/types.h>**  **#include <sys/msg.h>**  **#define MAXMSG 512**  **/\*定义消息长度\*/**  **/\*定义消息缓冲区队列中的数据结构\*/**  **struct my\_msg**  **{**  **long int my\_msg\_type;**  **char some\_text[MAXMSG];**  **}msg;**  **int main()**  **{**  **int msgid;**  **/\*定义消息缓冲区内部标识\*/**  **char buffer[BUFSIZ];**  **/\*用户缓冲区\*/**  **/\*创建消息队列\*/**  **msgid = msgget(1234, 0666|IPC\_CREAT);**  **/\*循环向消息队列中发送消息,直到输入 end 结束\*/**  **while(1)**  **{**  **puts("Enter some text:");**  **fgets(buffer, BUFSIZ, stdin);**  **msg.my\_msg\_type = 1;**  **strcpy(msg.some\_text, buffer);**  **msgsnd(msgid, &msg, MAXMSG, 0);**  **/\*发送消息到缓冲队列中\*/**  **if (strncmp(msg.some\_text, "end", 3) == 0)**  **break;**  **}**  **return 0;**  **}**  **（4）接收进程**  **代码:**  **/\*消息队列机制的接收程序 rcvfile.c\*/**  **#include <stdio.h>**  **#include <stdlib.h>**  **#include <string.h>**  **#include <unistd.h>**  **#include <sys/types.h>**  **#include <sys/msg.h>**  **#define MAXMSG 512**  **struct my\_msg**  **{**  **long int my\_msg\_type;**  **char some\_text[MAXMSG];**  **}msg;**  **int main()**  **{**  **int msgid;**  **long int msg\_to\_receive = 0;**  **msgid = msgget(1234, 0666|IPC\_CREAT);**  **/\*循环从消息队列中接收消息,读入 end 结束接收\*/**  **while (1)**  **{**  **msgrcv(msgid, &msg, BUFSIZ, msg\_to\_receive, 0);**  **printf("You wrote:%s", msg.some\_text);**  **if (strncmp(msg.some\_text, "end", 3) == 0)**  **break;**  **}**  **msgctl(msgid, IPC\_RMID, 0);**  **return 0;**  **}**  **结果：**    **3.高级通信之管道通信**  **（1）机制：管道是连接读写进程，并允许以生产者-消费者方式进行通信的共享文件（先进先出），称为pipe文件。即这种通信机制基于硬盘上保存的共享文件来实现的。** | | | | | |

**管道类型：有名管道，文件系统中长期存在，具有路径名的文件。系统调用mknod（）建立，其他进程可以通过路径名访问文件。无名管道，利用pipe（）建立起来的无名文件，只有调用pipe（）的进程及其子进程才能识别此文件标识符，再利用管道进行通信。**

**（2）**

**/\*管道文件 pipe.c\*/**

**#include <stdio.h>**

**#include <unistd.h>**

**int main()**

**{**

**int p1, fd[2];**

**char outpipe[50];**

**/\*定义读缓冲区\*/**

**char inpipe[50] = "This is a message from child!";**

**pipe(fd);**

**while ((p1 = fork()) == -1);**

**if (p1 == 0)**

**/\*子进程中写\*/**

**{**

**write(fd[1], inpipe, 50);**

**}**

**else**

**{**

**/\*父进程中读\*/**

**wait(0);**

**read(fd[0], outpipe, 50);**

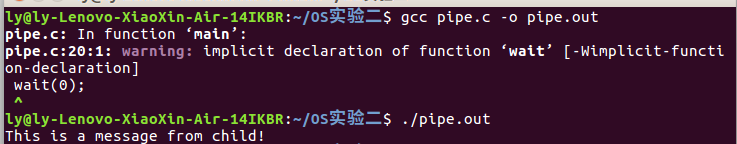
**printf("%s \n", outpipe);**

**}**

**return 0;**

**}**

**(3)结果**



**4. 低级通信之信号通信**

**(1）机制：进程间的通信以信号为单位，每个信号对应一个正整数的常量。**

**（2）第一个程序有信号处理机制**

**/\*signal.c\*/**

**#include <stdio.h>**

**#include <unistd.h>**

**#include <signal.h>**

**void int\_func(int sig);**

**int k;**

**/\*定义循环变量\*/**

**void int\_func(int sig)**

**{**

**k = 0;**

**}**

**int main()**

**{**

**signal(SIGINT, int\_func);**

**k = 1;**

**while (k == 1)**

**{**

**printf("Hello, world!\n");**

**}**

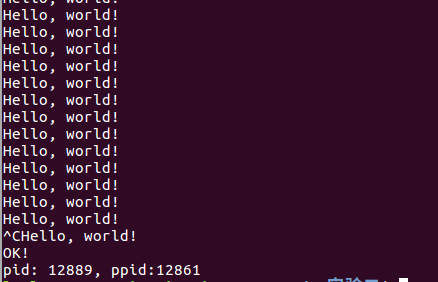
**printf("OK!\n");**

**printf("pid: %d,**

**ppid: %d \n", getpid(), getppid());**

**}**

**（3）结果:当按下ctrl+c，跳出while循环，结束进程。**



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **四、实验小结**（包括问题和解决方法、心得体会、意见与建议等）  这是操作系统B的第二个实验，通过实验熟悉了 Linux 的 C 语言开发环境， Linux 操作系统下的并发进程间同步， Linux 操作系统下的进程间通信。  1. 了解常见的消息通信方式:信号机制、消息队列机制、共享内存机制和管道机制。  2.了解了许多不同通信时的系统的函数调用.如消息队列机制的msgget（）,共享内存机制的shmget(),管道机制的pipe(),信号机制的signal()等。  问题及其解决：  1.问题：共享内存机制信息通信时，代码先执行了rcvshm.out，由于共享内存区中还没有通过sndshm.c写入内容，所以无法成功读取信息。  解决：先执行 sndshm.out,再执行 rcvshm.out。  意见与建议： | | | | | |
| **五、指导教师评语** | | | | | |
| **成 绩** |  | **批阅人** |  | **日 期** | 2019.09.19 |