实验6 进程通信（2）共享内存和消息队列通信

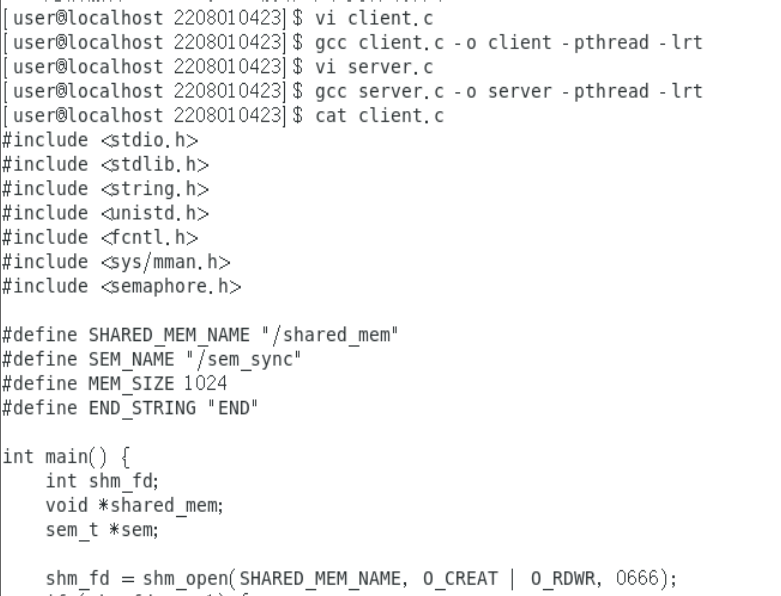
实验目的

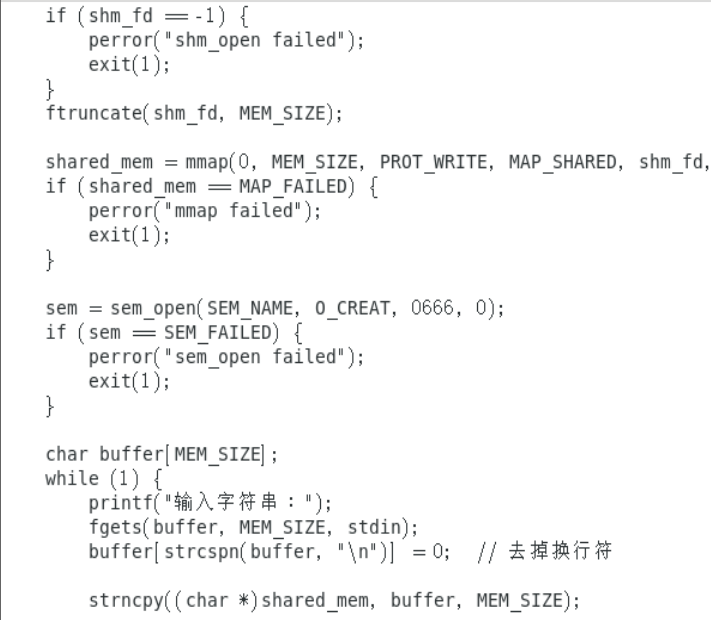
1. 能够利用共享内存通信原理设计程序
2. 掌握消息传送的原理
3. 能够利用消息队列原理设计进程通信程序
4. 掌握进程通信的不同方式之间的区别、特点、适用情况

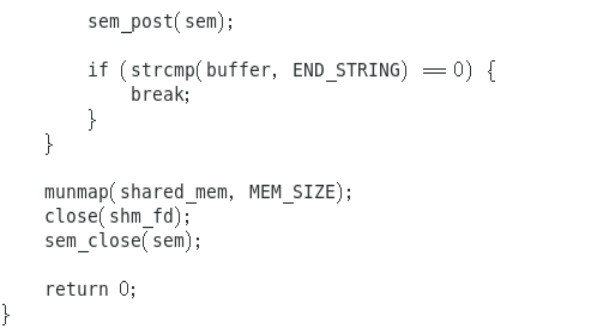
实验内容

1.设计两个程序，Client接受键盘输入的字符串，写入共享存储区；Server读出共享内存的内容输出。两个进程可以连续通信，设定结束字符串，两个进程的同步通过sleep或控制进程执行顺序实现（也可以考虑进程信号量）。

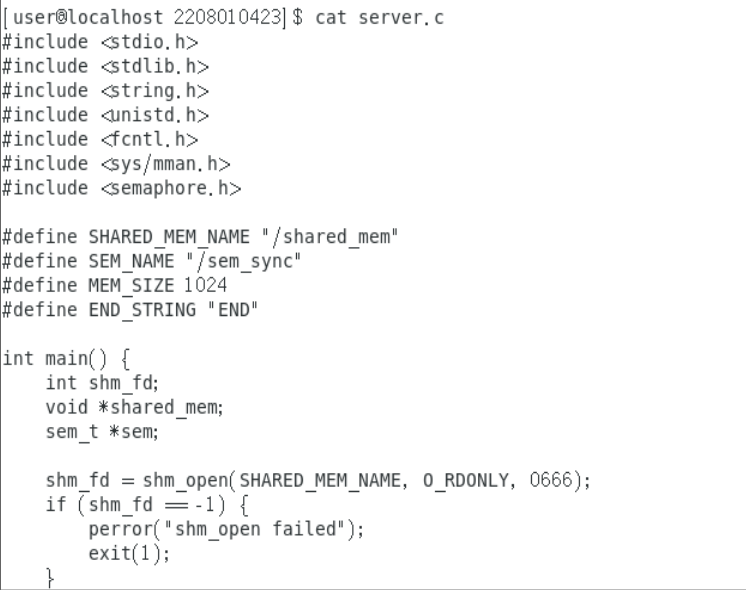
client.c 代码

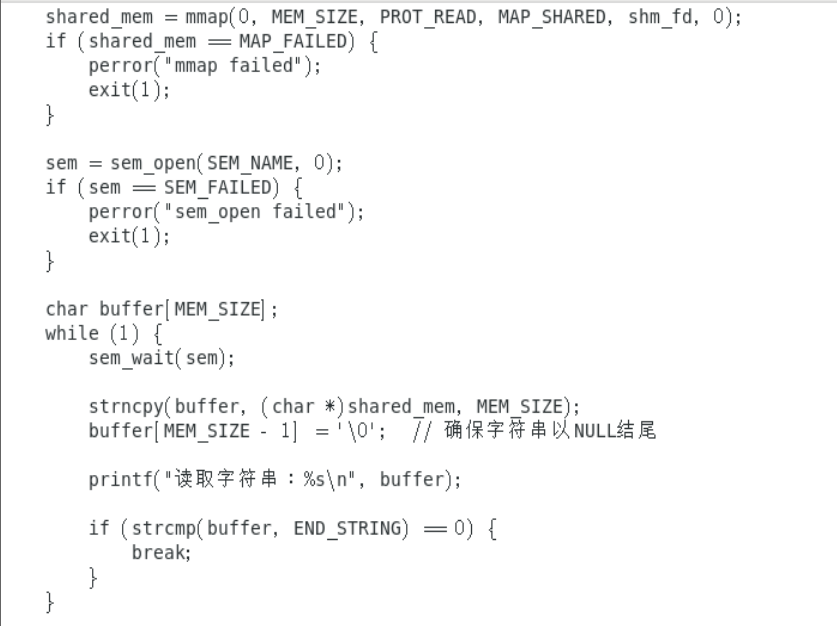


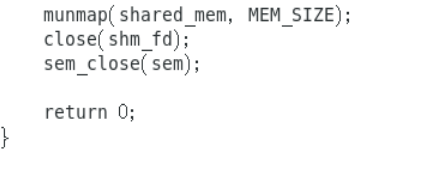




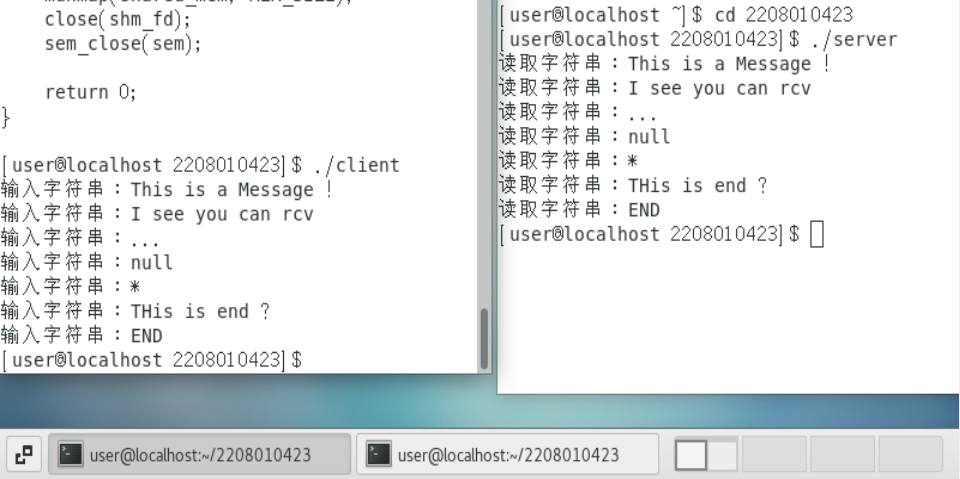
server.c 代码







运行截图



2.运行以下程序，分析程序的运行结果。

（1）send.c

#include <sys/types.h>

#include <sys/ipc.h>

#include <sys/msg.h>

#include <unistd.h>

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

struct msgbuf

{

long mtype;

char mtext[1000];

};

int main()

{

int qid;

key\_t key=75;

int len;

struct msgbuf msg;

qid=msgget(key,IPC\_CREAT|0666);

printf("Input some text to send:\n");

fgets((&msg)->mtext,512,stdin);

msg.mtype=getpid();

len=strlen(msg.mtext);

msgsnd(qid,&msg,len,0);

printf("message has been sent to message queue\n");

return 0;

}

（2） receive.c

#include <sys/types.h>

#include <sys/ipc.h>

#include <sys/msg.h>

#include <unistd.h>

#include <stdlib.h>

#include <stdio.h>

struct msgbuf

{

long mtype;

char mtext[1000];

};

int main()

{

int qid;

key\_t key=75;

int len;

struct msgbuf msg;

qid=msgget(key,0666);

msgrcv(qid,&msg,512,0,0);

printf("read from queue：%s\n",msg.mtext);

printf("The sending process id is:%ld\n",msg.mtype);

//msgctl(qid,IPC\_RMID,NULL)；

printf("message queue has been deleted\n");

return 0;

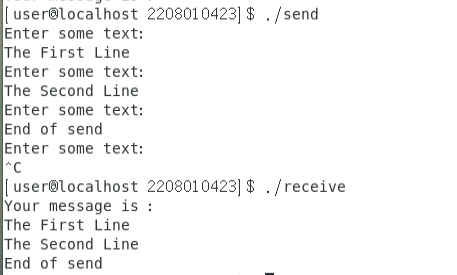
}

步骤：分别在前后台运行send和receive程序，观察两个进程的同步情况。

或者打开两个terminal，一个运行send，另一个运行receive，观察进程的同步情况。

要求：

1. 程序运行截图，分析程序实现的功能？



**实现功能 :**

**send.c 创建一个消息队列，如果消息队列已经存在则用现有消息队列。用户通过键盘输入将消息发到消息队列。**

**receive.c 打开消息队列，msgrcv 从队列中读取信息，并输出。**

1. 分析消息队列的如何控制消息发送接收的同步

**msgrcv接受消息会在队列为空时阻塞，直到有消息可以接受。**

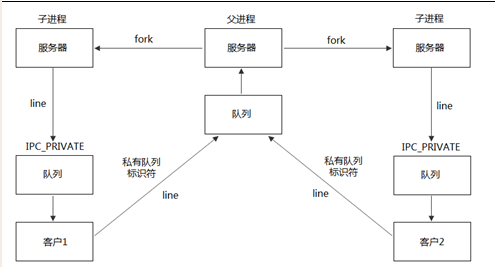
**msgsnd 发送消息会在队列满时阻塞，直到有空间可以存放消息。**

3.编写两个程序：client实现将键盘接收的一行字符串发送给server，输出接收到的应答消息；server接收client消息输出，发送”server received!”给client后结束。使用系统调用msgget( )，msgsnd( )，msgrcv( )，及msgctl( )编制程序。

（1）至少实现client和server的一次双向通信。

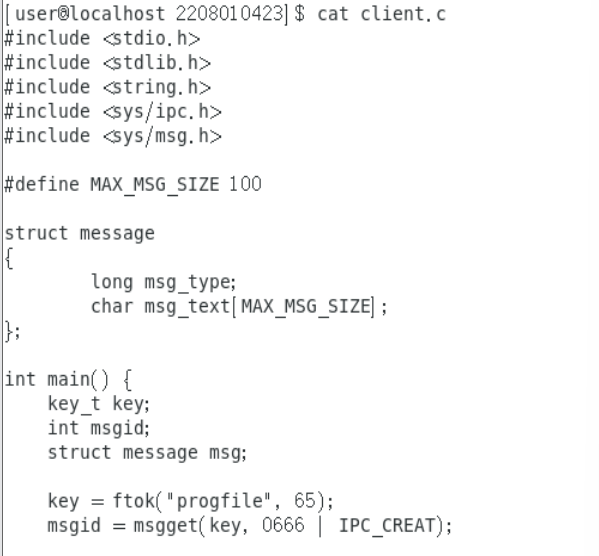
（2）选做的功能：将server程序设置为能够并发处理多个客户的程序（参考下图的程序结构），服务器无限循环，CTRL+C结束，设计捕捉信号，信号处理中删除消息队列，服务器收到客户机的消息后发给不同客户机不同的应答消息（内容自定）。

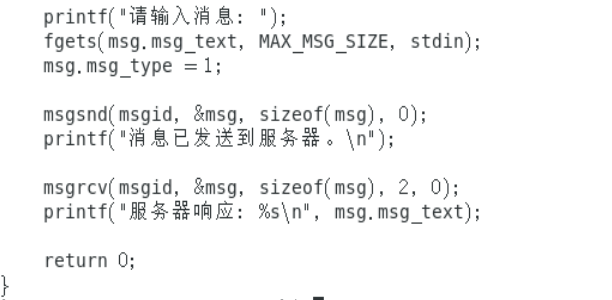
要求：程序代码及运行结果截图。



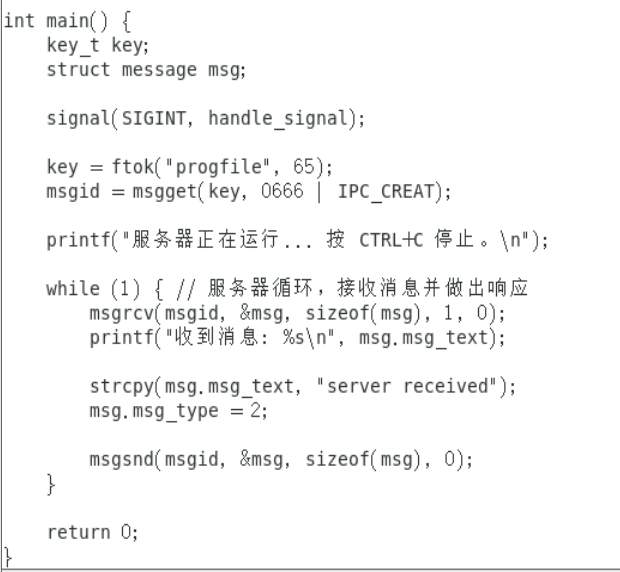
1. 代码及结果截图

client.c

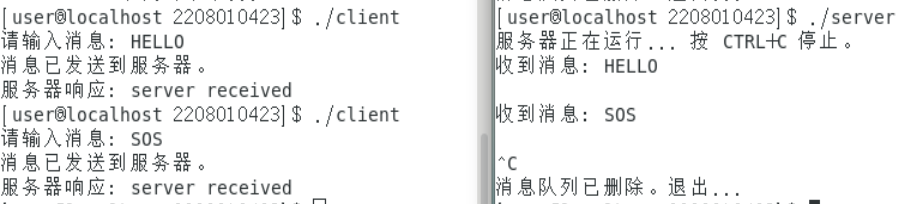




server.c

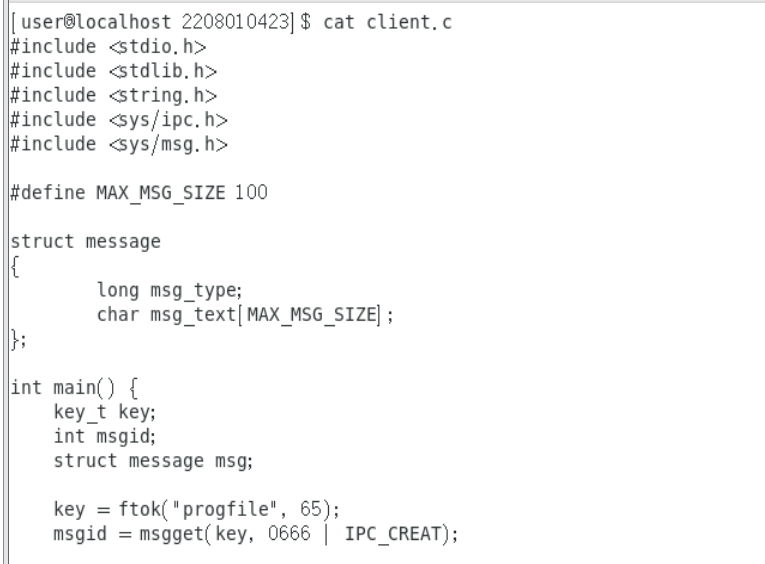


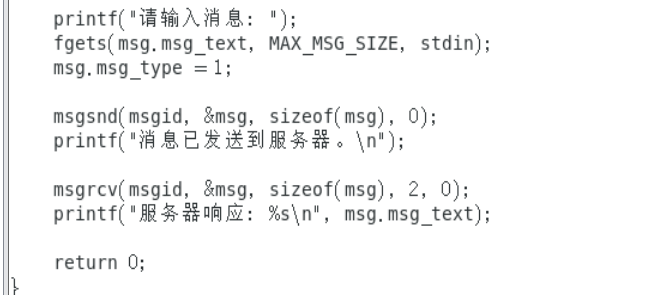
运行结果截图



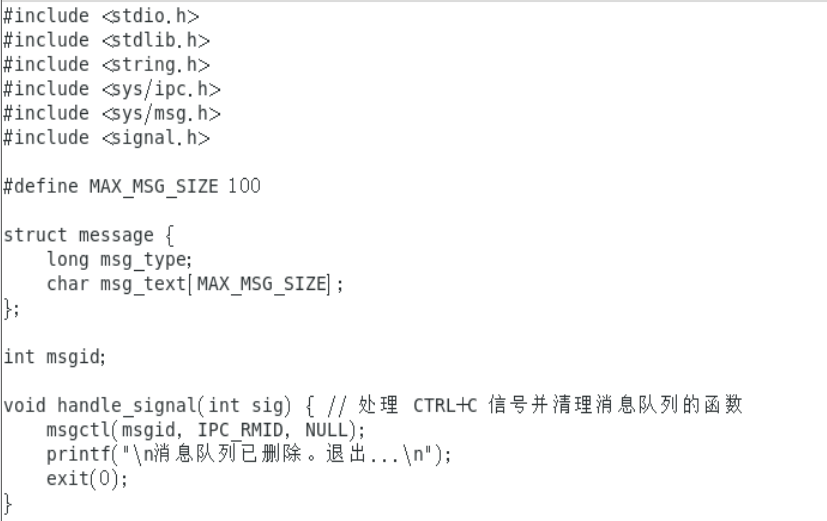
1. 代码及结果截图

client



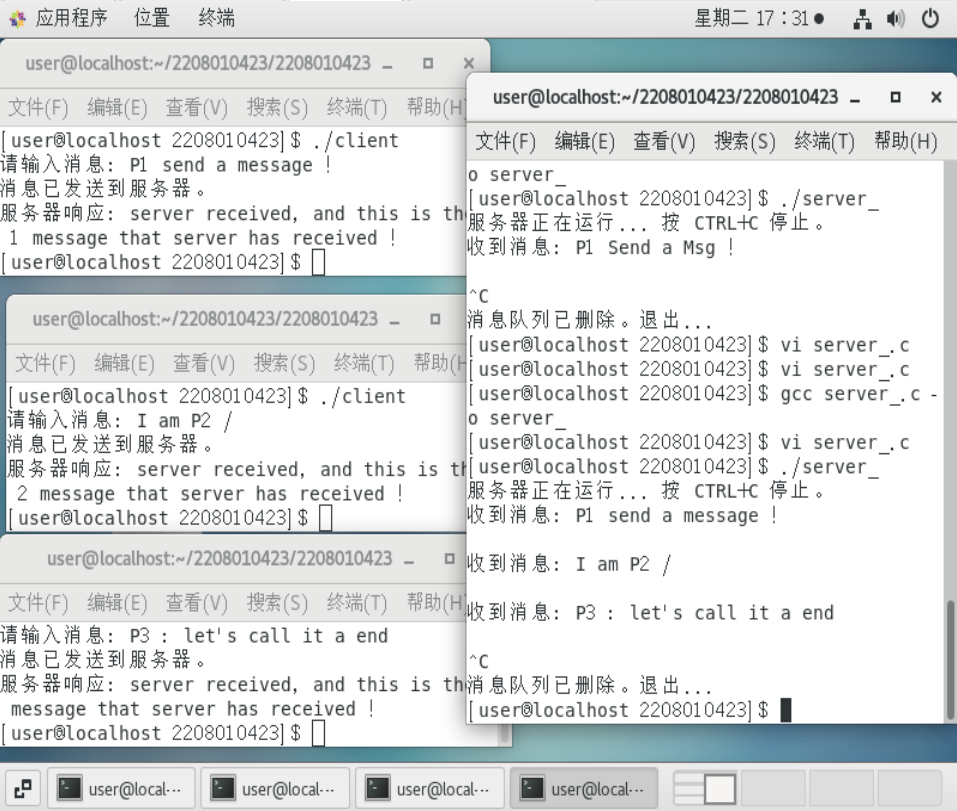


server









1. 总结无名管道、消息队列、共享内存的通信特点和适用场合。

无名管道的特点是只能在有亲缘关系的进程间使用，如父子进程之间，通信时数据以字节流的方式传输，具有单向性，适合在简单的父子进程数据交换中使用。

消息队列可以在任意进程之间通信，数据传输以消息为单位，不存在字节流的特性，因此更易于管理和控制。消息队列适用于需要顺序控制和独立于进程间结构的应用场合。

共享内存提供最快速的进程间通信方式，因为数据直接写入和读取内存，不需要经过操作系统的中转，适合在需要高效、快速的数据共享的场合，如多进程密集计算或大量数据交互的应用。需要同步机制来避免数据一致性问题。