实验四　Linux 下多线程间通信

一. 实验目的

学习和掌握 linux 下的多线程通信的基本原理和基本编程方法

二. 实验平台

基于 ATOM N270，NORCO(华北工控) EMB-3850 嵌入式开发板

Linux redhat kernel-2.4 嵌入式操作系统

三. 实验内容

编写 linux 下多线程间通信的程序

四. 参考资料

1、Linux 基础教程.吴学毅.北京交通大学出版社 出版日期：2005 年 9 月

2、红帽企业 Linux 4 X86-64 体系安装指南

3、Red Hat Enterprise Linux 4 System Administration Guide

4、Linux 下多线程技术分析及应用.金惠芳，陶利民，张基温.《计算机系统应用》.2003 年 09 期

五. 实验原理

线程间无需特别的手段进行通信，因为线程间可以共享数据结构，也就是一个全局变量可以被两个线程同时使用。不过要注意的是线程间需要做好同步，一般用 mutex。pthread_mutex_lock 声明开始用互斥锁上锁，此后的代码直至调用 pthread_mutex_unlock 为止，均被上锁，即同一时间只能被一个线程调用执行。当一个线程执行到 pthread_mutex_lock 处时，如果该锁此时被另一个线程使用，那此线程被阻塞，即程序将等待到另一个线程释放此互斥锁。

当两个或更多线程需要同时访问一个共享资源时，系统需要使用同步机制来确保一次只有一个线程使用该资源。**Mutex** 是同步基元，它只向一个线程授予对共享资源的独占访问权。如果一个线程获取了互斥体，则要获取该互斥体的第二个线程将被挂起，直到第一个线程释放该互斥体。

六. 实验步骤

a) 登录进入 linux 操作系统，打开命令窗口进入 myprojects 文件夹，新建一文件夹取名为 thread_comm 令如下：

```
cd /myprojects
mkdir thread_comm
```

b) 进入 thread_comm 文件夹，新建文件 thread_comm.c，并在其中输入以下代码：

```
cd thread_comm          //进入文件夹
vi thread_comm.c     //新建文件 thread_comm.c
//*************thread_comm.c**************//
//*Author:          manchen
//*Date:            2009-8-16
//****************************************//
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
pthread_mutex_t mutex=PTHREAD_MUTEX_INITIALIZER;
pthread_cond_t cond=PTHREAD_COND_INITIALIZER;
void *thread1(void *);
void *thread2(void *);
int i=1;
int main()
{
 pthread_t t_a;
 pthread_t t_b;
 pthread_create(&t_a,NULL,thread1,(void *)NULL);
 pthread_create(&t_b,NULL,thread2,(void *)NULL);
 pthread_join(t_b,NULL);
 pthread_mutex_destroy(&mutex);
 pthread_cond_destroy(&cond);
 exit(0);
}
void *thread1(void *junk)
{
 for(i=1;i<=4;i++)
  {
   printf("Enter thread1\n");
   pthread_mutex_lock(&mutex);
   if(i%3==0)
```

```
        {
          pthread_cond_signal(&cond);printf("thread2 is unlocked
                                by thread1\n");
        }
       else
         printf("thread1:%d\n",i);
         pthread_mutex_unlock(&mutex);
         printf("thread1 is unlocked by itself\n");
         sleep(1);
       }
    }
    void *thread2(void *junk)
    {
     while(i<4)
      {
       printf("Enter thread2\n");
       pthread_mutex_lock(&mutex);
       if(i%3!=0)
        {
          pthread_cond_wait(&cond,&mutex);
          printf("thread2 is waiting for work...\n");
        }
       printf("thread2:%d\n",i);
       pthread_mutex_unlock(&mutex);
       printf("thread2 is unlocked by itself\n");
       sleep(1);
      }
    }
```

c) 保存文件退出文件编辑模式回到命令行模式，对所编写程序进行编译，编译通过后执行所生成的文件 thread_comm，观看执行结果。命令如下：

gcc –lpthread –o thread_comm thread_comm.c

./thread_comm

执行结果如下：

Enter thread1

thread1:1

thread1 is unlocked by itself

Enter thread2

Enter thread1

thread1:2

thread1 is unlocked by itself

Enter thread1

thread2 is unlocked by itself

thread1 is unlocked by itself

thread2 is waiting for work…

thread2:3

thread2 is unlocked by itself

Enter thread1

thread1:4

thread1 is unlocked by itself