

转 pthread_create函数详解

2017年05月10日 10:05:04 阅读数：774

函数简介

编辑

头文件

1	#include<pthread.h>
---	---------------------

函数声明

1	int pthread_create(pthread_t *tidp,const pthread_attr_t *attr,
2	(void*)(*start_rtn)(void*),void *arg);

编译链接参数

-lpthread

返回值

若线程创建成功，则返回0。若线程创建失败，则返回出错编号，并且*thread中的内容是未定义的。
返回成功时，由tidp指向的内存单元被设置为新创建线程的线程ID。attr参数用于指定各种不同的线程属性。新创建的线程从start_rtn函数的地址开始运行，该函数只有一个万能指针参数arg，如果需要向start_rtn函数传递的参数不止一个，那么需要把这些参数放到一个结构中，然后把这个结构的地址作为arg的参数传入。
linux下用C语言开发多线程程序，Linux系统下的多线程遵循POSIX线程接口，称为pthread。
由 restrict 修饰的指针是最初唯一对指针所指向的对象进行存取的方法，仅当第二个指针基于第一个时，才能对对象进行存取。对对象的存取都限定于基于由 restrict 修饰的指针表达式中。由 restrict 修饰的指针主要用于函数形参，或指向由 malloc() 分配的内存空间。restrict 数据类型不改变程序的语义。编译器能通过作出 restrict 修饰的指针是存取对象的唯一方法的假设，更好地优化某些类型的例程。

参数

- 第一个参数为指向线程标识符的指针。
- 第二个参数用来设置线程属性。
- 第三个参数是线程运行函数的起始地址。
- 最后一个参数是运行函数的参数。

注意事项

因为pthread并非Linux系统的默认库，而是POSIX线程库。在Linux中将其作为一个库来使用，因此加上 -lpthread (或-pthread) 以显式链接该库。函数在执行错误时的错误信息将作为返回值返回，并不修改系统全局变量errno，当然也无法使用perror()打印错误信息。

示例

编辑

输出线程标识符

1	
---	--

```
2  #include <stdio.h>
3  #include <stdlib.h>
4  #include <pthread.h>
5  #include <unistd.h>
6  #include <string.h>
7
8  void printids(const char *s)
9  {
10     pid_t pid;
11     pthread_t tid;
12     pid = getpid();
13     tid = pthread_self();
14     printf("%s pid %u tid %u (0x%x)\n", s, (unsigned int) pid,
15           (unsigned int) tid, (unsigned int) tid);
16 }
17
18 void *thr_fn(void *arg)
19 {
20     printids("new thread: ");
21     return NULL;
22 }
23
24 int main(void)
25 {
26     int err;
27     pthread_t ntid;
28     err = pthread_create(&ntid, NULL, thr_fn, NULL);
29     if (err != 0)
30         printf("can't create thread: %s\n", strerror(err));
31     printids("main thread:");
32     pthread_join(ntid, NULL);
33     return EXIT_SUCCESS;
34 }
```

```
$ gcc main.c -o main -std=c99 -pthread
```

```
$ ./main
```

```
main thread: pid 13073 tid 3077572816 (0xb77008d0)
```

```
new thread: pid 13073 tid 3077569392 (0xb76ffb70)
```

简单的线程程序

```
1
2  #include <stdio.h>
3  #include <stdlib.h>
4  #include <pthread.h>
5  #include <unistd.h>
6
7  #define    NUM_THREADS    8
8
9  void *PrintHello(void *args)
10 {
11     int thread_arg;
12     sleep(1);
13     thread_arg = (int)((int*)args);
14     printf("Hello from thread %d\n", thread_arg);
15     return NULL;
16 }
17
18 int main(void)
19 {
20     int rc,t;
21     pthread_t thread[NUM_THREADS];
22
23     for( t = 0; t < NUM_THREADS; t++)
```

```
25     {
26         printf("Creating thread %d\n", t);
27         rc = pthread_create(&thread[t], NULL, PrintHello, &t);
28         if (rc)
29         {
30             printf("ERROR; return code is %d\n", rc);
31             return EXIT_FAILURE;
32         }
33     }
34     sleep(5);
35     for( t = 0; t < NUM_THREADS; t++)
36         pthread_join(thread[t], NULL);
37     return EXIT_SUCCESS;
38 }
```

```
$ gcc thread_test.c -o thread_test -std=c99 -pthread
```

```
$ ./thread_test
```

```
Creating thread 0
Creating thread 1
Creating thread 2
Creating thread 3
Creating thread 4
Creating thread 5
Creating thread 6
Creating thread 7
Hello from thread 8
Hello from thread 8
Hello from thread 8
Hello from thread 8
Hello from thread 8
Hello from thread 8
Hello from thread 8
Hello from thread 8
Hello from thread 8
```

pthread_create是UNIX环境创建线程函数

头文件

```
#include<pthread.h>
```

函数声明

```
int pthread_create(pthread_t*restrict tidp,const pthread_attr_t*restrict_attr,void* (*start_rtn)(void*),void *restrict arg);
```

返回值

若成功则返回0，否则返回出错编号

返回成功时，由tidp指向的内存单元被设置为新创建线程的线程ID。attr参数用于制定各种不同的线程属性。新创建的线程从start_rtn函数的地址开始运行，该函数只有一个万能指针参数arg，如果需要向start_rtn函数传递的参数不止一个，那么需要把这些参数放到一个结构中，然后把这个结构的地址作为arg的参数传入。

linux下用C开发多线程程序，Linux系统下的多线程遵循POSIX线程接口，称为pthread。

由 restrict 修饰的指针是最初唯一对指针所指向的对象进行存取的方法，仅当第二个指针基于第一个时，才能对对象进行存取。对对象的存取都限定于基于由 restrict 修饰的指针表达式中。由 restrict 修饰的指针主要用于函数形参，或指向由 malloc() 分配的内存空间。restrict 数据类型不改变程序的语义。编译器能通过作出 restrict 修饰的指针是存取对象的唯一方法的假设，更好地优化某些类型的例程。

参数

第一个参数为指向线程标识符的指针。

第二个参数用来设置线程属性。

第三个参数是线程运行函数的起始地址。

最后一个参数是运行函数的参数。

另外，在编译时注意加上-lpthread参数，以调用静态链接库。因为pthread并非Linux系统的默认库

示例

打印线程 IDs

```

#include <pthread.h>
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <string.h>
pthread_t ntid;
void printids(const char *s)
{
    pid_t pid;
    pthread_t tid;
    pid = getpid();
    tid = pthread_self();
    printf("%s pid %u tid %u (0x%x)\n", s,
        (unsigned int)pid, (unsigned int)tid, (unsigned int)tid);
} void *thr_fn(void *arg)
{
    printids("new thread: ");
    return((void *)0);
}
int main(void)
{
    int err;
    err = pthread_create(&ntid, NULL, thr_fn, NULL);
    if (err != 0)
        printf("can't create thread: %s\n", strerror(err));
    printids("main thread:");
    sleep(1);
    exit(0);
}
$ gcc main.c -lpthread
$ ./a.out

```

向线程函数传递参数详解:

向线程函数传递参数分为两种:

- (1) 线程函数只有一个参数的情况: 直接定义一个变量通过应用传给线程函数。

例子:

```

#include <iostream>
#include <pthread.h>
using namespace std;
pthread_t thread;
void fn(void *arg)
{
    int i = *(int *)arg;
    cout<<"i = "<<i<<endl;
    return ((void *)0);
}
int main()
{
    int err1;
    int i=10;
    err1 = pthread_create(&thread, NULL, fn, &i);
    pthread_join(thread, NULL);
}

```

- 2、线程函数有多个参数的情况: 这种情况就必须申明一个结构体来包含所有的参数, 然后在传入线程函数, 具体如下:

例子:

首先定义一个结构体：

```
struct parameter
```

```
{
    int size,
    int count;
    . . . . .
    . . .
};
```

然后在main函数将这个结构体指针，作为void
*形参的实际参数传递

```
struct parameter arg;
```

通过如下的方式来调用函数：

```
pthread_create(&ntid, NULL, fn,& (arg));
```

函数中需要定义一个parameter类型的结构指针来引用这个参数

```
void fn(void *arg)
{
    int i = *(int *)arg;
    cout<<"i = "<<i<<endl;
    return ((void *)0);
}

void thr_fn(void *arg)
{
    struct parameter *pstru;
    pstru = ( struct parameter *) arg;
    然后在这个函数中就可以使用指针来使用相应的变量的值了。
}
```

linux下用C开发多线程程序，Linux系统下的多线程遵循POSIX线程接口，称为pthread。

```
#include <pthread.h>
```

```
int pthread_create(pthread_t *restrict tidp,
const pthread_attr_t *restrict attr,
void *(*start_rtn)(void),
void *restrict arg);
```

Returns: 0 if OK, error number on failure

C99 中新增加了 restrict 修饰的指针：由 restrict 修饰的指针是最初唯一对指针所指向的对象进行存取的方法，仅当第二个指针基于第一个时，才能对对象进行存取。对对象的存取都限定于基于由 restrict 修饰的指针表达式中。由 restrict 修饰的指针主要用于函数形参，或指向由 malloc() 分配的内存空间。restrict 数据类型不改变程序的语义。编译器能通过作出 restrict 修饰的指针是存取对象的唯一方法的假设，更好地优化某些类型的例程。

第一个参数为指向线程标识符的指针。

第二个参数用来设置线程属性。

第三个参数是线程运行函数的起始地址。

最后一个参数是运行函数的参数。

下面这个程序中，我们的函数thr_fn不 需要参数，所以最后一个参数设为空指针。第二个参数我们也设为空指针，这样将生成默认属性的线程。当创建线程成功时，函数返回0，若不为0则说明创建线程 失败，常见的错误返回代码为EAGAIN和EINVAL。前者表示系统限制创建新的线程，例如线程数目过多了；后者表示第二个参数代表的线程属性值非法。创建线程成功后，新创建的线程则运行参数三和参数四确定的函数，原来的线程则继续运行下一行代码。

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

```
pthread_t ntid;

void printids(const char *s){
pid_t pid;
pthread_t tid;

pid = getpid();
tid = pthread_self();
printf("%s pid %u tid %u (0x%x)\n",s,(unsigned int)pid,(unsigned int)tid,(unsigned
int)tid);
}

void *thr_fn(void *arg){
printids("new thread:");
return ((void *)0);
}

int main(){
int err;

err = pthread_create(&ntid,NULL,thr_fn,NULL);
if(err != 0){
printf("can't create thread: %s\n",strerror(err));
return 1;
}

printids("main thread:");
sleep(1);
return 0;
}
```

把APUE2上的一个程序修改一下，然后编译。

结果报错:

```
pthread.c:(.text+0x85) : 对 ‘pthread_create’ 未定义的引用
```

由于pthread库不是Linux系统默认的库，连接时需要使用库libpthread.a,所以在使用pthread_create创建线程时，在编译中要加-lpthread参数:

```
gcc -o pthread -lpthread pthread.c
```

这是一个关于Posix线程编程的专栏。作者在阐明概念的基础上，将向您详细讲述Posix线程库API。本文是第一篇将向您讲述线程的创建与取消。

一、线程创建

1.1 线程与进程

相对进程而言，线程是一个更加接近于执行体的概念，它可以与同进程中的其他线程共享数据，但拥有自己的栈空间，拥有独立的执行序列。在串行程序基础上引入线程和进程是为了提高程序的并发度，从而提高程序运行效率和响应时间。

线程和进程在使用上各有优缺点：线程执行开销小，但不利于资源的管理和保护；而进程正相反。同时，线程适合于在SMP机器上运行，而进程则可以跨机器迁移。

1.2 创建线程

POSIX通过pthread_create()函数创建线程，API定义如下：

```
int pthread_create(pthread_t * thread, pthread_attr_t * attr,
void * (*start_routine)(void *), void * arg)
```

与fork()调用创建一个进程的方法不同，pthread_create()创建的线程并不具备与主线程（即调用pthread_create()的线程）同样的执行序列，而是使其运行start_routine(arg)函数。thread返回创建的线程ID，而attr是创建线程时设置的线程属性（见下）。pthread_create()的返回值表示线程创建是否成功。尽管arg是void *类型的变量，但它同样可以作为任意类型的参数传给start_routine()函数；同时，start_routine()可以返回一个void *类型的返回值，而这个返回值也可以是其他类型，并由pthread_join()获取。

1.3 线程创建属性

pthread_create()中的attr参数是一个结构指针，结构中的元素分别对应着新线程的运行属性，主要包括以下几项：

__detachstate，表示新线程是否与进程中其他线程脱离同步，如果置位则新线程不能用pthread_join()来同步，且在退出时自行释放所占用的资源。缺省为 PTHREAD_CREATE_JOINABLE状态。这个属性也可以在线程创建并运行以后用pthread_detach()来设置，而一旦设置为 PTHREAD_CREATE_DETACH状态（不论是创建时设置还是运行时设置）则不能再恢复到 PTHREAD_CREATE_JOINABLE状态。

`_schedpolicy`，表示新线程的调度策略，主要包括 `SCHED_OTHER`（正常、非实时）、`SCHED_RR`（实时、轮转法）和 `SCHED_FIFO`（实时、先入先出）三种，缺省为 `SCHED_OTHER`，后两种调度策略仅对超级用户有效。运行时可以用过 `pthread_setschedparam()` 来改变。

`_schedparam`，一个 `struct sched_param` 结构，目前仅有一个 `sched_priority` 整型变量表示线程的运行优先级。这个参数仅当调度策略为实时（即 `SCHED_RR` 或 `SCHED_FIFO`）时才有效，并可以在运行时通过 `pthread_setschedparam()` 函数来改变，缺省为 0。

`_inheritsched`，有两种值可供选择：`PTHREAD_EXPLICIT_SCHED` 和 `PTHREAD_INHERIT_SCHED`，前者表示新线程使用显式指定调度策略和调度参数（即 `attr` 中的值），而后者表示继承调用者线程的值。缺省为 `PTHREAD_EXPLICIT_SCHED`。

`_scope`，表示线程间竞争 CPU 的范围，也就是说线程优先级的有效范围。POSIX 的标准中定义了两个值：`PTHREAD_SCOPE_SYSTEM` 和 `PTHREAD_SCOPE_PROCESS`，前者表示与系统中所有线程一起竞争 CPU 时间，后者表示仅与同进程中的线程竞争 CPU。目前 LinuxThreads 仅实现了 `PTHREAD_SCOPE_SYSTEM` 一值。

`pthread_attr_t` 结构中还有一些值，但不使用 `pthread_create()` 来设置。

为了设置这些属性，POSIX 定义了一系列属性设置函数，包括 `pthread_attr_init()`、`pthread_attr_destroy()` 和与各个属性相关的 `pthread_attr_get---`/`pthread_attr_set---` 函数。

1.4 线程创建的Linux实现

我们知道，Linux 的线程实现是在核外进行的，核内提供的是创建进程的接口 `do_fork()`。内核提供了两个系统调用 `_clone()` 和 `fork()`，最终都用不同的参数调用 `do_fork()` 核内 API。当然，要想实现线程，没有核心对多进程（其实是轻量级进程）共享数据段的支持是不行的，因此，`do_fork()` 提供了很多参数，包括 `CLONE_VM`（共享内存空间）、`CLONE_FS`（共享文件系统信息）、`CLONE_FILES`（共享文件描述符表）、`CLONE_SIGHAND`（共享信号句柄表）和 `CLONE_PID`（共享进程 ID，仅对核内进程，即 0 号进程有效）。当使用 `fork` 系统调用时，内核调用 `do_fork()` 不使用任何共享属性，进程拥有独立的运行环境，而使用 `pthread_create()` 来创建线程时，则最终设置了所有这些属性来调用 `_clone()`，而这些参数又全部传给核内的 `do_fork()`，从而创建的“进程”拥有共享的运行环境，只有栈是独立的，由 `_clone()` 传入。

Linux 线程在核内是以轻量级进程的形式存在的，拥有独立的进程表项，而所有的创建、同步、删除等操作都在核外 `pthread` 库中进行。`pthread` 库使用一个管理线程（`_pthread_manager()`，每个进程独立且唯一）来管理线程的创建和终止，为线程分配线程 ID，发送线程相关的信号（比如 `Cancel`），而主线程（`pthread_create()`）的调用者则通过管道将请求信息传给管理线程。

二、线程取消

2.1 线程取消的定义

一般情况下，线程在其主体函数退出的时候会自动终止，但同时也可以因为接收到另一个线程发来的终止（取消）请求而强制终止。

2.2 线程取消的语义

线程取消的方法是向目标线程发 `Cancel` 信号，但如何处理 `Cancel` 信号则由目标线程自己决定，或者忽略、或者立即终止、或者继续运行至 `Cancellation-point`（取消点），由不同的 `Cancellation` 状态决定。

线程接收到 `CANCEL` 信号的缺省处理（即 `pthread_create()` 创建线程的缺省状态）是继续运行至取消点，也就是说设置一个 `CANCELED` 状态，线程继续运行，只有运行至 `Cancellation-point` 的时候才会退出。

2.3 取消点

根据 POSIX 标准，`pthread_join()`、`pthread_testcancel()`、`pthread_cond_wait()`、`pthread_cond_timedwait()`、`sem_wait()`、`sigwait()` 等函数以及 `read()`、`write()` 等会引起阻塞的系统调用都是 `Cancellation-point`，而其他 `pthread` 函数都不会引起 `Cancellation` 动作。但是 `pthread_cancel` 的手册页声称，由于 LinuxThread 库与 C 库结合得不好，因而目前 C 库函数都不是 `Cancellation-point`；但 `CANCEL` 信号会使线程从阻塞的系统调用中退出，并置 `EINTR` 错误码，因此可以在需要作为 `Cancellation-point` 的系统调用前后调用 `pthread_testcancel()`，从而达到 POSIX 标准所要求的目标，即如下代码段：

```
pthread_testcancel();
retcode = read(fd, buffer, length);
pthread_testcancel();
```

2.4 程序设计方面的考虑

如果线程处于无限循环中，且循环体内没有执行至取消点的必然路径，则线程无法由外部其他线程的取消请求而终止。因此在这样的循环体的必经路径上应该加入 `pthread_testcancel()` 调用。

2.5 与线程取消相关的pthread函数

```
int pthread_cancel(pthread_t thread)
```

发送终止信号给 `thread` 线程，如果成功则返回 0，否则为非 0 值。发送成功并不意味着 `thread` 会终止。

```
int pthread_setcancelstate(int state, int *oldstate)
```

设置本线程对 `Cancel` 信号的反应，`state` 有两种值：`PTHREAD_CANCEL_ENABLE`（缺省）和 `PTHREAD_CANCEL_DISABLE`，分别表示收到信号后设为 `CANCELED` 状态和忽略 `CANCEL` 信号继续运行；`old_state` 如果不为 `NULL` 则存入原来的 `Cancel` 状态以便恢复。

```
int pthread_setcanceltype(int type, int *oldtype)
```

设置本线程取消动作的执行时机，`type` 由两种取值：`PTHREAD_CANCEL_DEFERRED` 和 `PTHREAD_CANCEL_ASYNCRONOUS`，仅当 `Cancel` 状态为 `Enable` 时有效，分别表示收到信号后继续运行至下一个取消点再退出和立即执行取消动作（退出）；`oldtype` 如果不为 `NULL` 则存入原来的取消动作类型值。

void pthread_testcancel(void)
检查本线程是否处于Canceld状态，如果是，则进行取消动作，否则直接返回。

查看更多>>

想对作者说点什么？

我来说两句

Linux多线程函数pthread_create()函数

函数原型：#include int pthread_create(pthread_t *thread, const pthread_attr_t *attr,void *(*start_routi...

 luyee2010 2013-08-08 21:13:29 阅读数：21150

pthread_create函数的详细讲解(包括向线程函数传递参数详解)

pthread_create是UNIX环境创建线程函数 头文件 #include 函数声明 int pthread_create(pthread_t*restrict ti...

 liangxanhai 2012-07-20 14:52:57 阅读数：81669

pthread_create回调函数返回值 - CSDN博客

Linux平台中通过POSIX接口创建线程函数为: #include int pthread_create(pthread_t *restrict tidp, const pthread_attr_t *restrict attr, void *(*start_rtn)(
2018-5-24

pthread_create 的返回值 - CSDN博客

pthread_create()函数用法 linux下用C开发多线程程序,Linux系统下的多线程遵循...举报内容: pthread_create 的返回值 举报原因: 色情 政治 抄袭 广告 招聘 骂人...
2018-6-4

c++里的 pthread_create 函数小结

在C++的类中，普通成员函数不能作为pthread_create的线程函数，如果要作为pthread_create中的线程函数，必须是static！ 在C语言中，我们使用p...
 wh_19910525 2013-10-22 14:48:38 阅读数：10555


pthread_create()函数用法 - CSDN博客

pthread_create()的返回值表示线程创建是否成功。尽管arg是void *类型的变量,但它同样可以作为任意类型的参数传给start_routine()函数;同时,start_routine()可以返回...
2018-5-22

pthread_create() 返回 11 - CSDN博客


通过反复的 pthread_create() ----> pthread_exit (0) 一段时间后,会导致pthread_create() 失败,返回11 google后,发现,单纯地调用 pthread_...
2018-6-6

pthread_create传递线程参数需要注意的问题

程序如下void* thread_routine(void *arg) { int *cnt = (int*)arg; for (int i=0; i
 MonroeD 2017-03-05 16:09:12 阅读数：2891

pthread_create传递参数

#include #include using namespace std; pthread_t thread; void *fn(void *arg) { int...

 YEYUANGEN

2011-09-07 17:09:41

阅读数：43653

pthread_create()函数用法 - CSDN博客

pthread_create()的返回值表示线程创建是否成功。尽管arg是void *类型的变量,但它同样可以作为任意类型的参数传给start_routine()函数;同时,start_routine()可以返回...

2017-12-6

pthread_create() 返回 11 - CSDN博客

pthread_create()函数用法 xipiaoyouzi 7250次阅读 2012-11-21 10:59:24 pthread_create 的返回值 yang3wei 7686次阅读 2013-07-16 23:09:55 ...

2018-5-29

正确使用pthread_create，防止内存泄漏

原文出处：http://www.cnblogs.com/lidabo/p/5514100.html 近日，听说pthread_create会造成内存泄漏，觉得不可思议，因此对posi...

 whuzm08

2016-12-09 12:51:01

阅读数：1525

pthread_create返回11解决方法 - CSDN博客

pthread_t test_tid; while(1) { usleep(10000); ret = pthread_create(&...如果线程是joinable状态,当线程函数自己返回退出时或pthread_exit时都不会释放线程...

2018-5-8

pthread_create传递参数 - CSDN博客

参数func 表示代一个参数void *,返回值也为void *; 对于void *arg,参数传入...pthread_create是类Unix操作系统(Unix、Linux、Mac OS X等)的创建线程的函数,头...

2018-5-5

C++ pthread_create 线程创建与传参 (struct)

在JNI里边想用线程，结果发现C++线程的几个重要的点，暂时没找到合适的资料详细阐述这些知识点。花了点时间终于把线程的创建，参数传递搞清楚了，特意记录下来，希望对别人有帮助。...

 CJ_star

2017-01-12 16:45:42

阅读数：3295

pthread_create用法

linux下用C开发多线程程序，Linux系统下的多线程遵循POSIX线程接口，称为pthread。#include int pthread_create(pthreada...

 Tommy_wxie

2013-01-26 20:29:18

阅读数：24272

pthread_create()函数说明 - CSDN博客

The pthread_create() function is used to create a new thread, with attributes...pthread_create回调函数返回值 Linux平台中通过POSIX接口创建线程函数为: #inc...

2018-5-20


pthread_create函数 - CSDN博客

函数原型:int pthread_create(pthread_t *tid, const pthread_attr_t *tattr, void*(*start_routine)(void *), void *arg); 功能:创建一个新的线程,并将...

2018-5-22

pthread_create()

pthread_create函数 原型：int pthread_create ((pthread_t *thread, pthread_attr_t *attr, void * (*...

 youbang321

2012-07-31 19:57:36

阅读数：131420

线程创建 pthread_create 中自定义参数注意事项

1. 函数原型 int pthread_create(pthread_t thread, const pthread_attr_t attr, void (start_rou...

 chinaeran

2017-02-10 18:34:10

阅读数：1307

pthread_create()在C和C++使用区别

原址 [pthread_create的使用](#) [html] view plain copy print? int pthread_create(pthread_t*, c...

 u010164190 2017-06-21 11:43:58 阅读数：712

clone的fork与pthread_create创建线程有何不同&pthread多线程编程的学习小结

进程是一个指令执行流及其执行环境，其执行环境是一个系统资源的集合，这些资源在Linux中被抽象成各种数据对象：进程控制块、虚存空间、文件系统，文件I/O、信号处理函数。所以创建一个进程的过程就...

 rock_joker 2017-05-25 09:32:36 阅读数：1441

pthread_create返回11解决方法

一直以为，程序创建线程，线程运行结束会自动清空资源 最近在一个项目中用到了线程，除去业务逻辑，我把他简化出来是下面这样//pthread.c #include #include static i...

 cry1994 2016-09-24 11:41:55 阅读数：3437

pthread_create()函数用法

linux下用C开发多线程程序，Linux系统下的多线程遵循POSIX线程接口，称为pthread。 #include int pthread_create(pthread...

 xipiaoyouzi 2012-11-21 10:59:24 阅读数：7305


pthread_create()创建线程最大个数

线程应用程序最常见导致创建线程失败的原因是线程栈大小的设置。创建一个新的线程，默认情况下系统为线程栈预留了2MB的寻址空间。线程栈起始于进程虚拟内存的高端地址，并向虚拟内存底端地址方向扩展。取决于线...

 maopig 2016-08-20 16:00:46 阅读数：5122


多线程之pthread_create创建线程

pthreads定义了一套C程序语言类型、函数、与常量。以pthread.h和一个线程库实现。数据类型： pthread_t:线程句柄 pthread_attr_t:线程属性 线程操作函数： ...

 liuy5277 2013-01-20 23:26:08 阅读数：5689

linux下c++使用pthread_create时需要调用类成员

前几天自己写代码的时候，需要用到多线程的东西，但是由于需要运行的函数是一个类的成员，没有办法进行调用（将函数填入之后，编译报错。大致是参数格式不正确之类的提示），后来在网上查找了一些解决的办法，做下记...

 spark550 2016-05-30 10:34:43 阅读数：1702

线程创建之重要属性PTHREAD_CREATE_DETACHED

#include int pthread_join(pthread_t thread, void **value_ptr); int pthread_detach(pthread_t thr...

 xuyunzhang 2013-04-15 15:52:26 阅读数：4544

线程属性pthread_attr_t简介

本文编辑整理自：<http://hi.baidu.com/7828058/blog/item/256e16decd1a385e94ee3784.html> <http://www.ibm.com/d...>

 hudashi 2012-07-02 18:04:28 阅读数：49457

pthread_create 的返回值

转载自：http://blog.csdn.net/charlie_2010/article/details/6052898 上午做个测试程序，模拟多客户端测试服务器压力，开始时有客户端总是显示断...

 yang3wei 2013-07-16 23:09:55 阅读数：7756

Pthread使用总结

摘要最近由于开发需要用到多线程，以前看的ARTOOLKIT项目中有用到pthread,所以又重新看了一下，重点分析了它的多线程实现。它的实现方式的确很nice,以前没有注意到，现在整理在这里，一方面便于...

 liujiabin076 2016-12-04 22:02:28 阅读数：8949


关于pthread里面一些函数的使用心得！

第一次使用pthread,遇到的问题还真不少，现在我——记录一下： 1.关于编译时出现 对‘pthread_create’未定义的引用 之类的错误的解决：由于pthread库不是Linux系统默...

 lishuhuakai 2013-09-25 14:27:20 阅读数：24439

pthread_create()函数传递整型参数问题

pthread_create()函数传递参数问题根据我的尝试，线程的函数在定义的时候，必须传入参数，使用的时候第四个参数可以填入NULLvoid *func(void *t) { printf...

 jiangph1001 2016-11-10 18:21:15 阅读数：2315

对pthread_create未定义的引用

今天写一个线程的程序，已经在c文件中包含了线程的头文件，可是编译的时候却报错“对pthread_create未定义的引用”，原来时因为pthread库不是Linux系统默认的库，连接时需要使用库lib...

 yezhen910328 2014-03-11 08:40:26 阅读数：5829

Signal ()函数详细介绍 Linux函数

signal()函数理解 在这个头文件中。 signal (参数1, 参数2) ; 参数1：我们要进行处理的信号。系统的信号我们可以再终端键入 kill -l查看(共64个)。其实这些信号时系统定义...

 ta893115871 2012-04-18 20:27:24 阅读数：82692

Linux下"undefined reference to ‘pthread_create’"问题解决

实验环境Centos 7.0, gcc 4.8.5问题在运行一个多线程的c文件时候报了错： #include #include #include #include pthread_mutex_t...

 qq_35056292 2017-04-15 09:11:14 阅读数：780

多线程~~简单的线程创建,C语言实现

线程，是计算机中最小的执行单元。通常，在window应用程序运行时，操作系统都会为其自动创建一个线程，即主线程。通过主线程，可以创建多个线程或进程。使用多线程，可以提高程序的执行效率。 ...

 qq_25425023 2015-04-24 21:30:58 阅读数：25299


1.pthread_create()初体验

```
#include #include #include void *sayhello(void *arg) { printf("hello, world! I'm son\n"); } int m...
```

 sadjason 2013-08-01 17:49:57 阅读数：4418

Matplotlib 中文用户指南 7.3 事件处理及拾取

事件处理及拾取 原文：Event handling and picking 译者：飞龙 协议：CC BY-NC-SA 4.0 matplotlib 使用了许多用户界面工具包（ w...

 wizardforcel 2017-01-30 12:00:45 阅读数：3713

pthread_create线程创建的过程剖析

概述 在Linux环境下，pthread库提供的pthread_create()API函数，用于创建一个线程。线程创建失败时，它可能会返回ENOMEM或EAGAIN。这篇文章主要讨论线程创建过程...

 yetyongjin 2012-06-18 16:56:13 阅读数：20913

pthread_create如何传递多个参数

pthread_create如何传递多个参数 分类： C语言2012-06-19 08:44 98人阅读 评论(0) 收藏 举报 http://zhidao.baidu.com...

 jfkidear 2012-08-29 08:57:19 阅读数：13712

避免使用不当pthread_create函数造成内存泄露

pthread_create使用不当会造成内存泄漏，

 wuruixn 2014-09-05 15:07:31 阅读数：2582

Linux pthread_create

原文地址：[pthread_create](#)>Linux pthread_create作者：大头龙Linux pthread_create 2009-11-01 19:54:26| 分类：li...

 andgot 2015-02-03 11:05:11 阅读数：1457

Linux多线程学习（一）pthread_create

Linux系统下的多线程遵循POSIX线程接口，称为pthread。#include int pthread_create(pthread_t *restrict tidp, con...

 xiaoyeyopulei 2012-09-10 10:48:46 阅读数：6332

pthread_cond_broadcast使用示例

今天在使用pthread_cond_t时，发现pthread_cond_t使用pthread_cond_broadcast函数唤醒多个条件变量时，使用两个互斥量分别控制时，只能唤醒其中一个变量，最后通...

 pengrui18 2017-04-23 23:14:22 阅读数：3570

pthread 简要使用指南（一）pthread_create

POSIX thread 简称为pthread，Posix线程是一个POSIX标准线程。该标准定义内部API创建和操纵线程。Pthreads定义了一套 C程序语言类型、函数与常量，它以 pt hrea...

 hslinux 2012-08-29 17:01:23 阅读数：3430

pthread_create未定义

对pthread_create未定义的引用 今天写一个线程的程序，已经在c文件中包含了线程的头文件，可是编译的时候却报错“对pthread_create未定义的引用”，原来时因为pthread库...

 chenjianqi0502 2017-12-01 09:50:55 阅读数：49

g++编译pthread_create函数的注意点

1.在c++中函数的参数是和函数一起进行编译的，参数列表为：int pthread_create(pthread_t*, const pthread_attr_t*, void* (*)(void*)...

 sjiubt 2011-09-30 15:02:41 阅读数：3508

类的成员变量作为pthread_create的参数

方式一：使用this指针 方式二：直接传递。实例：“paraData.h”#include #include class paraData { public: ...

 lfxyan 2015-09-22 17:23:22 阅读数：1003


Linux中pthread线程使用详解

Linux下多线程详解pdf文档下载：点击这里！Linux中线程和进程的区别：http://blog.csdn.net/qq_21792169/article/details/5043...

 JoysonQin 2017-04-19 10:19:52 阅读数：10058

pthread

POSIX线程（POSIX threads），简称Pthreads，是线程的POSIX标准。该标准定义了创建和操纵线程的一整套API。在类Unix操作系统（Unix、Linux、Mac OS X等）中...

 hebbely 2016-07-07 14:33:02 阅读数：3838

Pthread API总结

Pthread API 函数名 说明 pthread_atfork fork前后的处理函数，一般不建议多线程下进行fork，见http://blog.csdn.net/anxuegang...

 gogdizzy 2016-11-14 19:50:33 阅读数：1156

在windows下配置pthread

Pthread是由POSIX提出的一套通用的线程库，在linux平台下，他被广泛的支持，而windows平台下，却并不被支持，而pthreads-w32为我们提供了解决方案，本文我们准备在我们的win...


 qianchenglenger

2013-11-24 01:09:33

阅读数：37000

pthread多线程编程的学习小结

pthread多线程编程整理=====pthre...


 sunboy_2050

2010-10-04 12:33:00

阅读数：35537

C++服务器（七）：Windows 下配置pthread

pthread POSIX线程（ POSIX threads ），简称Pthreads ，是线程的POSIX标准。[1] 这套接口在 Linux 下得到很好的支持。但是在 Windows 下却需要额外配...

 u014613043

2016-04-09 09:35:31

阅读数：656

Pthread Tutorial

Tutorials | Exercises | Abstracts | LC Workshops | Comments | Search | Privacy & Legal N...

 xyblog

2016-02-02 14:56:28

阅读数：623

linux的pthread_self与gettid的返回值和开销的区别

pthread_self()是POSIX的实现，它的返回值是pthread_t，pthread_t在linux中实际是无符号长整型，即unsigned long。gettid是系统调用，它的返回值...

 littlefang

2012-05-18 12:25:36

阅读数：14817

Linux系统下-进程间通信（消息队列-详解）

Linux下，进程间通信方式：#（1）管道（Pipe）及有名管道（named pipe）：#（2）信号量：#（3）共享内存：#（4）消息队列：#（5）套接口（Socket）#（6）信号（S...


 sty23122555

2016-04-12 12:31:52

阅读数：14127

Linux进程间通信——使用消息队列

下面来说说如何用不用消息队列来进行进程间的通信，消息队列与命名管道有很多相似之处。一、什么是消息队列 消息队列提供了一种从一个进程向另一个进程发送一个数据块的方法。 每个数据块都被认为含有一个...

 ljianhui

2013-08-25 00:09:57

阅读数：107818

C语言多线程编程为什么要用pthread_join函数

如果你用的是Linux/UNIX/MacOSX，那么我们已经可以开始了，如果你用的是WINDOWS，那么你需要从网站上下载PTHREAD的WINDOWS开发包，所幸他非常的小。网站地址是http://...

 boy8239

2008-03-06 11:31:00

阅读数：7604

没有更多推荐了，[返回首页](#)

个人资料



海边顽石

关注

原创	粉丝	喜欢	评论
30	3	2	5

等级：博客 4 访问：8万+

积分：1215

排名：4万+

最新文章

TCP/IP协议号

Errors were encountered while processing:
google-chrome-stable

hash算法原理详解

linux下cp，mv进行动态库覆盖问题分析

tcpdump命令抓包保存pcap文件wireshark分析

个人分类

C语言中的经典小程序

19篇

linux中的经典小问题

25篇

c语言中的经典小问题

2篇

关于系统问题的总结

37篇

归档

2018年6月

2篇

2018年5月

3篇

2018年4月

1篇

2018年3月

1篇

2017年12月

1篇

展开

热门文章

64G的EXFAT格式的U盘如何格式化为FAT32

阅读量：16651

Linux-dd命令详解

阅读量：5505

zsh的自动完成辅助工具：oh-my-zsh

阅读量：5185

win7+ubuntu14.04双系统，重装win7后，修复grub方法

阅读量：4433

安装 Oh My Zsh 插件

阅读量：4255

最新评论

64G的EXFAT格式的U盘如何格...

temryu：非常感谢！！！

Linux两种方法来处理传入TCP...

xfyunhanliumu：网上的博客看来看去 都一个样 好慌

win7+ubuntu14.04双...

u012341410：可以问下博主,制作ubuntu U盘镜像时,需要和以前的版本一致吗?比如16.04的,如果u盘是1...

win7+ubuntu14.04双...

u012341410：可以问下博主,制作ubuntu U盘镜像时,需要和以前的版本一致吗?比如16.04的,如果u盘

是1...
win7+ubuntu14.04双...
u012341410 : 可以问下博主,制作ubuntu U盘镜像
时,需要和以前的版本一致吗?比如16.04的,如果u盘
是1...

联系我们



请扫描二维码联系客服
✉ webmaster@csdn.net
☎ 400-660-0108
🗣 QQ客服 🗣 客服论坛

关于 招聘 广告服务 百度
©1999-2018 CSDN版权所有
京ICP证09002463号

经营性网站备案信息
网络110报警服务
中国互联网举报中心
北京互联网违法和不良信息举报中心