Bilgisayar İşletim Sistemleri, Uygulama 3

İplikler (Threads)

Istanbul Technical University
34469 Maslak, İstanbul

2017

Bugün

Bilgisayar İşletim Sistemleri, PS 3

- İplik Yaratma ve Sonlandırma
- İpliklerin Birleştirilmesi
- İpliklerde Global Değişkenlerin Kullanımı

İplik Yaratma

```
#include <pthread.h>

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

`pthread_t *thread`	: Yaratılacak ipliğe gösterge
`const pthread_attr_t *attr`	: Yaratılacak ipliğin özelliklerine gösterge
`void *(*start_routine)(void*)`	: İpliği başlatacak yordama gösterge
`void *arg`	: Başlangıç yordamının parametrelerine gösterge

Doğru çalıştığında 0, hatalı çalışma durumunda hata mesajı dönderir.

Örnek Program 1

```c
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>

//void * bir memory bloguna tip belirtmeden bir pointer ile
    erisilmek istendiginde kullanilir
void* print_message_function(void *ptr){
  char *message;
  // interpreting as char *
  message = (char *) ptr;
  printf("\n %s \n", message);
  // terminating the thread
  pthread_exit(NULL);
}

int main(){
  pthread_t thread1, thread2, thread3;
  char *message1 = "Hello";
  char *message2 = "World";
```

Örnek Program 1

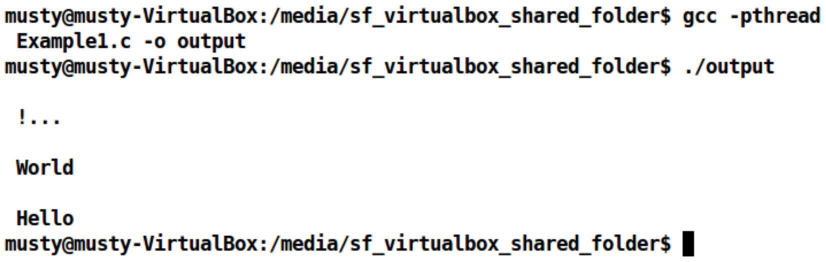
```
    char *message3 = "!...";
    // creating 3 threads with start routine as print_message_function
        and start routine arguments as message1, message2 and message3
    if(pthread_create(&thread1,NULL,print_message_function,(void *)
        message1)){
            fprintf(stderr,"pthread_create failure\n");
            exit(-1);
    }
    if(pthread_create(&thread2,NULL,print_message_function,(void *)
        message2)){
            fprintf(stderr,"pthread_create failure\n");
            exit(-1);
    }
    if(pthread_create(&thread3,NULL,print_message_function,(void *)
        message3)){
            fprintf(stderr,"pthread_create failure\n");
            exit(-1);
    }
    // to block main to support its threads until they terminate
    pthread_exit(NULL);
}
```

İplik/ler içeren bir programın derlenmesi

- Kaynak Dosya : `source.c`
- Çalıştırılabilir Dosya: `output`
- Bu iki dosya thread kütüphanesi ile bağlanmalıdır. Doğru bir derleme örneği:

```
gcc -pthread source.c -o output
```

Örnek Program 1'in çıktısı



```
musty@musty-VirtualBox:/media/sf_virtualbox_shared_folder$ gcc -pthread
 Example1.c -o output
musty@musty-VirtualBox:/media/sf_virtualbox_shared_folder$ ./output

 !...

 World

 Hello
musty@musty-VirtualBox:/media/sf_virtualbox_shared_folder$ █
```

Örnek Program 2

```c
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define NUM_THREADS 4

 void* BusyWork(void *t){
     int i;
     long tid;
     double result=0.0;
     tid = (long)t;
     printf("Thread %ld starting...\n", tid);
     for (i=0; i<1000000; i++){
         result = result + sin(i) * tan(i);
     }
     printf("Thread %ld done. Result = %e\n", tid, result);
     pthread_exit((void*) t);
 }
```

Barney B. (2013). POSIX Threads Programming. Retrieved March 03, 2014, from https://computing.llnl.gov/tutorials/pthreads/

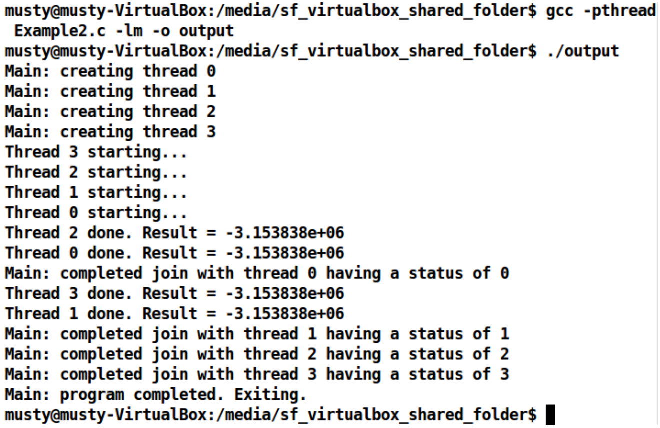
Örnek Program 2

```
int main (int argc, char *argv[]){
  pthread_t thread[NUM_THREADS];
  pthread_attr_t attr;
  int rc;
  long t;
  void *status;
  // Initialize and set thread detach state attribute
  // Only threads that are created as joinable can be joined
  // Threads created as PTHREAD_CREATE_DETACHED, cannot be joined
  pthread_attr_init(&attr);
  pthread_attr_setdetachstate(&attr, PTHREAD_CREATE_JOINABLE);
  for(t=0; t<NUM_THREADS; t++) {
    printf("Main: creating thread %ld\n", t);
    // creating thread t
    rc = pthread_create(&thread[t], &attr, BusyWork, (void *)t);
    if (rc) {
      printf("ERROR; return code from pthread_create() is %d\n", rc);
      exit(-1);
    }
  }
```

Örnek Program 2

```
// Free library resources used by the attribute
pthread_attr_destroy(&attr);

for(t=0; t<NUM_THREADS; t++) {
  // pthread_join fonksiyonu ile, bir thread'in sonlanmasini
   bekleyebiliriz. Bu fonksiyonun kullanildigi thread, sonlanmasi
   beklenen thread sonlanana kadar bloklanacaktir.
  rc = pthread_join(thread[t], &status);
//eger thread[t] durdurulursa bu thread in icerigi status un point
    ettigi yere yazilir.

  if (rc) {
    printf("ERROR; return code from pthread_join() is %d\n", rc);
    exit(-1);
  }
  printf("Main: completed join with thread %ld having a status of
  %ld\n",t,(long)status);
}
printf("Main: program completed. Exiting.\n");
// to block main to support its threads until they terminate
pthread_exit(NULL);
```

Örnek Program 2'nin çıktısı



```
musty@musty-VirtualBox:/media/sf_virtualbox_shared_folder$ gcc -pthread
 Example2.c -lm -o output
musty@musty-VirtualBox:/media/sf_virtualbox_shared_folder$ ./output
Main: creating thread 0
Main: creating thread 1
Main: creating thread 2
Main: creating thread 3
Thread 3 starting...
Thread 2 starting...
Thread 1 starting...
Thread 0 starting...
Thread 2 done. Result = -3.153838e+06
Thread 0 done. Result = -3.153838e+06
Main: completed join with thread 0 having a status of 0
Thread 3 done. Result = -3.153838e+06
Thread 1 done. Result = -3.153838e+06
Main: completed join with thread 1 having a status of 1
Main: completed join with thread 2 having a status of 2
Main: completed join with thread 3 having a status of 3
Main: program completed. Exiting.
musty@musty-VirtualBox:/media/sf_virtualbox_shared_folder$ 
```

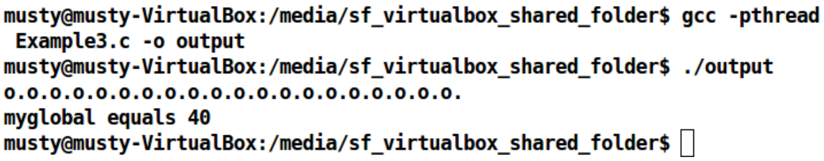
Örnek Program 3

```c
#include <pthread.h>
#include <stdlib.h>
#include <stdio.h>

int myglobal;

void* thread_function(void *arg){
  int i;
  // changing the value of myglobal in thread_function
  for(i=0;i<20;i++){
      myglobal++;
      printf(".");
      // to force writing all user-space buffered data to stdout
      fflush(stdout);
      sleep(1);
  }
  pthread_exit(NULL);
}

int main(void){
  pthread_t mythread;
```

Örnek Program 3

```
    myglobal=0;
    // creating a thread using thread_function as the start routine
    if(pthread_create(&mythread,NULL,thread_function,NULL)){
      printf("error creating thread");
      abort();
    }
    // changing the value of myglobal in main()
    for(i=0;i<20;i++){
      myglobal = myglobal+1;
      printf("o");
      // to force writing all user-space buffered data to stdout
      fflush(stdout);
      sleep(1);
    }
    printf("\nmyglobal equals %d\n",myglobal);
    // to block main to support its threads until they terminate
    pthread_exit(NULL);
}
```

Örnek Program 3’ün çıktısı



```
musty@musty-VirtualBox:/media/sf_virtualbox_shared_folder$ gcc -pthread
 Example3.c -o output
musty@musty-VirtualBox:/media/sf_virtualbox_shared_folder$ ./output
o.o.o.o.o.o.o.o.o.o.o.o.o.o.o.o.o.o.o.o.
myglobal equals 40
musty@musty-VirtualBox:/media/sf_virtualbox_shared_folder$ 
```