Bilgisayar İşletim Sistemleri

Uygulama IV

İplikler

İplik Yaratma

```
#include <pthread.h>

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

İplik

İplik özelliği

İplik tanımlanmış bir start_routine fonksiyonu çalıştırılarak yaratılır

İpliği start_routine fonksiyonun argümanı

Örnek Program

```
void print_message_function( void *ptr );

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

    pthread_create( &thread1, pthread_attr_default,
                    (void*)&print_message_function, (void*) message1);
    pthread_create(&thread2, pthread_attr_default,
                    (void*)&print_message_function, (void*) message2);

    exit(0);
}
void print_message_function( void *ptr )
{
    char *message;
    message = (char *) ptr;
    printf("%s ", message);
}
```

Derleme

- Kaynak dosyasi:
  - kaynak.c
- Çıkıştaki çalıştırılabilir dosya adi
  - cikti
- Bu uygulamaların çalıştırılabilmesi için *pthread* kitaplığına bağlanmaları gereklidir.
- Bu nedenele şu şekilde derlenmeliler:
  - gcc -D_POSIX_C_SOURCE -lpthread kaynak.c -o cikti

İplik Yaratma, Birleştirme, Sonlandırma

```
#define _GNU_SOURCE
#define _REENTRANT
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <sys/types.h>
#include <sys/time.h>
#include <unistd.h>

int MAX=5;

int    my_rand( int, int );
void  *say_it( void * );

int main(int argc, char *argv[])
{
    pthread_t   thread_id[MAX];
    int       status, *p_status = &status;
    int       i;

    setvbuf(stdout, (char *) NULL, _IONBF, 0);

    if ( argc > MAX+1 )
    {                          // check arg list
        fprintf(stderr, "%s arg1, arg2, ... arg%d\n", argv[0], MAX);
        return 1;
    }

    printf("Displaying\n");

    for (i = 0; i < argc-1; ++i)
    {  // generate threads
        if( pthread_create(&thread_id[i],NULL,say_it,(void *)argv[i+1]) > 0)
        {
            fprintf(stderr, "pthread_create failure\n");
            return 2;
        }
    }
```

İplik Yaratma, Birleştirme, Sonlandırma (Devam)

```
    for (i=0; i < argc-1; ++i)
    {       // wait for each thread
        if ( pthread_join(thread_id[i], (void **) p_status) > 0)
        {
            fprintf(stderr, "pthread_join failure\n");
            return 3;
        }
        printf("\nThread %d returns %d", thread_id[i], status);

    }

    printf("\nDone\n");
    return 0;
}
```

İplik Yaratma, Birleştirme, Sonlandırma (Devam)

```
void * say_it(void *word)
{
    int i;

    int numb = my_rand(2,6);

    printf("%s\t to be printed %d times.\n", (char *)word, numb);

    sleep(1);

    for (i=0; i < numb; ++i)
    {
        sleep(1);
        printf("%s ", (char *) word);
    }
    sleep(4);
    return (void *) NULL;
}

//   Generate a random # within given range
int my_rand(int start, int range)
{
    struct timeval t;
    gettimeofday(&t, (struct timezone *)NULL);
    return (int)(start+((float)range * rand_r((unsigned *)&t.tv_usec)) / (RAND_MAX+1.0));
}
```

Örnek Program Çıktısı

```
[root@koknar root]$ ./threadz 1 2 3 4 5
Displaying
1       to be printed 3 times.
2       to be printed 2 times.
3       to be printed 3 times.
4       to be printed 6 times.
5       to be printed 7 times.
1 2 3 4 5 1 2 3 4 5 1 3 4 5 4 5 4 5 4 5 5
Thread 1082395440 returns 0
Thread 1090788144 returns 0
Thread 1099180848 returns 0
Thread 1116957488 returns 0
Thread 1125350192 returns 0
Done
```

İpliklerle Global Değişkenlerin Kullanımı

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

int myglobal;

void *thread_function(void *arg)
{
    int i,j;
    for ( i=0; i<20; i++ )
    {
        j=myglobal;
        j=j+1;
        printf(".");
        fflush(stdout);
        sleep(1);
        myglobal=j;
    }
    return NULL;
}

int main(void)
{

    pthread_t mythread;
    int i;

    if ( pthread_create( &mythread, NULL, thread_function, NULL) )
    {
        printf("error creating thread.");
        abort();
    }

    for ( i=0; i<20; i++)
    {
        myglobal=myglobal+1;
        printf("o");
        fflush(stdout);
        sleep(1);
    }
```

İpliklerle Global Değişkenlerin Kullanımı

```
    if ( pthread_join ( mythread, NULL ) )
    {
        printf("error joining thread.");
        abort();
    }

    printf("\nmyglobal equals %d\n",myglobal);

    exit(0);
}
```

```
[root@koknar root]$ ./threadApp
o.o.o.o.o.o.o.o.o.o.o.o.o.o.o.o.o.o.o.o.o.o.
o.o
myglobal equals 20
```