Bilgisayar İşletim Sistemleri Uygulama VI

Unix’de Paylaşılan Bellek Alanları

Yapısı

```
struct shmid_ds {
struct ipc_perm shm_perm;          /* operation permission struct*/
size_t shm_segsz;                  /* size of segment in bytes */
__time_t shm_atime;                /* time of last shmat()*/
unsigned long int __unused1;
__time_t shm_dtime;                /* time of last shmdt() */
unsigned long int __unused2;
__time_t shm_ctime;                /* time of last change by shmctl() */
unsigned long int __unused3;
__pid_t shm_cpid;                  /* pid of creator */
__pid_t shm_lpid;                  /* pid of last shmop */
shmatt_t shm_nattch;               /* number of current attaches */
unsigned long int __unused4;
unsigned long int __unused5;
};
```

Fonksiyonlar

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>

int shmget(key_t key, int size, int flag);
int shmctl(int shmid, int cmd, struct shmid_ds *buf);

void *shmat(int shmid, void *addr, int flag);
int shmdt(void *addr);
```

Örnek 1

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/wait.h>
#include <sys/sem.h>
#include <sys/shm.h>
#include <unistd.h>
#include <signal.h>
#include <stdlib.h>
#include <stdio.h>

#define KEYSEM 123456
#define KEYSEM2 2345678
#define KEYSHM 345678


void sinyal12(void)
{}
```

```
void sem_signal(int semid, int val)
{
   struct sembuf semafor;

   semafor.sem_num=0;
   semafor.sem_op=val;
   semafor.sem_flg=1;
   semop(semid, &semafor,1);
}

void sem_wait(int semid, int val)
{
   struct sembuf semafor;

   semafor.sem_num=0;
   semafor.sem_op=(-1*val);
   semafor.sem_flg=1;
   semop(semid, &semafor,1);
}
```

Örnek 1

```
int main (void)
{
    int i, g, f, cocuk[2];
    int sonsem=0,
        kilit=0,
        siram=0,
        pba=0,
        *x=NULL;
    signal(12, (void *) sinyal12);
    for (i=0; i<2; i++)
    {
        f=fork();
        if (f==-1)
        {
            printf("FORK hata....\n");
            exit(1);
        }
        if (f==0)
            break;
        cocuk[i]=f;
    }
```

```
if (f!=0)
   {
      sonsem=semget(KEYSEM2, 1,
    0700|IPC_CREAT);
      semctl(sonsem, 0, SETVAL,0);


    kilit=semget(KEYSEM,1,0700|IPC
    _CREAT);
      semctl(kilit,0,SETVAL,1);


    pba=shmget(KEYSHM,sizeof(int),
    0700|IPC_CREAT);
      x=(int *)shmat(pba,0,0);
      *x=0;
      shmdt(x);
      sleep(2);
```

Örnek 1

```
printf("anne kaynaklari yaratti ve
    cocuklari baslatacak.\n");
      for (i=0; i<2; i++)
          kill(cocuk[i],12);

      sem_wait(sonsem,2);

      printf("Cocuklarin isi bitti\n");
      semctl(sonsem,0,IPC_RMID,0);
      semctl(kilit,0,IPC_RMID,0);
      shmctl(pba,IPC_RMID,0);
      exit(0);
  }
```

```
else
  {
     siram=i;
     pause();
     kilit=semget(KEYSEM,1,0);

   sonsem=semget(KEYSEM2,1,0);


   pba=shmget(KEYSHM,sizeof(int),
   0);
     x=(int *) shmat(pba,0,0);

     printf("cocuk %d basliyor ....\n",
   siram);
```

Örnek 1

```
 for (i=0; i<5; i++)
        {
            sem_wait(kilit,1);
            printf("   cocuk %d: degeri %d buldum \n", siram, *x);
            g=*x;
            sleep(1);
            g += i;
            *x=g;
            printf("   cocuk %d: yeni degeri %d yaptim \n", siram, *x);
            sem_signal(kilit,1);
            sleep(1);
        }
        shmdt(x);
        sem_signal(sonsem,1);
        exit(0);
    }
    return(0);
}
```

Örnek 1 Çıktısı

```
anne kaynaklari yaratti ve cocuklari baslatacak.
cocuk 0 basliyor ....
   cocuk 0: degeri 0 buldum
cocuk 1 basliyor ....
   cocuk 0: yeni degeri 0 yaptim i:0
   cocuk 1: degeri 0 buldum
   cocuk 1: yeni degeri 0 yaptim i:0
   cocuk 0: degeri 0 buldum
   cocuk 0: yeni degeri 1 yaptim i:1
   cocuk 1: degeri 1 buldum
   cocuk 1: yeni degeri 2 yaptim i:1
   cocuk 0: degeri 2 buldum
   cocuk 0: yeni degeri 4 yaptim i:2
   cocuk 1: degeri 4 buldum
   cocuk 1: yeni degeri 6 yaptim i:2
   cocuk 0: degeri 6 buldum
   cocuk 0: yeni degeri 9 yaptim i:3
   cocuk 1: degeri 9 buldum
   cocuk 1: yeni degeri 12 yaptim i:3
   cocuk 0: degeri 12 buldum
   cocuk 0: yeni degeri 16 yaptim i:4
   cocuk 1: degeri 16 buldum
   cocuk 1: yeni degeri 20 yaptim i:4
Cocuklarin isi bitti
```

Örnek 2

```
#include <stdio.h>
#include <sys/shm.h>
#include <sys/stat.h>

int main ()
{
    int segment_id;
    char* shared_memory;
    struct shmid_ds shmbuffer;
    int segment_size;
    const int shared_segment_size = 0x6400;

/*ortak bellek bölgesini al*/
    segment_id = shmget (IPC_PRIVATE, shared_segment_size,
                         IPC_CREAT | IPC_EXCL | S_IRUSR | S_IWUSR);
```

Örnek 2

```
 /* PBA bağlantı kur*/
    shared_memory = (char*) shmat (segment_id, 0, 0);
    printf (“PBA bağlantı adresi %p\n", shared_memory);

/* Segman büyüklüğünü öğren*/
    shmctl (segment_id, IPC_STAT, &shmbuffer);
    segment_size = shmbuffer.shm_segsz;
    printf (“Segman büyüklüğü: %d\n", segment_size);

/* PBA’ya bir katar yaz.  */
    sprintf (shared_memory, “Hello, World.");

 /*Bağlantıyı kopar  */
    shmdt (shared_memory);
```

Örnek 2

```
 /* Farklı bir adreste PBA bağlantısı kur  */
    shared_memory = (char*) shmat (segment_id, (void*) 0x5000000, 0);
    printf ("shared memory reattached at address %p\n", shared_memory);

/* PBA'dan katarı oku  */
    printf ("%s\n", shared_memory);

/* Bağlantıyı kopar*/
    shmdt (shared_memory);

/*PBA'yı iade et*/
    shmctl (segment_id, IPC_RMID, 0);

    return 0;
}
```

Örnek 2 Çıktı

PBA Bağlantı adresi 0xf6ff9000

Segman Büyüklüğü: 25600

PBA Bağlantı adresi 0x5000000

Hello, world.