İşletim Sistemleri Uygulama 5

Unix'de semafor işlemleri

Bilgisayar Mühendisliği

İstanbul Teknik Üniversitesi
34469 Maslak, İstanbul

March 15, 2011

Bugün


İSTANBUL TEKNİK ÜNİVERSİTESİ
1773


İTÜ
Bilgisayar Mühendisliği

Semafor yaratma

- Unix'de semafor işlemlerinde kullanılacak başlık dosyaları
  - sys/ipc.h
  - sys/sem.h
  - sys/types.h
- Semafor yaratma

```
int semget(key_t key, int nsems, int semflg);
semflg:  IPC_CREAT|0700
```

Semafor işlemleri

- `int semop(int semid, struct sembuf *sops, unsigned nsops);`
- ```
  struct sembuf{
  unsigned short sem_num; // numaralama 0'dan başlar
  short sem_op;
  short sem_flg;
  };
  ```
- sem_flg
  - `SEM_UNDO: process sonlanınca işlemi geri al`
  - `IPC_NOWAIT: Eksiltemeyince hata ver ve dön`
- sem_op
  - `== 0 : sıfır olmasını bekle (Okuma Hakkı olmalı)`
  - `!= 0 : değer semafor değerine eklenir(çıkarılır) (Değiştirme hakkı olmalı`


  ```

Semafor kontrolü

- Değer Kontrolü

```
int semctl(int semid, int semnum, int cmd, arg);
```

- cmd
  - IPC_RMID
  - GETVAL
  - SETVAL
  - SETALL
  - GETALL

Temel semafor işlemleri : Artırma

```
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);
}
```

Temel semafor işlemleri : Eksiltme

```
void sem_wait(int semid, int val){                                  1
        struct sembuf semafor;
        semafor.sem_num=0;
        semafor.sem_op=(-1*val);
        semafor.sem_flg=1;
        semop(semid, &semafor,1);                                   6
}
```

Örnek 1

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>                                    3
#include <sys/types.h>
#include <sys/wait.h>
#include <signal.h>
#include <sys/ipc.h>
#include <sys/sem.h>                                   8

#define SEMKEY 8

int sonsem;
                                                      13
void mysignal(void){
}
```

Örnek 1

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

        semafor.sem_num=0;                                          5
        semafor.sem_op=val;
        semafor.sem_flg=1;
        semop(semid, &semafor,1);
}
                                                                    10
void sem_wait(int semid, int val)
{
        struct sembuf semafor;

        semafor.sem_num=0;                                          15
        semafor.sem_op=(-1*val);
        semafor.sem_flg=1;
        semop(semid, &semafor,1);
}
                                                                    20
void mysigset(int num){
        struct sigaction mysigaction;
        mysigaction.sa_handler=(void *)mysignal;
        mysigaction.sa_flags=1;
        sigaction(num,&mysigaction,NULL);                           25
}
```

Örnek 1

```
int main(void)
{
        int f=1, i;
        int cocuklar[10];                                          4

        mysigset(12);

        for (i=0; i<10; i++){
                if (f>0)                                           9
                        f=fork();
                if (f==-1){
                        printf("fork error....\n");
                        exit(1);
                }                                                  14
                if (f==0)
                        break;
                else
                        cocuklar[i]=f;
        }                                                          19
```

Örnek 1

```
if (f>0){ /*anne */                                                    1
        sonsem=semget(SEMKEY, 1, 0700|IPC_CREAT);
        semctl(sonsem, 0, SETVAL,0);
        sleep(1);

        for (i=0; i<10; i++)                                           6
                kill(cocuklar[i], 12);

        sem_wait(sonsem,10);

        printf("Tum cocuklar oldu\n");                                 11

        semctl(sonsem,0,IPC_RMID,0);
}
```

Örnek 1

```
        else{ /*cocuk */                                                    1
                pause();
                sonsem=semget(SEMKEY, 1,0);
                printf("Ben %d. sirada yaratilan cocugum. Kimligim=%d\n",i, getpid());
                printf(" su an .... %d\n",
                semctl(sonsem,0,GETVAL,0));                                 6
                sem_signal(sonsem,1);
        }

        return 0;
}                                                                           11
```

Örnek 2 - Ölümcül kilitlenme

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/types.h>                                          4
#include <sys/wait.h>
#include <signal.h>
#include <sys/ipc.h>
#include <sys/sem.h>
                                                                9
#define SEMKEY_A 1
#define SEMKEY_B 2
#define SEMKEY_C 3

void mysignal(int signum){                                      14
}
```

Örnek 2 - Ölümcül kilitlenme

```
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);
}

void mysigset(int num){
        struct sigaction mysigaction;
        mysigaction.sa_handler=(void *)mysignal;
        mysigaction.sa_flags=0;
        sigaction(num,&mysigaction,NULL);
}
```

Örnek 2 - Ölümcül kilitlenme

```
int main (void)
{
        int semA,semB,semC,c[2],f,i,siram;                              3


        mysigset(12);

        for (i=0; i<2; i++){                                            8
                f=fork();
                if (f==0)
                        break;
                else
                        c[i]=f;                                         13
        }

        if (f==-1){
                printf("FORK hata....\n");
                exit(1);                                                18
        }
```

Örnek 2 - Ölümcül kilitlenme

```
if (f!=0){                                                          1
        printf("Anne kaynaklari yaratmaya basliyor....\n");

        semA=semget(SEMKEY_A, 1, 0700|IPC_CREAT);
        semctl(semA, 0, SETVAL,1);
                                                                    6
        semB=semget(SEMKEY_B,1,0700|IPC_CREAT);
        semctl(semB,0,SETVAL,1);

        semC=semget(SEMKEY_C,1,0700|IPC_CREAT);
        semctl(semC,0,SETVAL,0);                                    11

        sleep(2);

        printf("Anne cocuklari baslatiyor .......\n");
                                                                    16
        for (i=0; i<2; i++)
                kill(c[i],12);

        sem_wait(semC,2);
                                                                    21
        printf("Anne: Cocuklarin isi bitti, kaynaklar iade ediliyor...\n");

        semctl(semC,0,IPC_RMID,0);
        semctl(semA,0,IPC_RMID,0);
        semctl(semB,0,IPC_RMID,0);                                  26

        exit(0);
}
```

Örnek 2 - Ölümcül kilitlenme

```
else{                                                                    1
        siram=i;

        printf("cocuk %d anneden haber bekliyor ....\n", siram);

        pause();                                                         6

        semA=semget(SEMKEY_A,1,0);
        semB=semget(SEMKEY_B,1,0);
        semC=semget(SEMKEY_C,1,0);
                                                                         11
        printf("cocuk %d anneden haber aldi, basliyor ....\n", siram);
```

Örnek 2 - Ölümcül kilitlenme

```
if (siram==0){
        printf("cocuk %d: sem A eksiltiyorum.\n", siram);
        sem_wait(semA,1);                                          3

        sleep(1);

        printf("cocuk %d: sem A tamam, sem B eksiltiyorum.\n", siram);
        sem_wait(semB,1);                                          8

        printf("cocuk %d: kritik bolgemdeyim.\n", siram);
        sleep(5); /* K.B. islemleri */

        sem_signal(semB,1);                                        13
        sem_signal(semA,1);
        sem_signal(semC,1);
}
```

Örnek 2 - Ölümcül kilitlenme

```
                else if (siram==1){
                        printf("cocuk %d: sem B eksiltiyorum.\n", siram);
                        sem_wait(semB,1);
                                                                          4
                        sleep(1);

                        printf("cocuk %d: sem B tamam, sem A eksiltiyorum.\n", siram);
                        sem_wait(semA,1);
                                                                          9
                        printf("cocuk %d: kritik bolgemdeyim.\n", siram);
                        sleep(5); /* K.B. islemleri */

                        sem_signal(semA,1);
                        sem_signal(semB,1);                               14
                        sem_signal(semC,1);
                }
        }
        return 0;
}                                                                         19
```

Örnek 3 - Ölümcül kilitlenmeyi giderme

```
#include <stdio.h>                                          1
#include <unistd.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <signal.h>                                         6
#include <sys/ipc.h>
#include <sys/sem.h>

#define SEMKEY_AB 1
#define SEMKEY_C 2                                          11

void mysignal(void){
}
```

Örnek 3 - Ölümcül kilitlenmeyi giderme

```
void sem_signal(int semid, int val)                                    1
{
        struct sembuf semafor;

        semafor.sem_num=0;
        semafor.sem_op=val;                                            6
        semafor.sem_flg=1;
        semop(semid, &semafor,1);
}

void sem_multi_signal(int semid, int val, int nsems)                   11
{
        struct sembuf semafor[2];

        int i;
                                                                       16
        for (i=0; i<nsems; i++){
                semafor[i].sem_op=val;
                semafor[i].sem_flg=1;
                semafor[i].sem_num=i;
        }                                                              21

        //bir semafor seti uzerinde ayni anda iki islem yapiliyor
        semop(semid, semafor, 2);

        for (i=0; i<nsems; i++){                                       26
                printf("signal : %d su an .... %d\n", i,
                semctl(semid,i,GETVAL,0));
        }
}
```

Örnek 3 - Ölümcül kilitlenmeyi giderme

```
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);
}

void sem_multi_wait(int semid, int val, int nsems){
        struct sembuf semafor[2];
        int i;

        for (i=0; i<nsems; i++){
                semafor[i].sem_op=(-1*val);
                semafor[i].sem_flg=1;
                semafor[i].sem_num=i;
        }

        //bir semafor seti uzerinde ayni anda iki islem yapiliyor
        semop(semid, semafor,2);

        for (i=0; i<nsems; i++){
                printf("wait : %d su an .... %d\n", i,
                semctl(semid,i,GETVAL,0));
        }
}
```

Örnek 3 - Ölümcül kilitlenmeyi giderme

```
void mysigset(int num){
        struct sigaction mysigaction;                                   2
        mysigaction.sa_handler=(void *)mysignal;
        mysigaction.sa_flags=0;
        sigaction(num,&mysigaction,NULL);
}
                                                                        7
int main (void)
{
        int semAB,semC,c[2],f,i,siram;

        mysigset(12);                                                   12

        for (i=0; i<2; i++){
                f=fork();
                if (f==0)
                        break;                                          17
                else
                        c[i]=f;
        }

        if (f==-1){                                                     22
                printf("FORK hata....\n");
                exit(1);
        }
```

Örnek 3 - Ölümcül kilitlenmeyi giderme

```
if (f!=0){
        printf("Anne kaynaklari yaratmaya basliyor....\n");

        semAB=semget(SEMKEY_AB, 2, 0700|IPC_CREAT);
        semctl(semAB, 0, SETVAL,1);                                    5
        semctl(semAB, 1, SETVAL,1);

        semC=semget(SEMKEY_C,1,0700|IPC_CREAT);
        semctl(semC,0,SETVAL,0);
                                                                       10
        sleep(2);

        printf("Anne cocuklari baslatiyor .......\n");

        for (i=0; i<2; i++)                                            15
                kill(c[i],12);

        sleep(5);
        sem_wait(semC,2);
                                                                       20
        printf("Anne: Cocuklarin isi bitti, kaynaklar iade ediliyor...\n");

        semctl(semC,0,IPC_RMID,0);
        semctl(semAB,0,IPC_RMID,0);
                                                                       25
        exit(0);
}
```

Örnek 3 - Ölümcül kilitlenmeyi giderme

```
        else{
                siram=i;
                                                                        3
                printf("cocuk %d anneden haber bekliyor ....\n", siram);

                pause();

                semAB=semget(SEMKEY_AB,2,0);                            8
                semC=semget(SEMKEY_C,1,0);

                printf("cocuk %d anneden haber aldi, basliyor ....\n", siram);

                printf("cocuk %d: sem AB eksiltiyorum.\n", siram);      13
                sem_multi_wait(semAB,1,2);

                printf("cocuk %d: kritik bolgemdeyim.\n", siram);
                sleep(5);
                                                                        18
                sem_multi_signal(semAB,1,2);
                sem_signal(semC,1);

        }
        return 0;                                                       23
}
```