Operativni sistemi
2012

Sistemsko programiranje

Sinhronizacija procesa

Katedra za računarstvo
Elektronski fakultet u Nišu

Prof. dr Dragan Stojanović
mr Aleksandar Stanimirović
mr Bratislav Predić

Sadržaj

- System V semafor
- Primer

Sadržaj

- System V semafor
- Primer

System V semafori

Pojam

- Uslovne promenljive, mutex i POSIX semafori se koriste samo za sinhronizaciju niti. Izuzetak su POSIX semafori koji se mogu koristiti i za sinhronizaciju procesa (tu primenu nećemo obrađivati tokom kursa).
- Za sinhronizaciju procesa se obično koriste IPC System V semafori.
- Sinhronizacija procesa je znatno kompleksnija jer procesi imaju nezavistan adresni prostor a objekti koji se koriste za sinhronizaciju moraju biti vidljivi i kod jednog i kod drugog procesa.
- Objekti koji se koriste za sinhronizaciju procesa se obično kreiraju u jezgru operativnog sistema i postoje nezavisno od procesa u kojima se koriste.

System V semafori

Karakteristike

- System V semafori predstavljaju implementaciju generalnog koncepta semafora.
- Osnovne karakteristike ove implementacije su:
  - System V semafor ne predstavlja samo jednu celobrojnu vrednost već predstavlja niz celobrojnih vrednosti. Broj ovih vrednosti se specificira prilikom kreiranja System V semafora.
  - Kreiranje semafora i njegova inicijalizaciju su dva odvojena procesa. To može da dovede do problema da procesi pristupaju semaforu koji još uvek nije inicijalizovan.
  - Broj System V semafora u sistemu je ograničen tako da svaki semafor treba eksplicitno uništiti kada prestane potreba za njim.
- Sistemski pozivi za rad sa System V semaforima su definisani u zaglavlju <system/sem.h>

System V semafori

Kreiranje semafora

```
#include <sys/sem.h>
int semget (key_t key, int nsems, int flag);
```

Semantika

- Sistemski poziv koji kreira novi System V semafor.
- Prvi argument **key** predstavlja jedinstveni identifikator System V semafora na nivou čitavog sistema. Mora biti poznat svim procesima koji žele da koriste određeni semafor.
- Drugi argument **nsems** specificira broj celobrojnih vrednosti koje semafor sadrži.

System V semafori

Semantika

- Vrednost trećeg argumenta **flag** se definiše kao rezultat OR operacije nad različitim vrednostima i određuje :
  - prava pristupa semaforu (koristićemo vrednost 0666 koja svim korisnicima dodeljuje sve privilegije nad semaforom)
  - mod kreiranja semafora. Neke od mogućih vrednosti su:
    - IPC_CREAT – sistemski poziv kreira semafor ukoliko on već ne postoji u sistemu.
    - IPC_EXCL – koristi se u kombinaciji sa IPC_CREAT i zahteva da semafor sa zadatim identifikatorom ne postoji u sistemu.
- Sistemski poziv vraća celobrojni identifikator (referencu) semafora a u slučaju greške vraća (-1). Dobijeni identifikator (referenca) je važeći samo kod procesa koji je izvršio sistemski poziv i kod njegove dece.
- Ukoliko u sistemu ne postoji semafor sa zadatim identifikatorom a specificiran je flag IPC_CREAT kreira se novi System V semafor.
- Ukoliko u sistemu već postoji semafor sa zadatim identifikatorom (a nije specificiran flag IPC_CREAT | IPC_EXCL) ne kreira se novi semafor već se samo vraća referenca na postojeći System V semafor.

System V semafori

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

int main()
{
   int sid;

   sid = semget((key_t)333, 1,  0666 | IPC_CREAT);

   if (sid < 0)
   {
      perror("Greska prilikom kreiranja semafora 333");
      exit(1);
   }
}
```

Kreira se System V semafor čiji je identifikator 333. Semafor ima samo jednu celobrojnu vrednost.

System V semafori

Operacije nad semaforom

```
#include <sys/sem.h>
int semop (int semid, struct sembuf * semops, int nsops);
```

Semantika

- Prvi argument **semid** predstavlja identifikator (referencu) System V semafora koji je dobijen pozivom funkcije semget.
- Drugi argument **semops** predstavlja niz operacija koje treba izvršiti nad celobrojnim vrednostima System V semafora. Svaka operacija je zadata kao **sembuf** struktura.
- Treći argument **nsops** predstavlja broj elemenata niza sa operacijama.

System V semafor

Semantika

- Struktura sembuf je deklarisana u zaglavlju <sys/sem.h> i ima sledeći izgled:

```
struct sembuf
{
   ushort  sem_num;
   short   sem_op;
   short   sem_flg;
};
```

- Polje **sem_num** specificira indeks celobrojne vrednosti na koju se operacija odnosi.
- Polje **sem_op** definiše operaciju koja se obavlja. Moguće vrednosti su:
  - > 0 – pozitivne vrednosti se dodaju odgovarajućoj celobrojnoj vrednosti semafora odnosno ekvivalent je V operacija
  - < 0 – negativne vrednosti se oduzimaju od odgovarajuće celobrojne vrednosti semafora odnosno ekvivalent je P operacija. Nijedna celobrojna vrednost ne može biti negativna.
  - 0 – proces koji je izvršio sistemski poziv se blokira dok odgovarajuća celobrojna vrednost ne dobije vrednost 0.
- Polje **sem_flag** definiše način na koji se operacija obavlja. Najčešće ima vrednost NULL (kada se prihvata podrazumevani način izvršavanja operacije). Vrednost IPC_NOWAIT sprečava blokiranje procesa koji je izvršio sistemski poziv.
- Sistemski poziv vraća 0 ukoliko su sve operacije uspešno izvršene odnosno -1 ako je došlo do greške prilikom izvršavanja neke od operacija.

System V semafor

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
int main()
{
   int sid;
   struct sembuf sem_lock = { 0, -1, NULL};


   sid = semget((key_t)333, 1,  0666 | IPC_CREAT);


   if (sid < 0)
   {
      perror("Greska prilikom kreiranja semafora 333");
      exit(1);
   }

   ...

   if ((semop(sid, &sem_lock, 1) == -1)
   {
      perror("Greska prilikom P operacija");
      exit(1);
   }
}
```

Kreira se System V semafor čiji je identifikator 333. Semafor ima samo jednu celobrojnu vrednost.

P operacija nad System V semaforom. Operacija se izvršava nad celobrojnom vrednošću semafora na poziciji 0.

System V semafor

Kontrola semafora

```
#include <sys/sem.h>
int semctl (int semid, int semnum, int cmd, union semun arg);
```

Semantika

- Prvi argument **semid** predstavlja identifikator (referencu) System V semafora koji je dobijen pozivom funkcije semget.
- Drugi argument **semnum** predstavlja indeks celobrojne vrednosti na koju se operacija odnosi.
- Treći argument **cmd** definiše operaciju koju treba izvršiti nad semaforom. predstavlja broj elemenata niza sa operacijama.
- Četvrti argument **arg** omogućava definisanje parametara neophodnih za operaciju i definiše se kao unija semun.

System V semafor

Semantika

- Unija semun je deklarisana u zaglavlju <sys/sem.h> i ima sledeći izgled:

```
union semun
{
   int  val;
   struct semid_ds *buf;
   ushort *array;
   struct seminfo * __buf;
   void * __pad;
};
```

- Polje **val** se koristi za zadavanje vrednosti prilikom inicijalizacije celobrojne vrednosti semafora.
- Neke od mogućih operacija koje se mogu izvršiti nad System V semaforom:
  - SETVAL – definiše vrednost odgovarajuće celobrojne vrednosti System V semafora
  - IPC_RMID – brisanje System V semafora iz sistema.

System V semafor

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
int main()
{
   int sid;

   union semun semopts;
   semopts.val = 1;

   sid = semget((key_t)333, 1,  0666 | IPC_CREAT);
   if (sid < 0)
   {
      perror("Greska prilikom kreiranja semafora 333");
      exit(1);
   }
...
   if ((semctl(sid, 0, SETVAL, semopts) == -1)
      perror("Greska prilikom postavljanja vrednosti semafora");
      exit(1);
   }
...

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

Kreira se System V semafor čiji je identifikator 333. Semafor ima samo jednu celobrojnu vrednost.

Celobrojna vrednost semafora na poyiciji 0 se inicijalizuje na vrednost 1.

Semafor se briše iz sistema.

Sadržaj

- System V semafor
- Primer

Primer

```
//INIT.H
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#define MUTEX_KEY 10101
#define EMPTY_KEY 10102
#define FULL_KEY 10103
#define N 10

//INIT.C
#include  "init.h"
int main()
{
   int mutexid, emptyid, fullid;
   union semun semopts;

   //KREIRANJE SEMAFORA
   mutexid = semget((key_t)MUTEX_KEY, 1,  0666 |
IPC_CREAT);
   emptyid = semget((key_t)EMPTY_KEY, 1,  0666 |
IPC_CREAT);
   fullid = semget((key_t)FULL_KEY, 1,  0666 | IPC_CREAT);
```

```
//INICIJALIZACIJA SEMAFORA
    semopts.val = 1;
   semctl(mutexid, 0, SETVAL, semopts);
    semopts.val = N;
   semctl(emptyid, 0, SETVAL, semopts);
    semopts.val = 0;
   semctl(mutexid, 0, SETVAL, semopts);

//POKRETANJE PROIZVODJACA
   if (fork() == 0)
      execl("proizvodjac", "proizvodjac", NULLL);
//POKRETANJE POTROSACA
    if (fork() == 0)
       execl("potrosac", "potrosac", NULLL);

 wait(NULL);
 wait(NULL);
//BRISANJE SEMAFORA
   semctl(mutexid, 0, IPC_RMID, 0);
   semctl(emptyid, 0, IPC_RMID, 0);
   semctl(fullid, 0, IPC_RMID, 0);

}
```

Primer

```
//PROIZVODJAC.C
#include  "init.h"
int main()
{
   int mutexid, emptyid, fullid;
   struct sembuf sem_lock = { 0, -1, NULL};
   struct sembuf sem_unlock = { 0, 1, NULL};

   //PRIBAVLJANJE REFERENCE SEMAFORA
   mutexid = semget((key_t)MUTEX_KEY, 1,  0666);
   emptyid = semget((key_t)EMPTY_KEY, 1,  0666);
   fullid = semget((key_t)FULL_KEY, 1,  0666);

    while(TRUE)
    {
      semop(emptyid, &sem_lock, 1);
      semop(mutexid, &sem_lock, 1);
      //UPISIVANJE U BAFER
     semop(mutexid, &sem_unlock, 1);
     semop(fullid, &sem_unlock, 1);
    }
}
```

```
//POTROSAC.C
#include  "init.h"
int main()
{
   int mutexid, emptyid, fullid;
   struct sembuf sem_lock = { 0, -1, NULL};
   struct sembuf sem_unlock = { 0, 1, NULL};

   //PRIBAVLJANJE REFERENCE SEMAFORA
   mutexid = semget((key_t)MUTEX_KEY, 1,  0666);
   emptyid = semget((key_t)EMPTY_KEY, 1,  0666);
   fullid = semget((key_t)FULL_KEY, 1,  0666);

    while(TRUE)
    {
      semop(fullid, &sem_lock, 1);
      semop(mutexid, &sem_lock, 1);
      //CITANJE IZ BAFERA
     semop(mutexid, &sem_unlock, 1);
     semop(emptyid, &sem_unlock, 1);
    }
}
```