# Լաբորատոր աշխատանք 4

## Միջպրոցեսային հաղորդակցություն (IPC)
## Ընդհանուր հիշողություն (Shared memory)

Ընդհանուր հիշողությունը հնարավորություն է տալիս 2 կամ ավելի պրոցեսներին կիսել ֆիզիկական հիշողության միևնույն տարածքը (segment): Ընդհանուր հիշողության սեգմենտ օգտագործելու համար իրականացվում են հետևյալ քայլերը.

- Կանչել **shmget()** ֆունկցիան՝ ընդհանուր հիշողության նոր սեգմենտ ստեղծելու կամ գոյություն ունեցող սեգմենտի id-ն ստանալու համար,
- Կանչել **shmat()** ֆունկցիան՝ ընդհանուր հիշողության սեգմենտը կցելու համար: Այսինքն՝ սեգմենտը դարձնել կանչող պրոցեսի վիրտուալ հիշողության մաս,
- Կանչել **shmdt()** ֆունկցիան՝ ընդհանուր հիշողության սեգմենտը անջատելու համար: Այս կանչից հետո պրոցեսն այլևս չի կարող հղվել ընդհանուր հիշողությանը: Այս քայլը պարտադիր չէ և տեղի է ունենում ավտոմատ կերպով՝ պրոցեսն ավարտվելիս,
- Կանչել **shmctl()** ֆունկցիան՝ ընդհանուր հիշողության սեգմենտը ջնջելու համար: Սեգմենտը կջնջվի միայն երբ բոլոր կցված պրոցեսներն անջատեն այն: Այս քայլը պետք է իրականացվի միայն մեկ պրոցեսի կողմից:

### Ընդհանուր հիշողության սեգմենտի ստեղծումը

**shmget()** համակարգային կանչը ստեղծում է ընդհանուր հիշողության նոր սեգմենտ կամ ստանում է գոյություն ունեցող սեգմենտի id-ն: Նոր ստեղծված սեգմենտի պարունակությունը սկզբնավորվում է 0-ով: Ֆունկցիայի պրոտոտիպը.

```
#include <sys/shm.h>
int shmget(key_t key, size_t size, int shmflg);
```

- **key** – բանալին է, որը գեներացվում է **IPC_PRIVATE** հաստատունի կամ **ftok()** ֆունկցիայի միջոցով
- **size** – հատկացվող ընդհանուր հիշողության սեգմենտի չափը՝ արտահայտված բայթերով: ՕՀ միջուկը հիշողությունը հատկացնում է համակարգի էջի բազամատիկների չափով, ուստի ֆունկցիային փոխանցված չափը կլորացվում է մինչև հաջորդ էջի բազմապատիկ թիվը: Եթե *shmget()* կանչն օգտագործվում է գոյություն ունեցող սեգմենտի id-ն ստանալու համար, ապա *size* պարամետրն անտեսվում է, բայց այն պետք է փոքր կամ հավասար լինի սեգմենտի չափից,
- **shmflg** – իրականացնում է նույն գործառույթը, ինչ մյուս IPC get կանչերի համար՝ սահմանելով թույլտվության բիթերը: Բացի թույլտվության բիթերից կարող է ընդունել հետևյալ դրոշակները.

- **IPC_CREAT** – եթե տրված բանալիով սեգմենտ գոյություն չունի, ապա ստեղծում է նորը,
- **IPC_EXCL** – եթե նշված է *IPC_CREAT* դրոշակը, և տրված բանալիով սեգմենտ արդեն գոյություն ունի, ապա կանչը ձախողվում է *EEXIST* error-ով:

Ֆունկցիայի կիրառման օրինակը ներկայացված է shm_create.c ծրագրում, որը որպես հրամանային տողի արգումենտ ընդունում է ստեղծվող սեգմենտի չափը՝ բայթերով: Ծրագիրը կատարելու համար անհրաժեշտ քայլերն են.

```
gcc shm_create.c -o create
./create 1000
```

## Ընդհանուր հիշողության սեգմենտի կցումը

**shmat()** համակարգային կանչը կցում է *shmid* իդենտիֆիկատորով սահմանված ընդհանուր հիշողության սեգմենտը պրոցեսի վիրտուալ հիշողության տարածքին: Ֆունկցիայի պրոտոտիպը.

```
#include <sys/shm.h>
void *shmat(int shmid, const void *shmaddr, int shmflg);
```

**shmaddr** արգումենտը և **shmflg** արգումենտի **SHM_RND** բիթը ղեկավարում են կցման գործընթացը.

- Եթե *shmaddr == NULL*, ապա սեգմենտը կցվում է ՕՀ միջուկի կողմից ընտրված հասցեով: Սա սեգմենտի կցման նախընտրելի մեթոդն է:
- Եթե *shmaddr != NULL* և *SHM_RND* բիթը սահմանված չէ, ապա սեգմենտը կցվում է shmaddr-ում սահմանված հասցեով: Հասցեն պետք է լինի համակարգային էջի չափի բազմապատիկ, հակառակ դեպքում տեղի կունենա **EINVAL** error:
- Եթե *shmaddr != NULL* և *SHM_RND* բիթը սահմանված է, ապա սեգմենտը կցվում է shmaddr-ում սահմանված հասցեով, որը կլորացվում է դեպի ներքև մինչև *SHMLBA (shared memory low boundary address)* հաստատունի մոտակա բազմապատիկը:

Ֆունկցիայի կիրառման օրինակը ներկայացված է shm_attach.c ծրագրում, որը որպես հրամանային տողի արգումենտներ ընդունում է այն ընդհանուր հիշողության սեգմենտների id-ները, որոնք պետք է կցել պրոցեսին: Ծրագիրը կատարելու համար անհարժեշտ քայլերն են.

```
gcc shm_attach.c -o attach
./attach 0
```

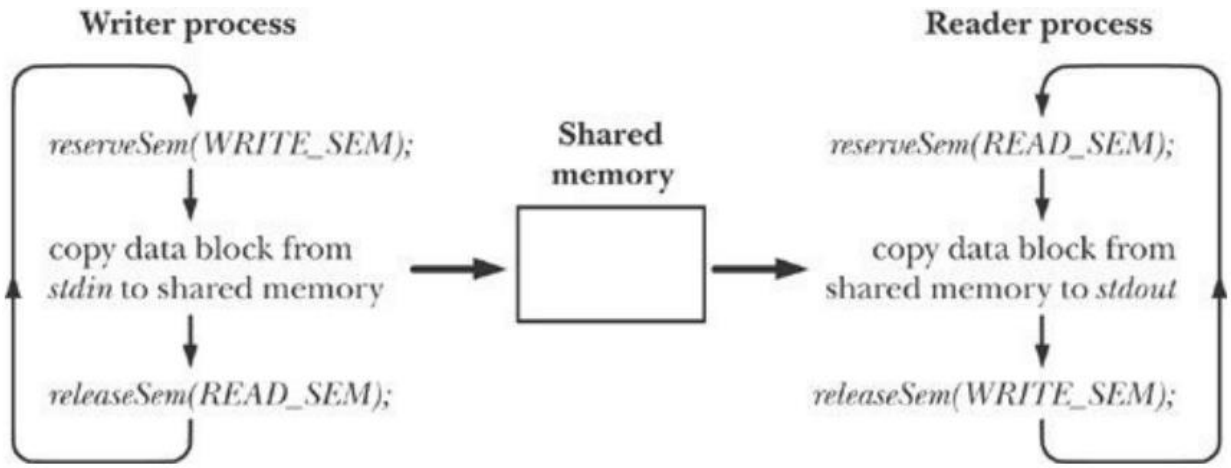
## **writer** և **reader ծրագրերը**

Writer ծրագիրը կարդում է տվյալները *stdin* հոսքից և դրանք պատճենում դրանք ընդհանուր հիշողության սեգմենտի մեջ: Reader ծրագիրը կարդում է տվյալները ընդհանուր հիշողության սեգմենտից և դրանք տեղադրում *stdout* հոսքի մեջ:

Կիրառվում է երկուական սեմաֆորների պրոտոկոլը (initSemAvailable(), initSemInUse(), reserveSem(), releaseSem() ֆունկցիաները) համոզվելու համար, որ.

- ժամանակի ցանկացած պահի միայն մեկ պրոցես հասանելիություն ունի ընդհանուր հիշողության սեգմենտին,
- պրոցեսները սեգմենտի հետ աշխատում են հերթով (1-ին պրոցեսը գրում է տվյալները, հետո 2-րդը կարդում է, հետո կրկին 1-ինը գրում է, և այսպես շարունակ):

Ծրագրում օգտագործվում են 2 սեմաֆորներ: Writer պրոցեսը սկզբնավորում է 2 սեմաֆորներն այնպես, որ writer պրոցեսի սեմաֆորը սկզբնապես ազատ է, իսկ reader պրոցեսի սեմաֆորը զբաղված է: Ծրագրի աշխատանքը պատկերված է Նկ. 1-ում:




Նկ. 1 Ընդհանուր հիշողության կիրառումը 2 պրոցեսների կողմից

Ծրագրերը կատարելու համար անհրաժեշտ քայլերն են.

```
gcc shm_writer.c binary_sems.c -o writer
./writer

gcc shm_reader.c binary_sems.c -o reader
./reader
```

## Առաջադրանքներ

1. Ստեղծել ընդհանուր հիշողության 2 սեգմենտներ՝ 100KB և 200KB ծավալով:
2. Կցել ստեղծված սեգմենտները պրոցեսի վիրտուալ հիշողության տարածքին: Մինչ պրոցեսը գտնվում է սպասման վիճակում, կանգնեցնել այն (Ctrl + Z): Բացել /proc/{pid}/maps ֆայլը և ցույց տալ ընդհանուր հիշողության սեգմենտների հասցեները:
3. Ջնջել ստեղծված սեգմենտները՝ shm_rm ծրագրի միջոցով:
4. Տարբեր հրամանային տողերով կատարել shm_writer և shm_reader ծրագրերը: shm_writer ծրագրի հրամանային տողում գրել կամայական հաղորդագրություն(ներ) և մուտքագրել: Ծրագրին ուղարկել EOF սիմվոլ (Ctrl + D) և բացատրել 2 ծրագրերի կողմից ցուցադրված հաղորդագրությունները: