KWANGWOON UNIVERSITY

2022년 2학기 **운영체제실습** 11주차

Shared Memory

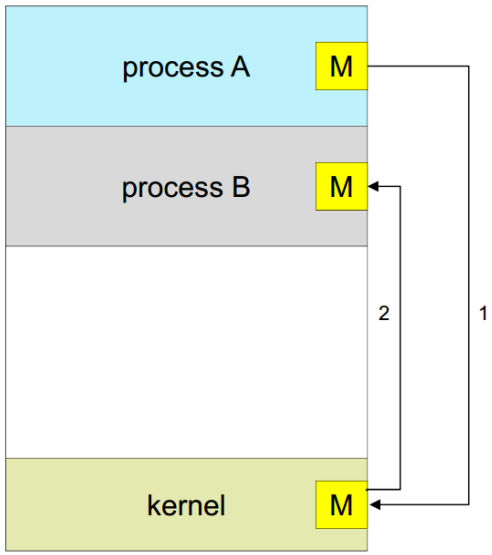
System Software Laboratory
School of Computer and Information Engineering
Kwangwoon Univ.

Contents

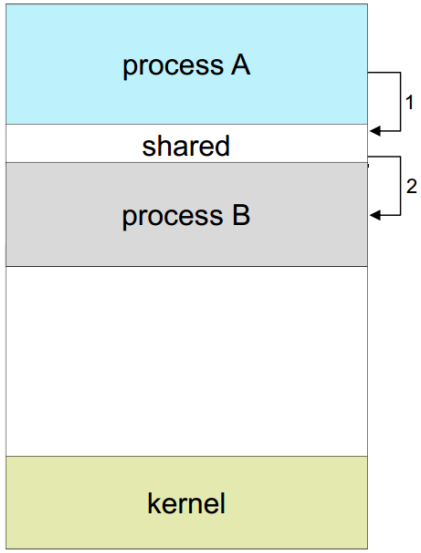
- IPC
- Shared Memory
- Lab. 1

IPC

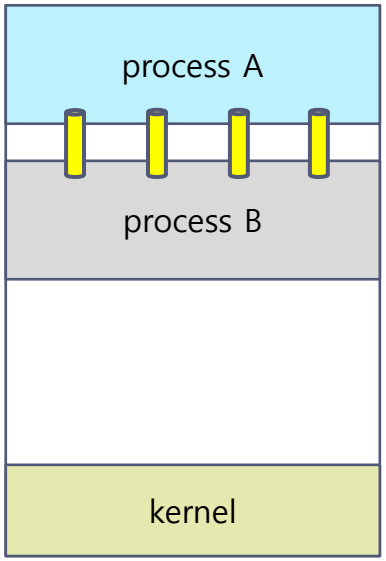
- **IPC 란?**
  - Inter-Process Communication의 약자.
  - 다중 thread 또는 process 간 데이터를 주고 받기 위한 방법이나 경로를 의미
- **종류**
  - Message passing, socket, pipe, shared memory, memory-mapped file(mmap() in Linux), …



Message passing



Shared memory



Pipe

Shared Memory (1/4)

- **Shared Memory 란?**
  - 메모리 상의 특정 공간에 여러 프로그램이 동시에 접근하여 데이터를 주고 받을 수 있는 IPC 기법 중 하나

- **특징**
  - IPC 기법 중 가장 빠른 수행 속도를 보임
  - Message Passing IPC들과 달리 불필요한 memory copy가 발생하지 않음
  - Shared memory에는 한 번에 하나의 프로세스가 접근하고 있음을 보증해야 함

  - 프로세스에 의해 shared memory가 요청되면 커널은 이를 위한 공간 할당 후 내부 자료구조(struct shmid_ds) 를 이용해 직접 관리한다.

Shared Memory (2/4)

Related functions

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

int  shmget (key_t key, int size, int shmflg)
void *shmat (int shmid, const void* shmaddr, int shmflg)
int  shmdt  (const void *shmaddr)
int  shmctl (int shmid, int cmd, struct shmid_ds* buf)
```

- **int shmget**(key_t key, int size, int shmflg)
  - 커널에 shared memory segment를 위한 공간 요청
  - key_t **key** – 공유 메모리 접근을 위한 key 값
  - int **size** – shared memory의 최소 크기(PAGE_SIZE의 배수)
  - int **shmflg** – 접근권한, 생성 방식
  - Shared memory segment를 위한 integer identifier를 return

Shared Memory (3/4)

- **Related functions (cont'd)**
  - **void *shmat**(int shmid, const void* shmaddr, int shmflg)
    - 현재의 프로세스가 생성된 shared memory를 사용할 수 있도록 id 값을 이용하여 덧붙임(attach) 작업 수행
    - int shmid – shmget()을 이용해 얻은 identifier
    - const void* shmaddr – memory가 붙을 주소 명시 (0: 커널이 명시)
    - int shmflg – 접근 권한
    - shared memory segment로 연결 된 pointer를 return
  - **int shmdt**(const void *shmaddr)
    - 프로세스가 더 이상 shared memory를 사용 할 필요가 없을 경우 프로세스와 shared memory를 분리(detach)하기 위해 사용
    - const void *shmaddr – shmat()에서 얻은 pointer

Shared Memory (4/4)

- **Related functions (cont'd)**
  - **int shmctl(**int shmid, int cmd, struct shmid_ds* buf**)**
    - shared memory의 권한 변경, 삭제, 잠금 설정 등을 위한 제어 함수
    - int shmid – shared memory segment identifier
    - int cmd – control command
      - IPC_STAT : get information of shared memory
      - IPC_SET : modification of permission
      - IPC_RMID : remove shared memory
    - struct shmid_ds* **buf** – shared memory 정보 획득을 위해 사용

Lab. 1 (1/8)

- **test1.c**

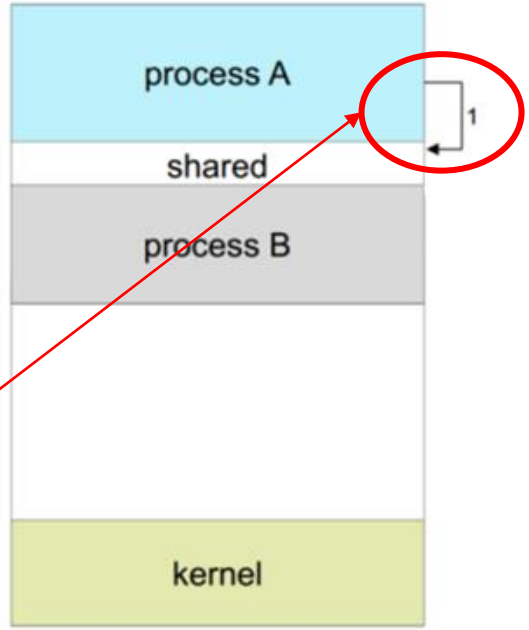
```c
#include <stdio.h>
#include <sys/shm.h>
#include <sys/stat.h>
#include <sys/user.h>


int main()
{
        int segment_id;
        char *shared_memory;
        int size =  PAGE_SIZE;

        printf("create shared memory\n");
        segment_id = shmget(1234, size, IPC_CREAT | S_IRUSR | S_IWUSR);
        shared_memory = (char*) shmat(segment_id, NULL, 0);
        sprintf(shared_memory, "hello shared memoey!");
        shmdt(shared_memory);
        printf("setting shared memory\n");

        return 0;

}

```

key에 해당하는 공유 메모리가 없다면 새로 생성

Lab. 1 (2/8)

- **test1.c**

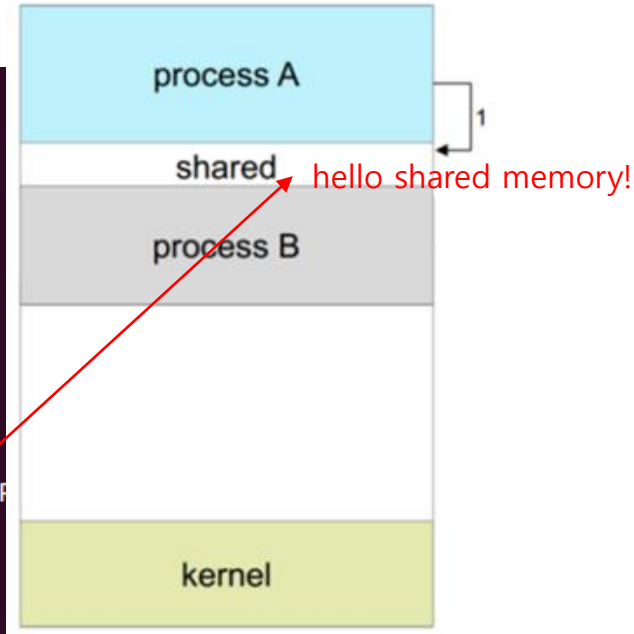
```c
#include <stdio.h>
#include <sys/shm.h>
#include <sys/stat.h>
#include <sys/user.h>


int main()
{
        int segment_id;
        char *shared_memory;
        int size =  PAGE_SIZE;

        printf("create shared memory\n");
        segment_id = shmget(1234, size, IPC_CREAT | S_IRUSR | S_IWUSR);
        shared_memory = (char*) shmat(segment_id, NULL, 0);
        sprintf(shared_memory, "hello shared memoey!");
        shmdt(shared_memory);
        printf("setting shared memory\n");

        return 0;

}

```



Lab. 1 (3/8)

- **test1.c**

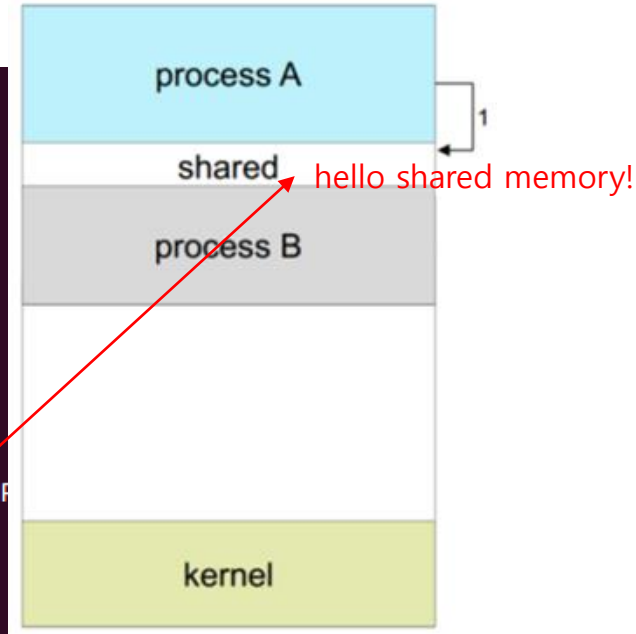
```c
#include <stdio.h>
#include <sys/shm.h>
#include <sys/stat.h>
#include <sys/user.h>


int main()
{
        int segment_id;
        char *shared_memory;
        int size =  PAGE_SIZE;

        printf("create shared memory\n");
        segment_id = shmget(1234, size, IPC_CREAT | S_IRUSR | S_IWUSR
        shared_memory = (char*) shmat(segment_id, NULL, 0);
        sprintf(shared_memory, "hello shared memoey!");
        shmdt(shared_memory);
        printf("setting shared memory\n");

        return 0;

}

```



Lab. 1 (4/8)

- **test1.c**

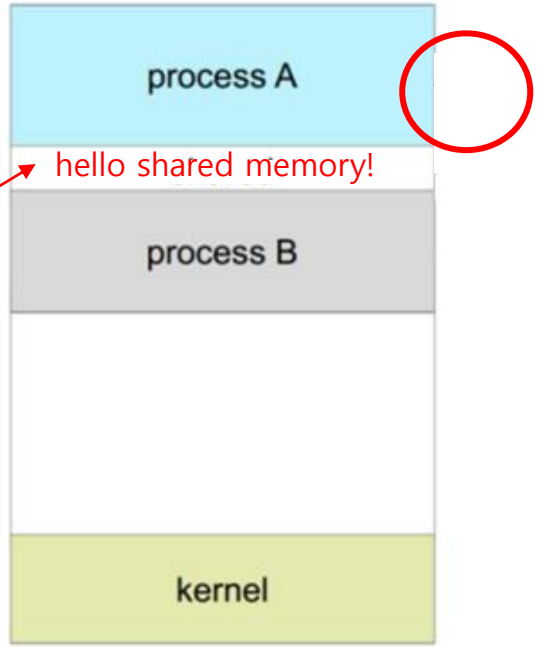
```c
#include <stdio.h>
#include <sys/shm.h>
#include <sys/stat.h>
#include <sys/user.h>


int main()
{
        int segment_id;
        char *shared_memory;
        int size =  PAGE_SIZE;

        printf("create shared memory\n");
        segment_id = shmget(1234, size, IPC_CREAT | S_IRUSR | S_IWUSR
        shared_memory = (char*) shmat(segment_id, NULL, 0);
        sprintf(shared_memory, "hello shared memoey!");
        shmdt(shared_memory);
        printf("setting shared memory\n");

        return 0;

}

```



Lab. 1 (5/8)

- **test1.c**

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


int main()
{
        int segment_id;
        char *shared_memory;
        int size =  PAGE_SIZE;

        printf("create shared memory\n");
        segment_id = shmget(1234, size, IPC_CREAT | S_IRUSR | S_IWUS
        shared_memory = (char*) shmat(segment_id, NULL, 0);
        sprintf(shared_memory, "hello shared memoey!");
        shmdt(shared_memory);
        printf("setting shared memory\n");

        return 0;

}

```



Lab. 1 (6/8)

- **test2.c**

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

int main()
{
        int segment_id;
        char *shared_memory;
        int size = PAGE_SIZE;

        segment_id = shmget(1234, size, 0);
        shared_memory = (char*)shmat(segment_id, NULL, 0);
        printf("contents of shared memory : %s\n", shared_memory);
        shmdt(shared_memory);
        shmctl(segment_id, IPC_RMID, NULL);
        printf("remove shared memory\n");


        return 0;
}
```

Lab. 1 (7/8)

- **Makefile**

```
1  default: test1 test2
2
3  test1: test1.o
4
5  test2: test2.o
6
7  clean:
8          $(RM) test1 test2 test1.o test2.o
9
10
```

Lab. 1 (8/8)

- **Result**

```
sslab@ubuntu:~/memory$ make          컴파일
cc   test1.o   -o test1
cc   test2.o   -o test2
sslab@ubuntu:~/memory$ ipcs -m       공유 메모리 확인

------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch     status

sslab@ubuntu:~/memory$ ./test1       공유 메모리 생성 & 값 저장
create shared memory
setting shared memory
sslab@ubuntu:~/memory$ ipcs -m       공유 메모리 확인

------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch     status
0x000004d2 1474569    sslab      600        4096       0

sslab@ubuntu:~/memory$ ./test2       공유 메모리 값 출력 & 제거
contents of shared memory : hello shared memoey!
remove shared memory
sslab@ubuntu:~/memory$ ipcs -m       공유 메모리 확인

------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch     status

sslab@ubuntu:~/memory$ ./test2       제거 된 공유 메모리 접근 에러
Segmentation fault (core dumped)
sslab@ubuntu:~/memory$
```

Assignment 4

- 제출 기한: 2022. 11.10(목) ~ 2022.12.1(목) 23:59:59
- Delay 없음
- 업로드 양식에 어긋날 경우 감점 처리
- Hardcopy 제출하지 않음