UNIX 시스템 프로그래밍

- 8장. 메모리 매핑

memory mapping

- memory mapping : file을 프로세스의 memory에 mapping

- 사용 방법 :

```
#include<sys/mman.h>
void *mmap(void *addr, size_t len, int prot, int flags,
 int fildes, off_t off);
```

memory mapping (2)

- fildes가 가리키는 파일에서 off로 지정한 위치부터 len만큼의 데이터를 읽어 addr이 가리키는 메모리 공간에 매핑한다.
  - addr : 매핑할 메모리 주소
  - len : 메모리 공간의 크기
  - prot : 보호모드
  - flags : 매핑된 데이터의 처리 방법을 지정하는 상수
  - fildes : file descriptor
  - off : file offset (페이지 크기의 배수)

memory mapping (3)

- prot 인자
  - PROT_READ : 읽기 허용
  - PROT_WRITE : 쓰기 허용
- flag 인자
  - MAP_SHARED : 변경 내용 공유
  - MAP_PRIVATE : 변경 내용 공유하지 않음
- 페이지 단위 메모리 매핑 실행
- 매핑된 영역을 벗어나면, SIGBUS, SIGSEGV 발생

memory mapping 예제

```
int main(void){
        int fd;
        char *addr;

       fd=open("data", O_RDWR);
       addr=mmap(NULL, 512, PROT_READ|PROT_WRITE,
                MAP_SHARED, fd, 0);

        printf("%s\n", addr);
}
```

memory mapping 해제

- 해제 방법

```
#include<sys/mman.h>
int munmap(void *addr,  size_t len);
```

- addr : mmap의 반환값

보호 모드 변경

- 변경 방법

```
#include<sys/mman.h>
int mprotect(void *addr, size_t len, int prot);
```

파일 크기 변경

- 변경 방법

```
#include<unistd.h>
int truncate(const char *path, off_t len);

#include<unistd.h>
int ftruncate(int fildes, off_t len);
```

매핑된 메모리 동기화

- 메모리 동기화 방법

  ```
  #include<sys/mman.h>
  int msync(void *addr, size_t len, int flags);
  ```

- flags 인자
  - MS_ASYNC : 비동기적 쓰기 작업
  - MS_SYNC : 동기적 쓰기 작업