운영체제 및 실습
- File I/O -

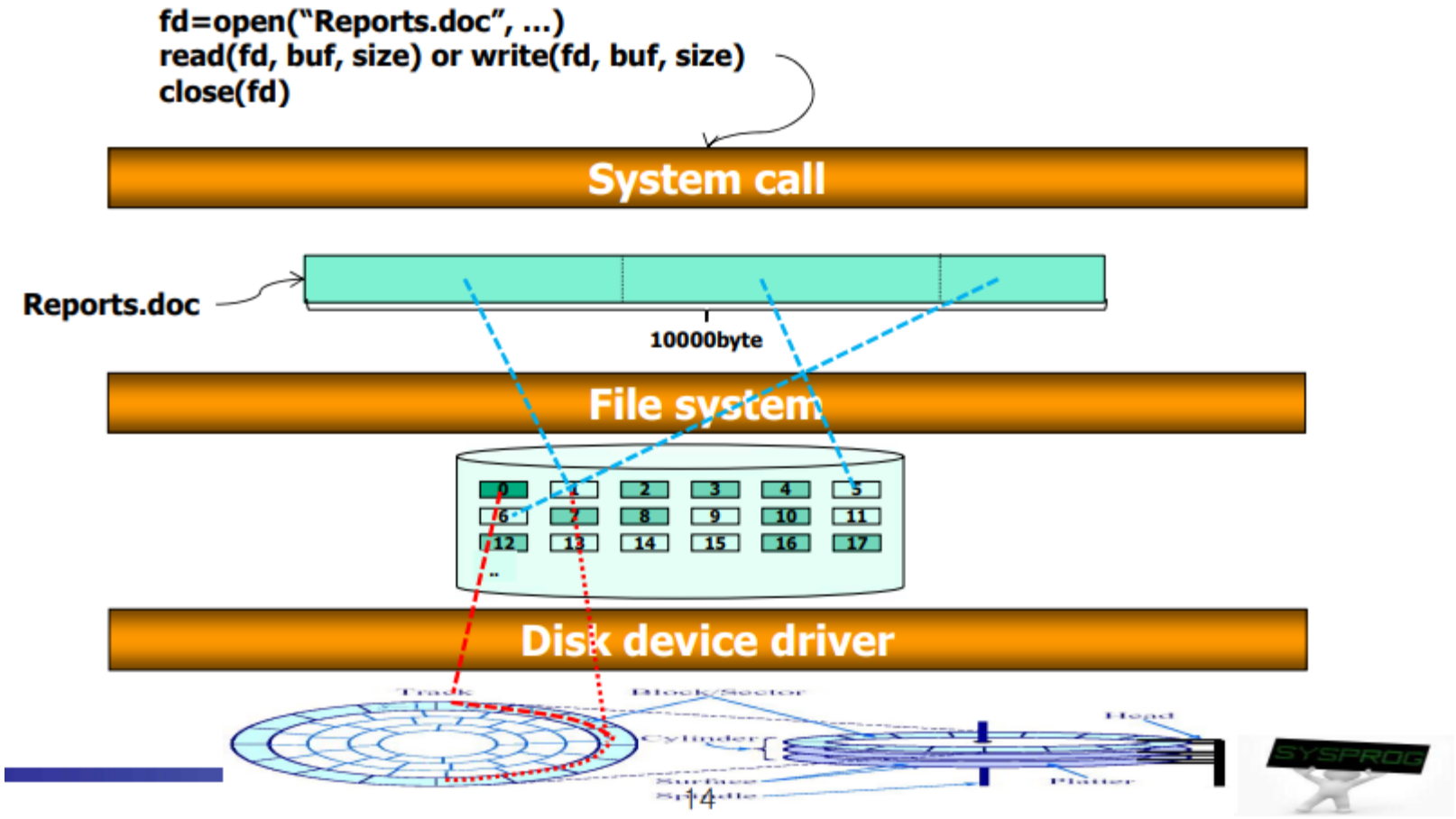
한예진, 김수창, 이승준

Dankook University

Index

- 실습1: open(),read()
- 실습2: write()
- 실습3: mycat
- 실습4: create new file
- 실습5: lseek

요약: 파일입출력 과정


fd=open("Reports.doc", ...)
read(fd, buf, size) or write(fd, buf, size)
close(fd)
System call
Reports.doc
10000byte
File system
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
Disk device driver
Track
Block/Sector
Cylinder
Head
Surface
Platter
SYSPROG

실습1: open(),read()


```
[centos@localhost ~]$ vim open.c
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <errno.h>
#define MAX_BUF 5
char fname[]="alphabet.txt";

int main(){
        int fd,size;
        char buf[MAX_BUF];

        fd = open(fname,O_RDONLY);
        if(fd<0){
                printf("Can't open %sfile with errno %d\n",fname,errno);
                exit(-1);
        }
        size = read(fd,buf,MAX_BUF);
        if(size < 0){
                printf("Can't read from file %s,size= %d\n",fname,size);
        }
        else
                printf("size of read data is %d\n",size);
        close(fd);
}
```

실습1: open(),read()


```
[centos@localhost ~]$ gcc -o open open.c
[centos@localhost ~]$ ls
Desktop    Downloads  open      Pictures  Templates
Documents  Music      open.c    Public    Videos
[centos@localhost ~]$ ./open
Can't open alphabet.txtfile with errno 2
```

오류코드: 파일 및 디렉토리 X

실습1: open(),read()


```
[centos@localhost ~]$ vim alphabet.txt
```

실습1: open(),read()


```
[centos@localhost ~]$ ./open
size of read data is 5
```

실습1: open(),read()

int open(const char *pathname, int flags, [mode_t mode])

- ✓ pathname : absolute path or relative path
- ✓ flags (see: /usr/include/asm/fcntl.h or Chapter 4.3 in the LPI)
  - O_RDONLY, O_WRONLY, O_RDWR
  - O_CREAT, O_EXCL
  - O_TRUNC, O_APPEND
  - O_NONBLOCK, O_SYNC
  - …
- ✓ mode
  - meaningful with the O_CREAT flag
  - file access mode (S_IRUSR, S_IWUSR, S_IXUSR, S_IRGRP, …, S_IROTH, …)
- ✓ return value
  - file descriptor if success
  - -1 if fail

int read(int fd, char *buf, int size) // same as the write(fd, buf, size)

- ✓ fd: file descriptor (return value of open())
- ✓ buf: memory space for keeping data
- ✓ size: request size
- ✓ return value
  - read size
  - -1 if fail

실습2: write()


```
[centos@localhost ~]$ vim write.c
```

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <errno.h>
#define MAX_BUF 5
char fname[]="alphabet.txt";

int main(){
        int fd,read_size,write_size;
        char buf[MAX_BUF];

        fd = open(fname,O_RDONLY);
        if(fd<0){
                printf("Can't open %sfile with errno %d\n",fname,errno);
                exit(-1);
        }
        read_size = read(fd,buf,MAX_BUF);
        if(read_size < 0){
                printf("Can't read from file %s,size= %d\n",fname,write_size);
        }
        write_size = write(STDOUT_FILENO,buf,MAX_BUF);
        close(fd);
}
~
~
~
"write.c" 24L, 506B                                   22,33-40      All
```

```
#define STDIN_FILENO    0   // Standard input
#define STDOUT_FILENO   1   // Standard output
#define STDERR_FILENO   2   // Standard error
```

실습2: write()


```
[centos@localhost ~]$ gcc -o write write.c
[centos@localhost ~]$ ./write
abcde[centos@localhost ~]$
```

실습3: mycat


```
[centos@localhost ~]$ vim mycat.c
```

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <errno.h>
#define MAX_BUF 64

int main(int argc, char *argv[]){
        int fd,read_size,write_size;
        char buf[MAX_BUF];

        if(argc != 2){
                printf("USAGE: %S file_name\n",argv[0]);
                exit(-1);
        }
        fd = open(argv[1], O_RDONLY);
        if(fd<0){
                //open error handling
        }
        while(1){
                read_size=read(fd,buf,MAX_BUF);
                if(read_size == 0)
                        break;
                write_size=write(STDOUT_FILENO,buf,read_size);
        }
        close(fd);
}
"mycat.c" 27L, 489B                                                    18,4-18       All
```

실습3: mycat


```
[centos@localhost ~]$ gcc -o mycat mycat.c
[centos@localhost ~]$ ./mycat alphabet.txt
abcdefgh
```

실습4: create new file


```
[centos@localhost ~]$ vim creat.c
```

```c
#include <fcntl.h>
#include <errno.h>
#define MAX_BUF 64
char fname[]="newfile.txt";
char dummy_data[]="abcdefg\n";


int main(){
        int fd,read_size,write_size;
        char buf[MAX_BUF];

        fd = open(fname,O_RDWR | O_CREAT | O_EXCL, 0664);
        if(fd<0){
                printf("Can't create %s file with errno %d\n",fname,errno);
                exit(1);
        }
        write_size=write(fd,dummy_data,sizeof(dummy_data));
        printf("write_size = %d\n",write_size);
        close(fd);

        fd=open(fname,O_RDONLY);
        read_size = read(fd,buf,MAX_BUF);
        printf("read_size = %d\n",read_size);
        write_size= write(STDOUT_FILENO,buf,read_size);

        close(fd);
}
```

```
                                                              30,1          Bot
```

실습4: create new file


```
[centos@localhost ~]$ gcc -o creat creat.c
[centos@localhost ~]$ ./creat
Can't create newfile.txt file with errno 17
[centos@localhost ~]$ rm -rf newfile.txt
[centos@localhost ~]$ ./creat
write_size = 9
read_size = 9
abcdefg
```

실습5: lseek()

- Using lseek()

off_t lseek(int fd, off_t offset, int whence)

- fd : file descriptor
- offset : offset position
- whence (/usr/include/unistd.h)
  - SEEK_SET : New offset is set to offset bytes.
  - SEEK_CUR: New offset is set to its current location plus offset bytes.
  - SEEK_END: New offset is set to the size of the file plus offset bytes
- return value
  - new offset if success
  - -1 if fail

Negative value is allowed

실습5: lseek()


```
[centos@localhost ~]$ vim lseek.c
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <errno.h>
#define MAX_BUF 64
char fname[]="newfile.txt";
char dummy_data[]="abcdefg\n";

int main(){
        int fd,read_size,write_size,new_offset;
        char buf[MAX_BUF];

        fd = open(fname,O_RDWR | O_CREAT | O_EXCL, 0664);
        if(fd<0){
                printf("Can't create %s file with errno %d\n",fname,errno);
                exit(1);
        }
        write_size=write(fd,dummy_data,sizeof(dummy_data));
        close(fd);

        fd=open(fname,O_RDONLY);
        new_offset = lseek(fd,3,SEEK_SET);
        read_size = read(fd,buf,MAX_BUF);
        printf("read_size = %d\n",read_size);
        write_size = write(STDOUT_FILENO,buf,read_size);

        close(fd);
}
```

```
                                                              9,0-1         All
```

Read and write

O_CREAT 또는create()

파일이 존재하는지 확인

접근 권한

실습5: lseek()


```
[centos@localhost ~]$ gcc -o lseek lseek.c
[centos@localhost ~]$ ./lseek
Can't create newfile.txt file with errno 17
[centos@localhost ~]$ rm -rf newfile.txt
[centos@localhost ~]$ ./lseek
read_size = 6
defg
[centos@localhost ~]$
```

그 이외 파일 입출력 관련 시스템콜

- creat()
- mkdir(), readdir(), rmdir()
- pipe()
- mknod()
- link(),unlink()
- dup(),dup2()
- stat(),fstat()
- chmod(), fchmod()
- Ioctl(), fcntl()
- Sync(), fsync()