File Descriptor

- **File descriptor**
  - open() returns a fd, an integer value
  - 열린 파일을 나타내는 번호
  - used in subsequent I/O operations on that file
  - close(fd) closes that file described by fd
  - all of a process's open files are automatically closed when it terminates

File Descriptor

- **file descriptor : 0 ~ 19**

Value	Meaning
0	standard input
1	standard output
2	standard error
3 .. 19	fds for users

Open File Table

- 파일 테이블 (file table)
  - 커널 자료구조
  - 열려진 모든 파일 목록
  - 열려진 파일 → 파일 테이블의 항목

열려진 모든 `file구조체`들을 원소로 가지고있는 linked list 자료구조!

- 파일 테이블 항목 (file table entry)
  - 파일 상태 플래그 `f_op` (read, write, append, sync, nonblocking,...)
  - 파일의 현재 위치 (current file offset) `f_pos`
  - i-node에 대한 포인터 `f_dentry`

한 개의 `file구조체`가 가지고 있는 정보들!

메모리에 생성되는 것

Active i-node table

- Active i-node table
  - 커널 내의 자료 구조
  - Open 된 파일들의 i-node를 저장하는 테이블

열린 파일들의 `inode구조체`를 원소로 가지고 있는 linked list 자료구조!

- i-node
  - 하드 디스크에 저장되어 있는 파일에 대한 자료구조
  - 한 파일에 하나의 i-node
  - 하나의 파일에 대한 정보 저장
    - 소유자, 크기
    - 파일이 위치한 장치
    - 파일 내용 디스크 블럭에 대한 포인터

해당 파일의 block 내 데이터가 저장된다.

Example <read() 함수 작동 원리>

- read(fd, buf1, 100);
  - copy the first block from disk into an buffer
  - copy the first 100 bytes from the buffer into buf1
  - offset <- 100
- read(fd, buf2, 200);
  - copy the next 200 bytes from the buffer into buf2
- read(fd, buf3, 5000);
  - copy the remaning (3796bytes) from the buffer to buf3
  - copy the second block from disk into an buffer
  - copy the remainig(1204 bytes) from the buffer into buf3

buf에는 offset을 저장

buffer에는 현재 first block 내 데이터들이 저장되어 있음.

buffer에는 현재 second block 내 데이터들이 저장되어 있음.

<여기서 offset의 값은 f_pos 임!!>

dup(), dup2()

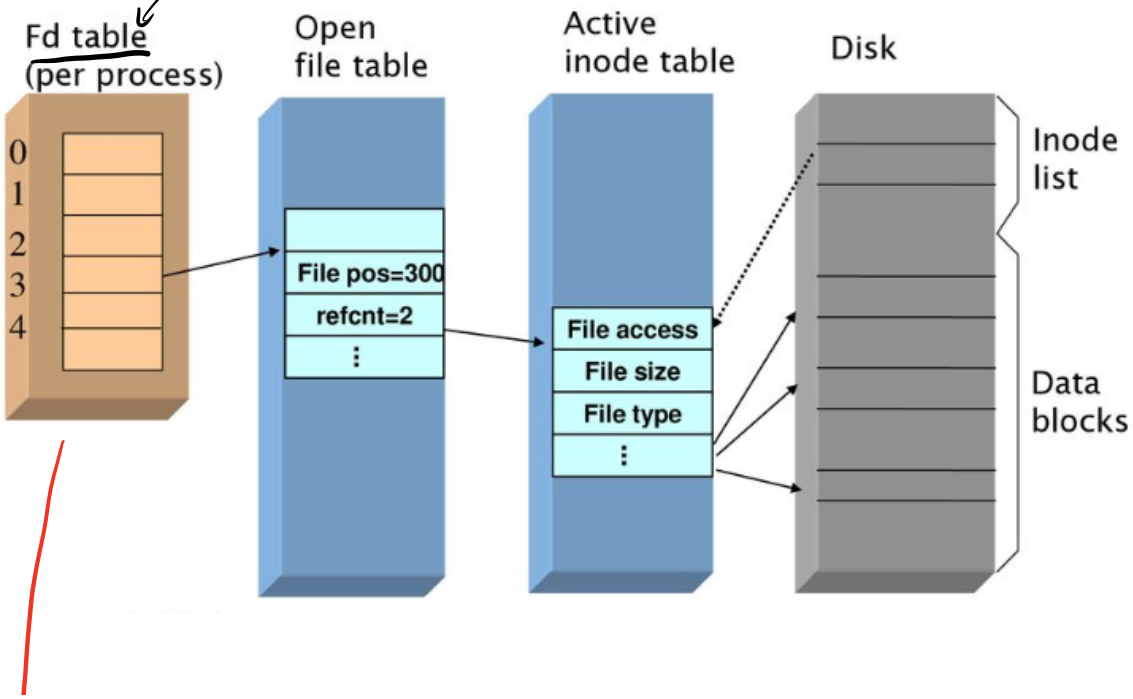
```
#include <unistd.h>
int dup (int fd);
int dup2 (int fd, int fd2);
```

- 사용 중인 파일 디스크립터의 복사본을 만듦
  - dup()는 새 파일 디스크립터 번호가 할당됨
  - dup2()는 *fd2* 를 사용
- 리턴 값
  - 성공하면 복사된 새 파일 디스크립터, 실패하면 -1
  - dup() 함수는 할당 가능한 가장 작은 번호를 리턴한다.
  - dup2() 함수는 *fd2* 를 리턴한다.

dup 구현

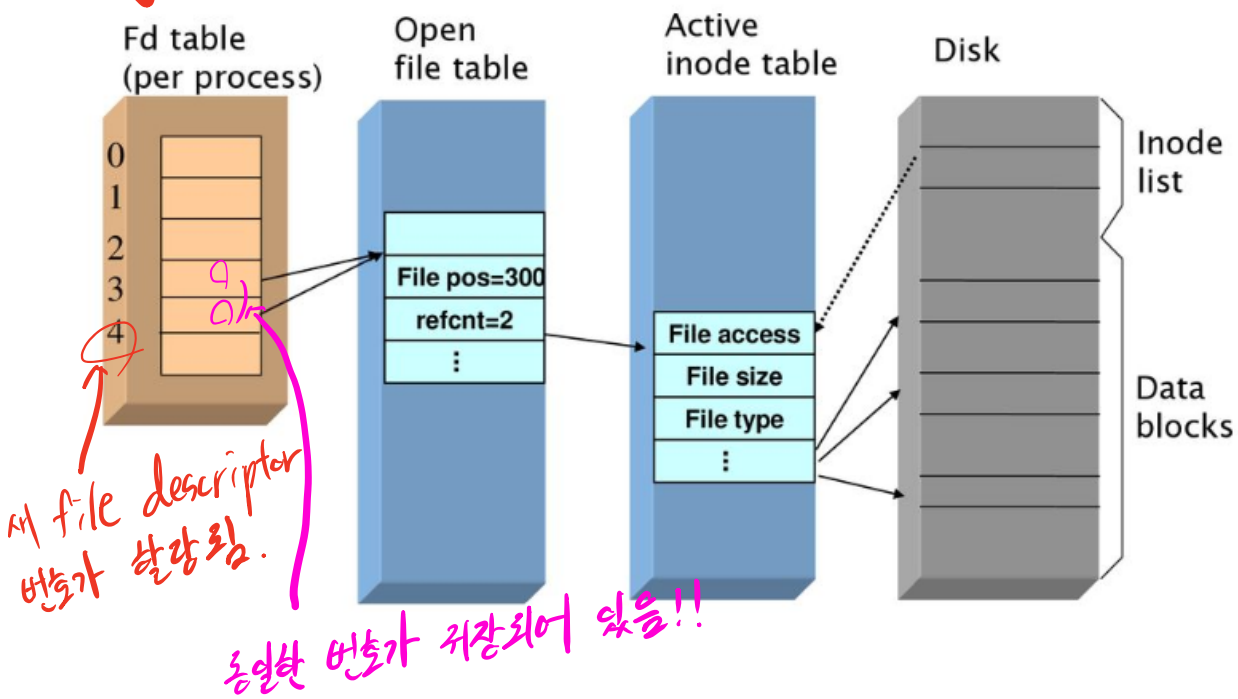
(= file descriptor table)

- fd2 = dup(3);

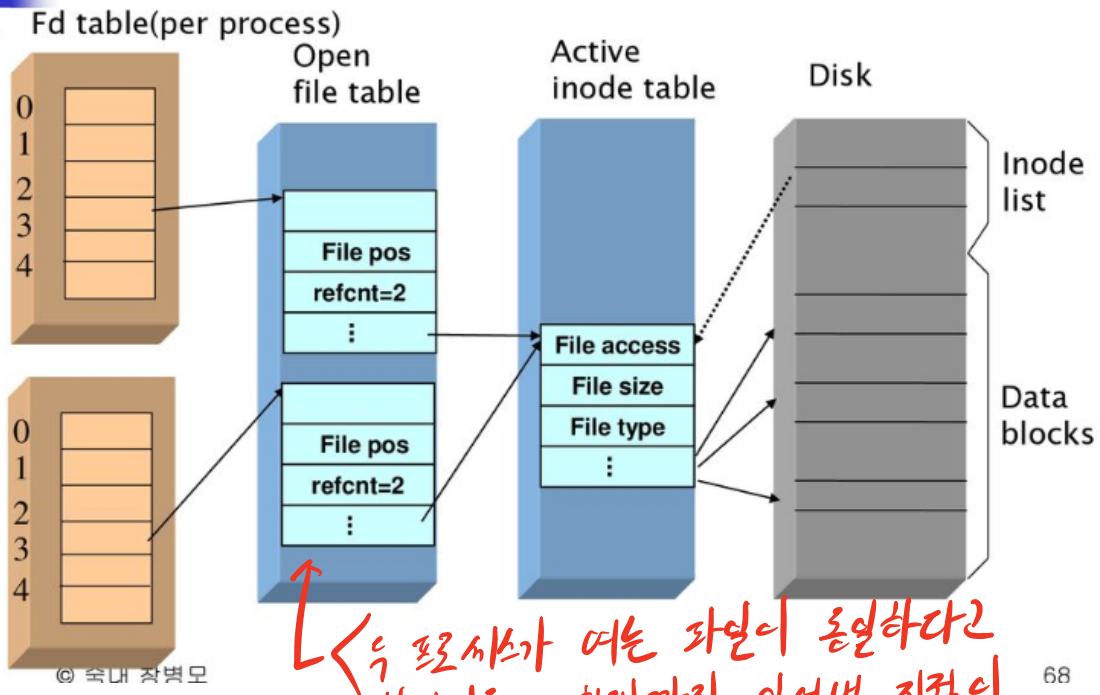


dup 구현

- fd2 = dup(3);



두 프로세스에서 같은 파일 open()



<두 프로세스가 여는 파일이 동일하다고 할지라도 현재까지 읽어낸 지점이 다르기 때문에 (즉, f_pos가 다르기 때문에), 동일한 inode를 가르키는 'file3' 구조체 두 개가 생성되어야 한다.>

fork()