11주차 1차시 파이프 개념

【학습목표】

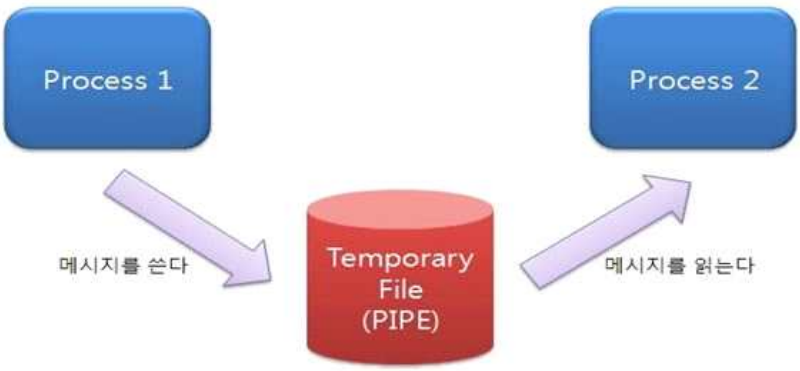
1. 파이프의 기본 개념을 설명할 수 있다.
2. PIPE와 FIFO를 이해한다

학습내용1 : 파이프 개념 및 종류

1. 파이프

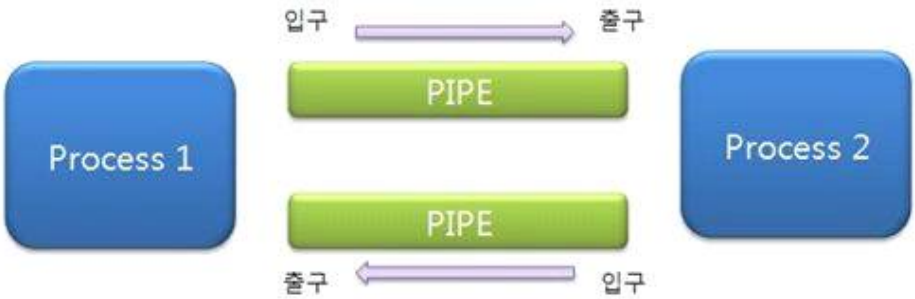
1) 개념
여러 개의 프로세스가 공통으로 사용하는 임시공간
임시 공간은 실제로 파일 시스템에 생성되는 임시 파일.
하나의 프로세스가 파이프에 쓰게 되면 다른 프로세스는 그 파이프에서 읽는 방식으로 쓰게 됩니다.
파이프는 시스템 내부에서 관리하는 임시 파일을 이용하므로, 다른 IPC 기법 중 하나인 SIGNAL 과는 다르게 대용량의 메시지도 전송이 가능합니다.

2) 흐름도



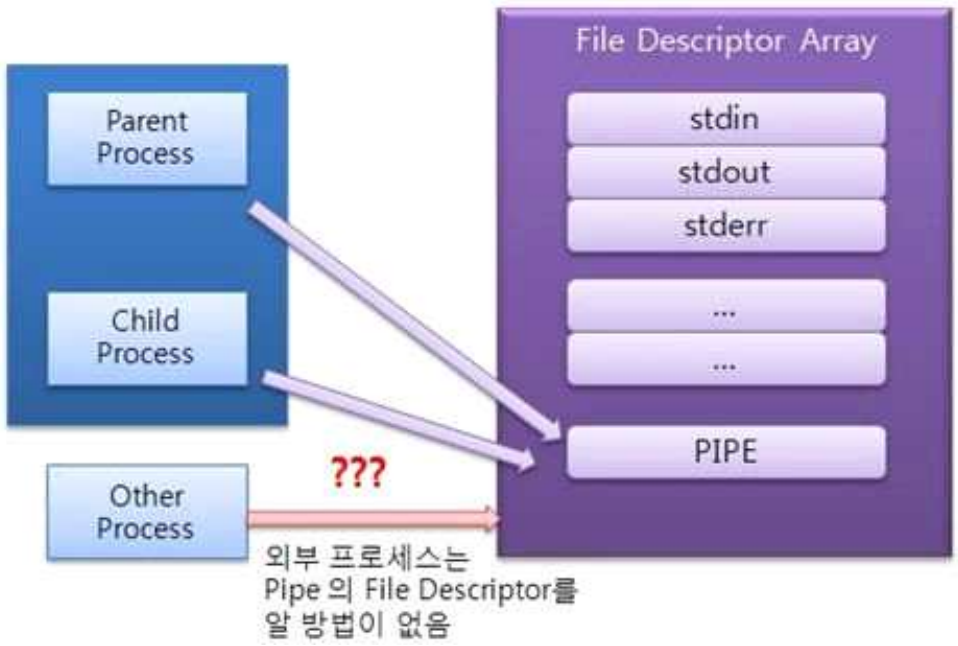
3) 종류
- 이름 없는 익명의 파이프
- 이름을 가진 Named Pipe

4) 이름 없는 익명의 파이프의 특성
* 파이프는 입구와 출구의 개념이 없다
- 방향성이 없기 때문에 자신이 쓴 메시지를 자신이 읽을 수 있다
- 두 개의 프로세스가 통신할 때는 읽기전용 파이프와 쓰기 전용 파이프의 두 개의 파이프를 사용, 두개의 파이프를 생성하고, 한쪽 파이프를 일부러 닫아 버리는 것이지요.
- 파이프를 두개 생성하여 하나의 방향으로만 갈 수 있도록 합니다.



* 파이프는 fork() 함수에 의해 복사 되지 않는다
- fork() 함수에 의해서 프로세스가 생성되면, 자식 프로세스는 부모가 사용하던 변수를 복사하게 됩니다. 하지만 파이프의 경우 복사되는 것이 아니라 File Descriptor 를 공유하게 됩니다. 즉 자식 프로세스와 부모 프로세스가 같은 파이프를 가리키게 됩니다.

* 부모 자식프로세스 사이에서만 파이프를 사용 가능하다는 것입니다.
- 파이프는 운영체제에서 임시로 생성되는 파일이고, 접근 가능한 방법은 File Descriptor(파일 디스크립터) 를 공유하는 방법만이 존재합니다.
- 익명의 파이프는 부모와 자식 프로세스만이 파일 디스크립터를 공유하므로 다른 프로세스는 파이프를 사용하여 통신이 불가능 합니다.



* Named PIPE(네임드 파이프) 의 경우 다른 프로세스도, 지정한 이름으로 파일 디스크립터를 열수 있기 때문에, 부모 자식 프로세스 간이 아니더라도 사용할 수 있다.

학습내용2 : PIPE/FIFO 개념

1. 파이프 생성: popen(3)

* popne 함수는 fork()함수를 실행해 자식 프로세스를 만들고 command에서 지정한 명령을 exec()함수를 실행해 자식 프로세스가 수행하도록 함.
- 성공 시 : 파일 포인터
- 실패 시 : NULL 포인터

2. 파이프 닫기: pclose(3)

- 지정한 파이프 닫음.
- 관련된 waitpid() 함수 수행 자식 프로세스들이 기다렸다가 리턴
- 성공 시 : 자식 프로세스의 상태 값 리턴
- 실패 시 : -1 리턴

```
#include <stdio.h>
int pclose(FILE *stream);
```

3. pipe(2)

* 리눅스에서는 파이프를 사용하기 위해서 pipe() 함수를 제공합니다.
* 파이프의 Descriptor 를 저장할 배열. filedes[0] 에는 PIPE 의 읽기전용 Descriptor, filedes[1] 에는 PIPE 의 쓰기전용 Descriptor 가 저장된다.

* pipe() 함수에 의해 생성되는 파일 디스크립터에는 일반 파일처럼 읽기/쓰기 작업이 가능
- 파이프로 사용할 파일기술자 2개를 인자로 지정
- fildes[0]는 읽기, fildes[1]은 쓰기용 파일 기술자

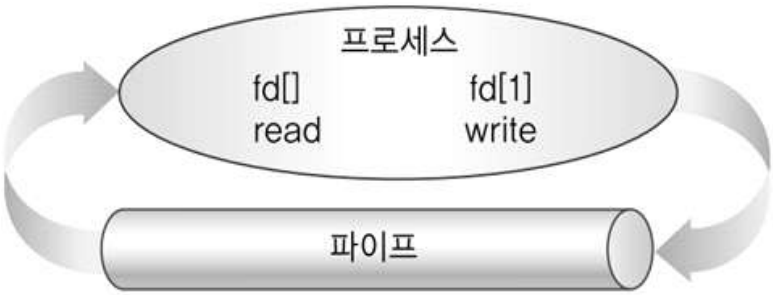
```
#include <unistd.h>
int pipe(int fildes[2]);
```

- 성공 시 : 0 을 반환
- 실패 시 : -1 반환

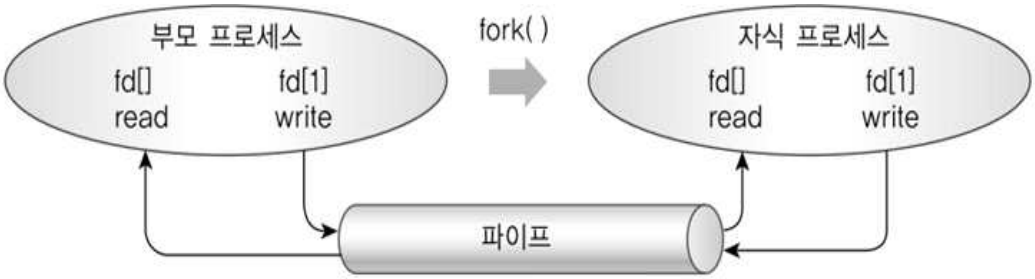
* 익명의 파이프는 외부 프로세스와 파일 디스크립터를 공유할 방법이 없으므로, 부모와 자식 프로세스 사이에서만 통신 가능합니다.

* fork() 함수를 이용한 통신방식

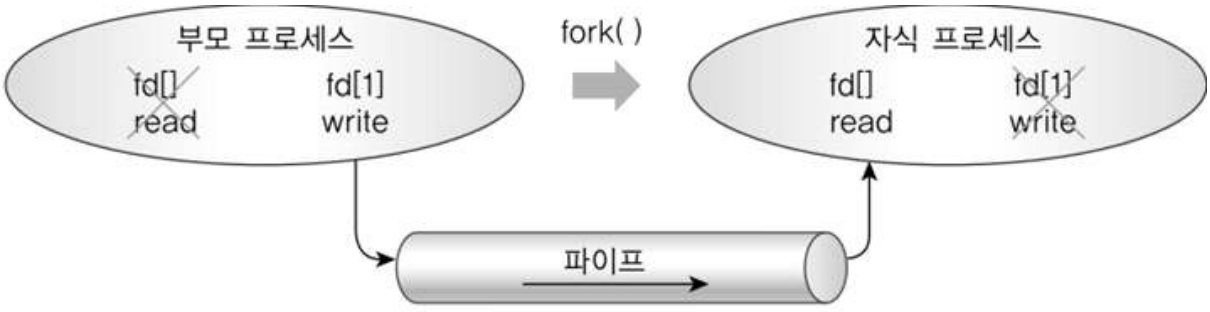
* pipe 함수로 통신과정
piep 함수를 호출하여 파이프로 사용할 파일기술자 생성



fork 함수로 자식 프로세스 생성. pipe도 자식 프로세스로 복사됨

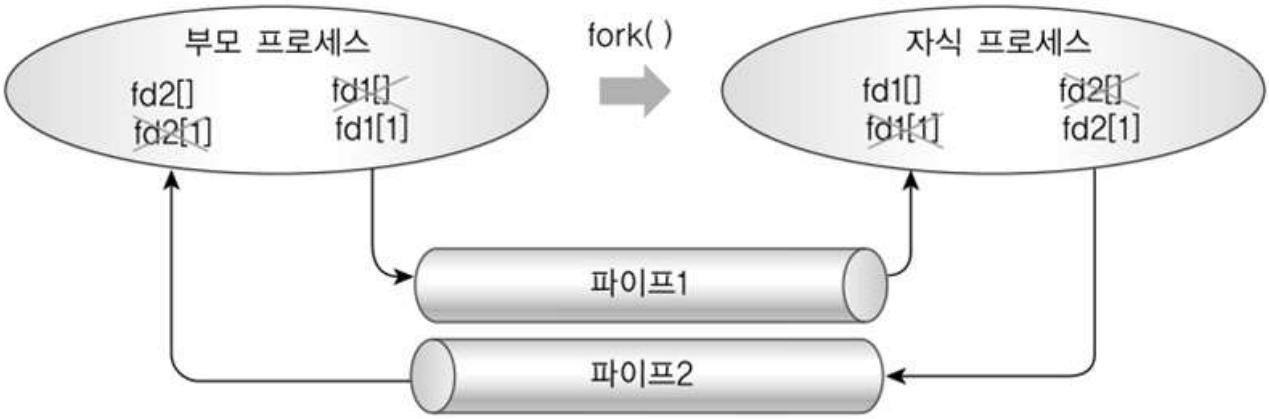


통신방향 결정(파이프는 기본적으로 단방향)



4. FIFO

양방향 파이프의 활용
파이프는 기본적으로 단방향이므로 양방향 통신을 위해서는 파이프를 2개 생성한다.



【학습정리】

1. 파이프 개념 및 종류
- 파이프 : 두 프로세스 사이에서 한 방향으로 통신 (기호 : |)
- 파이프 종류 : 이름없는 파이프(anonymous pipe)이름있는 파이프(named pipe)

- 이름없는 파이프 관련 함수

기능	함수원형
간단한 파이프 생성	FILE *popen(const char *command, const char *mode); int pclose(FILE *stream);
복작한 파이프 생성	int pipe(int fildes[2]);

- 이름있는 파이프 관련 함수

기능	함수원형
FIFO 생성 명령	mknod name p mkfifo [-m mode] path...
FIFO 생성 함수	int mknod(const char *paht, mode_t, dev_t dev); int mkfifo(const char *path, mode_t mode);

2. PIPE/FIFO 개념
- 익명의 파이프는 외부 프로세스와 파일 디스크립터를 공유할 방법이 없으므로, 부모와 자식 프로세스 사이에서만 통신 가능합니다.
- fork() 함수를 이용한 통신방식
- 파이프의 Descriptor를 저장할 배열. filedes[0]에는 PIPE의 읽기전용 Descriptor, filedes[1]에는 PIPE의 쓰기전용 Descriptor가 저장된다.
- pipe() 함수에 의해 생성되는 파일 디스크립터에는 일반 파일처럼 읽기/쓰기 작업이 가능 : 파이프로 사용할 파일기술자 2개를 인자로 지정