**Pintos Project 2: User Program (2)**

담당 교수 / 분반 : 김영재 / 2분반

이름 / 학번 : 정은교 / 20211590

개발 기간 : 2023.10.10~2023.10.28

1. **개발 목표**

* 해당 프로젝트에서 구현할 내용을 간략히 서술

File과 directory를 관리하는 kernel의 모듈을 file system이라고 한다. 그리고 프로젝트 1에서 구현하지 않았고 file system과 관련된 system call create, remove, open, close, filesize, seek, tell, read, write를 구현한다. 여기서, read, write system call의 경우 STDIN, STDOUT이 아니라 file에 접근하도록 다시 구현한다. 또, 프로세스 파일 간의 동기화 문제를 해결하기 위해 semaphore, file lock을 사용한다.

1. **개발 범위 및 내용**
   1. **개발 범위**

* 아래 각 항목을 구현해야 하는 이유, 혹은 구현 시 기대되는 결과를 간략히 서술

1. File Descriptor

System call을 호출하면 각각의 특정 파일에 접근해야하는데, 이때 file descriptor가 필요하다. 따라서, pintos에서 thread는 각각의 file descriptor를 얻고 관리하며 이를 구현하므로써, 각 thread가 파일에 접근할 수 있도록 해야한다.

2. (이번 프로젝트에서 구현해야 하는) System Calls

프로젝트 1에서 구현한 system call을 제외하고, create, remove, open, close, filesize, seek, tell, read, write를 구현한다. 이때, read, write는 프로젝트 1에서 구현하였으나, STDIN, STDOUT이 아니라 file에 접근하도록 다시 구현한다.

3. Synchronization in Filesystem

여러 개의 process가 동시에 실행되고 있는 도중, 한 process가 file을 읽거나, 쓸때, 다른 process가 같은 file을 읽거나 쓰게 되는 경우 데이터 문제가 발생할 수 있다. 따라서, 이를 위한 synchronization을 위해 semaphore를 사용하고, file synchronization을 위해 filelock을 사용한다.

* 1. **개발 내용**
* 아래 항목의 내용만 서술

1. File Descriptor: 구현에 이용할 자료구조와 선택한 이유를 서술

threads/thread.h의 thread 구조체에 file descriptor 구현을 위해 구조체 배열을 추가한다. 스레드가 생성될 때 이 구조체 배열은 NULL로 초기화되며, 이를 사용하여 파일에 접근할 수 있다. 이 구조체 배열에 파일 디스크립터를 저장함으로써 다른 동작을 수행할 때 해당 파일에 접근할 수 있도록 한다.

2. System Calls: 구현할 각 system call에 대해 간략히 서술 (하나의 system call 당 최대 3문장으로 간략히 설명; 3문장을 넘길 정도로 길게 작성하지 말 것)

1. create (bool create(const char \*file, unsigned initial\_size)) 'initial\_size' 바이트 크기를 가지는 'file' 파일을 생성. 성공 또는 실패 여부에 따라 true 또는 false를 반환. 파일은 열지 않음.

2. remove (bool remove(const char \*file)) 'file' 파일명을 가진 파일을 삭제. 성공 또는 실패 여부에 따라 true 또는 false를 반환. 파일은 열려 있던지 닫혀 있던지 삭제.

3. open (int open(const char \*file)) 'file' 파일명을 가진 파일을 열기. 파일 디스크립터를 반환하며, open에 실패한 경우 -1을 반환.

4. close (void close(int fd)) 'fd' 파일 디스크립터를 가진 파일을 닫기.

5. filesize (int filesize(int fd)) 'fd' 파일 디스크립터를 가진 파일의 크기를 바이트 단위로 반환.

6. read (int read(int fd, void \*buffer, unsigned size)) 'fd' 파일 디스크립터를 가진 열린 파일에서 'buffer'에 'size'만큼 읽어오기. 실제로 읽은 바이트 수를 반환하며, 읽기에 실패한 경우 -1을 반환.

7. write (int write(int fd, const void \*buffer, unsigned size)) 'fd' 파일 디스크립터를 가진 열린 파일에 'buffer'에서 'size'만큼 데이터를 기록. 실제로 기록한 바이트 수를 반환.

8. seek (void seek(int fd, unsigned position)) 'fd' 파일 디스크립터를 가진 열린 파일에서 읽거나 쓴 이후의 바이트 위치를 설정.

9. tell (unsigned tell(int fd)) 'fd' 파일 디스크립터를 가진 열린 파일에서 읽거나 쓴 이후의 바이트 위치를 반환.

3. Synchronization in Filesystem: Lock, Semaphore를 어떻게 이용할 수 있는지 각각에 대해 설명 (다른 방법을 서술해도 되지만 lock과 semaphore는 반드시 포함해야 함)

 - Lock

Race condition 및 다른 문제를 방지하기 위해 Lock 구조체를 정의하고, lock\_acquire 및 lock\_release를 사용한다. 파일 관련 시스템 콜 함수를 호출하기 전에 lock\_acquire 함수를 호출하여 작업을 시작하고, 작업이 완료된 후 lock\_release 함수를 호출하여 작업을 완전히 해제하는 방식이다.

- Semaphore

스레드 구조체에 file\_semaphore 변수를 선언하고 sema\_up 및 sema\_down 함수를 사용한다.

1. **추진 일정 및 개발 방법**
   1. **추진 일정**

* II. A. 개발 범위를 포함하여 구현 내용에 대한 일정 작성

2023.10.10~2023.10.15: create, remove, open, close, filesize, seek, tell system call 구현

2023.10.27~2023.10.28: read, write system call, synchronization 문제 해결 방안 구현

2023.10.28~2023.10.29: 보고서 작성

* 1. **개발 방법**
* II. B.의 개발 내용을 구현하기 위해 각각에 대해 다음 사항들을 포함하여 설명
  + 수정해야하는 소스코드
  + 수정하거나 추가해야 하는 자료구조
  + 수정하거나 추가해야 하는 함수

1. File descriptor

- pintos/src/threads/thread.h에 file descriptor를 위한 struct file \*fd[128]을 추가한다.

- pintos/src/threads/thread.c에 init\_thread 함수에서 생성한 fd값을 모두 NULL로 초기화해주는 기능을 가진 코드 작성한다.

- pintos/src/userprog/syscall.c exit 함수에서 exit하기 전 fd[i]가 NULL아니면 모두 닫아주는 코드를 추가한다.

-

1. System call

- pintos/src/userprog/syscall.h에 create, remove, open, close, filesize, seek, tell의 prototype을 선언한다.

- pintos/src/userprog/syscall.c에 create, remove, open, close, filesize, seek, tell 함수를 구현하고 read, write 함수를 file에 접근하도록 수정한다. 그리고 syscall\_handler 함수의 switch문에 SYS\_CREATE, SYS\_REMOVE, SYS\_OPEN, SYS\_CLOSE, SYS\_FILESIZE, SYS\_SEEK, SYS\_TELL일 때 각각의 system call을 처리할 수 있도록 추가한다. 이는 is\_user\_vaddr 함수를 사용하여 잘못된 메모리의 접근을 하지 않도록 해준다.

1. Synchronization in Filesystem

- pintos/src/thread/thread.h 내에 lock\_acquire과 lock\_release 함수를 실행하기 위한 lock 구조체 변수를 선언한다. 동기화를 위해 flock 변수(프로세스 상태를 확인하는 용도), flag 변수(프로세스가 비정상적으로 종료되는 지의 여부를 확인하는 용도), parent thread 변수(현재 프로세스의 부모 프로세스)를 선언해준다.

- pintos/src/thread/thread.c 내에 lock\_init 함수를 통해 lock\_acquire, lock release 함수 실행을 위한 lock 구조체 변수를 초기화를 해준다. 그리고 init\_thread 함수에서 위의 나머지 변수들을 초기화해준다.

- pintos/src/userprog/syscall.c에 open, read, write 함수에서 lock\_acquire, lock\_release를 사용하여 Synchronization 문제를 해결한다.

- pintos/src/userprog/exception.c에 page\_fault 함수 내에 not\_present 변수가 1일 때 프로그램을 종료하는 조건을 추가하여 예외 처리를 강화한다.

- pintos/src/userprog/process.c에 process\_execute, start\_process 함수에 shared data 처리를 위해 sema\_up, sema\_down 함수로 file\_semaphore 변수를 처리한다.

1. **연구 결과**
   1. **Flow Chart**

* II. B. 개발 내용의 각 3가지 항목에 대하여 Flow Chart 작성

1. File descriptor

텍스트, 도표, 스크린샷, 디자인이(가) 표시된 사진

자동 생성된 설명

2. System call

디자인이(가) 표시된 사진

낮은 신뢰도로 자동 생성된 설명

3. Synchronization in Filesystem

텍스트, 도표, 스크린샷, 디자인이(가) 표시된 사진

자동 생성된 설명

* 1. **제작 내용**
* II. B. 개발 내용의 각 3가지 항목에 대하여 실직적으로 구현한 코드의 관점에서 작성 (구현 내용, 알고리즘 등을 명확히 서술할 것)
* 구현에 있어 Pintos에 내장된 라이브러리나 자체 제작한 함수를 사용한 경우 이에 대해서도 설명
* 개발 중 발생한 문제나 이슈가 있으면 이를 간략히 설명하고 해결한 방식에 대해 설명
  + 1. pintos/src/threads/thread.h의 struct thread에 struct file \*fd[128]을 추가한다.

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

pintos/src/threads/thread.c의 init\_thread 함수에 thread 생성시 모든 fd[i]를 NULL 값으로 초기화하는 코드를 작성한다.

텍스트, 폰트, 스크린샷, 라인이(가) 표시된 사진

자동 생성된 설명

pintos/src/userprog/syscall.c exit 함수에서 exit하기 전 fd[i]가 NULL아니면 모두 닫아주는 코드를 추가한다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

* + 1. pintos/src/userprog/syscall.h에 create, remove, open, close, filesize, seek, tell의 prototype을 선언한다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

pintos/src/userprog/syscall.c에 syscall\_handler 함수 switch 문에 새롭게 추가할 함수들을 조건으로 넣는다.

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

pintos/src/userprog/syscall.c에 system call 함수 구현한다.

* + - * bool create (const char \*file, unsigned initial\_size)

텍스트, 스크린샷, 폰트, 일렉트릭 블루이(가) 표시된 사진

자동 생성된 설명

위의 SYS\_CREATE의 조건에 걸릴 때 실행되며 file == NULL, !is\_user\_vaddr(file) 인 경우 exit(-1)을 한다. filesys\_create를 사용해 구현된다.

* + - * bool remove (const char \*file)

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

위의 SYS\_REMOVE의 조건에 걸릴 때 실행되며 file == NULL, !is\_user\_vaddr(file) 인 경우 exit(-1)을 한다. filesys\_remove를 사용해 구현된다.

* + - * int open (const char \*file)

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

위의 SYS\_OPEN의 조건에 걸릴 때 실행되며 file == NULL인 경우 lock\_release(&flock)을 뒤 exit(-1)을 한다. filesys\_open(file)을 하여 fp == NULL 인 경우 lock\_release를 하고 -1을 반환한다. i=3부터 128전 까지 for문을 돌며 값이 NULL일때 fp 값을 넣어주고 lock\_release한다. 그 전에 thread\_current의 name 변수와 file 변수가 같으면 file\_deny\_write를 한다. synchronization 문제를 해결하기 위해서 lock\_acquire은 open, read, write 함수 모두 syscall\_handler 함수의 switch 문에서 해주었다.

* + - * void close (int fd)

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

위의 SYS\_CLOSE의 조건에 걸릴 때 실행되며 현재 thread의 fd가 NULL 인 경우 exit(-1)을 한다. 현재 thread의 fd 값을 fp에 저장하고, 해당 fd은 NULL 값을 넣어준다. file\_close(fp)를 호출한다.

* + - * int filesize (int fd)

텍스트, 스크린샷, 폰트, 일렉트릭 블루이(가) 표시된 사진

자동 생성된 설명

위의 SYS\_FILESIZE의 조건에 걸릴 때 실행되며 현재 thread의 fd가 NULL 인 경우 exit(-1)을 한다. file\_length(thread\_current()->fd[fd]) 함수를 호출하고 결과를 반환한다.

* + - * void seek (int fd, unsigned position)

텍스트, 폰트, 스크린샷, 일렉트릭 블루이(가) 표시된 사진

자동 생성된 설명

위의 SYS\_SEEK의 조건에 걸릴 때 실행되며 현재 thread의 fd가 NULL 인 경우 exit(-1)을 한다. file\_seek(thread\_current()->fd[fd]) 함수를 호출하고 결과를 반환한다.

* + - * unsigned tell (int fd)

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

위의 SYS\_TELL의 조건에 걸릴 때 실행되며 현재 thread의 fd가 NULL 인 경우 exit(-1)을 한다. file\_tell(thread\_current()->fd[fd]) 함수를 호출하고 결과를 반환한다.

* + - * int read (int fd, void \*buffer, unsigned length)

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

위의 SYS\_READ의 조건에 걸릴 때 실행되며 표준입력이 아닌 파일이 입력된 경우를 추가 구현한다. 파일 이름과 buffer, size를 입력 받고 file\_read 함수를 이용하여 파일을 size만큼을 읽어 buffer에 저장한다. synchronization 문제를 해결하기 위해서 lock\_acquire은 open, read, write 함수 모두 syscall\_handler 함수의 switch 문에서 해주었다.

* + - * int write (int fd, const void \*buffer, unsigned length)

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

위의 SYS\_WRITE의 조건에 걸릴 때 실행되며 표준출력이 아닌 파일에 출력하는 경우를 추가 구현한다. 파일 이름과 buffer, size를 입력 받고 file\_write 함수를 이용하여 파일을 size만큼을 write한다. synchronization 문제를 해결하기 위해서 lock\_acquire은 open, read, write 함수 모두 syscall\_handler 함수의 switch 문에서 해주었다.

* + 1. pintos/src/userprog/process.c에 process\_execute 함수와 start\_process 함수에 semaphore를 사용하여 synchronization 문제를 해결한다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

pintos/src/threads/thread.h에 lock\_acquire, lock\_release 함수를 실행하기 위한 lock 구조체 변수 flock을 선언해준다.

**폰트, 텍스트, 친필, 그래픽이(가) 표시된 사진

자동 생성된 설명**

pintos/src/threads/thread.c thread\_init 함수에 구조체 변수를 초기화해준다.

폰트, 텍스트, 그래픽, 스크린샷이(가) 표시된 사진

자동 생성된 설명

pintos/src/userprog/syscall.c에 deny\_write을 사용하기 위한 file 구조체를 선언해준다.

텍스트, 폰트, 스크린샷, 번호이(가) 표시된 사진

자동 생성된 설명

* 1. **시험 및 평가 내용**
* make check 수행 결과를 캡처하여 첨부
* 스크린샷, 텍스트, 패턴, 블루이(가) 표시된 사진

  자동 생성된 설명
* 텍스트, 스크린샷, 디자인이(가) 표시된 사진

  자동 생성된 설명