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

담당 교수 / 분반 :

이름 / 학번 :

OSHIMA ASUKA

20202106

1. **개발 목표**

* 파일 시스템을 구현해서 file descriptor의 중요성과 synchronization의 중요성을 배운다.

1. **개발 범위 및 내용**
   1. **개발 범위**

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

1. File Descriptor

프로세스가 파일이나 socket 등의 system resource에 접근하기 위한 포인터다. File Descriptor는 다수의 프로세스가 동시에 정보를 공유하는 것을 제어한다. File Descriptor를 구현함으로 파일을 열었을 때 시스템은 하나의 file description를 리턴하여 그 파일에 대해 작업을 하는 것을 허락한다.

2. (이번 프로젝트에서 구현해야 하는) System Calls

이번 실습에서는 파일 시스템을 이용하기 쉽게 만드는 System Calls를 만들었다. 유저는 파일에 관한 작업을 쉽게 효율적으로 수행할 수 있다.

3. Synchronization in Filesystem

여러 개의 프로세스과 스레드가 동시에 파일 시스템의 접근하려고 할 때 데이터를 concurrency과 무결성을 유지하기 위해서 동기화를 한다. 이것을 구현하지 않으면 프로세스나 스레드를 여러 개 실행하였을 때 데이터의 일관성을 유지하지 못해 잘못한 결과가 나올 위험성이 있다. 파일시스템에 동기화를 도입함으로 여러 개의 프로세르나 스레드를 안전하게 동시에 파일에 접근/수정하는 것이 가능해진다.

* 1. **개발 내용**
* 아래 항목의 내용만 서술

1. File Descriptor: 구현에 이용할 자료구조와 선택한 이유를 서술

File descriptor table를 관리하기 위한 배열을 만들었다. 원하는 file descriptor를 관리하기 쉽기 때문에 배열을 선택했다.

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)

Open: 파일을 연다. 성공하면 fd(file descriptor)를 리턴한다.

Close: file descriptor를 닫아 연결된 resource의 연결을 끊는다.

Read:해당하는 file descriptor를 이용해서 데이터를 읽는다. 읽은 데이터의 바이트 수를 리턴한다.

Write: 해당하는 file descriptor에 데이터를 쓴다. 쓰는 데이터의 바이트 수를 리턴한다.

Create: 해당하는 이름으로 파일을 생성한다.

Remove: 해당하는 이름의 파일을 삭제한다.

Tell: file descriptor의 offset를 리턴한다.

Seek: 해당하는 file descriptor의 위치를 변경한다.

Filesize: 해당하는 file descriptor의 크기를 리턴한다.

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

Lock: 공유하는 파일 시스템의 동시에 접근하는 것을 막는다. 동시에 같은 파일을 하나가 아닌 프로세스가 수정을 시도할 때 이를 이용해서 하나의 프로세스만 작업을 할 수 있게 된다.

Semaphore: 한번에 특정한 프로세스 또는 스레드를 수행할 수 있도록 제한한다.

1. **추진 일정 및 개발 방법**
   1. **추진 일정**

10월 28일부터 29일까지 코드 작성을 했다.

* 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. **연구 결과**
   1. **Flow Chart**

* II. B. 개발 내용의 각 3가지 항목에 대하여 Flow Chart 작성
  1. **제작 내용**
* II. B. 개발 내용의 각 3가지 항목에 대하여 실직적으로 구현한 코드의 관점에서 작성 (구현 내용, 알고리즘 등을 명확히 서술할 것)

**File descriptor**

File descriptor를 관리하기 위한 배열을 구조체를 추가했다. Process\_add\_file()를 이용해서 파일을 추가할 때 사용가능한 file descriptor를 찾아 그 위치에 파일 포인터를 저장한다.

**System call**

Open를 구현할 때 filesys\_open , read, write를 구현할 때 file\_read,file\_write를 사용했다.

Read-boundary, write-boundary, open-boundary의 테스트 케이스를 통과 못했다. Buf의 크기에 주목을 하여 수정을 했지만 해결하지 못했다. 좀 더 debug과 분석이 필요한 것 같다.

* 1. **시험 및 평가 내용**
* テキスト

  自動的に生成された説明