0. Pintos 설정하기

Pintos는 80x86 아키텍처를 위한 간단한 운영 체제 프레임워크입니다. 이 프레임워크는 커널 스레드, 사용자 프로그램 로딩 및 실행, 파일 시스템을 지원하지만, 이 모든 기능을 매우 간단한 방식으로 구현합니다. Pintos 프로젝트에서는 여러분과 팀이 이 세 가지 영역에 대한 지원을 강화할 것입니다.
Pintos 프로젝트는 시스템 시뮬레이터에서 실행됩니다. 시스템 시뮬레이터는 80x86 CPU와 주변 장치를 정확하게 시뮬레이션하여 수정되지 않은 운영 체제와 소프트웨어가 그 아래에서 실행될 수 있게 하는 프로그램입니다. 수업에서는 QEMU 시뮬레이터를 사용할 것입니다.

0-1 소스 트리 개요

GitHub private repository 초대에 응하여 이 repo를 clone하여 $HOME/pintos-KMA/src의 디렉토리 구조를 갖도록 저장하십시오.
하부 디렉토리 구조는 다음과 같습니다.
- threads/: 기본 커널 소스 코드, 프로젝트 1부터 수정합니다.
- userprog/: 사용자 프로그램 로더 소스 코드, 프로젝트 2부터 수정합니다. **변경 X**
- vm/: 거의 비어 있는 디렉토리. 가상 메모리 구현에 사용됩니다. **변경 X**
- filesys/: 기본 파일 시스템 소스 코드. **변경 X**
- devices/: I/O 장치 인터페이스 소스 코드. **변경 X**
- lib/: 표준 C 라이브러리의 하위 집합 구현. 이 디렉토리의 코드는 Pintos 커널과 사용자 프로그램에 컴파일됩니다. 커널 코드와 사용자 프로그램 모두에서 헤더를 #include <...>로 포함할 수 있습니다. **변경 X**
- lib/kernel/: Pintos 커널에 포함된 C 라이브러리의 일부. 비트맵, 이중 연결 리스트, 해시 테이블 등의 데이터 타입 구현이 포함되어 있으며, 커널 코드에서 사용할 수 있습니다. **변경 X**
- lib/user/: Pintos 사용자 프로그램에 포함된 C 라이브러리의 일부. 사용자 프로그램에서 헤더를 #include <...>로 포함할 수 있습니다. **변경 X**
- tests/: 각 프로젝트의 테스트. 제출을 테스트하는 데 도움이 되면 이 코드를 수정할 수 있지만, 테스트를 실행하기 전에 원본으로 교체합니다.
- examples/: 예제 사용자 프로그램.
- misc/, utils/: 자체 기계에서 Pintos를 작업할 때 유용할 수 있는 파일입니다. 그렇지 않으면 무시할 수 있습니다.

0-2 Pintos 빌드하기

다음 단계는 첫 번째 프로젝트를 위한 소스 코드를 빌드하는 것입니다. 그에 앞서 다음과 같은 작업이 필요합니다.
① WSL 및 Git 설치
 - https://learn.microsoft.com/ko-kr/windows/wsl/setup/environment
 - 링크대로 WSL에서 git을 설치하고, 윈도우에는 최신 git을 설치하여 git 명령을 실행하면 윈도우에 설치된 git-credential-manager가 실행되도록 합니다.

② 설치된 Linux subsystem 설정
 - 프로젝트 실행파일 경로를 ~/.bashrc에 추가
   => export PATH=$HOME/pintos-KMA/src/utils:$PATH
 - 필수 패키지 설치(sudo apt-get install ~)
   => build-essential / clang / gdb / qemu-system-x86
③ pintos 빌드
 - utils 디렉토리로 이동, make를 실행합니다. warning 메시지는 무시해도 되나, 다른 에러 메시지가 나온다면 화면과 함께 교수에게 문의 바랍니다.

이후 threads 디렉토리로 이동한 후, make 명령어를 실행합니다. 이 명령어는 threads 아래에 빌드 디렉토리를 생성하고, Makefile 및 몇 개의 하위 디렉토리로 채운 후, 그 안에 커널을 빌드합니다. 전체 빌드는 30초 이내에 완료되어야 합니다.
빌드 중에 실행된 명령어를 주의 깊게 살펴보세요. Linux 기계에서는 일반적인 시스템 도구를 사용합니다.
빌드 후, 빌드 디렉토리에서 다음 파일들이 있어야 합니다.
- Makefile: 커널을 빌드하는 방법을 설명하는 pintos/src/Makefile.build의 복사본입니다. 소스 파일 추가에 대한 자세한 내용은 'Adding Source Files'를 참조하세요.
- kernel.o: 전체 커널의 객체 파일. 각 개별 커널 소스 파일에서 컴파일된 객체 파일을 연결하여 생성된 파일입니다. 디버그 정보가 포함되어 있어 GDB(섹션 E.5 GDB) 또는 백트레이스(섹션 E.4 Backtraces)를 실행할 수 있습니다.
- kernel.bin: 커널의 메모리 이미지로, 메모리에 로드하여 Pintos 커널을 실행하기 위한 정확한 바이트입니다. 디버그 정보가 제거된 kernel.o로, 많은 공간을 절약하고 커널이 512kB의 크기 제한에 부딪히지 않도록 합니다.
- loader.bin: 커널 로더의 메모리 이미지로, 커널을 디스크에서 메모리로 읽어 시작하는 작은 조각의 어셈블리 코드입니다. PC BIOS에 의해 고정된 크기인 512바이트입니다.
build 디렉토리의 하위 디렉토리에는 컴파일러에 의해 생성된 객체 파일(.o)과 의존성 파일(.d)이 포함되어 있습니다. 의존성 파일은 다른 소스 또는 헤더 파일이 변경될 때 다시 컴파일해야 하는 소스 파일을 make에게 알려줍니다.

0-3 Pintos 실행하기

Pintos를 시뮬레이터에서 편리하게 실행할 수 있는 프로그램을 제공했습니다. 이 프로그램은 pintos라고 불립니다. 가장 간단한 경우, pintos를 pintos argument...와 같이 호출할 수 있습니다. 각 인자는 Pintos 커널에 전달되어 작동합니다.
다음과 같이 실행합니다. 먼저 새로 생성된 build 디렉토리로 이동합니다. 그런 다음 pintos run alarm-multiple 명령어를 입력합니다. 이 명령어는 run alarm-multiple 인자를 Pintos 커널에 전달합니다. 이 인자에서 run은 커널에 테스트를 실행하라는 지시를 주고, alarm-multiple은 실행할 테스트입니다. 이 명령어는 qemu를 호출하고, qemu는 시뮬레이션된 기계의 디스플레이를 나타내는 새 창을 열고, BIOS 메시지가 잠깐 깜박입니다. 그 다음 Pintos가 부팅되고 alarm-multiple 테스트 프로그램이 실행되어 몇 페이지의 텍스트를 출력합니다. 작업이 완료되면 qemu의 오른쪽 상단에 버튼을 클릭하여 종료할 수 있습니다.