달고나 문서 요약

윤준혁

8086 시스템의 메모리 구조에서 커널은 기본적으로 low address쪽, 64Kbyte 영역에 위치한다. 하지만 오늘날의 운영체제들은 더 큰 영역을 사용한다. 운영체제는 하나의 프로세스를 실행시키면 이 프로세스를 segment라는 단위로 묶어서 가용 메모리 영역에 저장시킨다. 또한 한번에 여러 개의 프로세스를 저장시켜 병렬적으로 작업을 수행한다.

이때, 실제로 프로그램을 구동시키면 공유 라이브러리 함수나 환경 변수, argc, argv pointer 등등에 의해 가용 메모리 범위는 조금 더 적다.

하나의 segment는 code segment, data segment, stack segment로 이루어 진다.

code segment에는 명령어 (instruction)이 들어있다. 명령어가 수행하는 작업 중 분기와 점프의 경우 메모리 상의 특정 위치를 알아야 하는데 segment는 컴파일 과정에서 자신이 메모리 상의 어느 위치에 저장될지 알 수 없기 때문에 logical address를 사용한다.

logical address는 실제 메모리 상의 주소(physical address)와 매핑되어 있으므로 segment는 segment selector에 의해서 자신의 시작 위치(offset)를 찾고 그 위치로부터 logical address에 있는 명령을 수행할 지를 결정한다. 따라서 physical address는 offset + logical address라고 할 수 있다.

data segment에는 프로그램이 실행시에 사용되는 데이터가 들어간다.(ex>전역변수) data segment는 다시 네 개의 data segment로 나뉘는데 각각 현재 모듈의 data structure, 상위 레벨로부터 받아들이는 데이터 모듈, 동적 생성 데이터, 다른 프로그램과 공유하는 공유 데이터 부분이다.

stack segment에는 버퍼, 지역 변수들이 자리잡고 프로그램이 사용하는 multiple 스택을 생성할 수 있고 각 스택들간의 switch가 가능하다.

스택은 처음 생성될 때 그 필요한 크기만큼 만들어지고 stack pointer(SP)라고 하는 레지스터가 스택의 맨 꼭대기를 가리키고 있다. 스택이 데이터를 저장하고 읽어 들이는 과정은 PUSH와 POP instruction에 의해서 수행되는데 가장 최근에 Push(저장)된 데이터를 POP(읽음) 명령을 통해서 가져오게 된다.

CPU가 프로세스를 실행하기 위해서는 프로세스를 CPU에 적재시켜야 한다. 그리고 명령어 집합

(instruction set)과 데이터들을 적절하게 집어내고 읽고 저장하기 위해서는 여러 가지 저장공간이 필요한데 이것들은 CPU가 재빨리 읽고 쓰기를 해야하는 데이터들이므로 CPU 내부에 존재하는 레지스터(register)라는 저장공간에 저장한다.

register는 그 목적에 따라서 범용 레지스터(General_Purpose register), 세그먼트 레지스터(segment register), 플래그 레지스터(Program status and control register), 그리고 인스트럭션 포인터(instruction pointer)로 구성된다.

범용 레지스터는 논리 연산, 수리 연산에 사용되는 피연산자, 주소를 계산하는데 사용되는 피연산자, 그리고 메모리 포인터가 저장되는 레지스터다. EAX, EBX, ECX, EDX 등이 있으며 (32bit) EAX 레지스터의 상위 부분을 AH라고 하고 하위 부분을 AL이라고 한다. 범용 레지스터는 프로그래머가 임의로 조작할 수 있는데 각각의 목적대로 사용해 주는 것이 좋다.

세그먼트 레지스터는 프로세스의 특정 세그먼트를 가리키는 포인터 역할을 한다. 즉 code segment, data segment, stack segment를 가리키는 주소가 들어있는 레지스터다. CS 레지스터는 code segment를, DS, ES, FS, GS 레지스터는 data segment를, SS 레지스터는 stack segment를 가리킨다.

플래그 레지스터는 프로그램의 현재 상태나 조건 등을 검사하는데 사용되는 플래그들이 있는 레지스터이다. (상태 플래그, 컨트롤 플래그, 시스템 플래그)

인스트럭션 포인터는 다음 수행해야 하는 instruction이 있는 메모리 상의 주소, 즉 현재 code segment의 offset 값이 들어있는 레지스터이다.