Computer System Structure

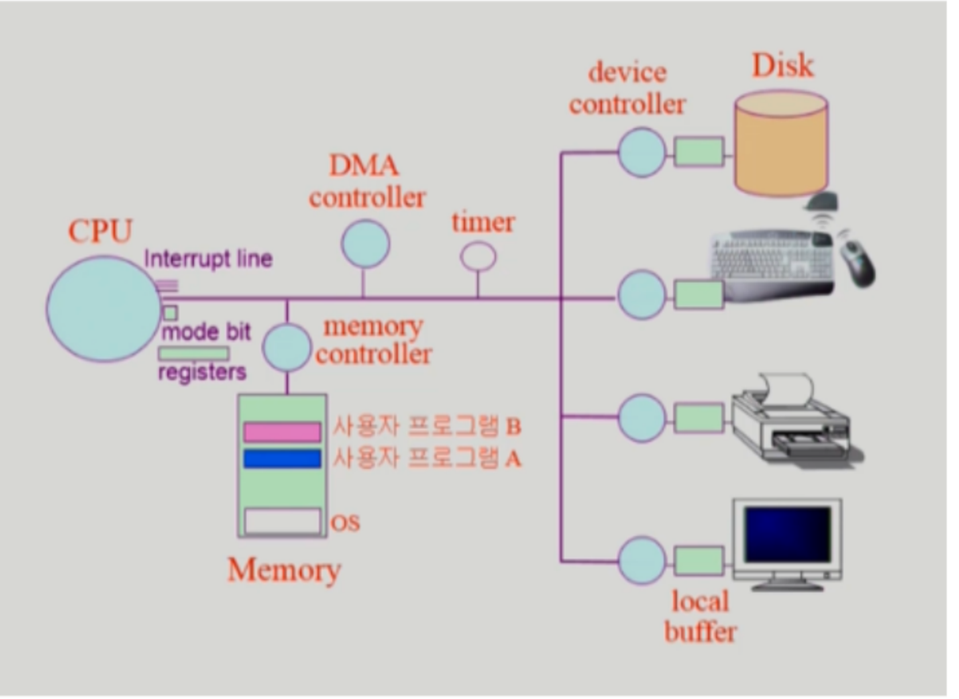
2023.06.15 **김하림**

Index

01 Computer System Structure

1. **CPU**: 매 clock cycle마다 memory에서 instruction을 읽어 실행
   - **mode bit**: kernel mode / user mode
   - **interrupt line**: instruction 한 줄 수행 후 interrupt line을 확인해 들어온 interrupt가 있는지 확인
   - **registers**: memory보다 빠르면서 정보를 저장할 수 있는 작은 공간
2. **Memory**: CPU의 작업 공간
3. **I/O Device**: disk, keyboard, ...
   - 각각 **device controller**와 **local buffer**가 존재
   - device controller: I/O Device의 작은 CPU 역할, 작업 완료 후 CPU에 interrupt
   - local buffer: device controller의 작업 공간
4. **timer**: 특정 프로그램이 CPU를 독점하는 것을 막음

Computer System Structure



02 Interrupt

interrupt 당한 시점의 register와 program counter(PC)를 save한 후 CPU의 제어를 interrupt handler에 넘긴다

Interrupt(hardware interrupt): 하드웨어(키보드, 하드디스크..)가 발생시킨 interrupt

Trap(software interrupt): Exception, System call(user program이 커널 함수 호출)

interrupt 관련 용어

- **interrupt vector**: 해당 interrupt handler의 주소를 가지고 있음
- **interrupt handler** = interrupt service routine: 해당 interrupt를 처리하는 kernel 함수

03 Mode bit

1. user program의 잘못된 수행으로 다른 프로그램 및 운영체제에 피해가 가지 않도록 하기 위한 보호 장치
2. mode bit = 0: **Kernel mode** - previleged instruction 수행
    - Interrupt나 Exception 발생 시 하드웨어가 mode bit을 0으로 바꿈
    - user program에게 CPU를 넘기기 전에 mode bit을 1로 set
3. mode bit = 1: **User mode** - 제한된 instruction 수행

04 Timer

- 정해진 시간이 흐른 뒤 OS에게 제어권이 넘어가도록 interrupt를 발생시킴
- 타이머 값이 매 clock 1씩 감소하다가 0이 되면 timer interrupt 발생
- **특정 프로그램이 CPU를 독점하는 것을 막음**
- **time sharing** 구현, 현재 시간 계산 등을 위해 사용

05 I/O Device Controller

- 해당 I/O device를 관리하는 일종의 작은 CPU
- control register, status register
- local buffer = 일종의 data register
- I/O는 실제 device와 local buffer 사이에서 일어남
- I/O가 끝났을 경우 interrupt로 CPU에 알림**(hardware interrupt)**
- **device driver**: OS 코드 중 각 장치별 처리 루틴(handler) → Software
- **device controller**: 각 장치를 통제하는 작은 CPU → Hardware

05 I/O Execution

- user program은 어떻게 I/O 을 실행하는지?

1. **system call**: OS에게 I/O request를 보냄(Trap - software interrupt)

2. **trap**을 사용하여 interrupt vector의 특정 위치로 이동

3. 제어권이 interrupt vector가 가리키는 interrupt handler로 이동

4. 올바른 I/O request인지 확인 후 I/O 진행

5. I/O 완료 시 제어권을 system call 다음 instruction으로 넘김

06 System Call

- user program이 OS의 서비스를 받기 위해 커널 함수를 호출
- CPU 제어권이 OS에게 넘어감