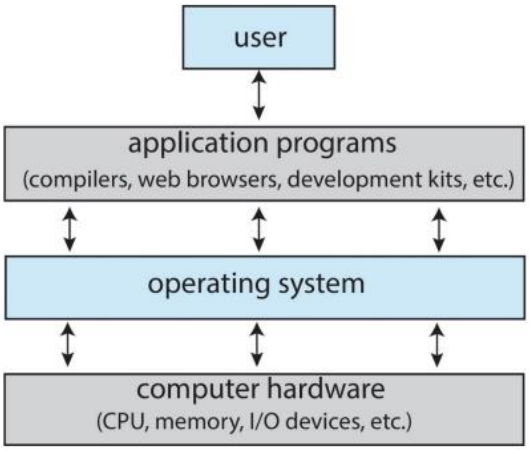
Os Introduction

Computer System Structure

컴퓨터 시스템은 4가지 구성 요소로 나눌 수 있다.

- 하드웨어(Hardware) – 기본 컴퓨팅 리소스 제공
  - CPU, 메모리, storage, I/O 디바이스
- 운영체제(Operating System)
  - 다양한 애플리케이션 및 사용자 간의 하드웨어 사용 제어 및 조정
  - Windows, MacOS, Linux 등
- 응용 프로그램(Application Programs)
  - 시스템 리소스를 사용하여 사용자의 컴퓨팅 문제를 해결하는 방법
  - 워드 프로세서, 컴파일러, 웹 브라우저, 데이터베이스 시스템, 비디오 게임
- 사용자
  - 사람, 기계, 컴퓨터 등



운영체제의 작업

- 사용자는 사용 편의성 및 우수한 성능을 원함 – 리소스 활용에 신경 쓰지 않음
- 운영체제는 하드웨어를 효율적으로 활용하고 사용자 프로그램의 실행을 관리하는 리소스 할당자 및 제어 프로그램.
- 모바일 장치는 사용성과 배터리 수명에 최적화되어 리소스가 부족

- 모바일 사용자 인터페이스 (터치 스크린, 음성 인식)

- 일부 컴퓨터는 사용자 인터페이스가 없음

- 사용자 개입 없이 실행

운영 체제의 정의

- **Kernel** : 컴퓨터에서 항상 실행되는 프로그램

- **Middleware**

- 데이터베이스, 멀티미디어, 그래픽과 같은 애플리케이션 개발자에게 추가 서비스를 제공하는 소프트웨어 프레임워크 세트

- 오늘날 범용 및 모바일 컴퓨팅용 OS에 포함

- iOS, Android

- **OS**

- 컴퓨터 사용자와 컴퓨터 하드웨어 사이의 중개자 역할을 하는 프로그램

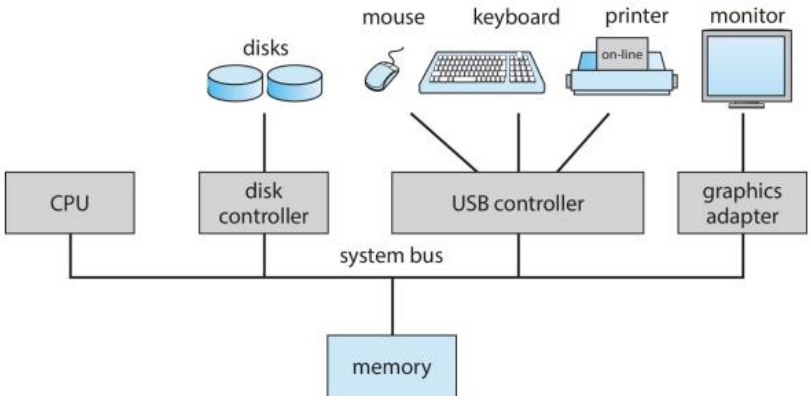
- 컴퓨터 하드웨어, 소프트웨어 리소스를 관리하고 컴퓨터 프로그램을 위한 공통 서비스를 제공하는 시스템 소프트웨어

컴퓨터 시스템 조직

- 컴퓨터 시스템 작동

- 공유 메모리에 대한 엑세스를 제공하는 공통 버스를 통해 하나 이상의 CPU, 장치 컨트롤러 연결

- 메모리 사이클을 위해 경쟁하는 CPU 및 장치의 동시 실행

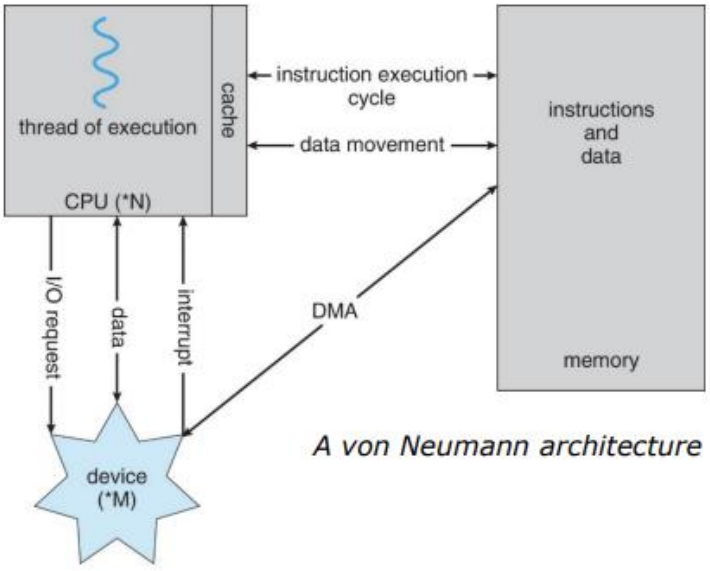


컴퓨터 시작 및 작동

- **시작**

- 전원을 켜거나 재부팅할 때 부트스트랩 프로그램이 로드
- 일반적으로 ROM 또는 EPROM에 저장되며, 펌웨어(firmware)라고 한다.
- 시스템의 모든 측면을 초기화
- 운영체제 커널을 로드하고 실행 시작

- **작동**

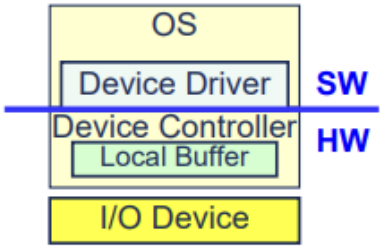


A von Neumann architecture

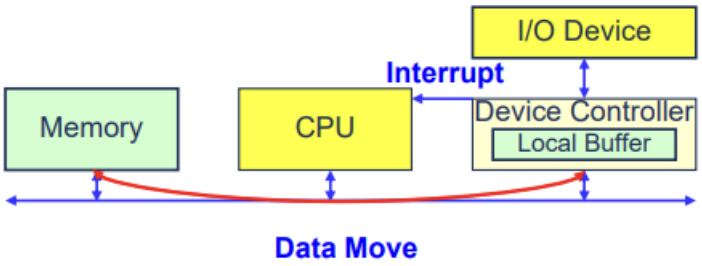
컴퓨터 시스템 동작

- **각 장치 컨트롤러**

- 특정 I/O 장치 유형 담당
- I/O 데이터를 저장할 로컬 버퍼가 있음
- 관리할 운영체제 장치 드라이버(device driver)가 있음.
- 장치 드라이버(device driver) : 컨트롤러와 커널 사이에 균일한 인터페이스 제공

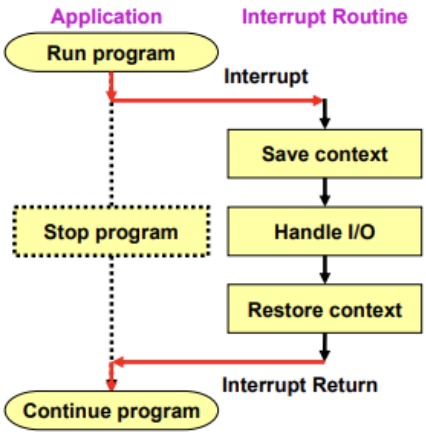


- CPU가 메인 메모리에서 로컬 버퍼로 데이터를 이동하거나 로컬 버퍼에서 데이터를 이동
- I/O가 컨트롤러의 로컬 버퍼에서 디바이스로 송수신됨
- 디바이스 컨트롤러가 인터럽트를 발생시켜 CPU에 작업을 완료했음을 알림



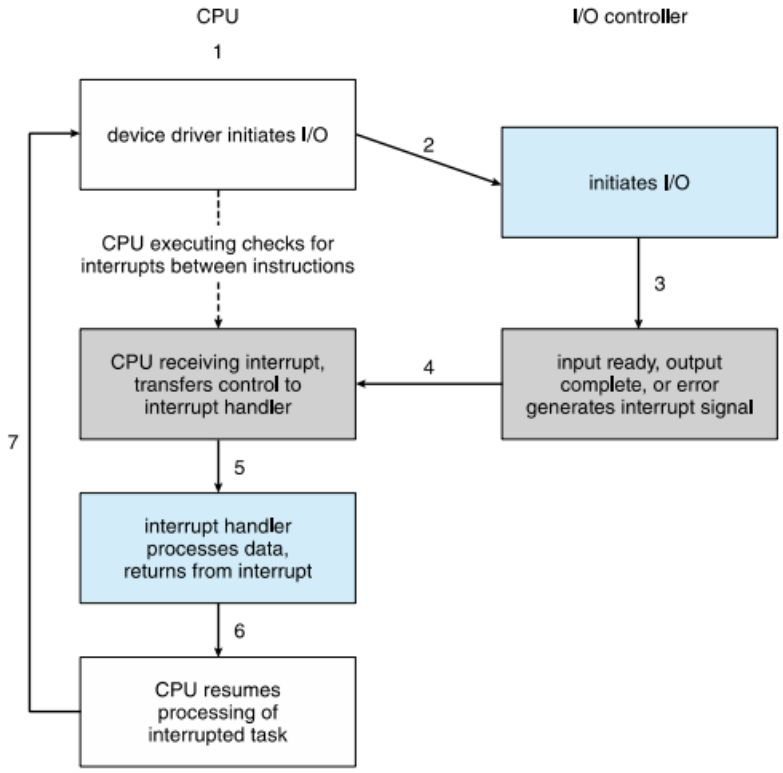
인터럽트 처리 (Interrupt Handling)

- 인터럽트가 ISR(인터럽트 서비스 루틴)에 제어 권한을 전달
- 인터럽트 벡터를 통해 (또는 소프트웨어에 의해) 모든 ISR의 주소를 포함
- 인터럽트 아키텍처는 인터럽트된 명령의 주소를 저장
- OS는 CPU 레지스터와 PC를 저장하여 CPU 상태(context)를 보존
- 코드는 각 인터럽트 유형에 대해 취해야 할 조치를 결정



- **trap, exception**
- 오류 또는 사용자 요청에 의해 발생하는 소프트웨어 생성 인터럽트
- 운영체제는 interrupt driven

Interrupt-driven I/O Cycle



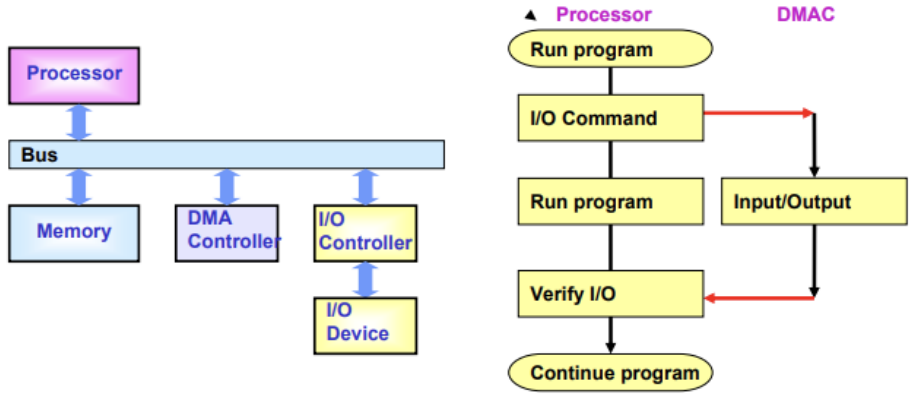
두 개의 I/O 구조

- I/O가 시작된 후 I/O가 완료될 때 사용자 프로그램으로 돌아감
  - 다음 인터럽트가 발생할 때까지 CPU를 유휴 상태로 유지 : Wait instruction
  - Wait loop : 메모리 엑세스에 대한 경합
  - 한 번에 최대 한 개의 I/O 요청이 처리되고 동시에 처리되지 않음
- I/O가 시작되면 I/O가 완료될 때까지 기다리지 않고 사용자 프로그램으로 돌아감
  - System call : 사용자가 I/O 완료를 기다릴 수 있도록 OS에 요청
  - Device-status table에는 각 I/O 디바이스의 유형, 주소 및 상태를 나타내는 항목이 포함됨.
  - 장치 상태를 확인하고 인터럽트를 포함하도록 테이블 항목을 수정하기 위해 I/O 장치 테이블로 OS 인덱싱

Direct Memory Access Structure

- 메모리 속도에 가까운 속도로 정보를 전송할 수 있는 고속 I/O 장치에 사용

- 장치 컨트롤러는 CPU 개입 없이 버퍼 스토리지에서 메인 메모리로 데이터 블록을 직접 전송
- 블록당 하나의 인터럽트만 생성



Storage Structure

- Main Memory – CPU가 직접 액세스할 수 있는 대용량 스토리지 미디어
  - Random access
  - 휘발성(volatile)
  - DRAM(Dynamic Random-Access Memory) 형태로 사용
- Secondary storage - 대용량 비휘발성 스토리지 용량을 제공하는 메인 메모리 확장
  - 하드 디스크 드라이브(HDD) – 자기 기록 재료로 덮인 단단한 금속 또는 유리 플래터 (platter)
    - 디스크 표면은 논리적으로 트랙으로 분할되고 섹터로 세분화
    - 디스크 컨트롤러는 장치와 컴퓨터 간의 논리적 상호 작용을 결정
  - 비휘발성 메모리(NVM) 장치 - 하드 디스크보다 빠름
    - SSD(Solid-State Disk)
    - NAND 플래시 메모리
    - 용량 및 성능이 증가함에 따라 가격이 하락
  - 광 디스크
    - CD, DVD, BD

Storage 표기법

- Bit: 컴퓨터 저장소의 기본 단위

- 0과 1의 두 값 중 하나를 포함.

- Byte: 8비트, 가장 작은 편리한 스토리지 청크
- Word: 주어진 컴퓨터 아키텍처의 기본 데이터 단위

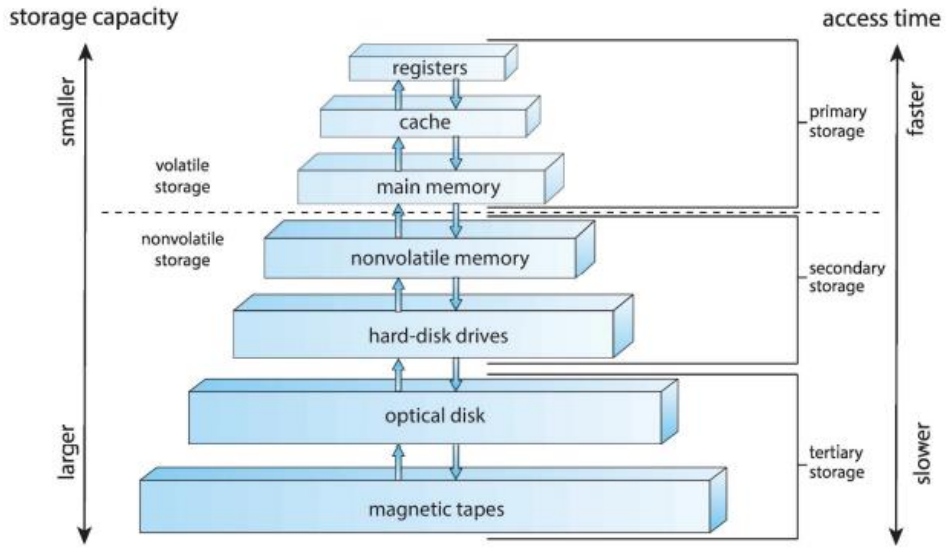
- 1바이트 이상: 64비트 아키텍처 컴퓨터에 64비트(8바이트) 단어가 있음
- 한 번에 바이트가 아닌 네이티브 워드 크기로 많은 작업을 수행

Storage Hierarchy

- 계층 구조로 구성된 스토리지 시스템

- 속도, 비용, 휘발성

- Caching – 정보를 더 빠른 스토리지 시스템에 복사, 메인 메모리를 보조 스토리지용 캐시로 볼 수 있음



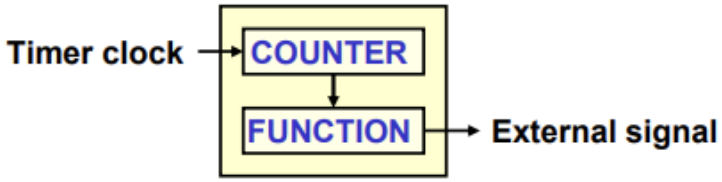
Timer

- 무한 루프에 빠지거나 시스템 서비스를 호출하지 못하고 운영 체제에 제어 권한을 반환하지 않도록 방지하려면

- Timer clock: 수정 발진기(crystal oscillator)에서 생성됨
- Counter: 위 또는 아래

- 기능: interrupt, reset, 일회성 또는 주기적, 주파수 분할기

- 외부 신호: interrupt, reset, 프로그래밍된 신호



- OS에 의해 일정 시간 후 컴퓨터를 중단하도록 타이머가 설정됨
- Linux에서, 일반적인 기간은 4 mS
- 카운터가 타이머 클록에 의해 감소
- 카운터 0이 인터럽트를 생성할 때
- 제어권을 되찾거나 할당된 시간을 초과하는 프로그램을 종료하도록 프로세스 예약 전에 설정

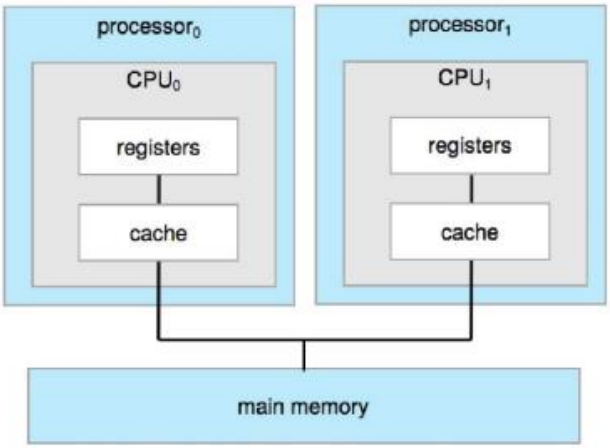
Computer-System Architecture

- Single-processor Systems: 현재 저사양 시스템에 사용
  - 하나의 코어가 있는 single general-purpose processor
  - 대부분의 시스템에는 특수 목적의 프로세서도 있음
- Multiprocessor systems의 사용 및 중요성 증가
  - 병렬 시스템 (parallel systems), 긴밀하게 연결된 (tightly-coupled) 시스템
  - 처리량 증가
  - Economy of scale– 생산량 증가, 비용 감소
  - 신뢰성 향상 – graceful degradation(일부 시스템이 고장이 나타났을 때 시스템을 축소 구성하여 운전을 계속함. 전체적인 시스템이 고장 나지 않도록 하기 위해서) 또는 fault tolerant(결함이 발생해도 부분적으로 기능을 수행)
  - 두 가지 유형:

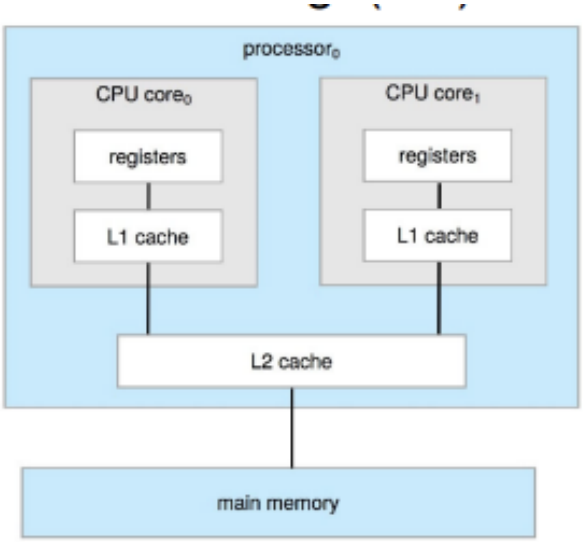
1. 비대칭 멀티프로세싱 – 각 프로세서에 특정 작업이 할당
2. 대칭형 멀티프로세싱 – 각 프로세서가 모든 작업을 수행

Multiprocessing Architecture

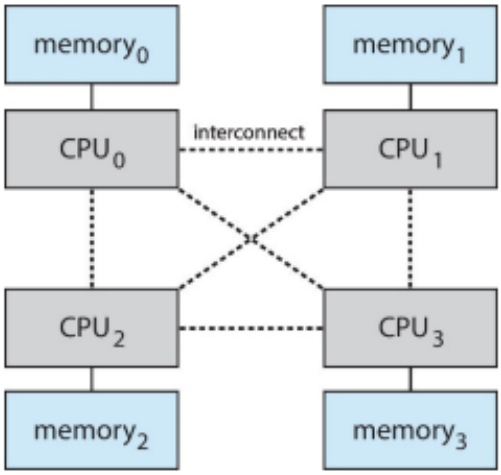
- Symmetric architecture



- Dual-core design(PC)

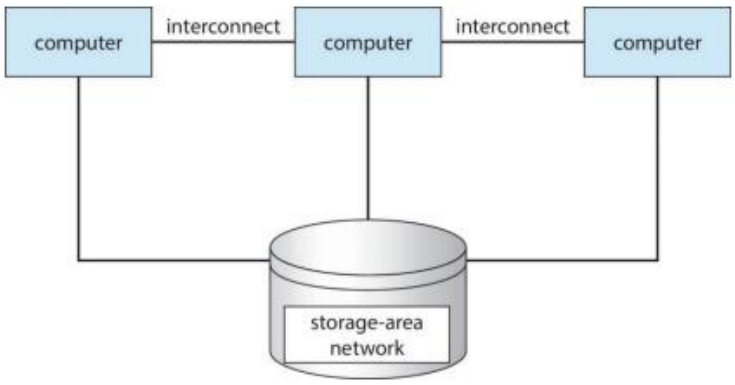


- Non-uniform memory access system



Clustered Systems

- multiprocessor system과 유사하지만 여러 시스템이 함께 작동함
  - storage-area network (SAN)를 통해 storage 공유
  - 고가용성 서비스 제공
    - 비대칭 클러스터링 (Asymmetric clustering)에 hot-standby mode의 시스템이 하나 있음
    - 대칭 클러스터링 (Symmetric clustering)에는 애플리케이션을 실행하고 서로를 모니터링하는 여러 노드가 있음.
  - 일부 클러스터는 고성능 컴퓨팅(HPC)용
    - 병렬화를 사용하려면 응용 프로그램을 작성
  - 충돌하는 작업을 방지하기 위해 DLM(분산 잠금 관리자)이 있음.



Operating-System Operations

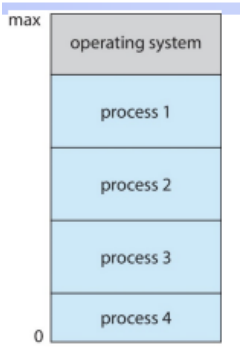
- Bootstrap program – 시스템을 초기화하는 간단한 코드,
  - 펌웨어로 저장
  - 하드웨어 검사 및 초기화
  - 커널 로드
- 커널 로드 후
  - system daemons (커널 외부에서 제공되는 서비스)을 시작
  - 외부 이벤트 대기 중
  - 커널이 인터럽트 기반 (하드웨어 및 소프트웨어)
    - 장치 중 하나에 의한 Hardware interrupt
    - Software interrupt (exception or trap):

    - 소프트웨어 오류(예: 0으로 나누기)
    - 운영 체제 서비스 요청 – system call
    - 무한 루프, 운영 체제를 수정하는 프로세스
- 애플리케이션 실행

Multiprogramming and Multitasking

- 효율성을 위해 Multiprogramming (Batch system) 필요

- 단일 사용자가 CPU 및 I/O 디바이스를 항상 사용할 수 없음
- CPU에 항상 실행할 작업이 있도록 작업(코드 및 데이터)을 구성
- 시스템의 총 작업 중 하위 집합이 메모리에 보관
- 작업 scheduling을 통해 하나의 작업을 선택하고 실행
- 대기해야 하는 경우(예: I/O) OS가 다른 작업으로 전환

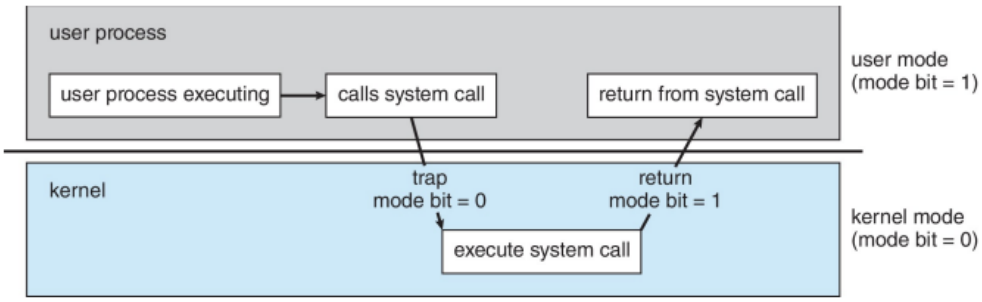


- Timesharing (multitasking) : multiprogramming의 논리적 확장

- CPU 전환 작업이 자주 수행되어 사용자가 각 작업과 상호 작용할 수 있음
    - Interactive computing 만들기
- 응답 시간은 1초 미만
- 사용자는 메모리에 실행 중인 프로그램을 가지고 있음 -> process
- 여러 작업을 동시에 실행할 준비가 된 경우 » CPU scheduling
- 프로세스가 메모리에 맞지 않는 경우 swap을 통해 프로세스가 실행되도록 안팎으로 이동
- 가상 메모리를 사용하면 메모리에 완전히 포함되지 않은 프로세스를 실행할 수 있음.

Dual-mode and Multimode Operation

- 듀얼 모드 작동으로 OS 자체 및 기타 시스템 구성 요소 보호 가능
  - 사용자 모드 및 커널 모드(supervisor, system, privileged)
  - 하드웨어에서 제공하는 Mode bit
- 시스템에서 사용자 코드 또는 커널 코드를 실행할 때 구분
- 커널 모드에서만 실행할 수 있는 privileged 명령으로 지정된 일부 지정
- System call이 모드를 커널로 변경하고, 사용자에게 재설정하여 return



- 점점 더 많은 CPU가 멀티 모드 작업을 지원
  - Intel CPU: 보호 링 4개,
  - ARMv8: 7 모드
  - 즉, 게스트 VMs에 대한 VMM(가상 시스템 관리자) 모드

Process Management

- 프로세스: 실행 중인 프로그램
  - 시스템 내 작업 단위
  - 프로그램은 수동적 엔티티, 프로세스는 능동적 엔티티
- 프로세스를 수행하는 데 리소스가 필요
  - CPU, 메모리, I/O, 파일
  - 초기화 데이터
- 프로세스 종료 시 재사용 가능한 리소스를 회수
- 단일 스레드(single-threaded) 프로세스에는 실행할 다음 명령의 위치를 지정하는 하나의 프로그램 카운터가 있다.
  - 프로세스가 완료될 때까지 한 번에 하나씩 순차적으로 명령을 실행

- 다중 스레드 프로세스에는 스레드당 하나의 프로그램 카운터가 있다.
- 시스템에는 프로세스, 사용자, 운영 체제가 하나 이상의 CPU에서 동시에 실행
  - 프로세스/스레드 간에 CPU를 multiplexing하여 동시 실행

Process Management Activities

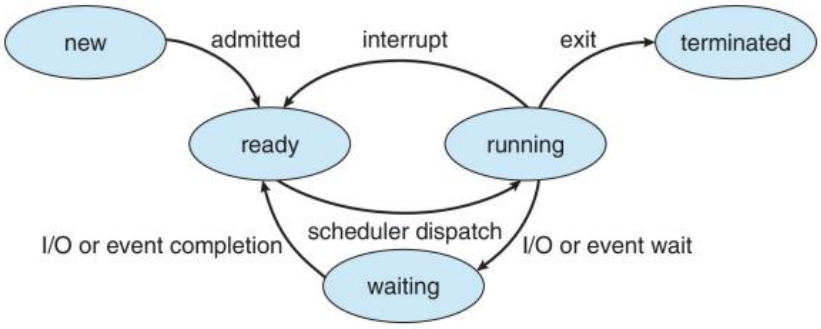
– 사용자 프로세스 및 시스템 프로세스 모두 생성 및 삭제

– 프로세스 일시 중단 및 재개

– 프로세스 동기화를 위한 메커니즘 제공

– 프로세스 통신을 위한 메커니즘 제공

– 교착 상태 처리(deadlock handling)를 위한 메커니즘 제공



Memory Management

- 프로그램을 실행
  - 지침의 모든(또는 일부) 및 데이터가 메모리에 있어야 합니다.
- CPU 활용률 및 사용자에 대한 시스템 응답을 향상
  - 여러 프로그램을 메모리에 보관하여 메모리 관리 필요성 발생
  - 메모리 관리는 메모리에 있는 항목과 시기를 결정합니다.
- 메모리 관리 활동
  - 메모리의 현재 사용 중인 부분과 사용 중인 프로세스 추적
  - 메모리 안팎으로 이동할 프로세스(또는 일부) 및 데이터 결정
  - 필요에 따라 메모리 공간 할당 및 할당 해제

File-System Management

- OS는 정보 스토리지에 대한 통일적이고 논리적 뷰를 제공
  - 스토리지의 물리적 속성을 논리적 스토리지 단위로 추상화 - file.
  - 각 매체는 장치(예: 디스크 드라이브, 테이프 드라이브)에 의해 제어
  - 다양한 속성에는 액세스 속도, 용량, 데이터 전송 속도, 액세스 방법(순차 또는 임의)이 포함
- 파일 시스템 관리
  - 디렉터리(폴더)로 구성
  - 대부분의 시스템에서 액세스 제어를 통해 누가 액세스할 수 있는지 결정
  - OS 활동
    - 파일 및 디렉터리 만들기 및 삭제
    - 파일 및 디렉터리 조작을 위한 기본 요소
    - 대용량 저장소에 파일 매핑
    - 안정적인(비휘발성) 저장 매체에 파일 백업

Mass-Storage Management

- 디스크는 메인 메모리에 맞지 않는 데이터나 "장시간" 동안 보관해야 하는 데이터를 저장하는 데 사용됨
- 컴퓨터 성능은 디스크 서브시스템과 알고리즘에 달려 있다.
- OS 활동
  - Mounting, Unmounting
  - 여유 공간 관리
  - 스토리지 할당
  - Disk Scheduling
  - Partitioning
  - 보호
- 일부 스토리지는 빠를 필요가 없음.
  - 3차 스토리지에는 광학 스토리지, 자기 테이프 등이 포함됩니다.
  - OS 또는 애플리케이션에 의해 여전히 관리되어야 함

Caching

- 컴퓨터의 여러 수준에서 수행되는 중요한 원칙(하드웨어, 운영 체제, 소프트웨어)
- 사용 중인 정보가 느린 스토리지에서 빠른 스토리지로 일시적으로 복사됨
- 더 빠른 스토리지(캐시 cache)를 먼저 확인하여 정보가 있는지 확인
  - 이 경우 캐시에서 직접 사용되는 정보(빠른 속도)
  - 그렇지 않은 경우 데이터가 캐시에 복사되어 캐시에 사용됨
- 캐시되는 스토리지보다 작은 캐시
  - 캐시 관리 중요 설계 문제
  - 캐시 크기 및 교체 정책

다양한 유형의 스토리지 특성

Level	1	2	3	4	5
Name	registers	cache	main memory	solid-state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25-0.5	0.5-25	80-250	25,000-50,000	5,000,000
Bandwidth (MB/sec)	20,000-100,000	5,000-10,000	1,000-5,000	500	20-150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape

- 스토리지 계층 간의 이동은 명시적 또는 암묵적

Migration of data “A” from Disk to Register

- 멀티태스킹 환경에서는 스토리지 계층에서 저장되는 위치에 관계없이 최신 값을 사용
- 멀티프로세서 환경에서는 모든 CPU가 캐시에서 최신 값을 가질 수 있도록 하드웨어에 캐시 일관성(cache coherency)을 제공해야 합니다.
- 분산 환경 상황은 더욱 복잡해짐
  - 기준점의 여러 복사본이 존재할 수 있습니다.

I/O Subsystem

- 사용자에게 하드웨어 장치의 특수성 숨김
- I/O의 메모리 관리
- Buffering (데이터를 전송하는 동안 일시적으로 저장),
- Caching (성능을 위해 데이터 일부를 더 빠른 스토리지에 저장)
- Spooling (작업의 출력이 다른 작업의 입력과 중복됨)
- 일반 장치-드라이버 인터페이스
- 특정 하드웨어 장치용 드라이버

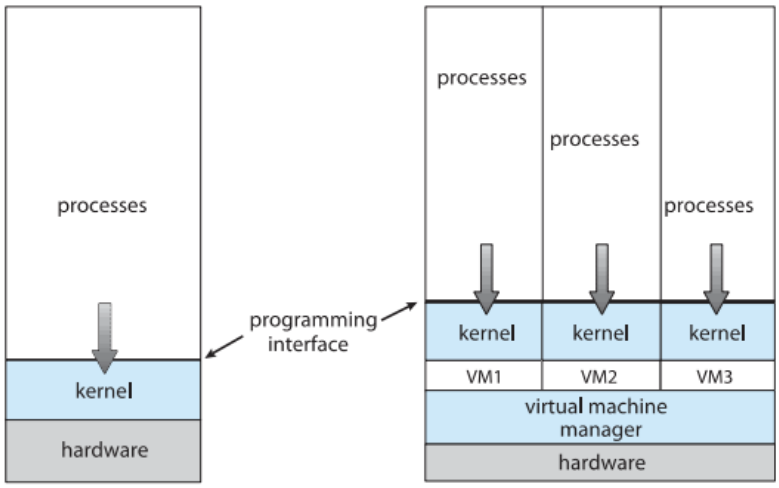
Protection and Security

- 보호(Protection) – OS에서 정의한 리소스에 대한 프로세스 또는 사용자의 액세스를 제어하는 모든 메커니즘
- 보안(Security) – 내부 및 외부 공격에 대한 시스템 방어
- 서비스 거부, 웜, 바이러스, 신원 도용, 서비스 도용을 포함한 광범위한 범위
- 시스템은 일반적으로 먼저 사용자를 구분하여 누가 무엇을 할 수 있는지 결정합니다.
- 사용자 ID(사용자 ID, 보안 ID)에는 이름과 관련 번호가 포함됩니다(사용자당 하나씩).
- 액세스 제어를 결정하기 위한 해당 사용자의 프로세스, 모든 파일과 연결된 사용자 ID
- 그룹 식별자(그룹 ID)를 통해 사용자 집합을 정의하고 제어한 후 각 프로세스, 파일에도 연결할 수 있습니다.
- 권한 확대를 통해 사용자는 더 많은 권한을 가진 유효한 ID로 변경할 수 있습니다.

Virtualization

- 단일 컴퓨터의 하드웨어를 여러 다른 실행 환경으로 추상화
- OS가 다른 OS 내에서 애플리케이션으로 실행되도록 허용
- 소스 CPU 유형이 대상 유형과 다를 때 사용되는 에뮬레이션
- Ex) Intel x86에서 ARM으로의 Apple CPU 마이그레이션
- 가장 느린 방법
- 컴퓨터 언어가 네이티브 코드로 컴파일되지 않은 경우 – Interpretation

- 가상화(Virtualization) – CPU용으로 기본적으로 컴파일된 OS, 실행 중인 게스트 OS도 기본적으로 컴파일됨



- 사용 사례: 탐색 또는 호환성을 위한 여러 OS
  - Mac OS X 호스트를 실행하는 Apple 노트북, 게스트로 Windows
  - 여러 시스템을 사용하지 않고 여러 OS용 앱 개발
  - 여러 시스템을 사용하지 않고 애플리케이션을 테스트하는 QA
  - 데이터 센터 내 컴퓨팅 환경 실행 및 관리

Distributed Systems

- 분산 컴퓨팅 (Distributed computing)
  - 서로 네트워크로 연결된 개별 또는 이종 시스템 모음
- 네트워크는 통신 경로이며 TCP/IP가 가장 일반적입니다.
  - LAN (Local Area Network)
  - 광역 네트워크(WAN)
  - 대도시 지역 네트워크(MAN)
  - 개인 영역 네트워크(PAN)
  - 네트워크 운영 체제는 네트워크를 통해 시스템 간에 기능을 제공합니다.
- 통신 체계를 통해 시스템이 메시지를 교환할 수 있습니다.
- 단일 시스템의 환상

Computing Environments - 기존 방식

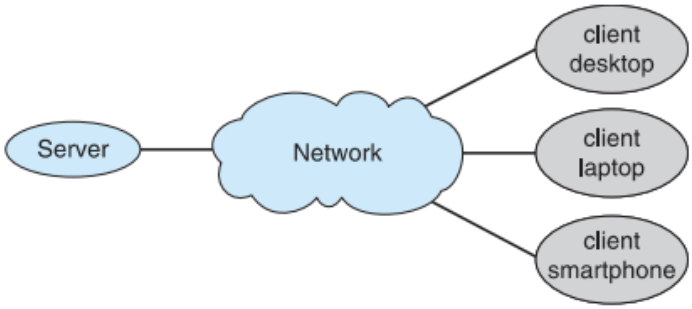
- 독립 실행형 범용 시스템
- 그러나 대부분의 시스템이 다른 시스템(즉, 인터넷)과 상호 연결
- 포털은 내부 시스템에 대한 웹 액세스를 제공합니다.
- 네트워크 컴퓨터(thin clients)는 웹 터미널과 같습니다.
- 무선 네트워크를 통해 상호 연결된 모바일 컴퓨터
- 어디서나 볼 수 있는 네트워킹 (ubiquitous) – 가정용 시스템도 방화벽을 사용하여 인터넷 공격으로부터 가정용 컴퓨터를 보호합니다.

Computing Environments – Mobile

- 휴대용 스마트폰, 태블릿 등
  - 가동 및 경량
- "기존" 노트북과 기능적인 차이는 무엇일까?
  - 작은 화면과 제한된 사용자 인터페이스
  - 정보 브라우징, 게임, 비디오/오디오 처리, 통신, 콘텐츠 생성 등
  - 추가 기능 – GPS, 자이로스코프
- 내비게이션, 게임 인터페이스는 증강 현실과 같은 새로운 유형의 앱을 가능하게 함
  - 연결에 IEEE 802.11 무선 또는 셀룰러 데이터 네트워크 사용
- Apple iOS와 Google Android가 선도

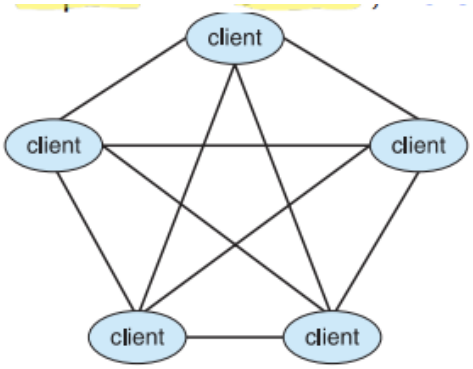
Computing Environments – Client-Server

- Client-Server Computing
  - 스마트 PC로 대체된 dumb terminals
  - 많은 시스템이 서버에서 클라이언트가 생성한 요청에 응답합니다.
- 컴퓨팅 서버 시스템은 클라이언트에게 서비스(즉, 데이터베이스)를 요청할 수 있는 인터페이스를 제공합니다.
- 파일 서버 시스템은 클라이언트가 파일을 저장하고 검색할 수 있는 인터페이스를 제공합니다.



Computing Environments - Peer-to-Peer

- 분산 시스템의 다른 모델
- P2P는 클라이언트와 서버를 구분하지 않는다.
  - 대신 모든 노드가 peer로 간주됨.
  - 각각 클라이언트, 서버 또는 둘 다 역할을 할 수 있음
  - 노드가 P2P 네트워크에 가입.
    - 네트워크의 중앙 조회 서비스에 서비스를 등록.
    - 서비스 요청을 broadcast하고 검색 프로토콜을 통해 서비스 요청에 응답
  - 예를 들어 Napster와 Gnutella, Skype와 같은 VoIP(Voice over IP) 등이 있다.

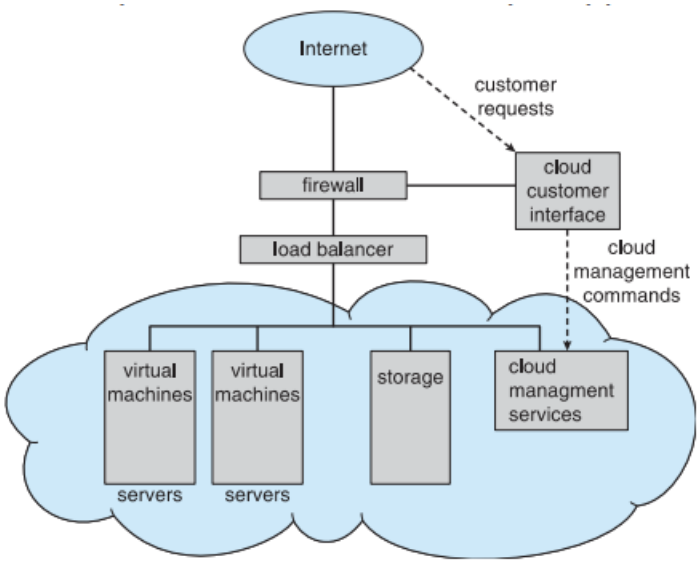


Computing Environments – Cloud Computing

- 네트워크를 통해 컴퓨팅, 스토리지, 애플리케이션을 서비스로 제공
- 가상화를 기능의 기반으로 사용하기 때문에 가상화의 논리적 확장입니다.
  - Amazon EC2에는 수천 대의 서버, 수백만 대의 가상 머신, 페타바이트급 스토리지가 있으며 사용량에 따라 비용을 지불합니다.
- 종류

- Public cloud – 지불 의사가 있는 모든 사람이 인터넷을 통해 사용 가능
- Private cloud – 회사 자체 사용을 위해 회사에서 운영
- Hybrid cloud – public 및 private 클라우드 구성 요소 모두 포함
- SaaS(서비스로서의 소프트웨어) – 인터넷을 통해 사용할 수 있는 하나 이상의 애플리케이션(예: 워드 프로세서)
- 서비스형 플랫폼(PaaS) – 인터넷을 통해 애플리케이션을 사용할 수 있는 소프트웨어 스택(즉, 데이터베이스 서버)
- IaaS(서비스형 인프라) – 인터넷을 통해 사용 가능한 서버 또는 스토리지(즉, 백업용으로 사용 가능한 스토리지)

- 기존 OS와 VMM, 클라우드 관리 툴로 구성된 클라우드 컴퓨팅 환경
- 인터넷 연결에는 방화벽과 같은 보안이 필요.
- 로드 밸런싱 장치가 여러 애플리케이션에 걸쳐 트래픽을 분산시킵니다.



Computing Env. – Real-Time Embedded Systems

- 가장 일반적인 형태의 컴퓨터 실시간 임베디드 시스템
- 다양한, 특수 목적, 제한된 목적의 OS
- 실시간 OS
- 확장 사용
- 일부는 OS를 사용하고 일부는 OS 없이 작업을 수행.

- 실시간 OS에 명확한 고정 시간 제약이 있음
  - 처리는 제약 조건 내에서 수행.
  - 제약 조건이 충족된 경우에만 올바른 작동