

혼자 공부하며 함께 만드는

---

혼공 용어 노트

# 목차

가나다 순

경로 path	24	보조기억장치 secondary storage	06
고급 언어 high-level programming language	08	부모 프로세스 parent process	18
고립형 입출력 isolated I/O	16	비순차적 명령어 처리 기법 OoOE; Out-of-order execution	11
공유 자원 shared resource	20	비트 bit	07
교착 상태 deadlock	21	비휘발성 저장장치 non-volatile memory	12
교착 상태 검출 후 회복 deadlock detection & recovery	21	산술연산장치 ALU; Arithmetic Logic Unit	09
교착 상태 예방 deadlock prevention	21	색인 할당 indexed allocation	24
교착 상태 회피 deadlock avoidance	21	선점형 스케줄링 preemptive scheduling	19
논리 주소 logical address	13	세마포 semaphore	20
대기 큐 waiting queue	19	섹터 sector	14
데이터 data	06	슈퍼스칼라 superscalar	11
데이터 접근 시간 data access time	15	스래싱 thrashing	23
동기화 synchronization	20	스레드 thread	11
디렉터리 directory	24	스왑핑 swapping	21
레지스터 register	09	스택 주소 지정 방식 stack addressing mode	10
멀티스레드 multithread	19	시스템 버스 system bus	07
멀티스레드 프로세서 multithread processor	11	시스템 호출 system call	17
멀티프로세스 multi-process	19	실린더 cylinder	14
메모리 관리 장치 MMU; Memory Management Unit	13	십육진법 hexadecimal	07
메모리 맵 입출력 memory-mapped I/O	16	연결 할당 linked allocation	24
메인보드 main board	07	연속 할당 contiguous allocation	24
명령어 instruction	06	예외 exception	10
명령어 사이클 instruction cycle	10	외부 단편화 external fragmentation	22
명령어 집합 구조 ISA; Instruction Set Architecture	12	요구 페이징 demand paging	23
명령어 파이프라이닝 instruction pipelining	11	우선순위 priority	19
모니터 monitor	20	운영체제 operating system	17
문맥 교환 context switching	18	유닉스 파일 시스템 UNIX file system	24
문자 집합 character set	07	이중 모드 dual mode	17
물리 주소 physical address	13	이진법 binary	07
뮤텍스 락 mutex lock	20	인터럽트 interrupt	10
변위 주소 지정 방식 displacement addressing mode	10	인터럽트 기반 입출력 interrupt-driven I/O	16
변환 색인 버퍼 TLB; Translation Lookaside Buffer	23		

인터프리터 언어 interpreter language	08	프레임 할당 frame allocation	23
임계 구역 critical section	20	프로그램 입출력 programmed I/O	16
입출력 버스 input/output bus	16	프로세스 process	17
입출력장치 I/O(input/output) device	06	프로세스 상태 process state	18
장치 드라이버 device driver	15	프로세스 제어 블록 PCB; Process Control Block	18
장치 컨트롤러 device controller	15	플래터 platter	14
저장 장치 계층 구조 memory hierarchy	13	하드웨어 인터럽트 hardware interrupt	10
제어장치 Control Unit	09	한계 레지스터 limit register	13
주기억장치 main memory	06	휘발성 저장장치 volatile memory	12
주소 지정 방식 addressing mode	09		
준비 큐 ready queue	19		
중앙처리장치 CPU; Central Processing Unit	06		
참조 지역성의 원리 locality of reference	14		
최악 적합 worst fit	22		
최적 적합 best fit	22		
최초 적합 first fit	22		
캐시 메모리 cache memory	14		
커널 kernel	17		
컴파일 언어 compile language	08		
코어 core	11		
클럭 속도 clock speed	11		
트랙 track	14		
파일 file	23		
파일 속성 file attribute	24		
파티셔닝 partitioning	24		
페이지 교체 알고리즘 page replacement algorithm	23		
페이지 테이블 page table	22		
페이지 테이블 베이스 레지스터			
PTBR; Page Table Base Register	22		
페이지 폴트 page fault	23		
페이징 paging	22		
포매팅 formatting	24		

# 목차

ABC 순

addressing mode 주소 지정 방식	09	directory 디렉터리	24
ALU; Arithmetic Logic Unit 산술연산장치	09	displacement addressing mode 변위 주소 지정 방식	10
best fit 최적 적합	22	DRAM; Dynamic Random Access Memory	12
binary 이진법	07	dual mode 이중 모드	17
bit 비트	07	exception 예외	10
cache memory 캐시 메모리	14	exec system call exec 시스템 호출	18
character set 문자 집합	07	external fragmentation 외부 단편화	22
CISC; Complex Instruction Set Computer	12	FAT file system FAT 파일 시스템	24
clock speed 클럭 속도	11	file 파일	23
compile language 컴파일 언어	08	file attribute 파일 속성	24
context switching 문맥 교환	18	first fit 최초 적합	22
contiguous allocation 연속 할당	24	fork system call fork 시스템 호출	18
Control Unit 제어장치	09	formatting 포매팅	24
core 코어	11	frame allocation 프레임 할당	23
CPU scheduling CPU 스케줄링	19	hardware interrupt 하드웨어 인터럽트	10
CPU scheduling algorithm CPU 스케줄링 알고리즘	19	hexadecimal 십육진법	07
CPU; Central Processing Unit 중앙처리장치	06	high-level programming language 고급 언어	08
critical section 임계 구역	20	I/O(input/output) device 입출력장치	06
cylinder 실린더	14	indexed allocation 색인 할당	24
data 데이터	06	input/output bus 입출력 버스	16
data access time 데이터 접근 시간	15	instruction 명령어	06
DDR SDRAM; Double Data Rate SDRAM	13	instruction cycle 명령어 사이클	10
deadlock 교착 상태	21	instruction pipelining 명령어 파이프라이닝	11
deadlock avoidance 교착 상태 회피	21	interpreter language 인터프리터 언어	08
deadlock detection & recovery 교착 상태 검출 후 회복	21	interrupt 인터럽트	10
deadlock prevention 교착 상태 예방	21	interrupt-driven I/O 인터럽트 기반 입출력	16
demand paging 요구 페이징	23	ISA; Instruction Set Architecture 명령어 집합 구조	12
device controller 장치 컨트롤러	15	isolated I/O 고립형 입출력	16
device driver 장치 드라이버	15	kernel 커널	17
Direct Memory Access I/O DMA 입출력	16		

limit register 한계 레지스터	13	process 프로세스	17
linked allocation 연결 할당	24	process state 프로세스 상태	18
locality of reference 참조 지역성의 원리	14	programmed I/O 프로그램 입출력	16
logical address 논리 주소	13	PTBR; Page Table Base Register	
main board 메인보드	07	페이지 테이블 베이스 레지스터	22
main memory 주기억장치	06	RAID; Redundant Array of Independent Disks	15
memory-mapped I/O 메모리 맵 입출력	16	ready queue 준비 큐	19
memory hierarchy 저장 장치 계층 구조	13	register 레지스터	09
MMU; Memory Management Unit		RISC; Reduced Instruction Set Computer	12
메모리 관리 장치	13	SDRAM; Synchronous DRAM	13
monitor 모니터	20	secondary storage 보조기억장치	06
multi-process 멀티프로세스	19	sector 섹터	14
multithread 멀티스레드	19	semaphore 세마포	20
multithread processor 멀티스레드 프로세서	11	shared resource 공유 자원	20
mutex lock 뮉텍스 락	20	SLC; Single Level Cell	15
non-volatile memory 비휘발성 저장장치	12	SRAM; Static Random Access Memory	12
OoOE; Out-of-order execution		stack addressing mode 스택 주소 지정 방식	10
비순차적 명령어 처리 기법	11	superscalar 슈퍼스칼라	11
operating system 운영체제	17	swapping 스와핑	21
page fault 페이지 폴트	23	synchronization 동기화	20
page replacement algorithm 페이지 교체 알고리즘	23	system bus 시스템 버스	07
page table 페이지 테이블	22	system call 시스템 호출	17
paging 페이징	22	thrashing 스래싱	23
parent process 부모 프로세스	18	thread 스레드	11
partitioning 파티셔닝	24	TLB; Translation Lookaside Buffer	
path 경로	24	변환 색인 버퍼	23
PCB; Process Control Block 프로세스 제어 블록	18	track 트랙	14
physical address 물리 주소	13	UNIX file system 유닉스 파일 시스템	24
platter 플래터	14	volatile memory 휘발성 저장장치	12
preemptive scheduling 선점형 스케줄링	19	waiting queue 대기 큐	19
priority 우선순위	19	worst fit 최악 적합	22

# 01 장 <sup>✓</sup> 컴퓨터 구조 시작하기

□ 데이터	<b>data</b> [01장 37쪽]
	컴퓨터와 주고받는 숫자, 문자, 이미지, 동영상과 같은 정보나 컴퓨터에 저장된 정보
□ 명령어	<b>instruction</b> [01장 37쪽]
	데이터를 움직이고 컴퓨터를 실질적으로 동작시키는 정보
□ 주기억장치	<b>main memory</b> <span style="background-color: #f08080;">참고 용어</span> 메모리 [01장 40쪽]
	실행되는 프로그램의 명령어와 데이터를 저장하는 부품 크게 RAM(Random Access Memory)과 ROM(Read Only Memory) 두 가지가 있으며, 메모리라는 용어는 보통 RAM을 지칭
□ 중앙처리장치	<b>CPU; Central Processing Unit</b> [01장 41쪽]
	메모리에 저장된 명령어를 읽어 들이고, 읽어 들인 명령어를 해석하고, 실행하는 부품 <u>CPU 내부 구성 요소 중 가장 중요한 세 가지</u> → 산술연산장치, 레지스터, 제어장치
□ 보조기억장치	<b>secondary storage</b> [01장 45쪽]
	전원이 꺼져도 저장된 내용을 기억할 수 있는 장치 하드 디스크 드라이브, SSD, USB 메모리, DVD, CD-ROM 등
□ 입출력장치	<b>I/O(input/output) device</b> [01장 46쪽]
	컴퓨터 외부에 연결되어 컴퓨터 내부와 정보를 교환할 수 있는 장치 마이크, 스피커, 프린터, 마우스, 키보드 등

□ 메인보드 main board [01장 47쪽]  
 여러 컴퓨터 부품을 부착할 수 있는 부품  
 마더보드(mother board)라고도 부름

□ 시스템 버스 system bus [01장 47쪽]  
 컴퓨터 네 가지 핵심 부품(CPU, 메모리, 보조기억장치, 입출력장치)을 연결하는  
 가장 주요한 버스  
 → 주소 버스, 데이터 버스, 제어 버스

## 02장 데이터

CPU가 한 번에 처리할 수 있는 데이터 크기. CPU가 처리할 수 있는 비트 수에 따라 워드 크기는 달라질 수 있음

□ 비트 bit **참고 용어** 워드(word) [02장 55쪽]

0과 1을 나타내는 가장 작은 정보 단위. 여덟 개의 비트를 묶어 바이트라고 함

1바이트(1byte) = 8비트(8bit)  
 1킬로바이트(1KB) = 1,000바이트(1,000byte)  
 1메가바이트(1MB) = 1,000킬로바이트(1,000KB)  
 1기가바이트(1GB) = 1,000메가바이트(1,000MB)  
 1테라바이트(1TB) = 1,000기가바이트(1,000GB)

□ 이진법 binary **참고 용어** 이진수 [02장 57쪽]

1을 넘어가는 시점에 자리 올림하여 0과 1만으로 모든 숫자를 표현하는 방법

이진수. 십진수 8을 이진수로 표현하면 1000('일영영영'으로 읽습니다)

□ 십육진법 hexadecimal [02장 60쪽]

15를 넘어가는 시점에 자리 올림하여 수를 표현하는 방법

십육진법 체계에서는 10, 11, 12, 13, 14, 15를 A, B, C, D, E, F로 표기

□ 문자 집합 character set [02장 67쪽]

컴퓨터가 인식하고 표현할 수 있는 문자들의 모음

- 문자 인코딩: 문자 집합에 속한 문자를 컴퓨터가 이해할 수 있는 0과 1로 변환하는 것
- 문자 디코딩: 인코딩의 반대. 0과 1로 이루어진 문자 코드를 사람이 이해할 수 있는 문자로 변환하는 것

## 03 장 명령어

### □ 고급 언어

C, C++, Java, Python 등

high-level programming language

[03장 79쪽]

프로그램을 만들 때 컴퓨터가 이해하는 언어가 아닌 사람이 이해하고 작성하기 쉽게 만들어진 언어 ↔ 저급 언어(low-level programming language)

기계어, 어셈블리어

### □ 컴파일 언어

compile language

[03장 84쪽]

컴파일러에 의해 소스 코드 전체가 저급 언어로 변환되어 실행되는 고급 언어

### □ 인터프리터 언어

interpreter language

[03장 84쪽]

인터프리터에 의해 소스 코드를 한 줄씩 저급 언어로 변환하여 실행하는 고급 언어

#### 그것이 알고싶다 컴파일러 vs 인터프리터

인터프리터	컴파일러
컴퓨터와 대화하듯 한 줄씩 실행	소스 코드 전체를 저급 언어로 변환하여 실행
N번째 줄에 문법 오류가 있어도 N-1번째까지는 올바르게 수행	코드 내에 오류가 하나라도 있으면 컴파일 불가
대표 언어 Python	대표 언어 C



□ 명령어

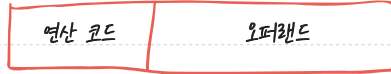
instruction

[03장 91쪽]

연산 코드와 오퍼랜드로 구성

명령어가 수행할 연산 = 연산자

연산에 사용할 데이터 혹은 저장된 위치 = 피연산자



□ 주소 지정 방식

addressing mode

[03장 95쪽]

오퍼랜드 필드에 데이터가 저장된 위치를 명시할 때 연산에 사용할 데이터 위치를 찾는 방법

## 04장 CPU의 작동 원리

□ 산술연산장치

ALU; Arithmetic Logic Unit

[04장 105쪽]

CPU 내부에서 레지스터와 제어장치로부터 받아들인 피연산자와 제어 신호로 산술 연산, 논리 연산 등 다양한 연산을 수행하는 장치

□ 제어장치

Control Unit

[04장 107쪽]

제어 신호를 내보내고 명령어를 해석하는 부품

- 제어장치가 받아들이는 정보: 클럭 신호, 명령어, 플래그 값, 제어 신호 등

□ 레지스터

register

[04장 113쪽]

프로그램 속 명령어와 데이터가 실행 전후로 저장되는 CPU 내부의 작은 임시 저장 장치

- 레지스터의 종류: 프로그램 카운터, 명령어 레지스터, 메모리 주소 레지스터, 메모리 버퍼 레지스터, 범용 레지스터, 플래그 레지스터 등

□ 스택 주소

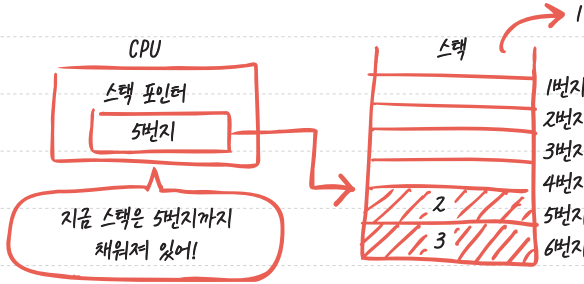
stack addressing mode

[04장 118쪽]

지정 방식

스택과 스택 포인터를 이용한 주소 지정 방식

한쪽 끝이 막혀 있는 통과 같은 저장 공간으로 나  
중에 저장한 데이터를 가장 먼저 빼내는 데이터  
관리 방식(푸입선출, LIFO; Last In First Out)



□ 변위 주소

displacement addressing mode

[04장 120쪽]

지정 방식

오퍼랜드 필드의 값과 특정 레지스터 값을 더해 유효 주소를 얻어내는 방식

□ 명령어 사이클

instruction cycle

[04장 127쪽]

하나의 명령어가 처리되는 주기  
인출, 실행, 간접, 인터럽트 사이클로 구성

□ 인터럽트

interrupt

[04장 129쪽]

CPU의 정상적인 작업을 방해하는 신호  
• 동기 인터럽트 = 예외, 비동기 인터럽트 = 하드웨어 인터럽트

□ 하드웨어  
인터럽트

hardware interrupt

[04장 131쪽]

입출력장치에 의해 발생하는 인터럽트  
세탁기 완료 알림, 전자레인지 조리 완료 알림과 같이 알림 역할을 하는 인터럽트

□ 예외

exception

[04장 138쪽]

CPU가 명령어들을 수행하다가 프로그래밍 오류와 같은 예상치 못한 상황에 마주  
쳤을 때 발생하는 인터럽트 → 폴트, 트랩, 중단, 소프트웨어 인터럽트

# 05장 CPU 성능 향상 기법

- 클럭 속도      clock speed      참고 용어      클럭      [05장 145쪽]  
1초에 클럭이 몇 번 반복되는지를 나타내는 단위. 헤르츠(Hz)로 측정  
클럭 속도가 높은 CPU가 일반적으로 빠르게 동작  
컴퓨터의 모든 부품을 일사불란하게 움직일 수 있게 하는 시간 단위
- 코어      core      [05장 147쪽]  
CPU 내에서 명령어를 실행하는 부품 → 멀티코어 프로세서란 여러 개의 코어를 포함하고 있는 CPU
- 스레드      thread      [05장 148쪽]  
실행 흐름의 단위로, 하드웨어적 스레드와 소프트웨어적 스레드가 있다.
- 멀티스레드 프로세서      multithread processor      [05장 149쪽]  
멀티스레드 CPU라고도 하며, 하나의 코어로 여러 명령어를 동시에 처리하는 CPU
- 명령어 파이프라이닝      instruction pipelining      [05장 158쪽]  
동시에 여러 개의 명령어를 겹쳐 실행하는 기법
- 슈퍼스칼라      superscalar      [05장 160쪽]  
여러 개의 명령어 파이프라인을 두는 기법
- 비순차적 명령어 처리 기법      OoOE; Out-of-order execution      [05장 161쪽]  
파이프라인의 중단을 방지하기 위해 명령어를 순차적으로 처리하지 않는 기법

□ 명령어 집합 구조      **ISA; Instruction Set Architecture**      [05장 167쪽]  
CPU의 언어이자 하드웨어가 소프트웨어를 어떻게 이해할지에 대한 약속

□ CISC      **Complex Instruction Set Computer**      [5장 170쪽]  
CISC는 복잡하고 다양한 수의 **가변 길이** 명령어 집합을 활용  
    ↓  
    파이프라이징에 불리

□ RISC      **Reduced Instruction Set Computer**      [5장 172쪽]  
RISC는 단순하고 적은 수의 **고정 길이** 명령어 집합을 활용  
    ↓  
    파이프라이징에 유리

## 06장 <sup>✓</sup> 메모리와 캐시 메모리

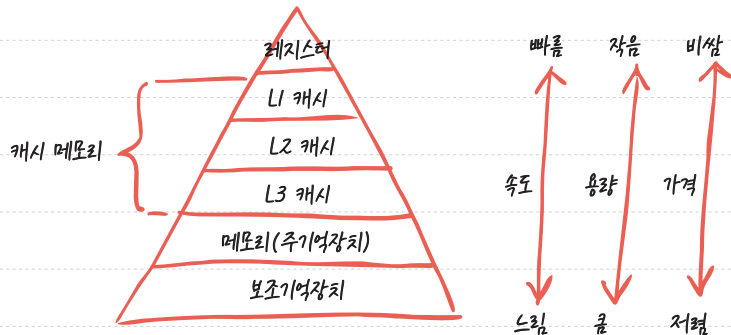
□ 휘발성 저장장치      **volatile memory**      [6장 179쪽]  
전원을 끄면 저장된 내용이 사라지는 저장 장치

□ 비휘발성 저장장치      **non-volatile memory**      [6장 179쪽]  
전원이 꺼져도 저장된 내용이 유지되는 저장 장치

□ DRAM      **Dynamic Random Access Memory**      [6장 181쪽]  
시간이 지나면 저장된 데이터가 점차 사라지는 RAM  
    ↓  
    주 기억장치로 사용

□ SRAM      **Static Random Access Memory**      [6장 182쪽]  
시간이 지나도 저장된 데이터가 사라지지 않는 RAM  
    ↓  
    캐시 메모리로 사용

□ SDRAM	Synchronous DRAM 클럭과 동기화된 DRAM	[6장 182쪽]
□ DDR SDRAM	Double Data Rate SDRAM SDR SDRAM에 비해 대역폭이 두 배 넓은 SDRAM 한 클럭에 한 번 데이터를 주고받는 RAM	[6장 183쪽]
□ 물리 주소	physical address 메모리 하드웨어상의 주소	[6장 187쪽]
□ 논리 주소	logical address CPU와 실행 중인 프로그램이 사용하는 주소	[6장 187쪽]
□ 메모리 관리 장치	MMU; Memory Management Unit 논리 주소를 물리 주소로 변환하는 장치	[6장 189쪽]
□ 한계 레지스터	limit register 실행 중인 프로그램의 논리 주소의 최대 크기를 저장 for. 메모리 보호	[6장 191쪽]
□ 저장 장치 계층 구조	memory hierarchy 각기 다른 용량과 성능의 저장 장치들을 계층화하여 표현한 구조	[6장 197쪽]



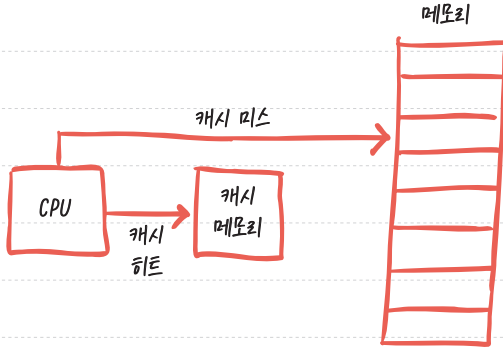
캐시 적중률이 높으면 CPU의 메모리 접근 횟수를 줄일 수 있다.

□ 캐시 메모리

cache memory **참고 용어** 캐시 적중률

[6장 198쪽]

CPU의 연산 속도와 메모리 접근 속도의 차이를 줄이기 위한 저장 장치



□ 참조 지역성의 원리

locality of reference

[6장 202쪽]

CPU가 메모리에 접근할 때의 주된 경향을 바탕으로 만들어진 원리

- CPU가 최근에 접근했던 메모리 공간에 다시 접근하려는 경향
- CPU가 접근한 메모리 공간 근처를 접근하려는 경향

# 07장 보조기억장치

□ 플래터

platter

[7장 209쪽]

하드 디스크에서 데이터가 저장되는 원판

□ 트랙

track

[7장 210쪽]

플래터를 여러 동심원으로 나누었을 때 그중 하나의 원

□ 섹터

sector

[7장 210쪽]

트랙을 여러 조각으로 나눈 한 조각. 하드 디스크의 가장 작은 전송 단위

□ 실린더

cylinder

[7장 210쪽]

여러 겹의 플래터상에서 같은 트랙이 위치한 곳을 모아 연결



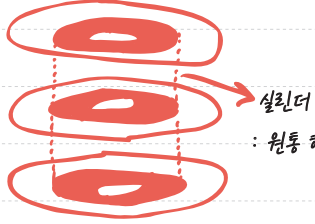
플래터



트랙



섹터



실린더

: 원통 형태의 논리적 단위

□ 데이터 접근 시간	<b>data access time</b> <span style="float: right;">[7장 211쪽]</span> 탐색 시간, 회전 지연, 전송 시간으로 구분 <ul style="list-style-type: none"> <li>• 탐색 시간: 헤드를 접근하고자 하는 데이터가 저장된 트랙까지 이동하는 시간</li> <li>• 회전 지연: 헤드가 있는 곳으로 플래터를 회전시키는 시간</li> <li>• 전송 시간: 하드 디스크와 컴퓨터 간에 데이터를 전송하는 시간</li> </ul>
-------------	---

□ SLC	<b>Single Level Cell</b> <span style="color: red; border: 1px solid red; border-radius: 5px; padding: 2px;">참고 용어</span> MLC, TLC <span style="float: right;">[7장 215쪽]</span> 한 셀에 <u>한 비트</u> 를 저장할 수 있는 플래시 메모리의 종류
-------	---

□ RAID	<b>Redundant Array of Independent Disks</b> <span style="float: right;">[8장 221쪽]</span> 데이터의 안전성 혹은 성능을 위해 여러 하드 디스크나 SSD를 하나의 장치처럼 사용하는 기술
--------	---

## 08장 장 입출력장치

□ 장치 컨트롤러	<b>device controller</b> <span style="float: right;">[8장 234쪽]</span> 입출력장치와 CPU 사이의 통신을 증개하는 장치
-----------	---

□ 장치 드라이버	<b>device driver</b> <span style="float: right;">[8장 236쪽]</span> 장치 컨트롤러가 컴퓨터 내부와 정보를 주고받을 수 있게 하는 프로그램
-----------	---

<p>□ 프로그램 입출력</p>	<p>programmed I/O [8장 241쪽]</p> <p>프로그램 속 명령어로 입출력 작업을 하는 방식</p>
<p>□ 메모리 맵 입출력</p>	<p>memory-mapped I/O [8장 242쪽]</p> <p>메모리에 접근하기 위한 주소 공간과 입출력장치에 접근하기 위한 주소 공간을 <u>하나의 주소 공간으로 간주하는 입출력 방식</u></p>
<p>□ 고립형 입출력</p>	<p>isolated I/O [8장 243쪽]</p> <p>메모리에 접근하기 위한 주소 공간과 입출력장치에 접근하기 위한 주소 공간을 <u>별도의 주소 공간으로 분리하는 입출력 방식</u></p> <div data-bbox="333 685 1089 1052" style="text-align: center;"> </div>
<p>□ 인터럽트 기반 입출력</p>	<p>Interrupt-Driven I/O [8장 245쪽]</p> <p>인터럽트로써 입출력을 수행하는 방법</p>
<p>□ DMA 입출력</p>	<p>Direct Memory Access I/O [8장 249쪽]</p> <p>CPU를 거치지 않고 메모리와 입출력장치 간에 데이터를 주고받는 입출력 방법</p>
<p>□ 입출력 버스</p>	<p>input/output bus [8장 252쪽]</p> <p>입출력장치와 컴퓨터 내부를 연결 짓는 통로</p> <p>입출력 작업의 시스템 버스 사용 횟수를 줄인다.</p>



# 09장 <sup>✓</sup> 운영체제 시작하기

□ 운영체제	operating system	[9장 261쪽]
	실행할 프로그램에 필요한 자원을 할당하고, 프로그램이 올바르게 실행되도록 돕는 특별한 프로그램	
□ 커널	kernel	[9장 269쪽]
	운영체제의 핵심 기능을 담당하는 영역	
□ 이중 모드	dual mode	[9장 272쪽]
	CPU가 명령어를 실행하는 모드를 <u>커널 모드</u> 와 <u>사용자 모드</u> 로 구분하는 방식	
	• 사용자 모드: 운영체제 서비스를 제공받을 수 <u>없는</u> 실행 모드	
	• 커널 모드: 운영체제 서비스를 제공받을 수 <u>있는</u> 실행 모드	
□ 시스템 호출	system call	[9장 273쪽]
	운영체제의 서비스를 제공받기 위해 커널 모드로 전환하기 위한 요청	

# 10장 <sup>✓</sup> 프로세스와 스레드

□ 프로세스	process	[10장 284쪽]
	실행 중인 프로그램	

<p>□ 프로세스 제어 블록</p>	<p><b>PCB; Process Control Block</b> [10장 287쪽]</p> <p>운영체제가 여러 프로세스를 관리하기 위한 자료 구조</p> <p>프로세스 ID(PID), 사용한 레지스터 값, 프로세스 상태, CPU 스케줄링 정보, 메모리 정보 등을 포함</p>
<p>□ 문맥 교환</p>	<p><b>context switching</b> <span style="color: red;">참고 용어</span> <span style="border: 1px solid red; border-radius: 50%; padding: 2px;">문맥</span> [10장 290쪽]</p> <p>프로세스 간에 실행을 전환하는 것</p> <p style="color: red; font-size: small;">→ 하나의 프로세스 수행을 재개하기 위해 기억해야 할 정보</p>
<p>□ 프로세스 상태</p>	<p><b>process state</b> [10장 297쪽]</p> <p>대표적으로 생성, 준비, 실행, 대기, 종료 상태가 있다.</p> <div style="text-align: center;"> <pre> graph LR     A[생성] --&gt; B[준비]     B -- "타이머 인터럽트" --&gt; C[실행]     C -- "디스패치" --&gt; B     C -- "입출력 요청" --&gt; D[대기]     D -- "입출력 완료" --&gt; B     C --&gt; E[종료] </pre> </div>
<p>□ 부모 프로세스</p>	<p><b>parent process</b> <span style="color: red;">참고 용어</span> <span style="border: 1px solid red; border-radius: 50%; padding: 2px;">자식 프로세스</span> <span style="color: red;">→ 부모 프로세스로부터 생성</span> [10장 298쪽]</p> <p>프로세스를 생성한 프로세스</p> <div style="border: 1px solid red; border-radius: 15px; padding: 10px; margin-top: 10px;"> <p style="color: red; font-weight: bold; font-size: small;">그것이 알고싶다    프로세스 계층 구조</p> <p style="font-size: small;">많은 운영체제는 프로세스가 프로세스를 낳는 계층적인 구조로 프로세스들을 관리한다.</p> </div>
<p>□ fork 시스템 호출</p>	<p><b>fork system call</b> [10장 301쪽]</p> <p><u>자신의 복제본을 자식 프로세스로 생성하는 시스템 호출</u></p>
<p>□ exec 시스템 호출</p>	<p><b>exec system call</b> [10장 301쪽]</p> <p>자신의 메모리 공간을 <u>다른 프로그램으로 교체하는 시스템 호출</u></p>

□ 멀티프로세스      multi-process      기본적으로 자원 공유하지 않음      [10장 309쪽]  
 여러 프로세스를 동시에 실행하는 것

□ 멀티스레드      multithread      [10장 309쪽]  
 여러 스레드로 프로세스를 동시에 실행하는 것  
기본적으로 자원 공유함

# 11장 CPU 스케줄링

□ CPU 스케줄링      CPU scheduling      [11장 316쪽]  
 공정하고 합리적으로 CPU 자원을 배분하는 방법

□ 우선순위      priority      [11장 317쪽]  
 CPU를 할당받을 수 있는 우선순위  
 운영체제는 프로세스별로 부여된 우선순위를 보고 CPU 스케줄링을 수행

□ 준비 큐      ready queue      [11장 321쪽]  
 CPU 할당을 기다리는 프로세스들을 위한 큐

□ 대기 큐      waiting queue      [11장 321쪽]  
 입출력장치를 기다리는 프로세스들을 위한 큐

□ 선점형 스케줄링      preemptive scheduling ↔ 비선점형 스케줄링      [11장 324쪽]  
 프로세스가 이용 중인 자원을 빼앗을 수 있는 스케줄링 방식

□ CPU 스케줄링 알고리즘      CPU scheduling algorithm      [11장 328쪽]  
 선입 선처리, 최단 작업 우선, 라운드 로빈, 최소 잔여 시간 우선, 우선순위, 다단계 큐, 다단계 피드백 큐 스케줄링 알고리즘 등

# 12장 <sup>✓</sup> 프로세스 동기화

□ 동기화	<b>synchronization</b> [12장 341쪽]
	프로세스들 사이의 수행 시기를 맞추는 것 <i>상호 배제를 위한 동기화</i> 특정 자원에 접근할 때 <u>하나의 프로세스만 접근하게 하거나 프로세스를 올바른 순서대로 실행하게 하는 것</u> <i>실행 순서 제어를 위한 동기화</i>
□ 공유 자원	<b>shared resource</b> [12장 345쪽]
	<u>여러 프로세스가 공동으로 사용하는 자원</u> 전역 변수가 될 수도, 파일이 될 수도, 입출력장치, 보조기억장치가 될 수도 있다.
□ 임계 구역	<b>critical section</b> [12장 346쪽]
	공유 자원에 접근하는 코드 중 <u>동시에 실행하면 문제가 발생하는 코드 영역</u> <i>레이스 컨디션</i>
□ 뮤텝스 락	<b>mutex lock</b> [12장 351쪽]
	임계 구역을 잠금으로써 프로세스 간의 상호 배제를 이루는 동기화 도구
□ 세마포	<b>semaphore</b> [12장 353쪽]
	공유 자원이 여러 개 있는 임계 구역 문제도 해결할 수 있는 동기화 도구
□ 모니터	<b>monitor</b> [12장 358쪽]
	세마포에 비해 사용자가 사용하기 편리한 동기화 도구로 조건 변수를 사용

# 13장 교착 상태

□ 교착 상태	deadlock	[13장 366쪽]
	일어나지 않을 사건을 기다리며 무한히 대기하는 현상	자원 할당 그래프를 이용해 교착 상태를 표현
□ 교착 상태 예방	deadlock prevention	[13장 377쪽]
	교착 상태의 발생 조건 중 하나를 충족하지 못하게 하는 방법	상호 배제, 점유와 대기, 비선점 원형 대기
□ 교착 상태 회피	deadlock avoidance	[13장 380쪽]
	안전 상태를 유지할 수 있는 경우에만 자원을 할당하는 방법	
□ 교착 상태 검출 후 회복	deadlock detection & recovery	[13장 384쪽]
	교착 상태 발생 여부를 주기적으로 검사하고, 발생 시 그때그때 회복하는 방식	선점을 통한 회복과 프로세스 강제 종료를 통한 회복이 있다

# 14장 가상 메모리

□ 스와핑	swapping	[14장 391쪽]
	메모리에서 사용되지 않는 일부 프로세스를 보조기억장치로 내보내고(스왑 아웃) 실행할 프로세스를 메모리로 들여보내는(스왑 인) 메모리 관리 기법	
□ 최초 적합	first fit	[14장 393쪽]
	최초로 발견한 적재 가능한 빈 공간에 프로세스를 배치하는 방식	

<p>□ 최적 적합</p>	<p><b>best fit</b></p> <p>프로세스가 적재될 수 있는 <u>가장 작은 공간에</u> 프로세스를 배치하는 방식</p>	<p>[14장 394쪽]</p>
<p>□ 최악 적합</p>	<p><b>worst fit</b></p> <p>프로세스가 적재될 수 있는 <u>가장 큰 공간에</u> 프로세스를 배치하는 방식</p>	<p>[14장 395쪽]</p>
<p>□ 외부 단편화</p>	<p><b>external fragmentation</b></p> <p>프로세스를 할당하기 어려울 만큼 작은 메모리 공간으로 인해 메모리가 낭비되는 현상</p>	<p>[14장 397쪽]</p>
<p>□ 페이징</p>	<p><b>paging</b></p> <p>물리 주소 공간을 프레임 단위로 자르고 프로세스의 논리 주소 공간을 페이지 단위로 자른 뒤 각 페이지를 프레임에 할당하는 가상 메모리 관리 기법</p>	<p>[14장 403쪽]</p>
<p>□ 페이지 테이블</p>	<p><b>page table</b></p> <p>페이지 번호와 프레임 번호뿐만 아니라 유효 비트, 보호 비트, 접근 비트, 수정 비트 등이 있다.</p>	<p>[14장 405쪽]</p>
<p>□ 페이지 테이블 베이스 레지스터</p>	<p><b>PTBR; Page Table Base Register</b></p> <p>각 프로세스의 페이지 테이블이 적재된 주소를 가리킨다.</p>	<p>[14장 407쪽]</p>
<p>□ 변환 색인 버퍼</p>	<p><b>TLB; Translation Lookaside Buffer</b></p> <p>페이지 테이블의 캐시 메모리 역할을 수행(페이지 테이블의 일부를 저장)</p>	<p>[14장 409쪽]</p>

□ 페이지 폴트	page fault	[14장 413쪽]
	메모리에 적재되지 않은 페이지를 참조할 경우 발생하는 인터럽트	
	<i>TLB에 원하는 페이지가 있으면 TLB 히트, 없으면 TLB 미스</i>	
□ 요구 페이징	demand paging	[14장 425쪽]
	페이지가 필요할 때에만 메모리에 적재하는 기법	
	<i>유틸 비트가 0인 페이지</i>	
□ 페이지 교체 알고리즘	page replacement algorithm	[14장 426쪽]
	사용되지 않은 페이지를 보조기억장치로 내보내고 적재될 페이지를 메모리에 적재하는 알고리즘 → FIFO, 최적, LRU 페이지 교체 알고리즘 등	
□ 스래싱	thrashing	[14장 431쪽]
	지나치게 빈번한 페이지 교체로 인해 CPU 이용률이 낮아지는 문제	
□ 프레임 할당	frame allocation	[14장 433쪽]
	<i>균등 할당, 비례 할당, 작업 집합 모델 기반 프레임 할당, 페이지 폴트를 기반 프레임 할당 등</i>	
	프로세스에게 적절한 프레임을 배분하는 <b>방법</b>	

# 15장 <sup>✓</sup> 파일 시스템

□ 파일	file	[15장 441쪽]
	의미 있고 관련 있는 정보를 모은 논리적인 단위	
□ 파일 속성	file attribute	[15장 442쪽]
	파일과 관련된 다양한 부가 정보	
	<i>파일의 유형, 크기, 생성 날짜, 마지막 접근/수정 날짜 등</i>	
□ 디렉터리	directory	[15장 443쪽]
	파일 또는 디렉터리들을 한데 묶어 관리할 수 있다.	

□ 경로	path	루트 디렉터리부터 시작하는 경로	[15장 444쪽]
		디렉터리를 이용해 파일의 위치를 특정 짓는 정보. 크게 절대 경로와 상대 경로가 있다. 상대 경로는 현재 디렉터리부터 시작하는 경로입니다.	
□ 파티셔닝	partitioning		[15장 453쪽]
		하드 디스크나 SSD처럼 용량이 큰 저장 장치를 하나 이상의 논리적인 여러 단위로 구획하는 작업 파티션(partition)	
□ 포매팅	formatting		[15장 454쪽]
		파일 시스템을 결정하는 작업	
□ 연속 할당	contiguous allocation	외부 단편화 야기	[15장 457쪽]
		파일을 보조기억장치에 연속적인 블록으로 할당하는 방식	
□ 연결 할당	linked allocation		[15장 458쪽]
		각 블록 일부에 다음 블록의 주소를 저장하여 블록들을 연결 리스트로 관리	
□ 색인 할당	indexed allocation		[15장 460쪽]
		파일의 모든 블록 주소를 색인 블록에 모아 관리하는 방식	
□ FAT 파일 시스템	FAT file system		[15장 463쪽]
		FAT를 이용하는 연결 할당 기반의 파일 시스템	
□ 유닉스 파일 시스템	UNIX file system		[15장 467쪽]
		i-node를 이용하는 색인 할당 기반의 파일 시스템	