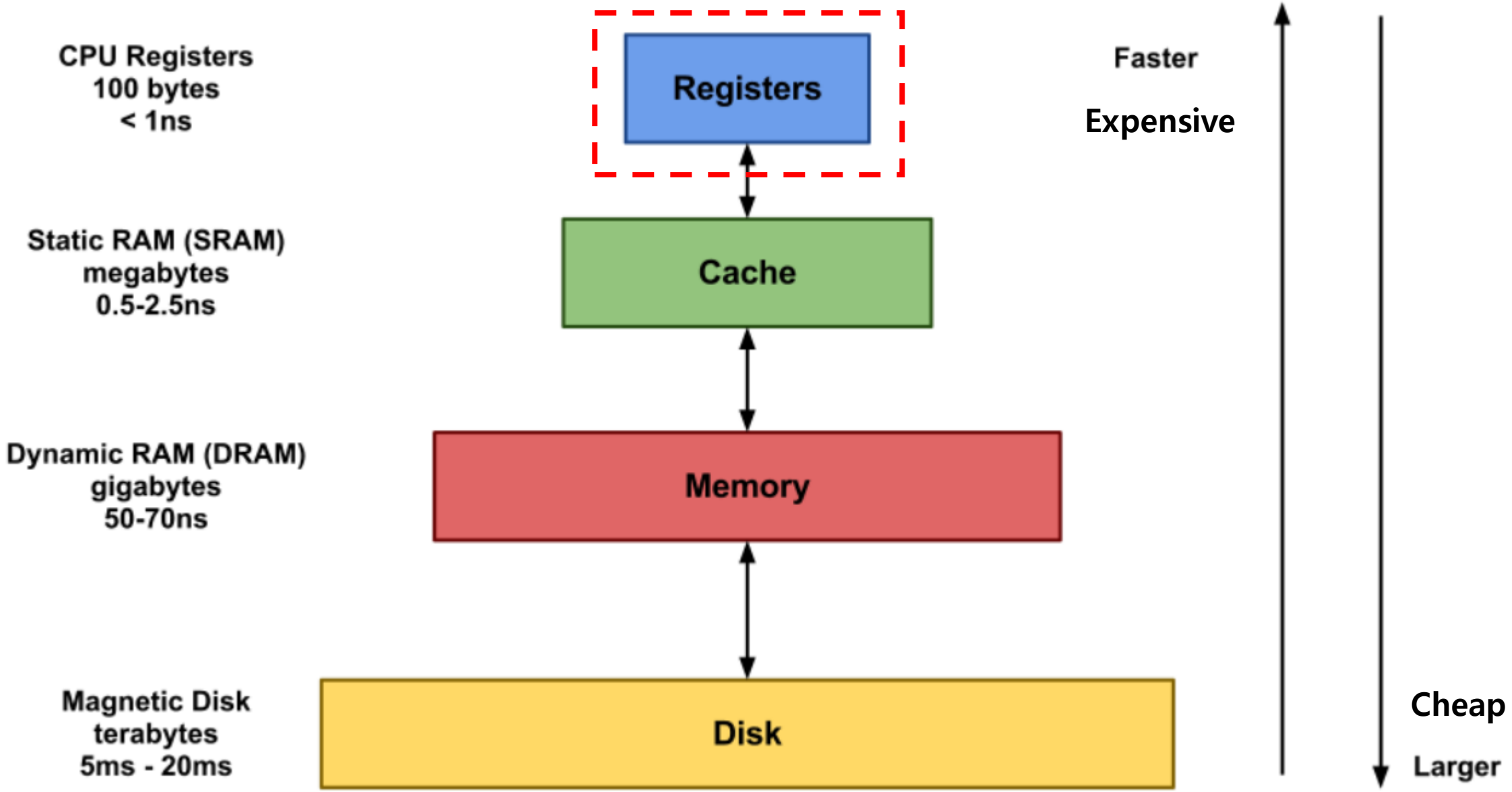
레지스터(Register)

박시원

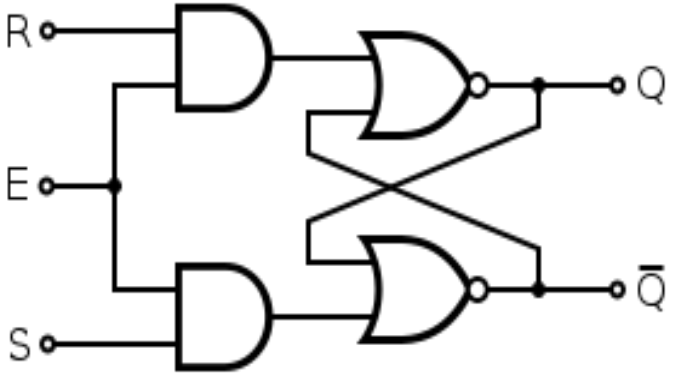
목차

- 레지스터의 정의
- 레지스터의 종류
- 레지스터의 구성
- 레지스터의 필요성
- 프로그램 카운터

CPU 내부 기억 장치

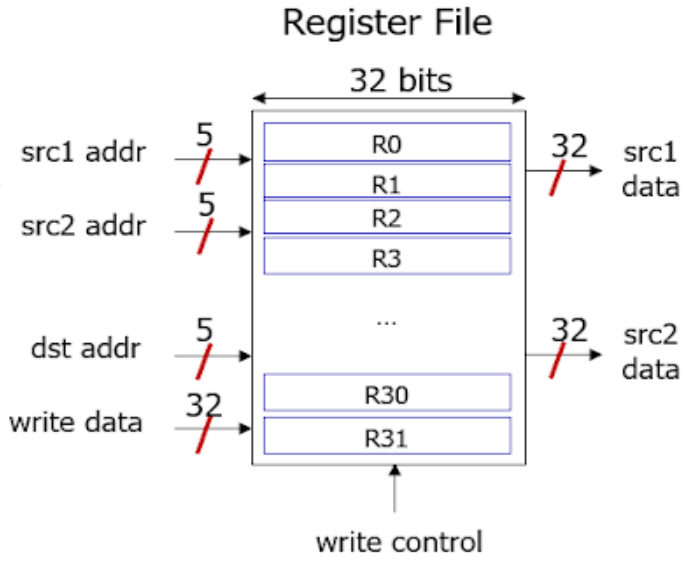

CPU Registers
100 bytes
< 1ns
Registers
Static RAM (SRAM)
megabytes
0.5-2.5ns
Cache
Dynamic RAM (DRAM)
gigabytes
50-70ns
Memory
Magnetic Disk
terabytes
5ms - 20ms
Disk
Faster
Expensive
Cheap
Larger

레지스터란?



정의 및 특징

- N bit의 정보를 저장할 수 있는 N개의 Flip Flop 집합
  => Flip Flop은 1bit를 저장 가능함
- CPU 내부에 존재하며, 기억 장치 중 가장 빠르며 용량이 작음
- 과거에는 플립플롭, 마그네틱 코어, 박막 필름 메모리 등으로 구현했지만, 최신 프로세서에서는 대개 레지스터 파일로 구현함

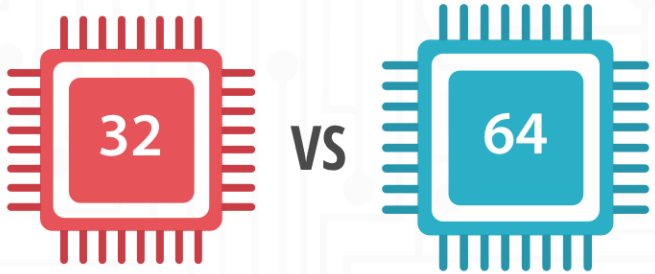


무엇을 저장하는가?

- CPU가 요청을 처리하는 데 필요한 데이터
  => CPU가 명령어를 처리하는 과정에서 임시로 처리할 데이터를 저장하거나 메모리의 주소를 저장할 목적으로 사용함
  => 예) 복잡한 계산 식 중 1+1의 값을 저장해야 할 때, 메모리로 해당 저장 결과를 보내지 않고 현재 계산을 수행 중인 값을 레지스터에 임시로 저장해 뒀다가 모든 연산이 끝나면 메모리 주소 데이터를 토대로 메모리로 전달

32bit 컴퓨터, 64bit 컴퓨터?

- 32bit, 64bit라는 용어의 의미는 컴퓨터의 프로세서, 즉 CPU가 정보를 처리하는 방식과 크기를 의미함
- bit는 CPU가 처리하는 데이터의 최소 단위인 동시에, 레지스터의 크기를 의미
- 즉, 32bit 컴퓨터는 CPU의 레지스터의 크기가 32bit라는 말이고, 64bit 컴퓨터는 레지스터의 크기가 64bit라는 의미이다



구분	데이터	GB	호환성
32비트	2^32 (= 4,294,967,296)	4GB	32bit에서 64bit 요구 프로그램은 실행 못함
64비트	2^64 (=18,446,744,073,709,551,616)	16GB	64bit에서 32bit 요구 프로그램은 실행 가능

※ TMI: x86과 x64의 차이?

x86 == 32bit, x64 == 64bit이고, 32bit를 x86이라고 부르게 된 이유는 인텔이 32bit 칩셋의 품번을 80-86으로 붙였기 때문

레지스터의 종류

- 데이터 레지스터 : 정수 값을 저장할 수 있는 레지스터
- 주소 레지스터 : 메모리 주소를 저장하여, 메모리 접근에 사용되는 레지스터
- **범용 레지스터** : 데이터와 주소를 모두 저장할 수 있는 레지스터, 우리가 흔히 레지스터라고 하면 범용레지스터를 말한다
- 부동 소수점 레지스터 : 많은 시스템에서 부동소수점 값을 저장하기 위해 사용된다
- 상수 레지스터 : 0이나 1 등 고정된 값을 저장하고 있는 레지스터
- 특수 레지스터 :
  - 명령 레지스터; 현재 실행중인 명령어를 저장,
  - 인덱스(색인) 레지스터; 실행 중에 피연산자의 주소를 계산하는 데 사용

레지스터의 구성

CPU의 구성

- 프로그램 카운터(PC, Program Counter) : 다음 번에 실행할 명령어의 주소를 기억하는 레지스터
- 명령 레지스터(IR, Instruction Register) : 현재 실행 중인 명령의 내용을 기억하는 레지스터
- 누산기(AC, Accumulator) : 연산된 결과를 일시적으로 저장하는 레지스터
- 메모리 주소 레지스터(MAR, Memory Address Register) : 읽기/쓰기를 수행할 때 필요한 주기억장치(메모리)의 주소를 저장하는 레지스터
- 메모리 버퍼 레지스터(MBR, Memory Buffer Register) : 메모리에 출입하는 데이터가 일시적으로 기억되는 레지스터

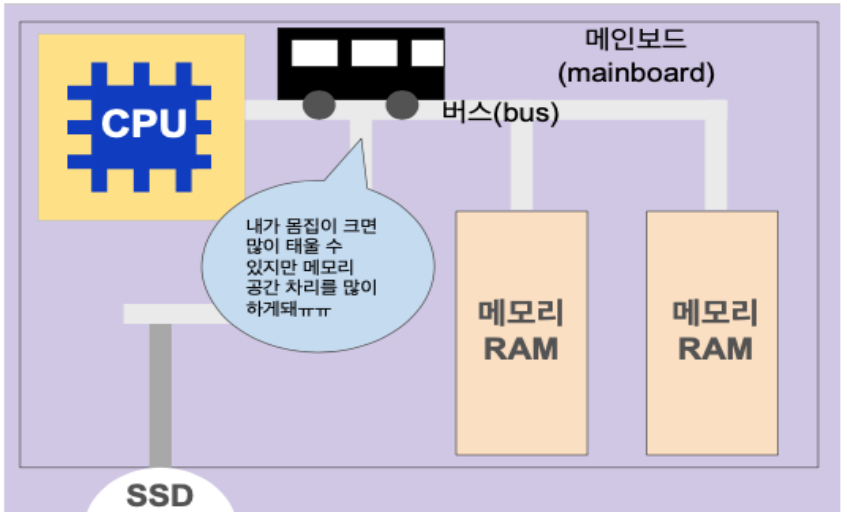
세부 구성

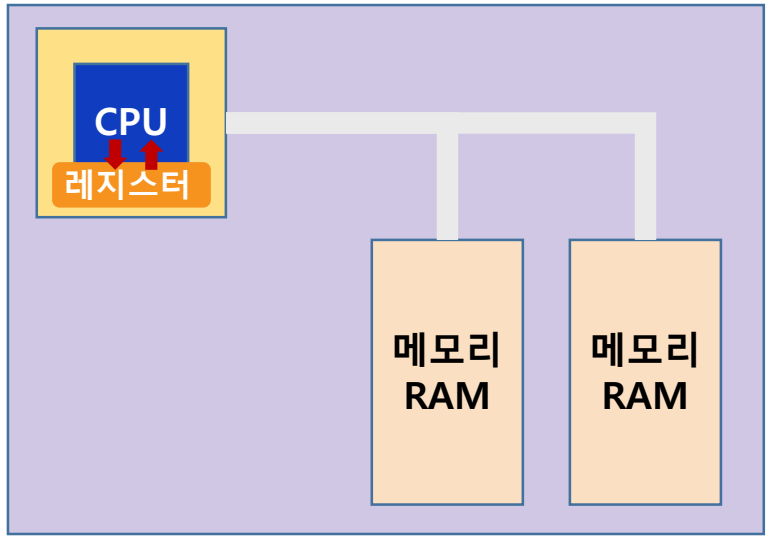
- 연산장치 (ALU)
  - 가산기(Adder)
  - 누산기 (Accumulator)
  - 데이터 레지스터
  - 상태 레지스터
  - 인덱스 레지스터
- 제어장치 (CU)
  - 메모리 주소 레지스터(MAR)
  - 메모리 버퍼 레지스터(MBR)
  - 프로그램 카운터(PC)
  - 명령 레지스터(IR)
  - 주소 레지스터(AR)
- 레지스터 (Register)
  - 플립플롭(FF)

레지스터의 필요성?

레지스터가 없다면?

- Access Time = Seek Time + Rotation Delay + Transfer Delay
- CPU가 RAM에 있는 데이터에 접근하기 위해서 물리적으로 먼 길을 돌아가야 함
- CPU의 연산 결과를 DRAM으로 매번 보내고 다시 주고 받고 해야함. 그러나 이런 식으로 동작한다면 CPU는 매우 느릴 수 밖에 없음
  예) 3000 + 3000 => (1+1+...+1) + (2+2+....+2) + (n + n + ... n)





- 이동할 필요가 없는 CPU만의 저장 공간을 만들어서 CPU 옆에 두자
- 대부분의 현대 프로세서는 메인 메모리에서 레지스터로 데이터를 옮겨와 데이터를 처리한 후, 그 내용을 다시 레지스터에서 메인 메모리로 저장하는 로드-스토어(Load-Store) 설계를 사용하고 있다.

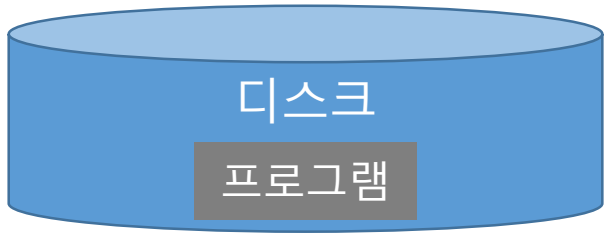
프로그램 카운터(PC; Program Counter)

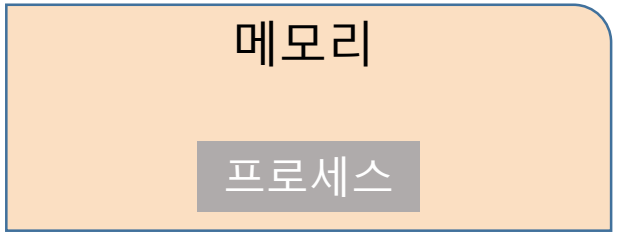
정의

- 다음에 실행될 명령어가 저장된 주기억 장치의 주소를 저장하는 레지스터
- 프로그램(프로세스)와 밀접한 관계가 있음

프로그램? 프로세스?

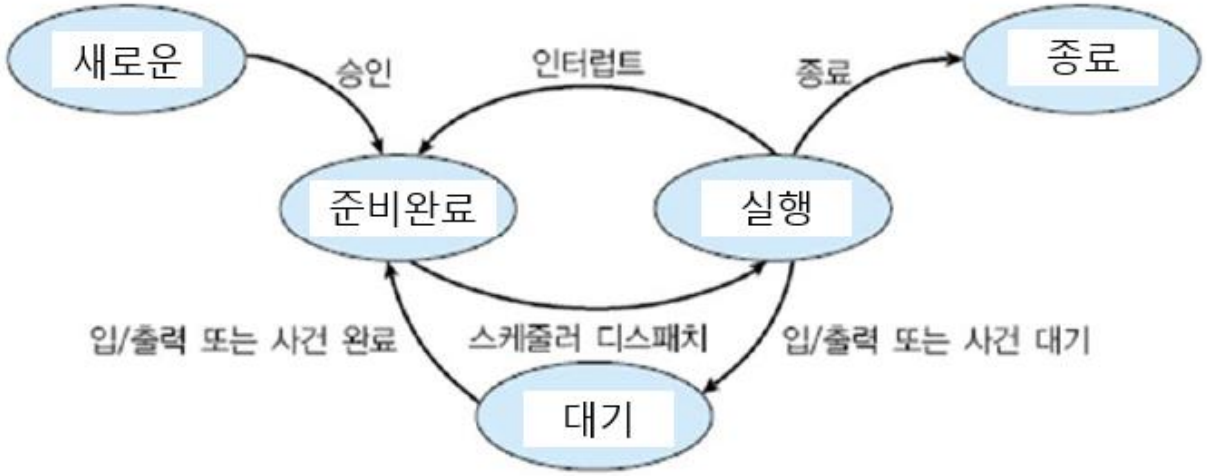
- 프로그램은 디스크에 저장된 수동적 존재
- 프로세스는 프로그램이 메인 메모리에 올라간 상태, 관련된 자원의 집합을 가진 능동적인 존재





프로세스의 상태

- 프로세스는 실행되면서 그 상태가 변하고, 현재 활동에 따라 상태가 다르다
  - 새로운 : 프로세스가 생성 중
  - 실행 : 명령어들이 실행되고 있음
  - 대기 : 프로세스가 어떤 사건이 일어나기를 기다림
  - 준비 완료 : 프로세스가 CPU에 할당 되기를 기다림
  - 종료 : 프로세스의 실행이 종료됨



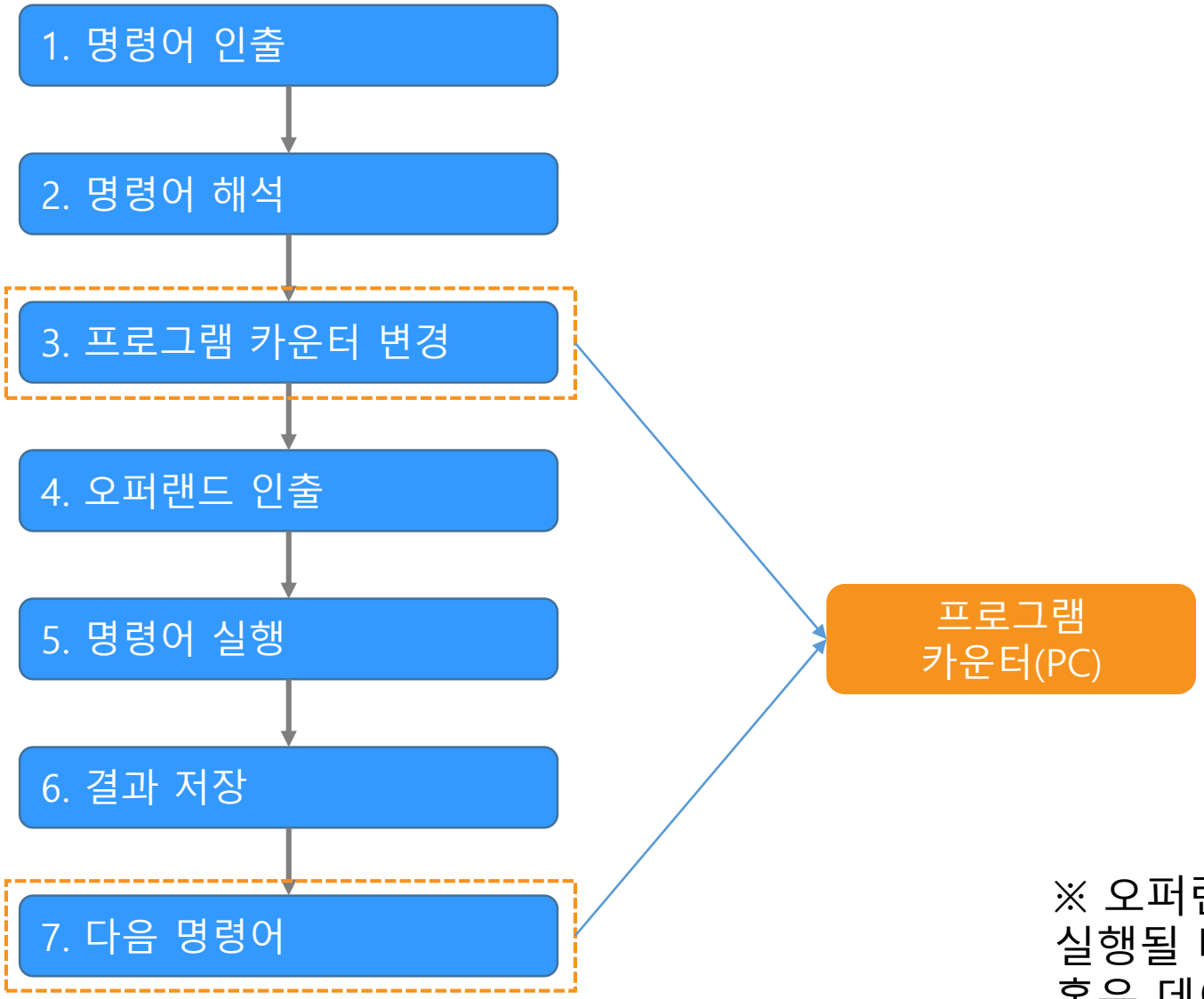
누가, 어떻게 제어하는가?

프로세스 제어 블록(PCB; Process Control Block)

- 프로세스 제어 블록에는 특정 프로세스와 연관된 여러 정보가 있음
  - 프로세스 상태 : 새로운, 준비 완료, 실행, 대기, 준비 완료, 종료
  - 프로그램 카운터 : 이 프로세스 다음에 실행할 명령어 주소를
  - CPU 레지스터 : 컴퓨터의 구조에 따라 다양한 수와 타입의 레지스터 정보
  - CPU 스케줄링 정보 : 프로세스 우선 순위, 스케줄 큐에 대한 포인터와 스케줄 매개변수
  - 메모리 관리 정보 : 기준 레지스터와 한계 레지스터의 값, 페이지 테이블, 세그먼트 테이블 등
  - 입출력 상태 정보 : 프로세스에게 할당된 입출력 장치들과 열린 파일의 목록
  - 그 외 : CPU 사용 시간, 경과된 시간, 시간 제한, 계정 번호, 잡 또는 프로세스 번호

PCB

명령어 실행 사이클



※ 오퍼랜드(Operand): 각 명령어가 CPU에 의해 실행될 때, 연산을 수행하는데 필요한 데이터 혹은 데이터 주소를 말함

Q&A