리눅스 프로그래밍

컴퓨터시스템 구조 이해

인공지능소프트웨어과
이혜정 교수

* 본 교안의 일부 그림과 텍스트는 '운영체제 – 그림으로 배우는 구조와 원리' (개정3판, 한빛미디어)으로 발췌하였음

목차

- **컴퓨터 하드웨어의 구성**
  - 프로세서, CPU
  - 시스템버스
  - 기억장치(메모리), 저장장치
  - 입출력 장치

- **컴퓨터 시스템의 동작**
  - 작업 처리 순서
  - 명령어 구조 및 실행 방식
  - 인터럽트 동작 방식

컴퓨터 하드웨어의 구성

- 프로세서, CPU
- 시스템버스
- 기억장치(메모리), 저장장치
- 입출력 장치

컴퓨터 하드웨어의 구성

▪ 컴퓨터 시스템

- 데이터를 처리하는 물리적인 기계장치인 하드웨어(HW)와 어떤 작업을 지시하는 명령어로 작성한 프로그램인 소프트웨어(SW)로 구성

▪ 컴퓨터 하드웨어

- 프로세서, 메모리(기억장치), 주변장치로 구성되고, 이들은 시스템 버스로 연결

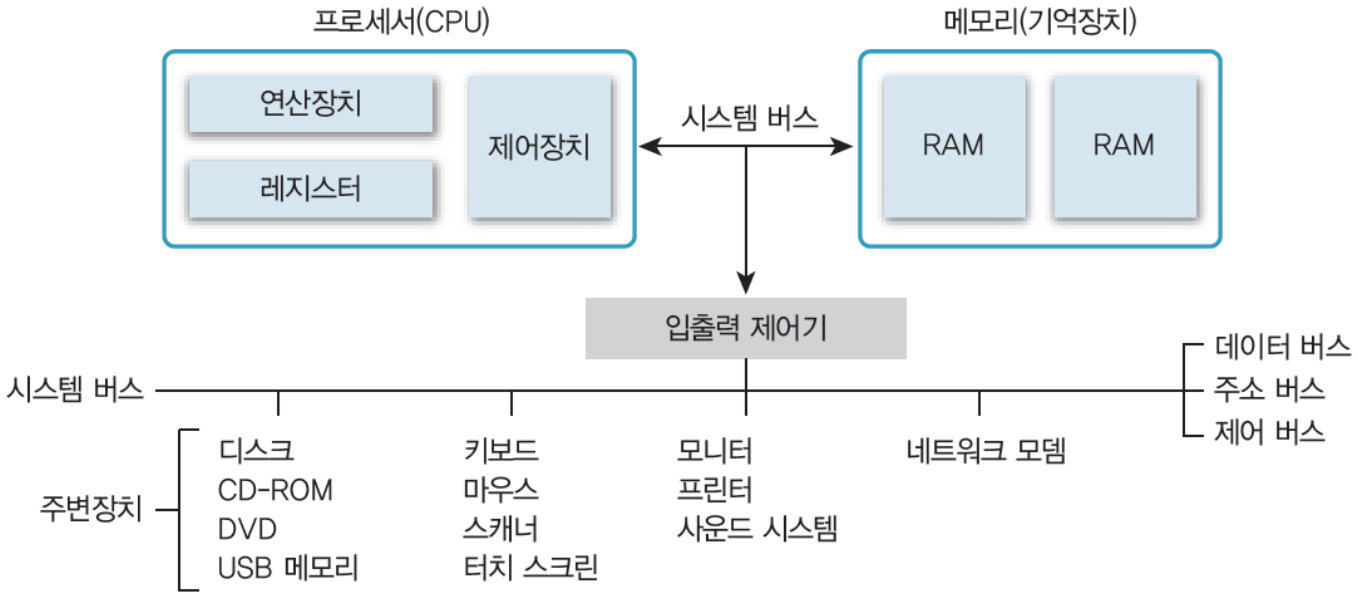
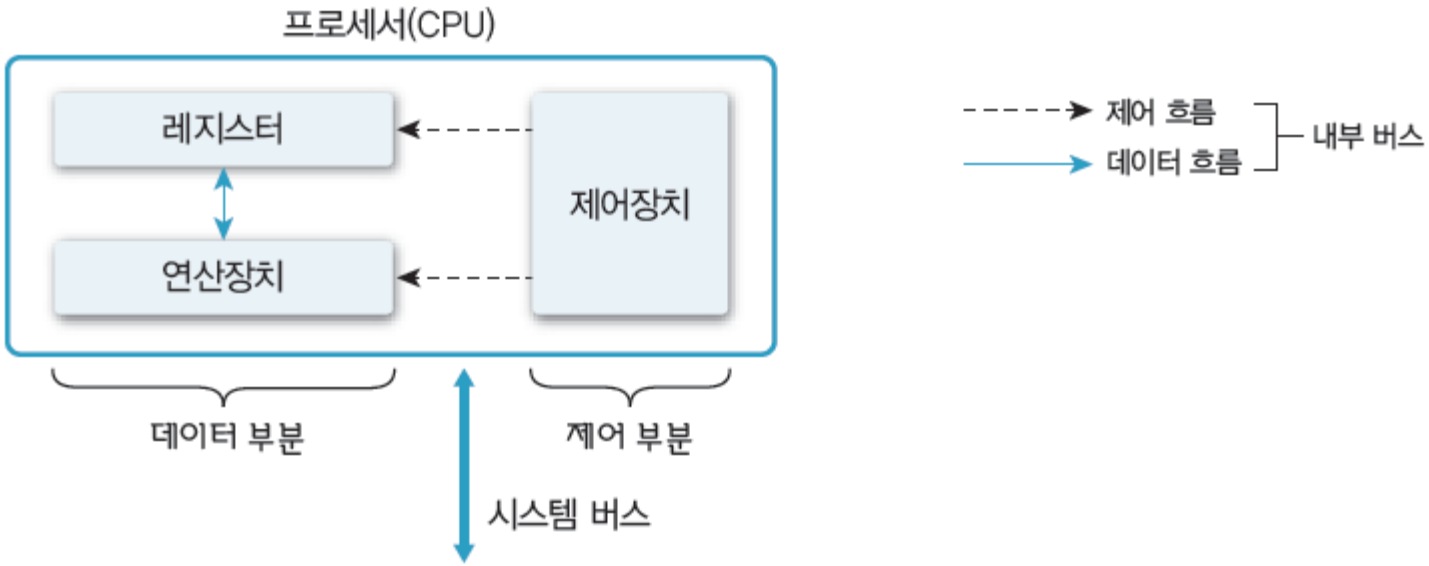


그림 1-1 컴퓨터 하드웨어의 구성

프로세서, CPU

- **프로세서(Processor), CPU(Central Processing Unit, 중앙처리장치)**
  - 컴퓨터를 구성하는 모든 장치의 동작을 제어하고 연산 수행

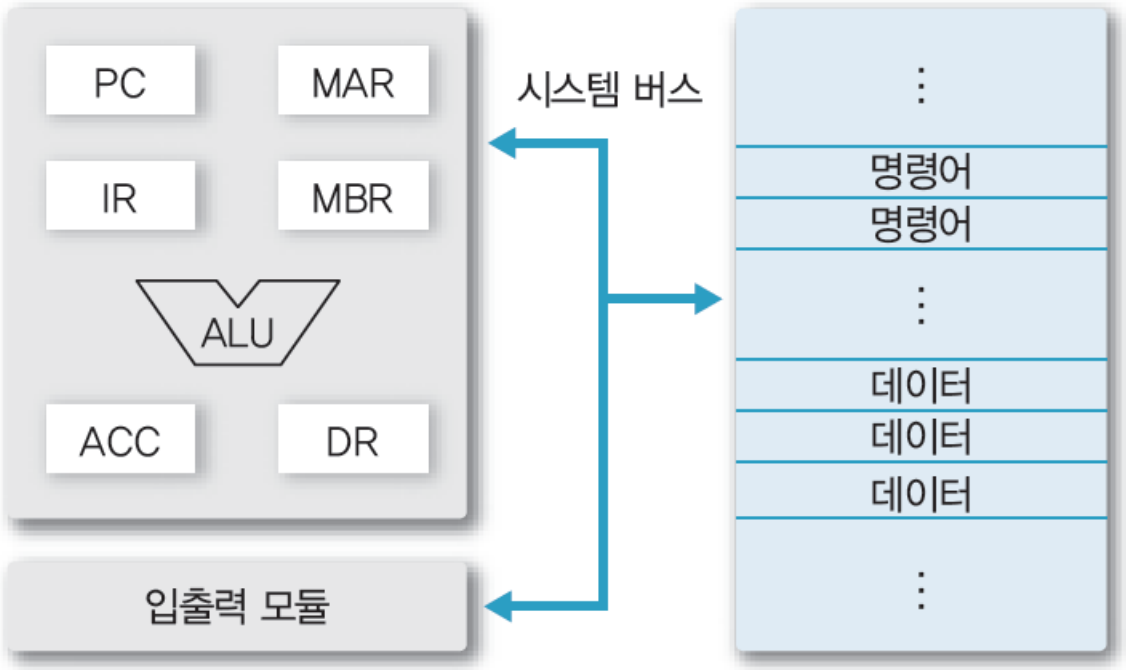


  - 구성요소
    - 레지스터 (Register) : 작업에 필요한 데이터를 CPU 내부에 저장
    - 연산장치 (ALU, Arithmetic & Logic Unit) : 산술 연산 , 논리 연산
    - 제어장치 (CU, Control Unit) : 프로세서에서 작업을 지시

레지스터

- 프로세서 내부에 있으며, 프로세서가 사용할 데이터를 보관하는 가장 빠른 메모리

- 프로세서의 기본 레지스터



• ALU Arithmetic Logic Unit : 산술 · 논리 연산장치

메모리, Memory

- **메모리 계층 구조** 자격증
  - 1950~1960년대 너무 비싼 메인 메모리의 가격 문제 때문에 제안한 방법
  - 메모리를 계층적으로 구성하여 비용, 속도, 용량, 접근시간 등을 상호 보완

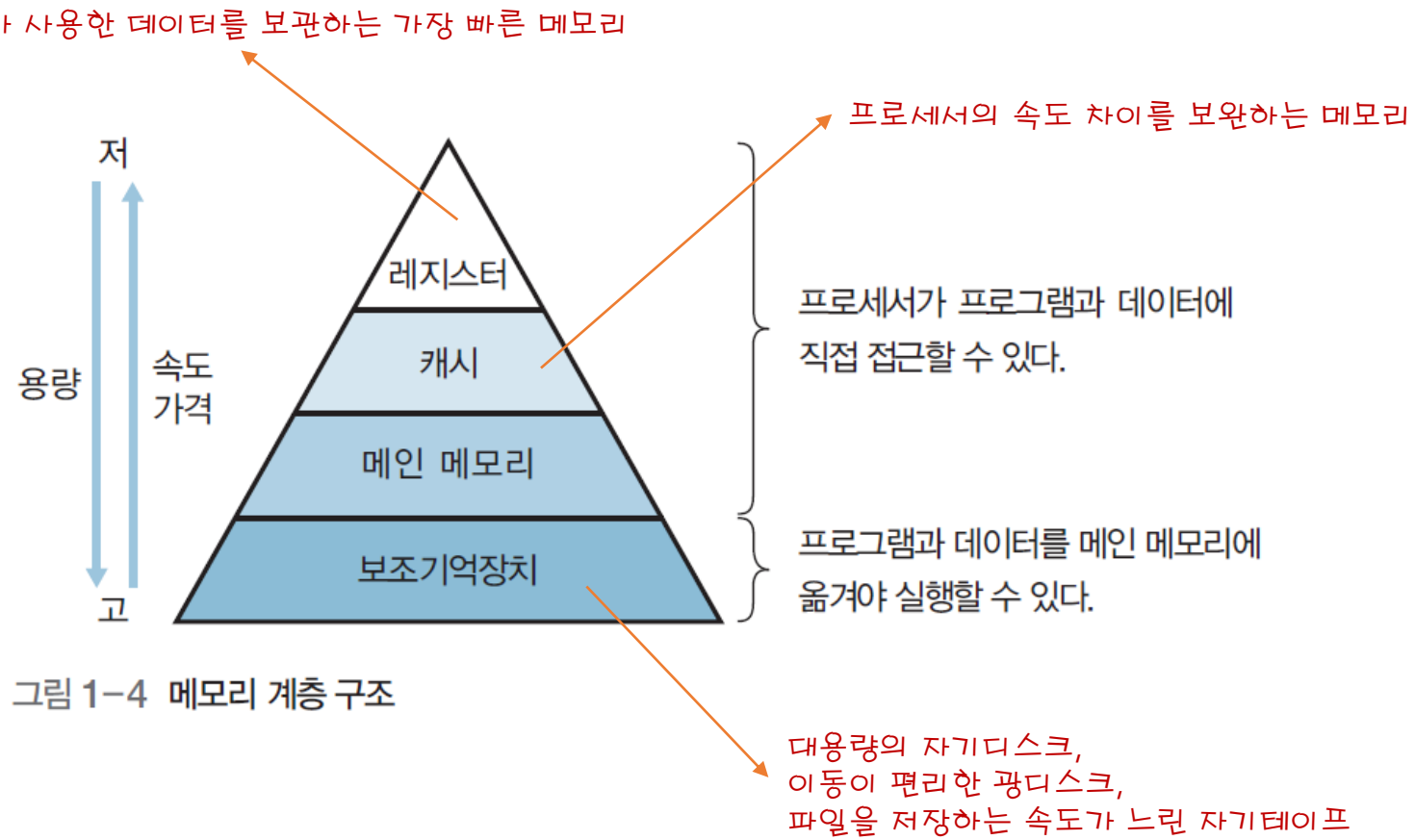
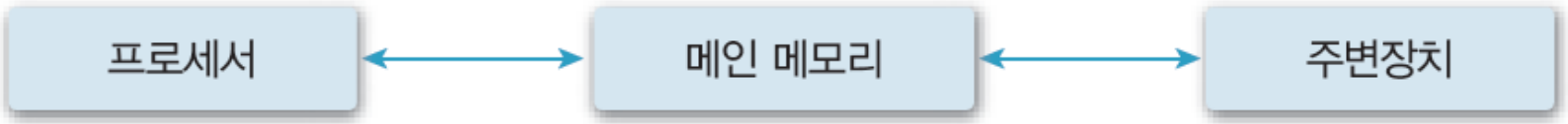


그림 1-4 메모리 계층 구조

메인 메모리

- Main Memory
- 주기억장치

- 프로세서 외부에 있으면서 프로세서에서 수행할 프로그램과 데이터를 저장하거나 프로세서에서 처리한 결과 저장



캐시(Cache)

- **프로세서 내부나 외부에 있으며, 처리 속도가 빠른 프로세서와 상대적으로 느린 메인 메모리의 속도 차이를 보완하는 고속 버퍼**
  - 메인 메모리에서 데이터를 블록 단위로 가져와 프로세서에 워드 단위로 전달하여 속도를 높임

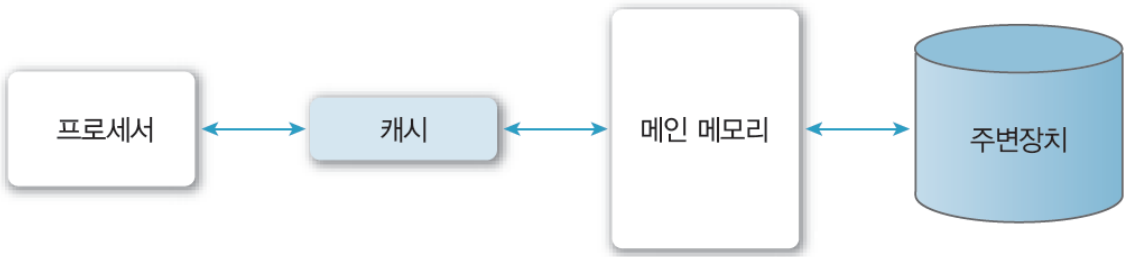


그림 1-9 메인 메모리의 역할 2 : 캐시 추가

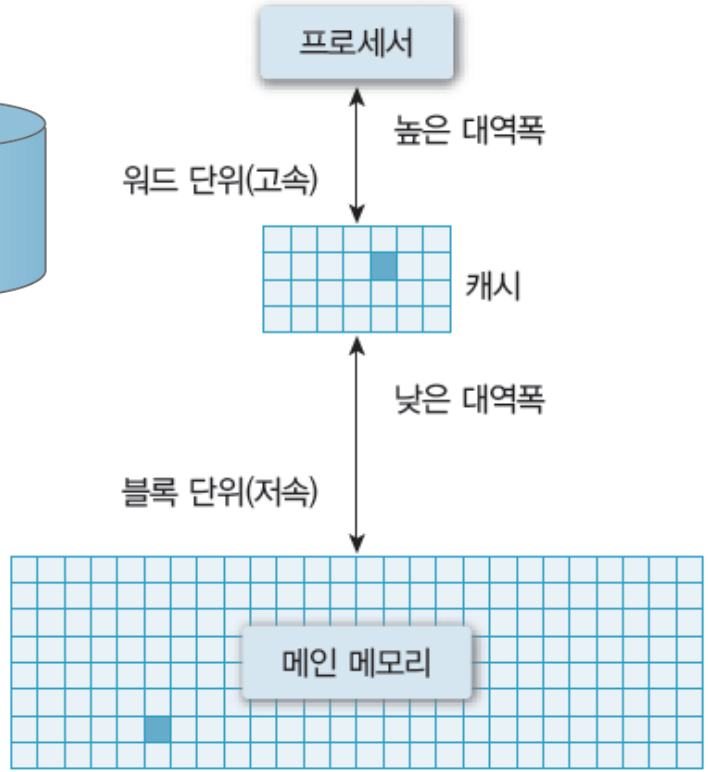
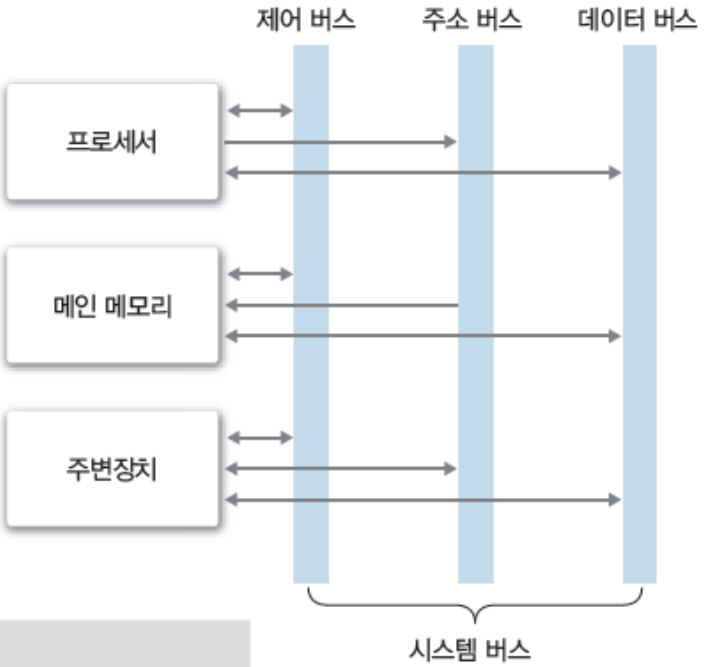


그림 1-10 캐시의 역할

시스템 버스(Bus)

- **하드웨어를 물리적으로 연결하여 서로 데이터를 주고받을 수 있게 하는 통로**
  - 컴퓨터 내부의 다양한 신호(데이터 입출력 신호, 프로세서 상태 신호, 인터럽트 요구와 허가 신호, 클록clock 신호 등)를 시스템 버스로 전달



- **종류 : 데이터 버스, 주소 버스, 제어 버스**

표 1-3 시스템 버스의 종류

종류	설명
데이터 버스	프로세서와 메인 메모리, 주변장치 사이에서 데이터를 전송한다. 데이터 버스를 구성하는 배선 수는 프로세서가 한 번에 전송할 수 있는 비트 수를 결정하는데, 이를 워드라고 한다.
주소 버스	프로세서가 시스템의 구성 요소를 식별하는 주소 정보를 전송한다. 주소 버스를 구성하는 배선 수는 프로세서와 접속할 수 있는 메인 메모리의 최대 용량을 결정한다.
제어 버스	프로세서가 시스템의 구성 요소를 제어하는 데 사용한다. 제어 신호로 연산장치의 연산 종류와 메인 메모리의 읽기나 쓰기 동작을 결정한다.

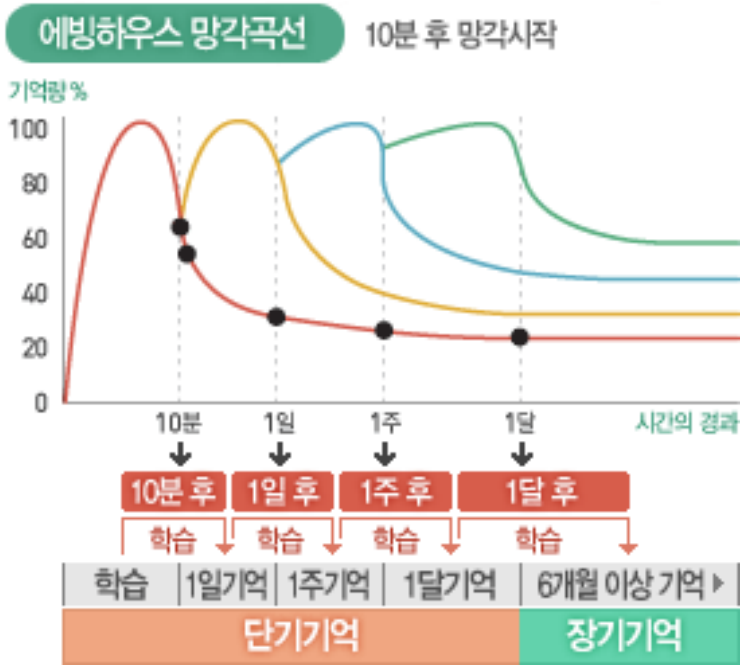
주변 장치

- **저장장치 (보조기억장치)**
  - 메인 메모리와 달리 거의 영구적으로 데이터를 저장하는 장치
  - 데이터를 입력하여 저장하며, 저장한 데이터를 출력하는 공간이므로 입출력장치에 포함하기도 함

- **입력장치**
  - 컴퓨터에서 처리할 데이터를 외부에서 입력하는 장치

- **출력장치**
  - 입력장치와 반대로 컴퓨터에서 처리한 데이터를 외부로 보내는 장치

망각 곡선 되살리기


에빙하우스 망각곡선
10분 후 망각시작
기억량 %
100
80
60
40
20
0
10분
1일
1주
1달
시간의 경과
10분 후
1일 후
1주 후
1달 후
학습
학습
학습
학습
학습
1일기억
1주기억
1달기억
6개월 이상 기억
단기기억
장기기억

컴퓨터 시스템의 동작

- 작업 처리 순서
- 명령어 구조 및 실행 방식
- 인터럽트 동작 방식

컴퓨터의 작업 처리 순서

- **컴퓨터 시스템으로 작업을 처리할 때는 다음 순서에 따라 동작, 제어장치(CU)가 이 동작을 제어**
  - ❶ 입력장치로 정보를 입력받아 메모리에 저장한다.
  - ❷ 메모리에 저장한 정보를 프로그램 제어에 따라 인출하여 연산장치에서 처리한다.
  - ❸ 처리한 정보를 출력장치에 표시하거나 보조기억장치에 저장한다.

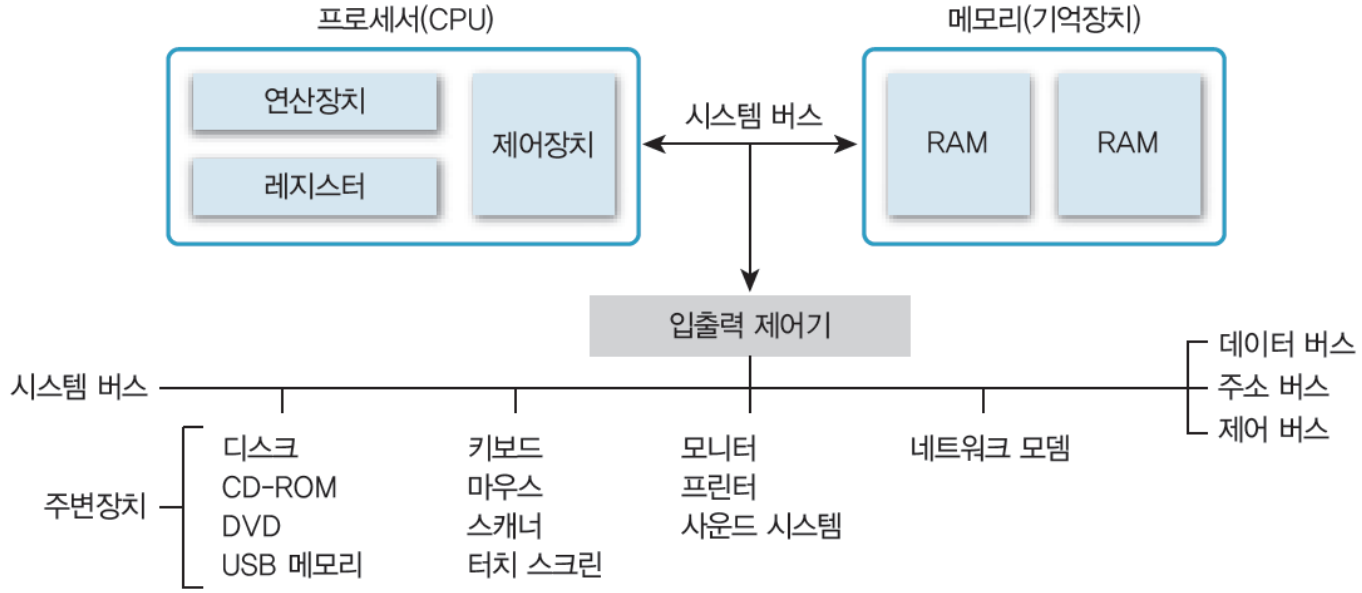


그림 1-1 컴퓨터 하드웨어의 구성

명령어의 구조

■ 명령어의 기본 구조

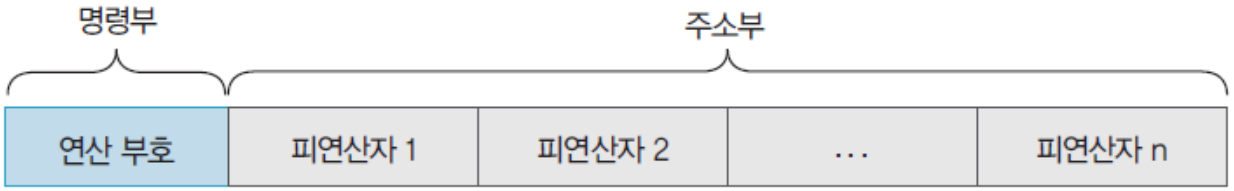


그림 1-13 명령어의 기본 구조

- 연산 부호OPcode, OPeration code (오피코드)
  - 프로세서가 실행할 동작인 연산 지정
  - 산술 연산(+, -, *, /), 논리 연산(AND, OR, NOT), 시프트shift, 보수 등 연산 정의
  - 연산 부호가 n비트이면 최대 2n개 연산이 가능
- 피연산자operand (오퍼랜드)
  - 연산할 데이터 정보 저장
  - 데이터는 레지스터나 메모리, 가상 기억장치, 입출력장치 등에 위치할 수 있는데 보통 데이터 자체보다는 데이터의 위치 저장

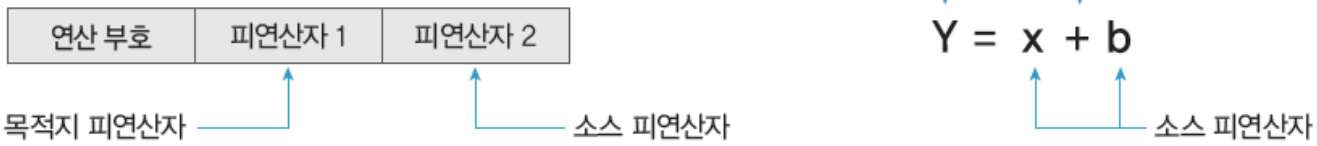


그림 1-14 소스 피연산자와 목적지 피연산자

명령어의 실행

▪ 명령어의 실행 과정

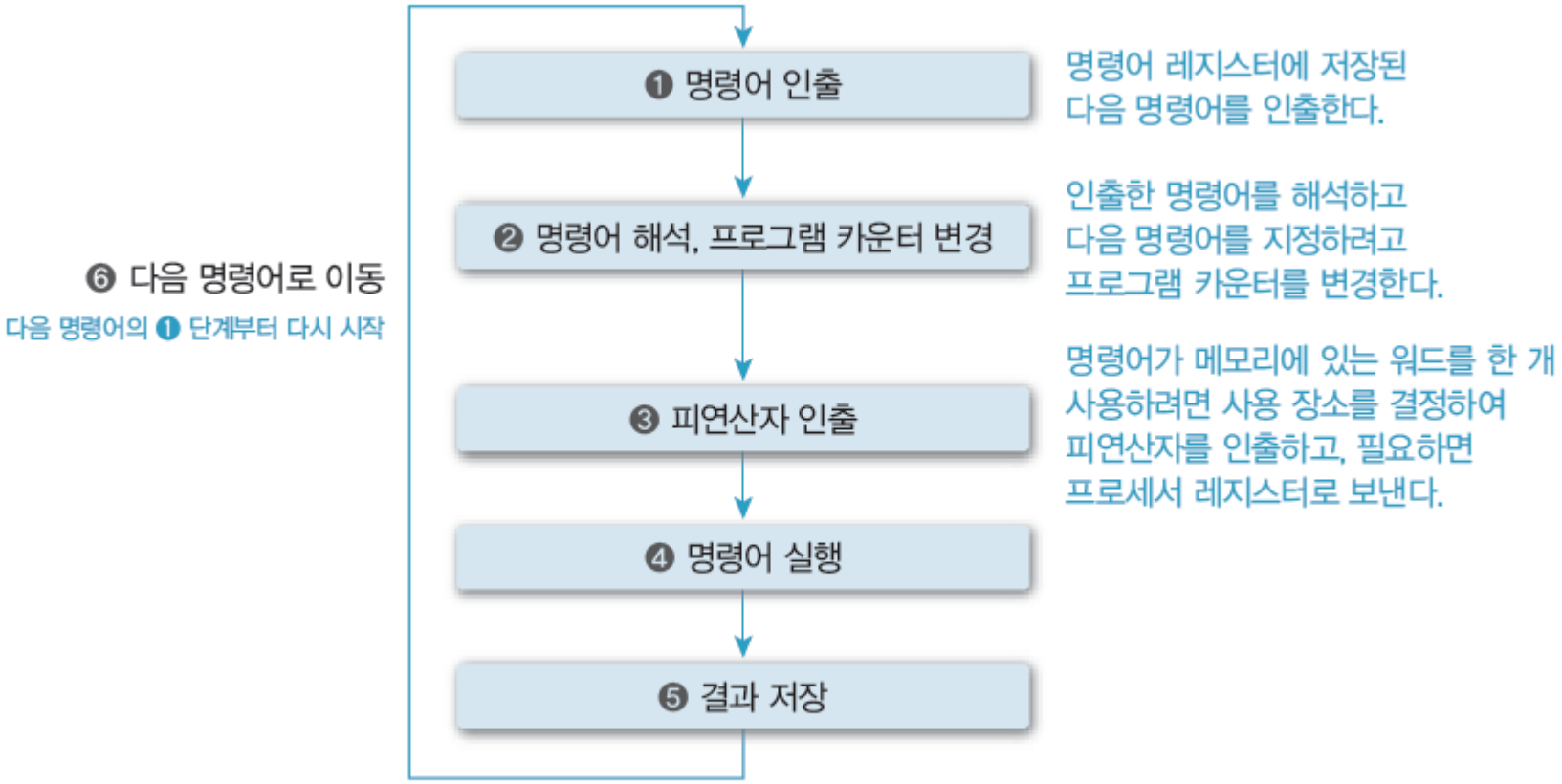
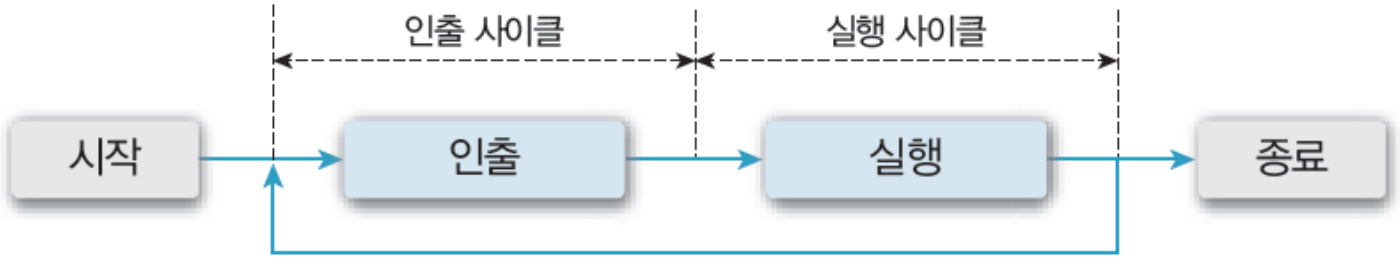


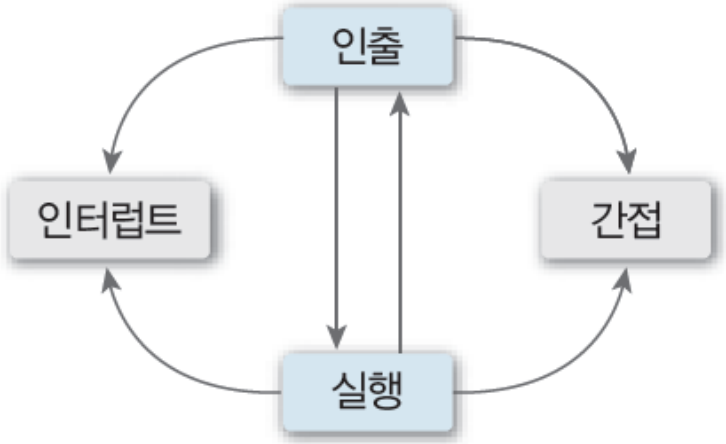
그림 1-18 명령어 실행 과정

명령어의 실행

▪ 명령어의 실행 사이클



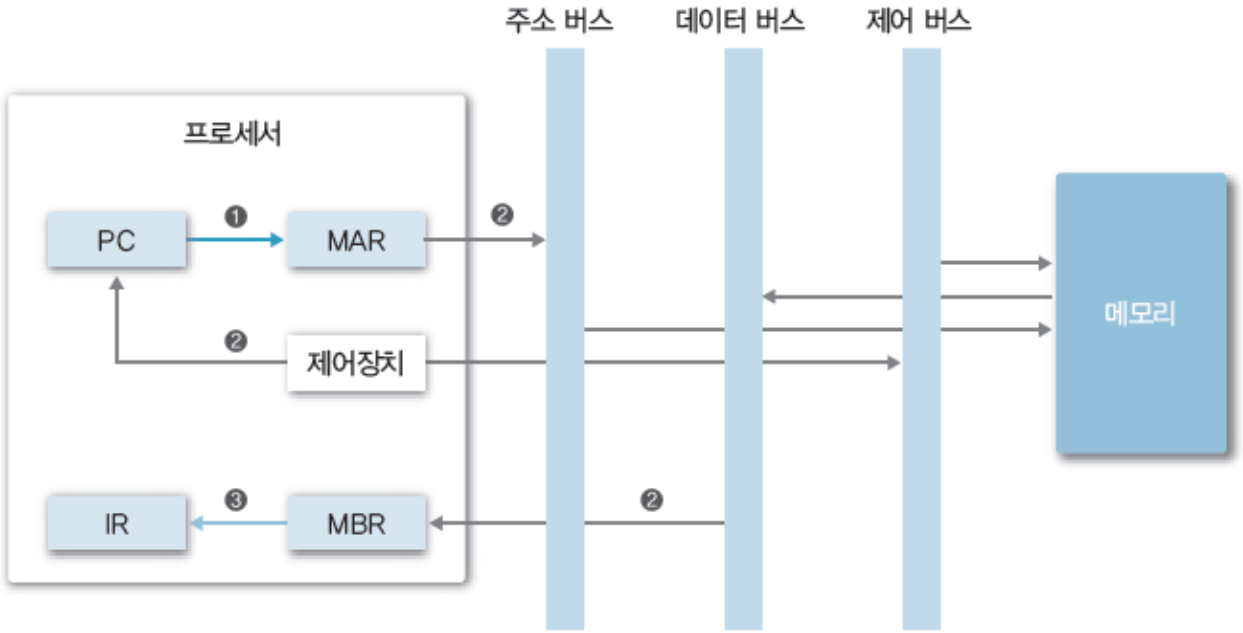
(a) 일반적인 명령어 사이클



(b) 세분화된 명령어 사이클

그림 1-19 명령어 실행 사이클

명령어의 실행 – 인출 사이클

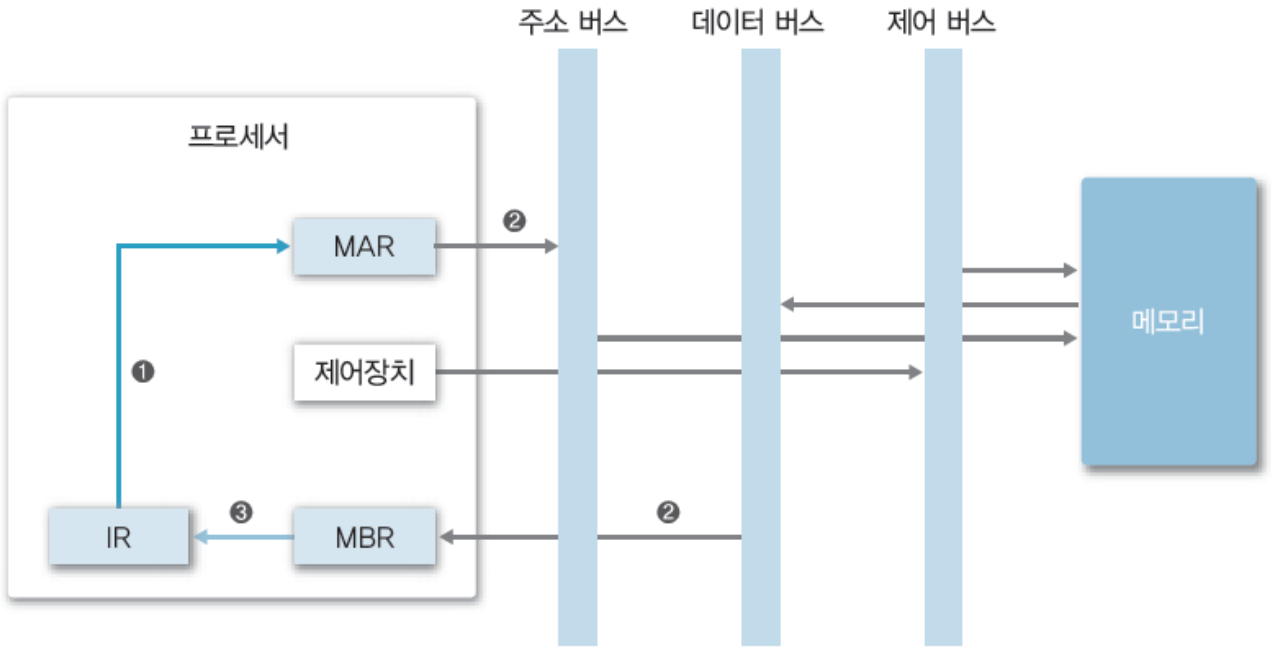


시간	레지스터 동작	설명
❶	PC → MAR	PC에 저장된 주소를 프로세서 내부 버스를 이용하여 MAR에 전달한다.
❷	Memory^{MAR} → MBR	MAR에 저장된 주소에 해당하는 메모리 위치에서 명령어를 인출한 후 이 명령어를 MBR에 저장한다. 이때 제어장치는 메모리에 저장된 내용을 읽도록 제어 신호를 발생시킨다.
	PC + 1 → PC	다음 명령어를 인출하려고 PC를 증가시킨다.
❸	MBR → IR	MBR에 저장된 내용을 IR에 전달한다.

- PC : 프로그램 카운터
- MBR : 메모리 버퍼 레지스터
- MAR : 메모리 주소 레지스터
- IR : 명령어 레지스터

그림 1-20 인출 사이클 과정

명령어의 실행 – 간접 사이클

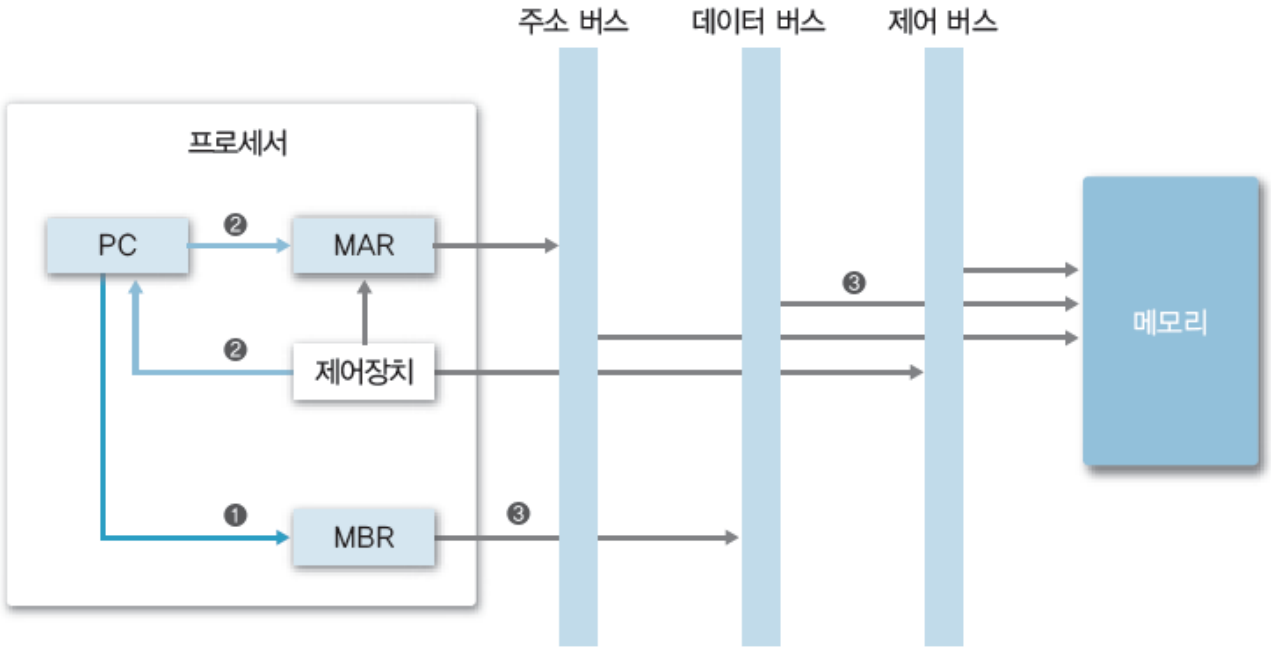


시간	레지스터 동작	설명
❶	$IR^{addr} \rightarrow MAR$	IR에 저장된 명령어의 피연산자(주소부)를 MAR에 전달한다.
❷	$Memory^{MAR} \rightarrow MBR$	MAR에 저장된 주소에 해당하는 메모리 위치에서 데이터를 인출한 후 이 데이터를 MBR에 저장한다. 이때 제어장치는 메모리에 저장된 내용을 읽도록 제어 신호를 발생시킨다.
❸	$MBR \rightarrow IR^{addr}$	MBR에 저장된 내용을 IR에 전달한다.

- IR : 명령어 레지스터
- MAR : 메모리 주소 레지스터
- MBR : 메모리 버퍼 레지스터

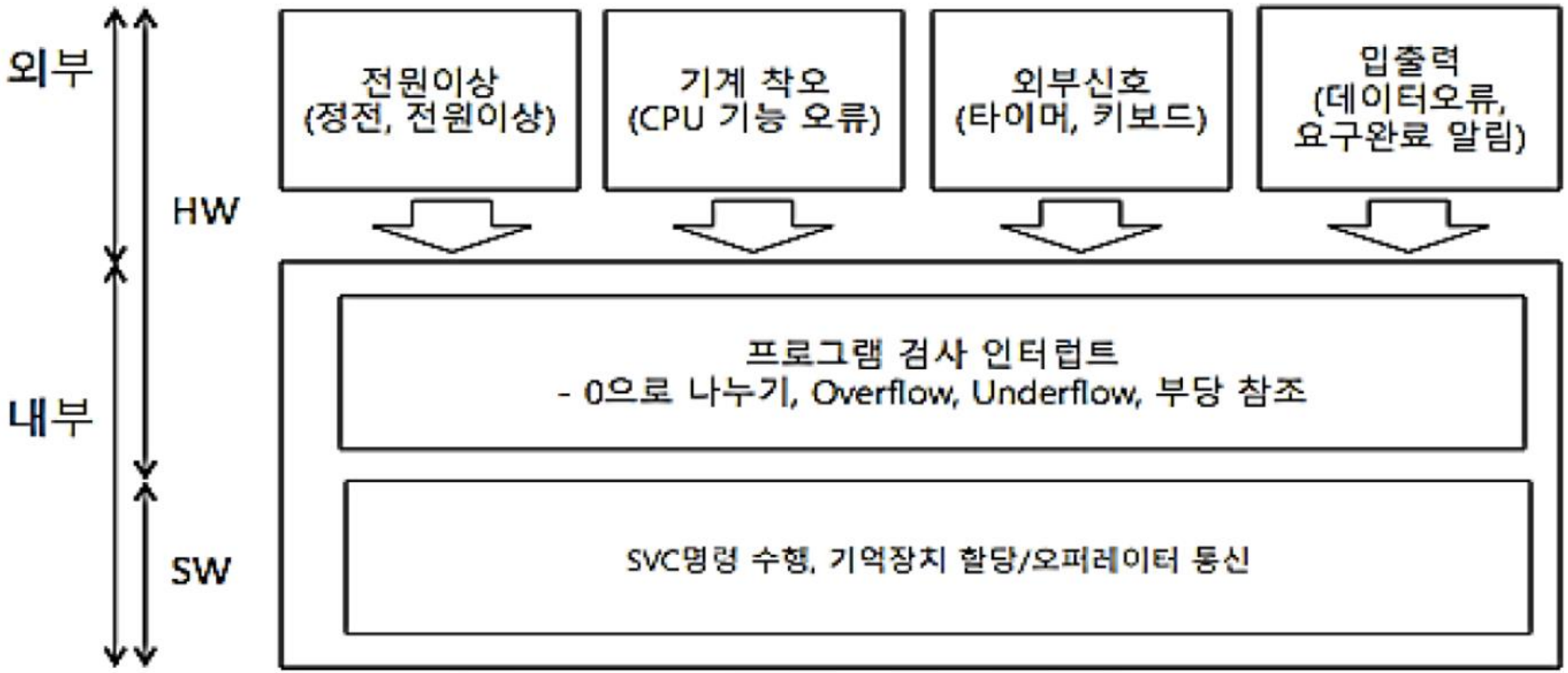
명령어의 실행 – 인터럽트 사이클

- **인터럽트**
  - 현재 실행 중인 프로그램을 중단하고 다른 프로그램의 실행을 요구하는 명령어

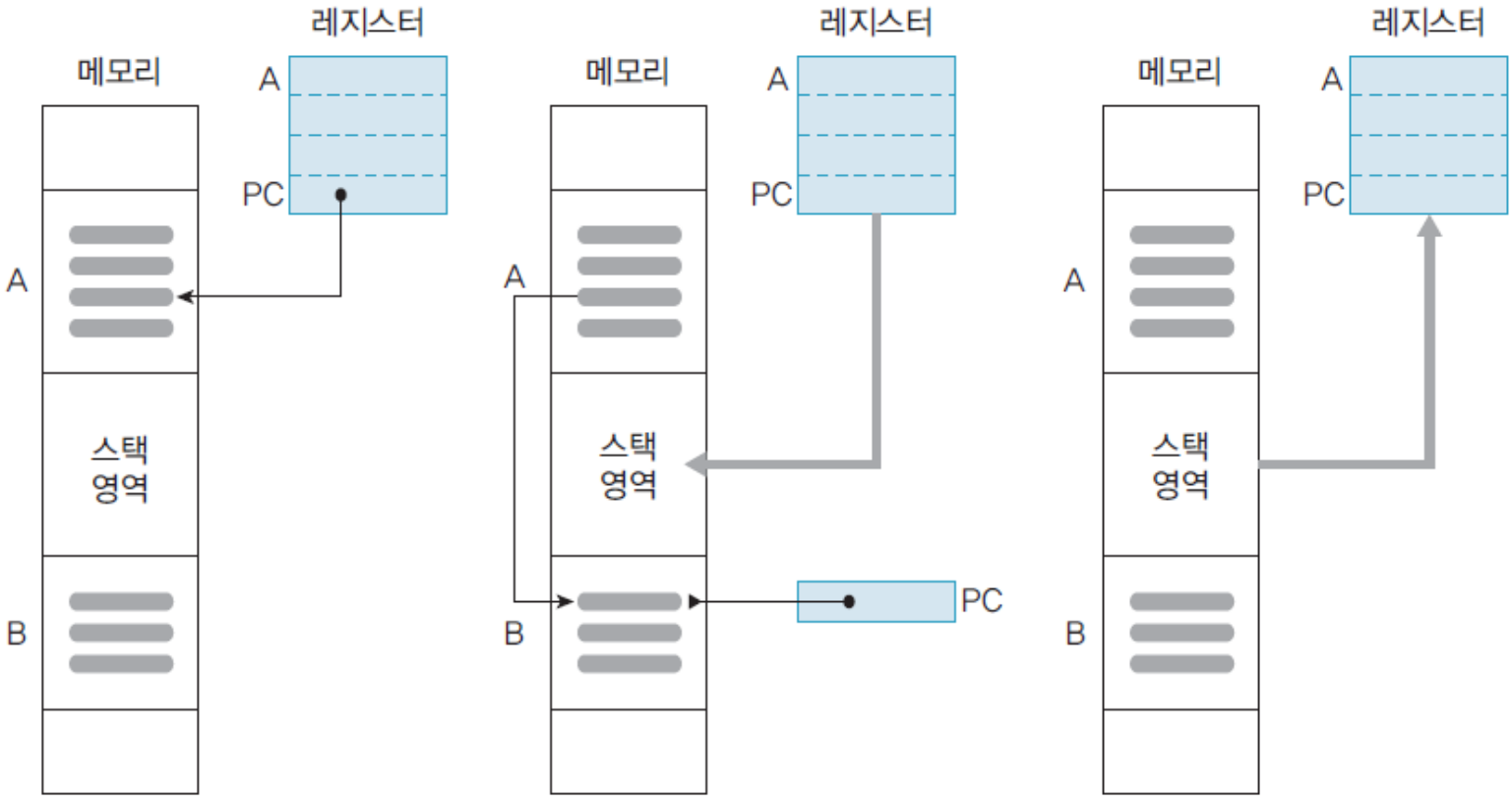


시간	레지스터 동작	설명
❶	PC → MBR	PC의 내용을 MBR에 저장한다.
❷	IntRoutine_Address → PC	인터럽트 루틴 주소를 PC에 저장한다.
	Save_Address → MAR	PC에 저장된 인터럽트 루틴 주소를 MAR에 저장한다.
❸	MBR → Memory^{MAR}	MBR의 주소에 있는 내용을 지시된 메모리 셀로 이동한다.

인터럽트 유형


외부
내부
HW
SW
전원이상
(정전, 전원이상)
기계 착오
(CPU 기능 오류)
외부신호
(타이머, 키보드)
입출력
(데이터오류,
요구완료 말림)
프로그램 검사 인터럽트
- 0으로 나누기, Overflow, Underflow, 부당 참조
SVC명령 수행, 기억장치 할당/오퍼레이터 통신

인터럽트 처리 과정



(a) 인터럽트 발생 전

프로그램 A를 실행, 프로그램 카운터PC는 현재 명령어를 가리킴

(b) 인터럽트 발생

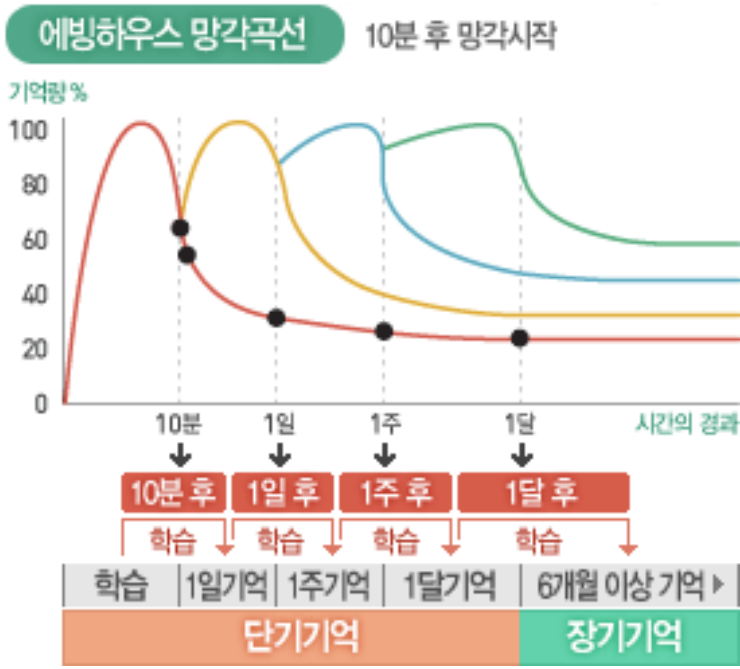
현재 명령어를 종료, 레지스터의 모든 내용을 스택 영역(또는 프로세스 제어 블록)에 보내고 pc에는 인터럽트 처리 프로그램(프로그램 B)의 시작 위치를 저장하고 제어를 넘긴

(c) 인터럽트 처리 후

스택 영역에 있던 내용을 레지스터에 다시 저장하며, 프로그램 A가 다시 시작하는 위치를 저장하고 중단했던 프로그램 A를 재실행

그림 1-24 인터럽트 처리 과정

망각 곡선 되살리기


에빙하우스 망각곡선
10분 후 망각시작
기억량 %
100
80
60
40
20
0
10분
1일
1주
1달
시간의 경과
10분 후
1일 후
1주 후
1달 후
학습
학습
학습
학습
학습
1일기억
1주기억
1달기억
6개월 이상 기억
단기기억
장기기억