idu 인덕대학교

컴퓨터 시스템 구조
– 기본 구성

컴퓨터소프트웨어학과

김병국 교수

학습목표

- 컴퓨터를 구성하는 하드웨어의 특성을 이해한다.
- 폰노이만 구조와 하버드 구조를 이해한다.
- CPU의 구성 및 동작 방식을 이해한다.

목차

1. 하드웨어의 구성 (1/5)

❑ 컴퓨터의 구성

- 필수 장치
  - 컴퓨팅(**연산 및 처리**)를 위한 필수 장치들
  - 중앙처리장치(CPU), 주 메모리(대부분의 작업이 이루어 짐)

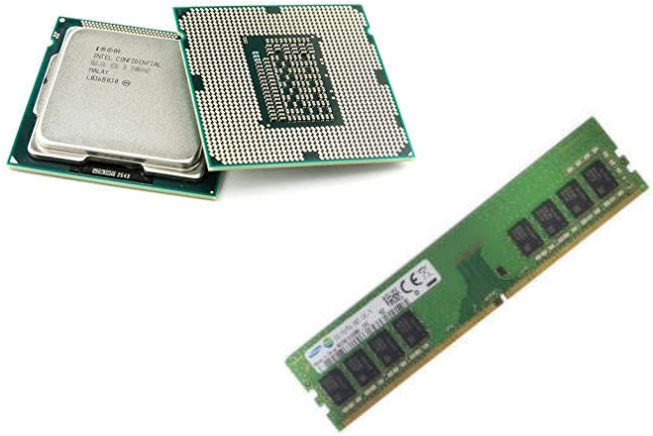
- 주변장치
  - 사용자와 대화를 위한 장치 및 데이터 저장을 위한 장치들
  - 입력장치, 출력장치, 저장장치

[컴퓨터를 구성하는 장치]

1. 하드웨어어의 구성 (2/5)

❑ 용어

- 중앙처리장치 = CPU, 주 프로세서(Main Processor)
- 주 메모리 = 메인 메모리(Main Memory) 또는 메모리
- 보조저장장치 = 저장장치 또는 데이터 스토리지



❑ CPU와 주 메모리

- CPU(Central Process Unit)
  - 명령어를 해석하여 실행하는 장치
  - 각종 연산을 수행
- 주 메모리(Main Memory)
  - 작업에 필요한 프로그램과 데이터를 저장하는 장소
  - 데이터의 접근 단위 : 워드(Word)

1. 하드웨어의 구성 (3/5)

❑입 · 출력장치

- 입력장치
  - 외부의 데이터를 컴퓨터에 입력하는 장치
  - 예: 키보드, 마우스, 마이크, 스캐너 등

- 출력장치
  - 처리 결과를 사용자가 원하는 형태로 출력하는 장치
  - 예: 모니터, 스피커, 프린터 등

1. 하드웨어의 구성 (4/5)

❑ 저장장치(디스크)

- 메모리보다 느리지만 **저렴하고 용량이 큼**
- 전원의 온·오프와 상관없이 **데이터를 영구적으로 저장**
- 느린 저장장치를 사용하는 이유 : **저장 용량에 비해 가격이 저렴**
- 종류
  - 자성(magnetic)을 이용하는 장치 : 카세트테이프, 플로피디스크, 하드디스크 등
  - 레이저를 이용하는 장치 : CD, DVD, 블루레이디스크 등
  - 메모리를 이용하는 장치 : USB 드라이브, SD 카드, SSD 등

[자성 기반]

[광 또는 레이저 기반]

[메모리 기반]

1. 하드웨어의 구성 (5/5)

❑ 메인보드(Main Board)

- 동일어: **마더보드**(Mother Board)
- CPU와 메모리 등 다양한 부품을 연결하는 커다란 판
- 다양한 장치들을 버스(bus)로 연결
  - 버스는 데이터가 지나다니는 통로
- 그래픽카드, 사운드카드, 랜카드 등의 기능이 기본으로 탑재되거나, 별도 장착이 가능

[메인보드 예]

2. 시스템 아키텍처(1/4)

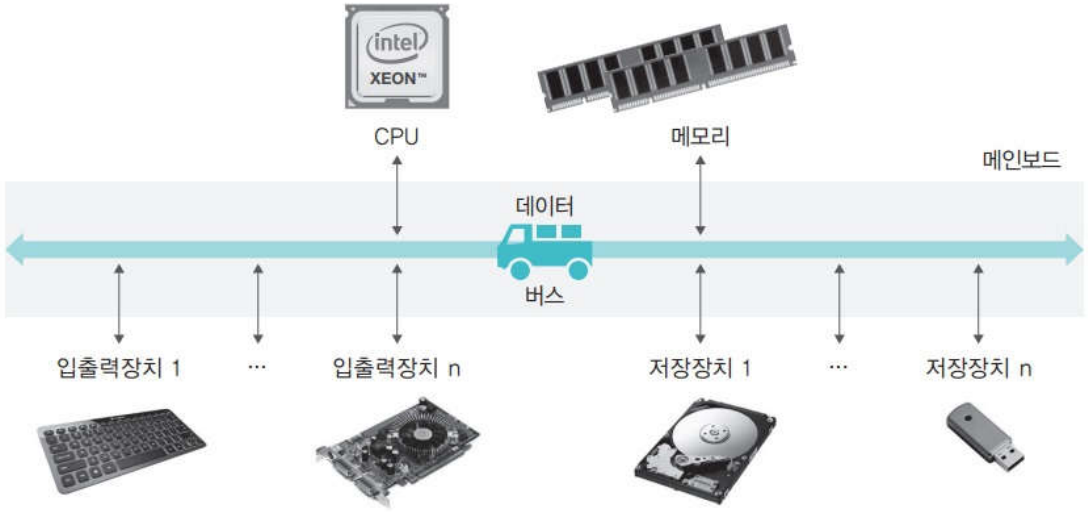
❑대표적인 종류

- 폰노이만 아키텍처
- 하버드 아키텍처

2. 시스템 아키텍처(2/4)

❑ 폰노이만 아키텍처 (1/2)

- Johann Ludwig Von-Neumann(1903.12~1957.02, 헝가리 출생)
- EDSAC(1949) 컴퓨터 개발 때 개념을 적용
- CPU, 메모리, 입출력 장치, 저장장치가 **하나의 버스로 연결**되어 있는 구조
- **명령어와 데이터를 위한 메모리 인터페이스가 하나임**
- **명령어를 읽을 때 데이터를 읽거나 쓸 수 없음**
- IBM 계열 PC(개인용 PC), ARM7 등

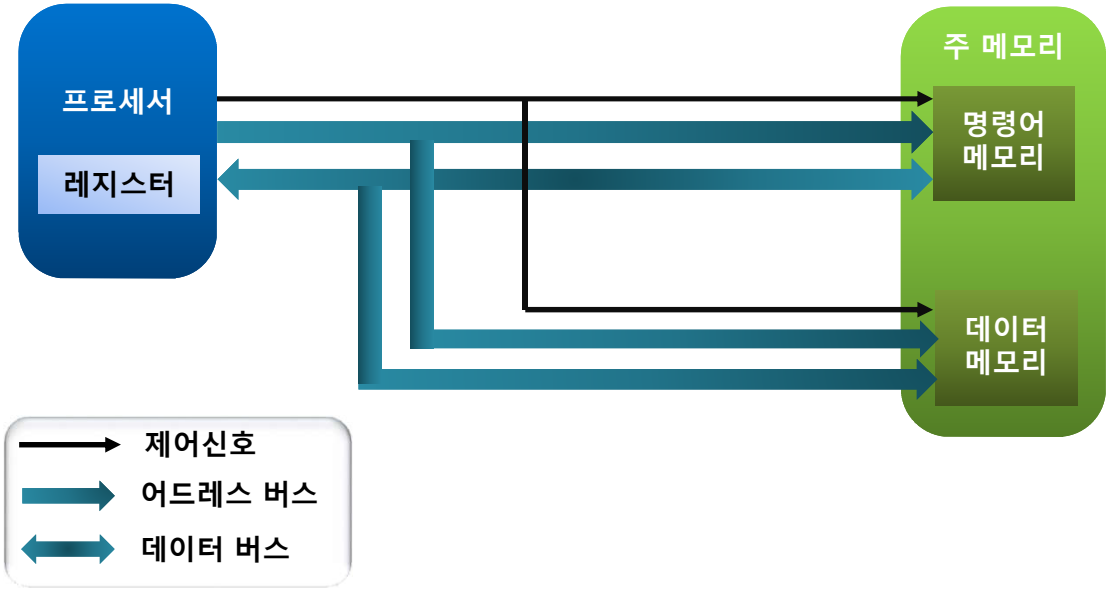


[폰노이만 구조 예]

2. 시스템 아키텍처(3/4)

❑폰노이만 아키텍처 (2/2)

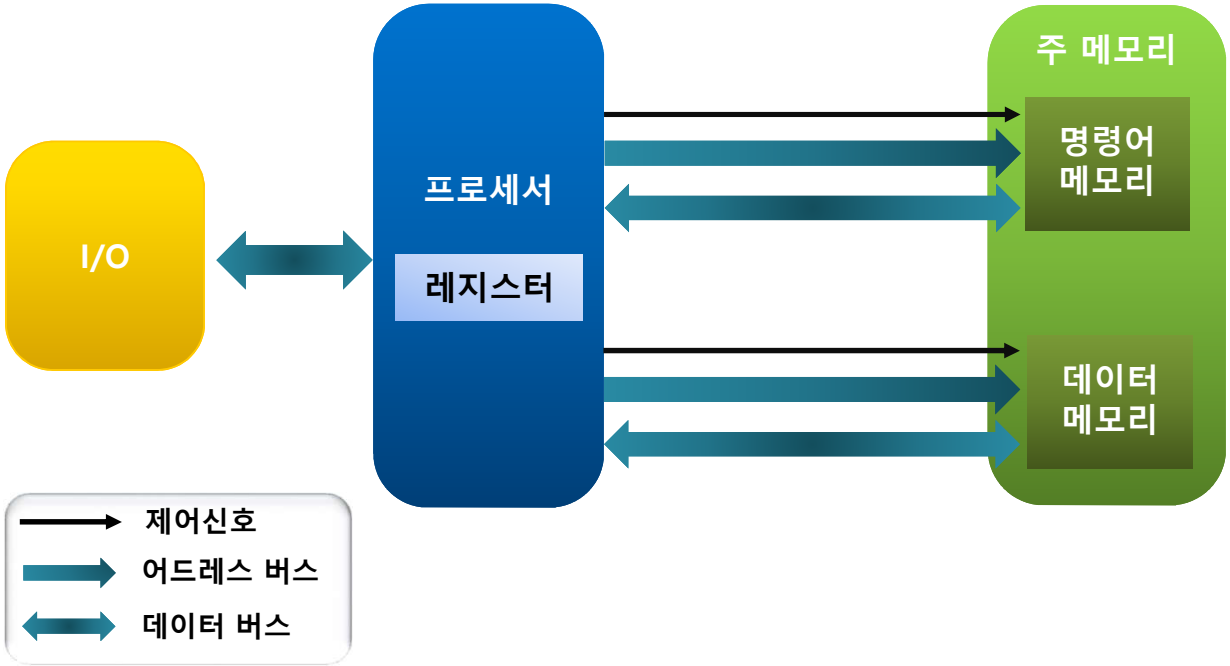
- 프로그램은 하드디스크와 같은 외부 저장 장치에 기록
- 프로그램의 구동을 위해서는 기록된 데이터를 주 메모리(RAM)로 읽어와야 함
- 메모리의 효율적인 접근에 따라 시스템의 성능에 큰 영향을 미침



2. 시스템 아키텍처(4/4)

❑ 하버드 아키텍처

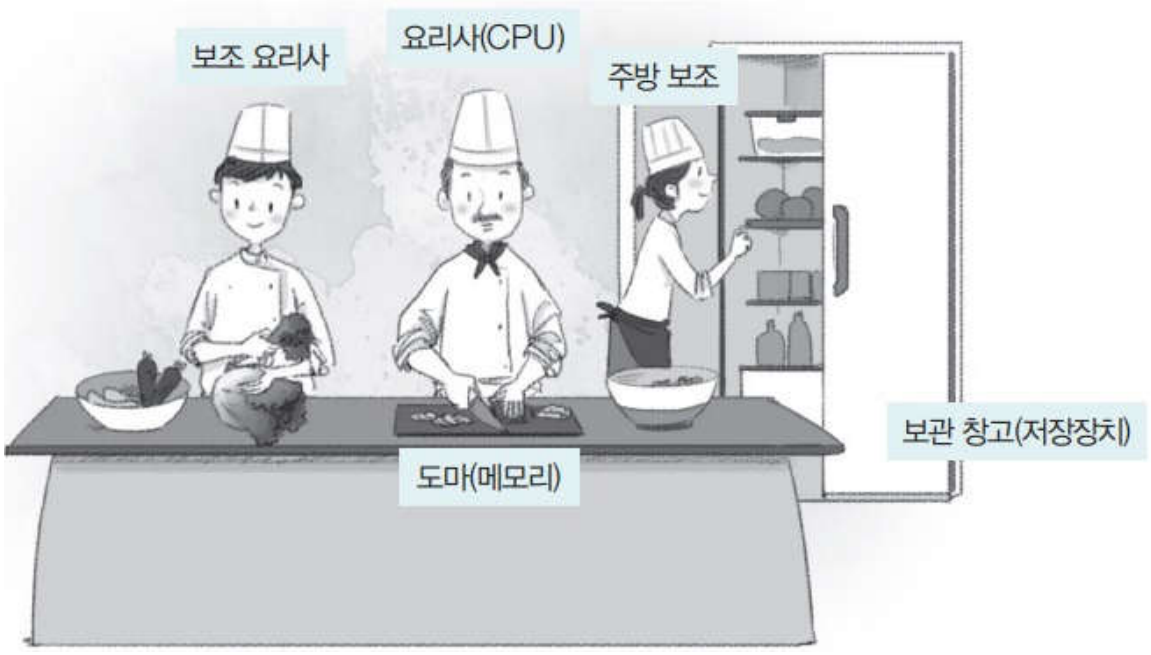
- 1944년 하버드 마크I(Harvard Mark I)을 개발할 때 설계 및 적용
- 명령어를 위한 메모리 인터페이스와 데이터를 위한 메모리 인터페이스가 분리되어 있음
- 버스 시스템이 복잡하여 설계가 복잡함(➔단가 상승)
- ARM9, ARM10, XScale 등



3. 요리사 모형 (1/3)

❑ 요리사 모형

- 운영체제의 여러 가지 현상에 대한 이해를 돕기 위한 것
- 요리사 → CPU, 도마 → 메모리, 보관 창고 → 저장장치에 각각 비유



[요리사 모형]

3. 요리사 모형 (2/3)

❑ 폰노이만 구조와 요리사 모형

- 요리사(CPU)가 요리(처리)를 하기 위해서는 보관 창고(저장장치)에 있는 재료를 도마(메모리) 위에 놓아야 함
- 요리사에게는 도마(메모리)가 **핵심적인 작업 공간임**
- 보관 창고(저장장치)는 **보조적인 공간임**

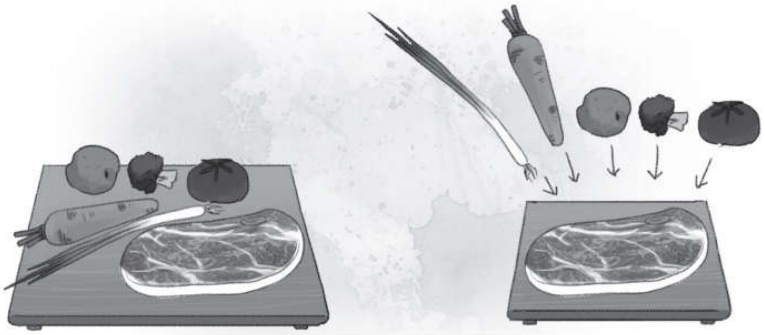
요리사 모형	운영체제 작업
요리 방법 결정	프로세스 관리
도마 정리	메모리 관리
보관 창고 정리	저장장치 관리

[요리사 모형과 운영체제 작업의 비교]

3. 요리사 모형 (3/3)

❑ 낮은 용량의 메모리의 컴퓨터가 느린 이유

- 도마가 크면 더 많은 재료를 올려놓고 요리할 수 있음
  - 반대로, 도마가 작으면 재료를 조금만 올려놓고 작업해야 함
- 작은 도마에서 작업을 하기 위해서는 보관 창고에 가져다 놓은 뒤 다른 재료를 가져와야 하므로 재료를 손질하는 시간보다 보관 창고로 옮기는 시간이 많이 걸림 → 처리 속도 저하
- 도마의 크기가 전체 재료를 놓을 수 있을 만큼 충분히 크다면 작업 속도에 영향을 미치지 않음
  - 전체 재료보다 큰 도마는 더 큰 성능을 발휘할 수 없음



[도마의 크기와 작업 속도]

4. 하드웨어 사양 관련 용어 (1/2)

❑ 클록(Clock)

- CPU의 속도와 관련된 단위
- 클록이 일정 간격으로 틱(tick)을 만들면 거기에 맞추어 CPU안의 모든 구성 부품이 작업을 수행함
- 틱(Tick) = 펄스(Pulse) = 클록 틱(Clock Tick)

❑ 헤르츠(Hz)

- 클록틱이 발생하는 속도를 나타내는 단위
  - 1초에 클록틱이 한 번이면 1Hz,
  - 1,000번이면 1kHz(1,000Hz)
  - 3.4GHz는 1초에 틱이 3.4×10^9번 발생(34억)

부품	사양
CPU	인텔 코어 i7(4코어, 3.4GHz, 캐시 4MB)
메인보드	FSB 1,333MHz
메모리	DDR3 SDRAM 4GB(1,333MHz)

4. 하드웨어 사양 관련 용어 (2/2)

❑ 시스템 버스

- 메모리와 주변장치를 연결하는 버스
- FSB(Front-Side Bus, 전면 버스)라고 함

❑ CPU 내부 버스

- CPU 내부에 있는 버스(CPU 내 구성 요소 간 연결)
- BSB(Back-Side Bus, 후면 버스)라고 함

❑ CPU와 메모리의 속도

- CPU는 CPU 내부 버스(BSB)의 속도로 작동
- 메모리는 시스템 버스(FSB)의 속도로 작동
- 두 버스의 속도 차이로 인하여 작업이 지연될 수 있음
  - 캐쉬(cache)로 해결

5. CPU의 구성과 동작 (1/5)

❑ 산술논리 연산장치(ALU: Arithmetic Logic Unit)

- 데이터의 덧셈, 뺄셈, 곱셈, 나눗셈 같은 산술 연산과 AND, OR 같은 논리 연산을 수행

❑ 제어장치(Control Unit)

- CPU에 작업을 지시

❑ 레지스터(Register)

- CPU 내에 데이터를 임시로 보관

❑ 버스(Bus)

- CPU내 구성요소들과의 데이터 전달을 위한 통로

[CPU의 구성요소]

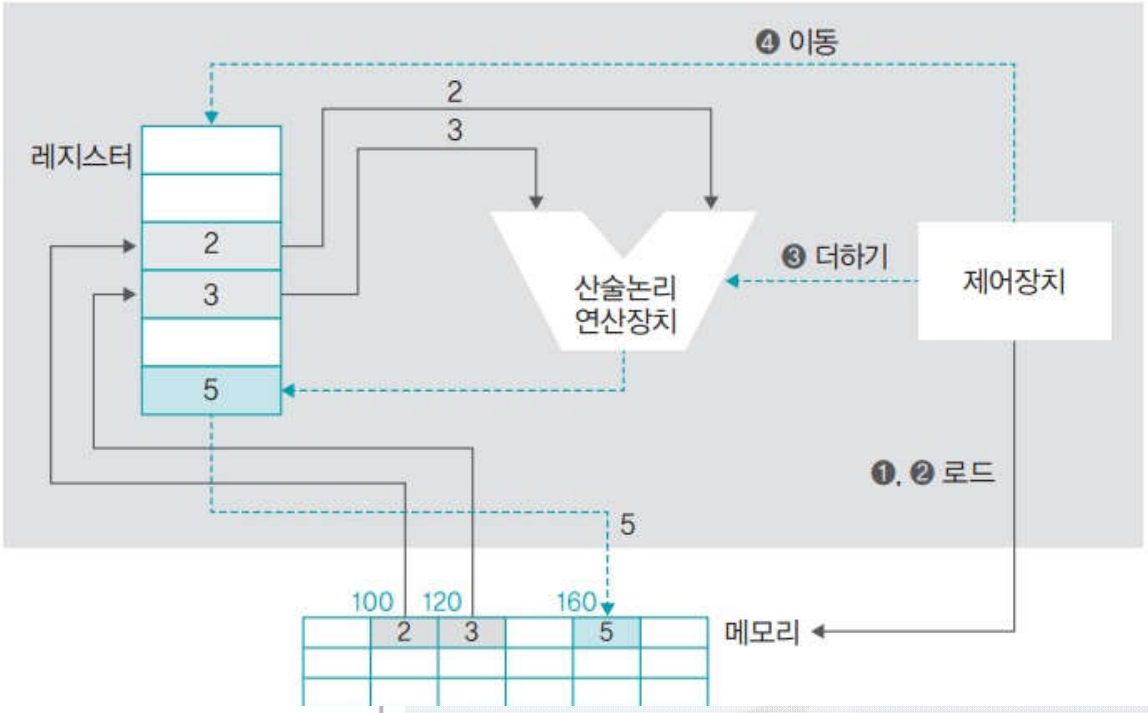
5. CPU의 구성과 동작 (2/5)

❑ CPU의 명령어 처리 과정

int D2=2, D3=3, sum;

sum=D2+D3;

- 위 코드를 어셈블리어로 바꾸면 다음과 같음




```
01  LOAD mem(100), register 2;
02  LOAD mem(120), register 3;
03  ADD register 5, register 2, register 3;
04  MOVE register 5, mem(160);
```

- 01행 : 메모리 100번지(D2)에 있는 값을 레지스터 2로 가져옴
- 02행 : 메모리 120번지(D3)에 있는 값을 레지스터 3으로 가져옴
- 03행 : 레지스터 2와 3에 저장된 값을 더한 결과를 레지스터 5에 저장
- 04행 : 레지스터 5의 값을 메모리 160번지(sum)에 저장

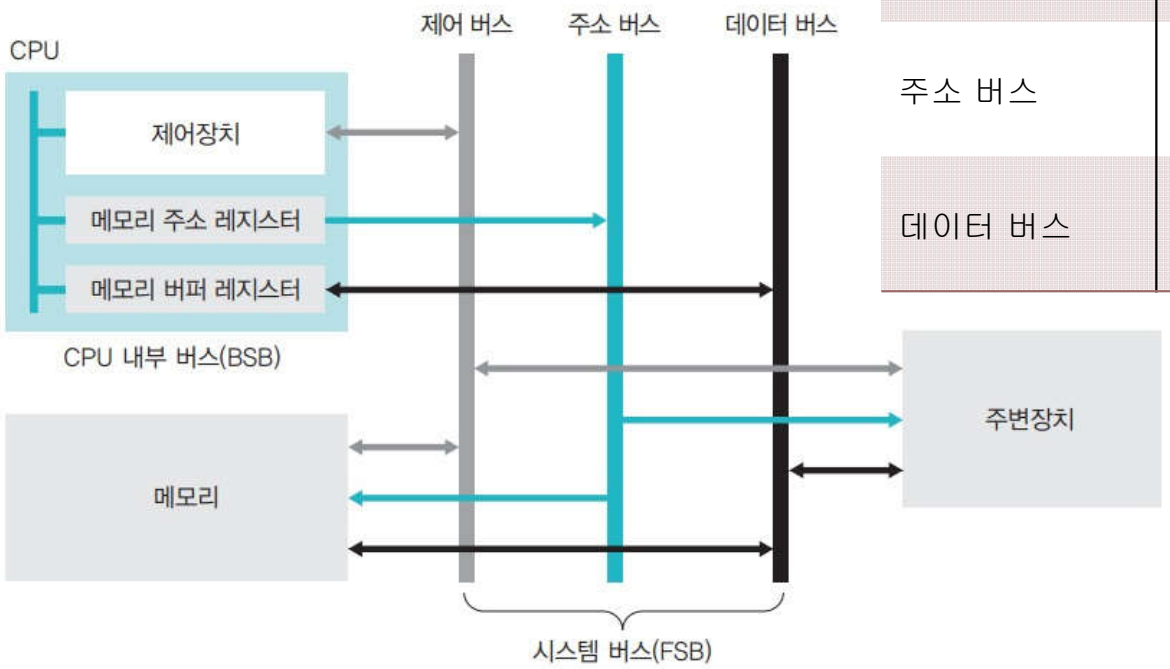
5. CPU의 구성과 동작 (3/5)

레지스터	기능
데이터 레지스터 (DR: Data Register)	피연산자(Operand)의 값들이 임시로 저장되는 메모리 공간
주소 레지스터 (AR: Address Register)	명령어(Instruction Set) 또는 데이터가 저장된 메모리의 주소를 저장
프로그램 카운터 (PC: Program Counter)	다음에 실행할 명령어의 위치 코드(일반적으로 메모리 주소)를 저장
명령어 레지스터 (IR: Instruction Register)	실행할 명령어를 저장
메모리 주소 레지스터 (MAR: Memory Address Register)	메모리 관리자(MMU: Memory Management Unit)가 접근해야 할 메모리의 주소
메모리 버퍼 레지스터 (MBR: Memory Buffer Register)	메모리 관리자(MMU)가 가져온 데이터를 임시로 저장
프로그램 상태 레지스터 (PSR: Program State/Status Register)	연산/처리 결과를 저장하는 공간

[주요 레지스터의 종류]

5. CPU의 구성과 동작 (4/5)

❑ 버스의 종류



버스	특징
제어 버스	• 제어장치와 연결된 버스 • CPU가 메모리와 주변장치에 제어 신호를 보내기 위해 사용 • 양방향
주소 버스	• 메모리 주소 레지스터와 연결된 버스 • 접근할 메모리의 주소를 알려주기 위해 사용 • 단방향
데이터 버스	• 메모리 버퍼 레지스터와 연결된 버스 • 데이터의 이동 • 양방향

5. CPU의 구성과 동작 (5/5)

❑ 버스의 대역폭

- 한 번에 전달할 수 있는 데이터의 최대 크기
- CPU가 한 번에 처리할 수 있는 데이터의 크기와 같음

- CPU는 워드(word)단위로 데이터를 처리
  - 예: 32bit CPU
    - 메모리에서 데이터를 읽거나 쓸 때 한 번에 최대 32bit 처리
    - 레지스터의 크기 : 32bit,
    - 버스의 대역폭 : 32bit

- 일반적으로 버스의 대역폭, 레지스터의 크기, 메모리에 한 번에 저장할 수 있는 데이터의 크기는 동일하게 설계

수고하셨습니다.