Chapter 1

컴퓨터 시스템 개요

Computer system overview

컴퓨터 하드웨어

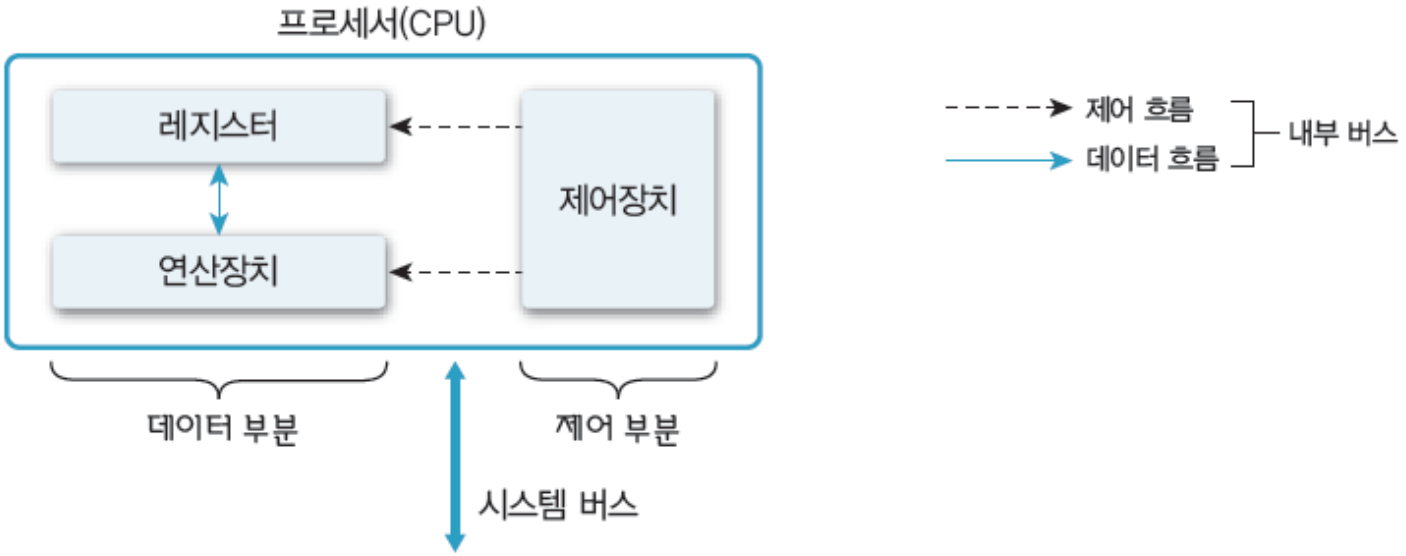
- **프로세서 (Processor)**
  - CPU
  - 그래픽카드 (GPU)
  - 응용 전용 처리장치 등

- **메모리 (Memory)**
  - 주 기억장치
  - 보조 기억장치 등

- **주변장치**
  - 키보드/마우스
  - 모니터, 프린터
  - 네트워크 모뎀 등

프로세서 (Processor)

- **컴퓨터의 두뇌** (중앙처리장치)
  - 연산 수행
  - 컴퓨터의 모든 장치의 동작 제어



레지스터 (Register)

- **프로세서 내부에 있는 메모리**
  - 프로세서가 사용할 데이터 저장
  - 컴퓨터에서 가장 빠른 메모리

- **레지스터의 종류**
  - 용도에 따른 분류
    - 전용 레지스터, 범용 레지스터
  - 사용자가 정보 변경 가능 여부에 따른 분류
    - 사용자 가시 레지스터, 사용자 불가시 레지스터
  - 저장하는 정보의 종류에 따른 분류
    - 데이터 레지스터, 주소 레지스터, 상태 레지스터

레지스터 (Register)

register.c

```c
#include <stdio.h>
#include <stdlib.h>    // malloc, free 함수가 선언된 헤더 파일

int main()
{
    register int num1 = 10;      // 변수 num1은 CPU의 레지스터를 사용

    printf("%d\n", num1);
```

출처: https://dojang.io/mod/page/view.php?id=804

표 1-1 사용자 가시 레지스터

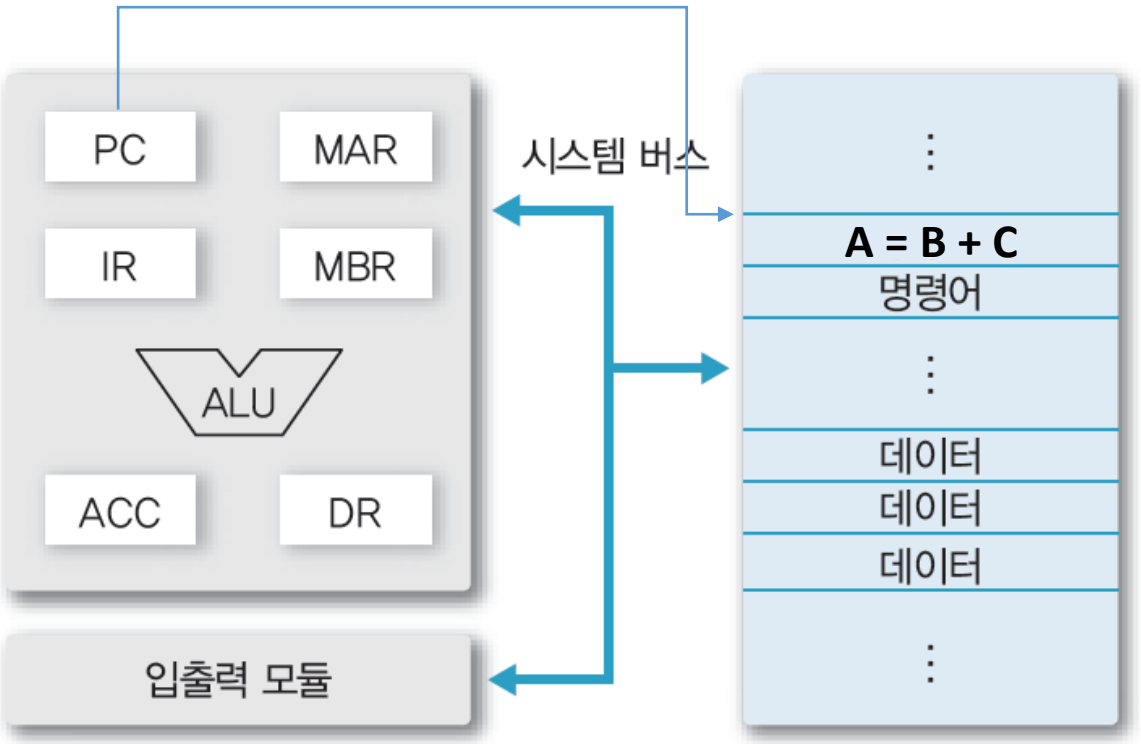
종류		
데이터 레지스터 DR, Data Register	함수 연산에 필요한 데이터	
주소 레지스터 AR, Address Register	주소나 유효 주소를 계산하는 데 필요한 주소의 일부분을 저장한다. 주소 레지스터에 저장한 값(값 데이터)을 사용하여 산술 연산을 할 수 있다.	
	기준 주소 레지스터	프로그램을 실행할 때 사용하는 기준 주소 값을 저장한다. 기준 주소는 하나의 프로그램이나 일부처럼 서로 관련 있는 정보를 저장하며, 연속된 저장 공간을 지정하는 데 참조할 수 있는 주소이다. 따라서 기준 주소 레지스터는 페이지나 세그먼트처럼 블록화된 정보에 접근하는 데 사용한다.
	인덱스 레지스터	유효 주소를 계산하는 데 사용하는 주소 정보를 저장한다.
	스택 포인터 레지스터	메모리에 프로세서 스택을 구현하는 데 사용한다. 많은 프로세서와 주소 레지스터를 데이터 스택 포인터와 큐 포인터로 사용한다. 보통 반환 주소, 프로세서 상태 정보, 서브루틴의 임시 변수를 저장한다.

레지스터 (Register)

표 1-2 사용자 불가시 레지스터

종류	설명
프로그램 카운터PC, Program Counter	다음에 실행할 명령어의 주소를 보관하는 레지스터이다. 계수기로 되어 있어 실행할 명령어를 메모리에서 읽으면 명령어의 길이만큼 증가하여 다음 명령어를 가리키며, 분기 명령어는 목적 주소로 갱신할 수 있다.
명령어 레지스터IR, Instruction Register	현재 실행하는 명령어를 보관하는 레지스터이다.
누산기ACC, ACCumulator	데이터를 일시적으로 저장하는 레지스터이다.
메모리 주소 레지스터MAR, Memory Address Register	프로세서가 참조하려는 데이터의 주소를 명시하여 메모리에 접근하는 버퍼 레지스터이다.
메모리 버퍼 레지스터MBR, Memory Buffer Register	프로세서가 메모리에서 읽거나 메모리에 저장할 데이터 자체를 보관하는 버퍼 레지스터이다. 메모리 데이터 레지스터MDR, Memory Data Register라고도 한다.

프로세서의 동작



- ALU Arithmetic Logic Unit : 산술 · 논리 연산장치

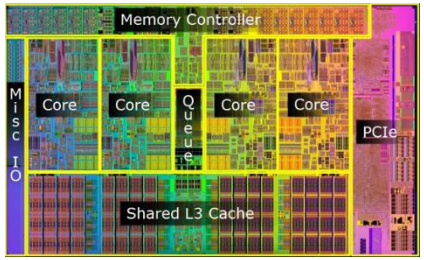
운영체제와 프로세서

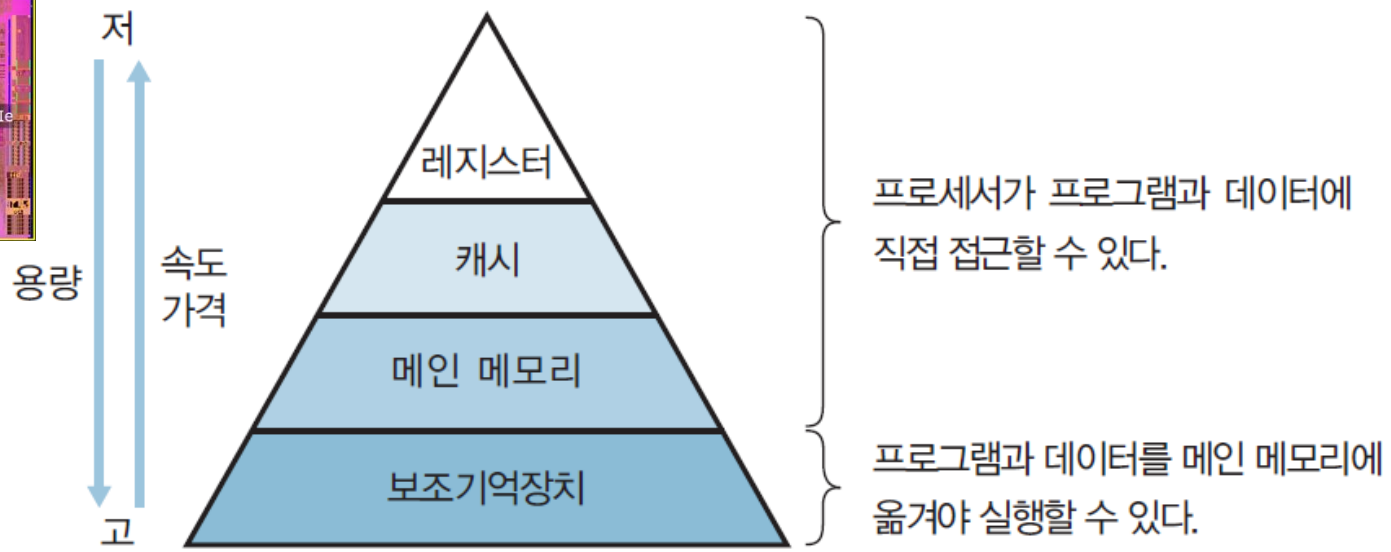
- **프로세서에게 처리할 작업 할당 및 관리**
  - 프로세스(Process) 생성 및 관리
  - 3장
- **프로그램의 프로세서 사용 제어**
  - 프로그램의 프로세서 사용 시간 관리
  - 복수 프로그램간 사용 시간 조율 등
  - 4~5장

메모리 (Memory)

- **데이터를 저장하는 장치** (기억장치)
  - 프로그램(OS, 사용자SW 등), 사용자 데이터 등

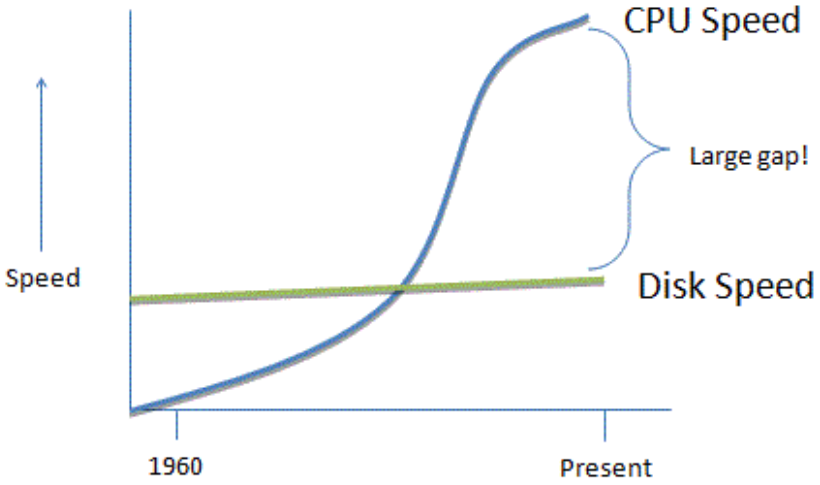
- **메모리의 종류**

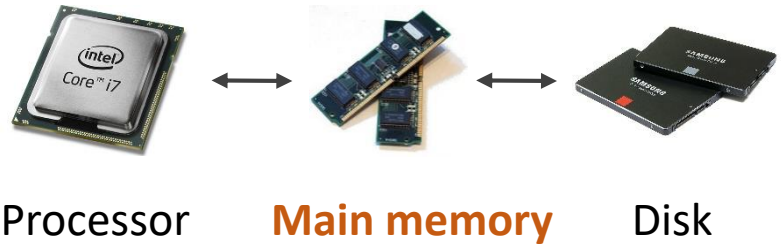




메모리의 종류

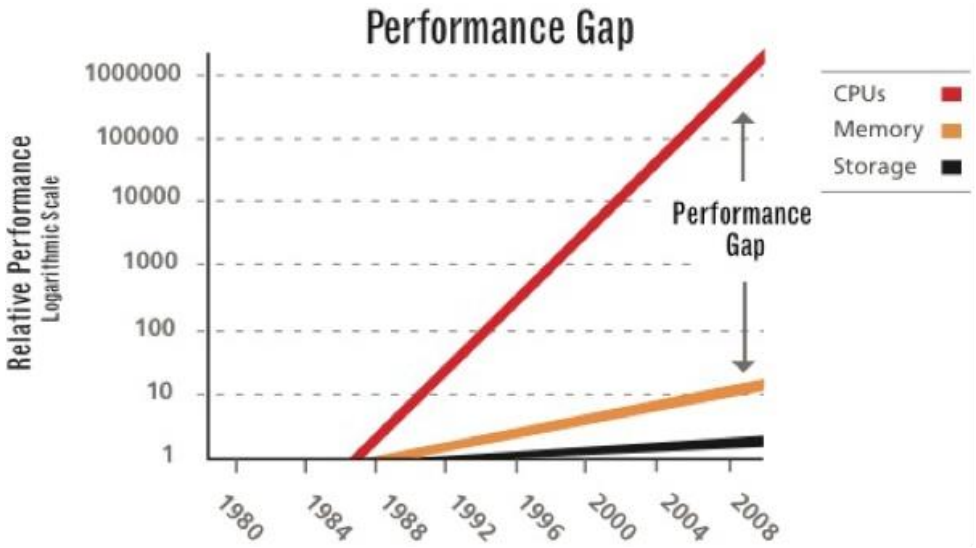
- **주기억장치 (Main memory)**
  - 프로세서가 수행할 프로그램과 데이터 저장
  - DRAM을 주로 사용
    - 용량이 크고, 가격이 저렴
  - **디스크 입출력 병목현상(I/O bottleneck) 해소**

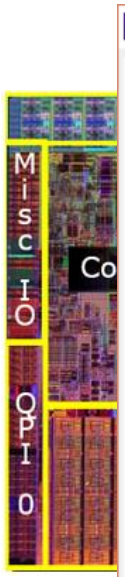


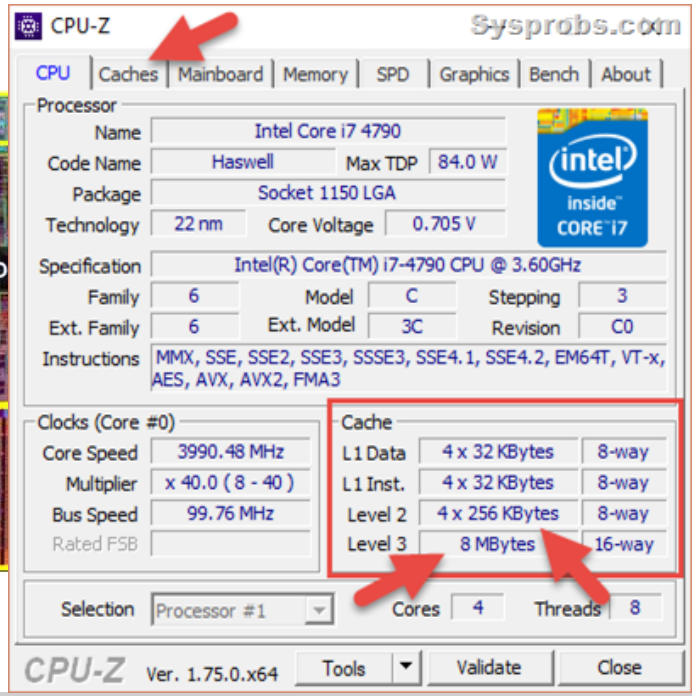


메모리의 종류

- **캐시 (Cache)**
  - 프로세서 내부에 있는 메모리 (L1, L2 캐시 등)
    - 속도가 빠르고, 가격이 비쌈
  - **메인 메모리의 입출력 병목현상 해소**

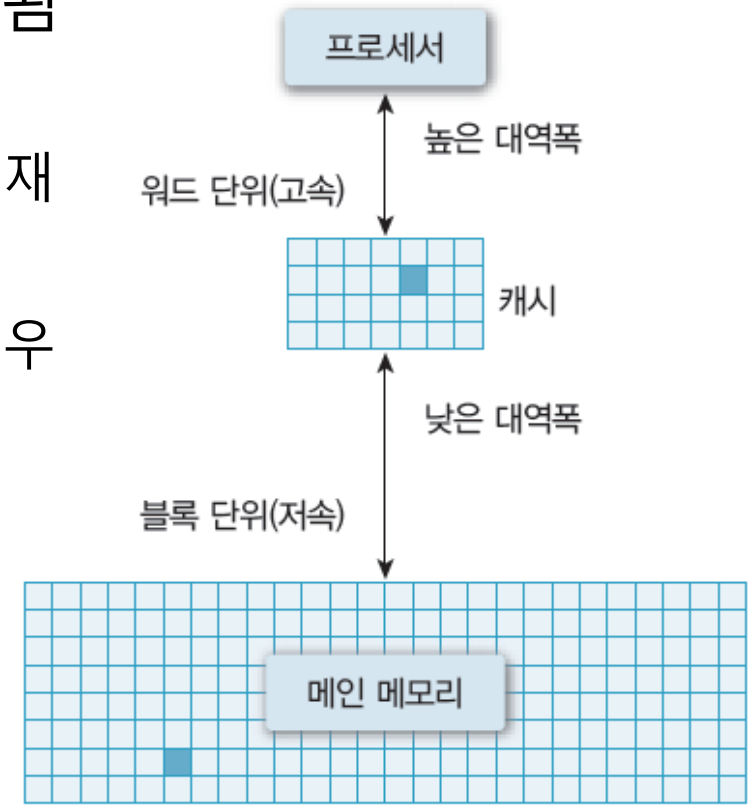






캐시의 동작

- **캐시의 동작**
  - 일반적으로 HW적으로 관리 됨
  - **캐시 히트 (Cache hit)**
    - 필요한 데이터 블록이 캐시 존재
  - **캐시 미스 (Cache miss)**
    - 필요한 데이터 블록이 없는 경우

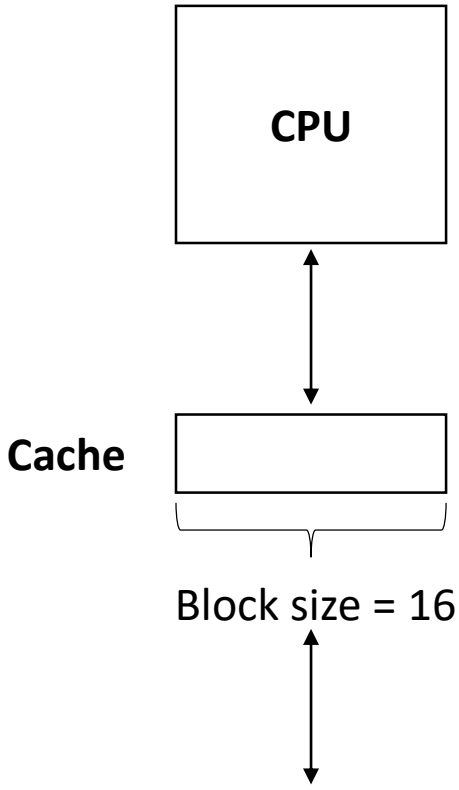


지역성 (Locality)

- **공간적 지역성 (Spatial locality)**
  - 참조한 주소와 인접한 주소를 참조하는 특성
    - 예) 순차적 프로그램 수행
- **시간적 지역성 (Temporal locality)**
  - 한 번 참조한 주소를 곧 다시 참조하는 특성
    - 예) For 문 등의 순환 문
- **지역성은 캐시 적중률(cache hit ratio)과 밀접**
  - 알고리즘 성능 향상 위한 중요한 요소 중 하나

지역성 (Locality)

```
for (i = 0 ; i <= n ; i++) {
    for (j = 0 ; j < m ; j++) {
        x = x + (a[i][j]) ; // A
            or
        x = x + (a[j][i]) ; // B
    }
}
```



Main memory

A[0][0~15]	A[0][16~31]	…	A[0][~ m]
…	…	…	…
A[n][0~15]	A[n][16~31]	…	A[n][~ m]

메모리의 종류

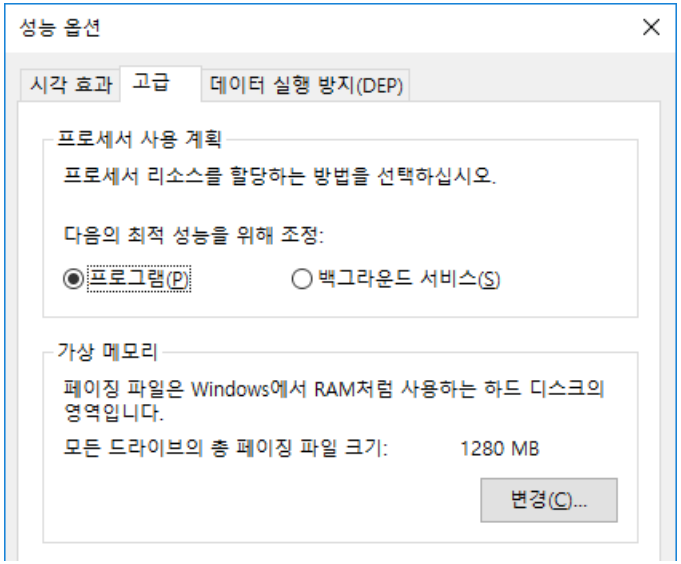
- **보조기억 장치**
  (Auxiliary memory / secondary memory / storage)
  - 프로그램과 데이터를 저장
  - 프로세서가 직접 전근할 수 없음 (주변장치)
    - 주기억장치를 거쳐서 접근
    - (프로그램/데이터 > 주기억장치) 인 경우는 ?
      - 가상 메모리(Virtual memory
  - 용량이 크고, 가격이 저렴

메모리와 운영체제

- **메모리 할당 및 관리**
  - 프로그램의 요청에 따른 메모리 할당 및 회수
  - 할당된 메모리 관리
  - 7장

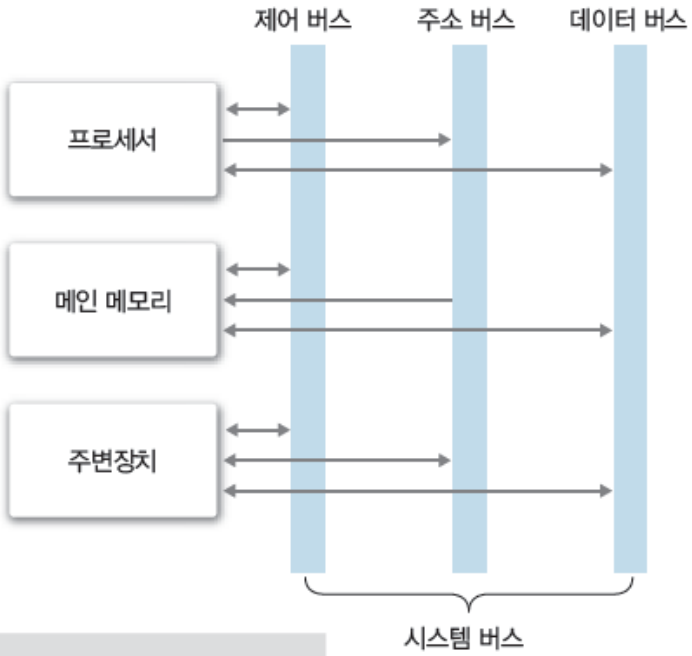
```
int main( void )
{
   char *string;

   // Allocate space for a path name
   string = malloc( _MAX_PATH );
```

- **가상 메모리 관리**
  - 가상메모리 생성 및 관리
  - 논리주소 → 물리주소 변환
  - 8장



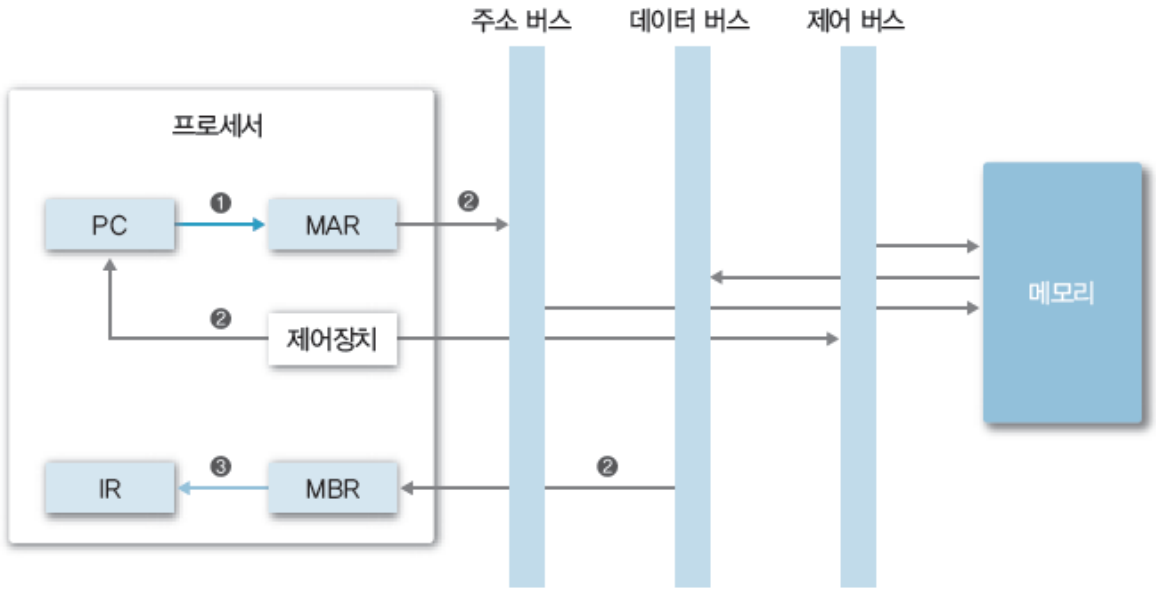
시스템 버스 (System Bus)

- 하드웨어들이 데이터 및 신호를 주고 받는 물리적인 통로



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.

시스템 버스 (System Bus)



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

주변 장치

- **프로세서와 메모리를 제외 하드웨어들**
  - 입력장치



  - 출력장치



  - 저장장치


주변장치와 운영체제

- **장치드라이버 관리**
  - 주변 장치 사용을 위한 인터페이스 제공
- **인터럽트 (Interrupt) 처리**
  - 주변 장치의 요청 처리
  - 3장
- **파일 및 디스크 관리**
  - 파일 생성 및 삭제
  - 디스크 공간 관리 등
  - 9~10장