# 레지스터 전송과 마이크로 연산

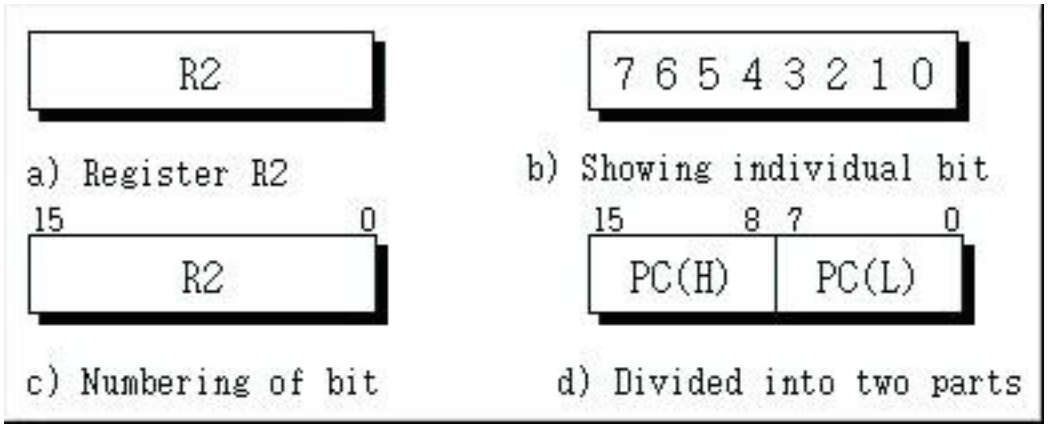
# 레지스터 전송과 마이크로 연산

- 마이크로 연산(micro-operation) : 레지스터에 저장된 데이터를 가지고 실행되는 동작
  - shift, count, clear, load, etc
- 디지털 컴퓨터의 구조 정의를 위한 규정
  - 레지스터의 종류와 그 기능
  - 레지스터에 저장된 이진 정보를 가지고 수행되는 마이크로 연산
  - 마이크로 연산을 시작시키는 제어 기능
- 레지스터 전송 언어( register transfer language)
  - 레지스터 간의 마이크로 연산을 간단하고 명료하게 표시하기 위해 사용하는 기호

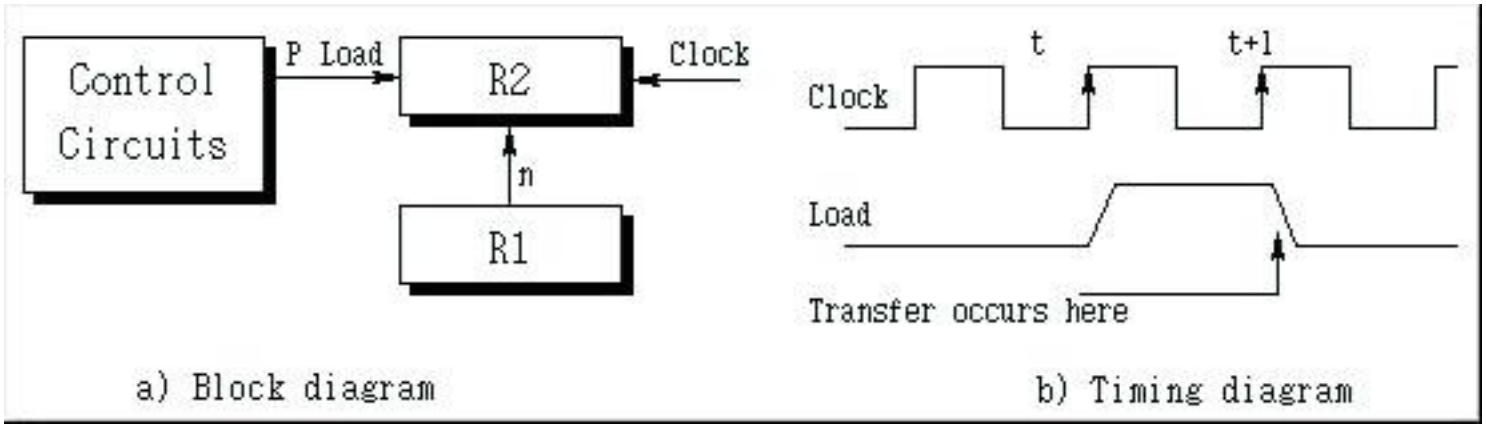
Register : CPU 내부에 있는 특수한 목적의 메모리

: 레지스터는 어떻게 구성(설계)되어 있는가?

- 레지스터의 표시
  - 기능을 나타내기 위해 머리글자를 대문자로 표시
  - 예) MAR(memory address register), PC(program counter), IR(instruction register), R1 등
- 레지스터의 각 플립 플롭들은 맨 오른쪽 것을 0으로 하여 차례로 번호를 붙인다.
- 레지스터의 블럭도



- 레지스터간의 마이크로 연산 표시
  - 레지스터들 사이의 정보 전송 표시 : R2 ← R1
  - 제어 조건에 의한 레지스터 전송 표시 : If (P=1) then (R2←R1)
  - 제어 조건을 제어 함수로 표시, 제어 함수는 0또는 1의 값을 갖는 부울 변수 : P : R2 ← R1
  - P=1일 때 R1으로부터 R2로의 전송 블럭도와 타이밍도
    - Load 신호가 high이고 클럭이 positive edge trigger시 R1 데이터가 R2로 전송



- 레지스터 전송을 표시하는 기본기호

표4-1 레지스터 전송언어의 기본기호

| Symbol | Description | Examples |
|---|---|---|
| Letters (& numerals) | Denotes a register | MAR, R2 |
| Parenthese() | Denotes a part of a register | R2(0-7), R2(L) |
| Arrow ← | Denotes transfer of information | R2←R1 |
| Comma , | Separates two microoperation | R2←R1, R1←R2 |

- 숫자가 뒤에 붙는 영어 대문자: 레지스터 표시
- 괄호: 레지스터의 일부분 표시
- 쉼표: 동시에 일어나는 여러 개의 동작
- 화살표: 전송의 방향 표시
  - 예) T=1일 때, 두 레지스터의 내용이 동시에 교체
  - T : R2 ← R1, R1 ← R2

# 버스와 메모리 전송

- 버스 시스템
  - bus : 레지스터와 레지스터들 사이의 정보 전송을 위한 경로
  - 전송 회선의 수를 줄이고 효율적으로 방법으로 전송
  - 한 번에 하나의 전송만이 가능
    - 제어 신호를 이용하여 전송 레지스터를 선택
- 공통버스를 구성하는 방법
  - 멀티플렉서의 이용
  - 4개의 레지스터에 대한 버스 시스템

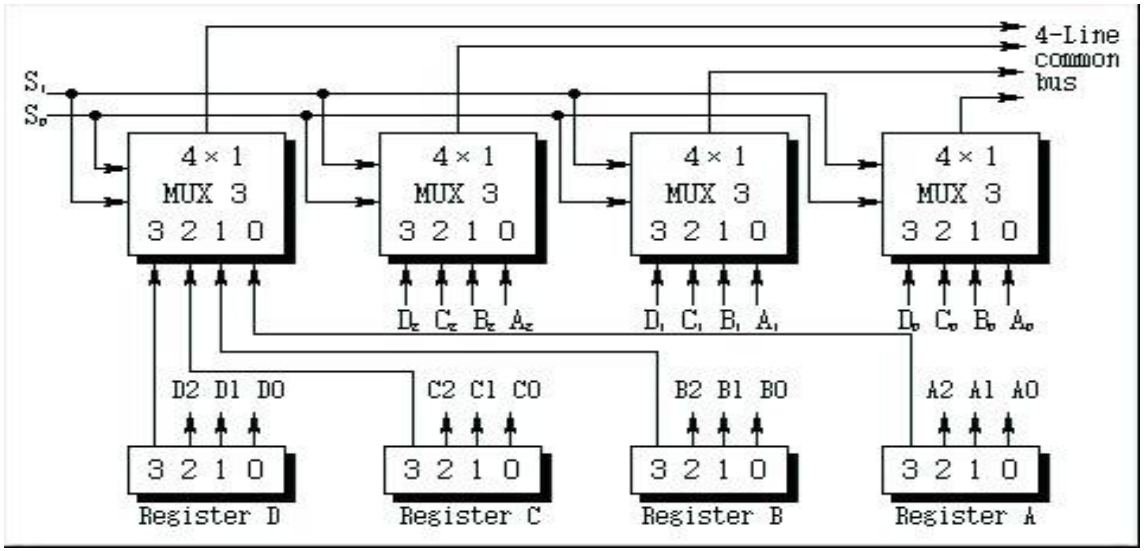
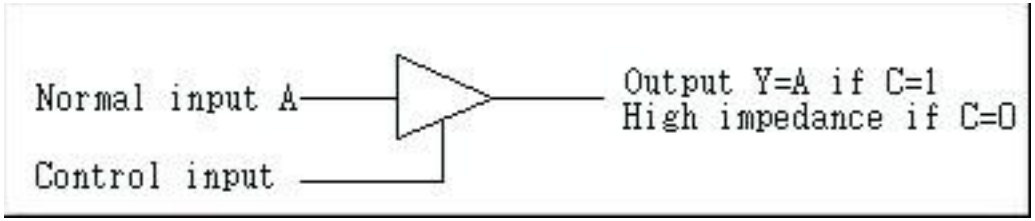


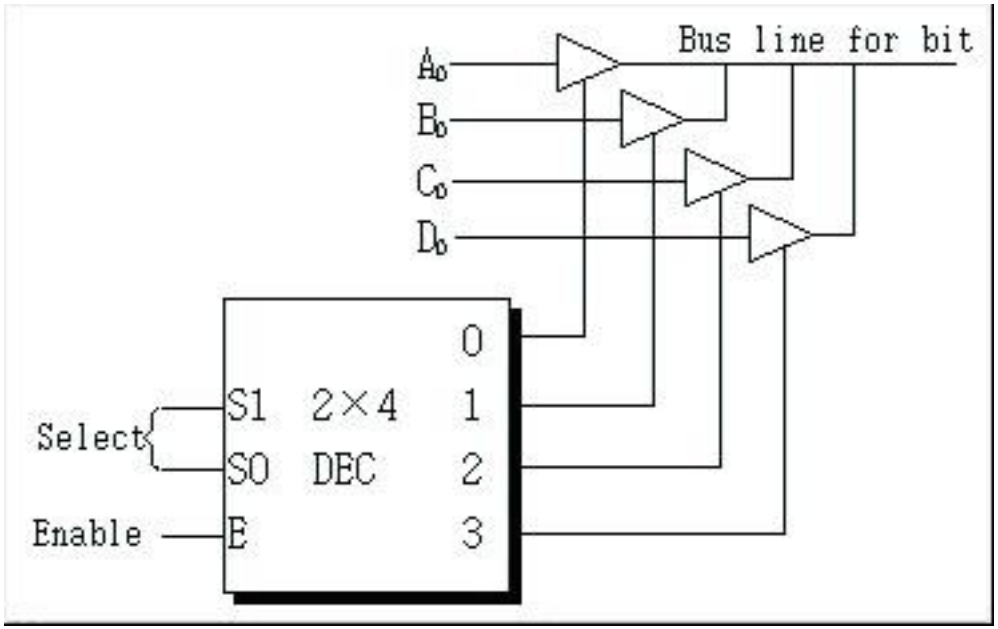

표4-2 <그림4-3>의 버스에 대한 함수표

| S1 S0 | Register Select |
|---|---|
| 0 0 | A |
| 0 1 | B |
| 1 0 | C |
| 1 1 | D |

# 3-상태 버스 버퍼

- 3-상태(Tri-State) 버퍼
  - 0의 상태, 1의상태, 고저항 상태(high-impedance state)
    - 고저항 상태: 개회로와 같은 상태로서 출력이 차단되는 상태




**S1,S0**의 디코더 입력에 의해 디코더 출력의 **0-3**중 하나가 활성화되어 **3**-상태 버퍼를 동작시킨다.

# 메모리 전송

- 메모리로부터 외부로의 정보전송
  - read 동작:
    - MAR(memory address register)에 주소를 넣고 읽은 데이타를 DR(data reigister)에 넣는다.
  - read 동작의 기호 표시
    - Read : DR ← M[AR]
- 메모리로 새로운 정보의 저장
  - write 동작:
    - R1에 있는 데이타를 AR에서 지정하는 메모리로 전송
  - write 동작의 기호 표시
    - Write : M[AR] ← R1

# 기본 컴퓨터 구조와 설계

하드와이어드 제어장치와 기본 컴퓨터 설계 이해를 위해 M. Morris Mano의 Computer System Architecture 교재 5장~7장의 내용으로 학습합니다.

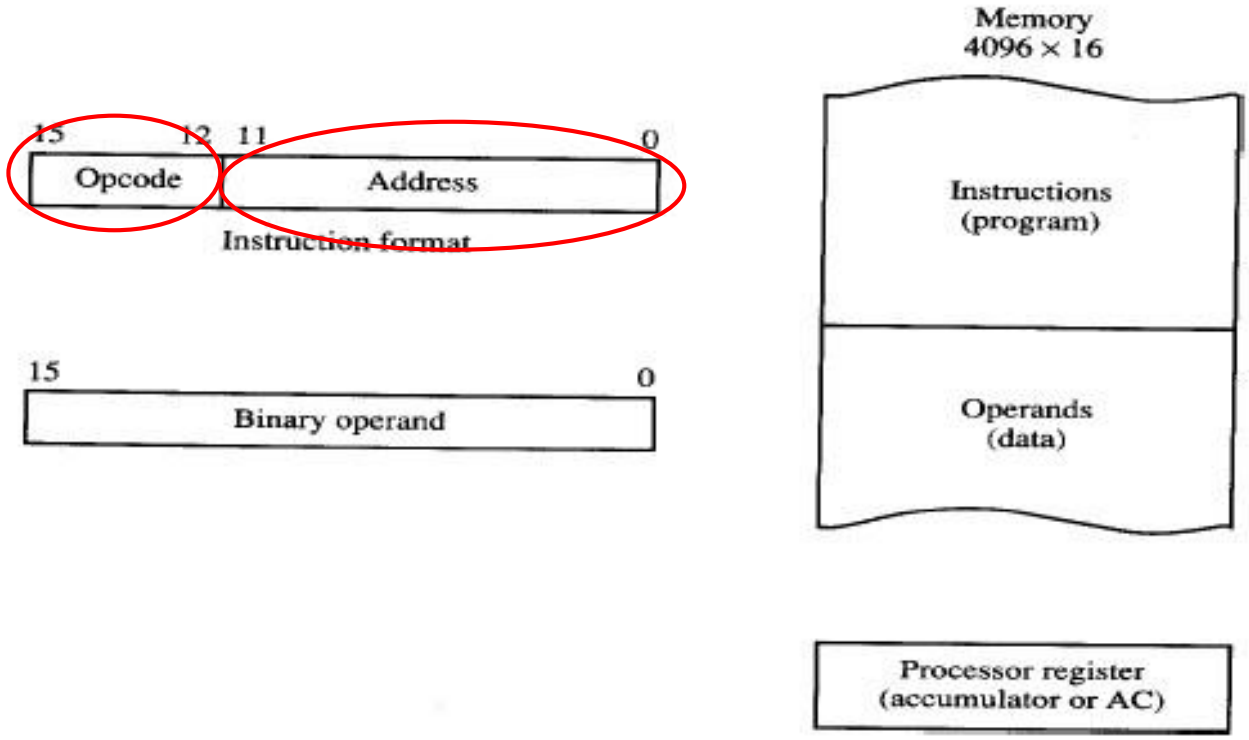
# Ch5. 기본 컴퓨터의 구조와 설계

- 5.1 명령어 코드
  - 컴퓨터 구조 : 내부 레지스터, 타이밍과 제어구조, 명령어 집합에 의해 정의됨
- 디지털 시스템의 내부조직
  - 마이크로 연산의 시이퀀스(sequence)에 의해 정의됨
- 컴퓨터 명령어
  - 컴퓨터에 대한 일련의 마이크로 연산을 기술한 이진 코드
- operation(연산) : macro operation
- microoperation

# 저장 프로그램 구조

- Instruction format = opcode + address
- 4096 = $2^{12}$ (4096워드를 위해 12 비트의 주소가 필요)

**Figure 5-1** Stored program organization.



- Immediate : 명령어 코드의 피연산자 주소 부분이 직접 피연산자의 내용을 닮고있음.
- Direct address : 피연산자의 내용이 담겨있는 메모리 주소
- Indirect address : 주소의 내용을 담고있는 메모리 워드의 주소
- 직접주소와 간접주소의 구분 : 1 비트를 사용

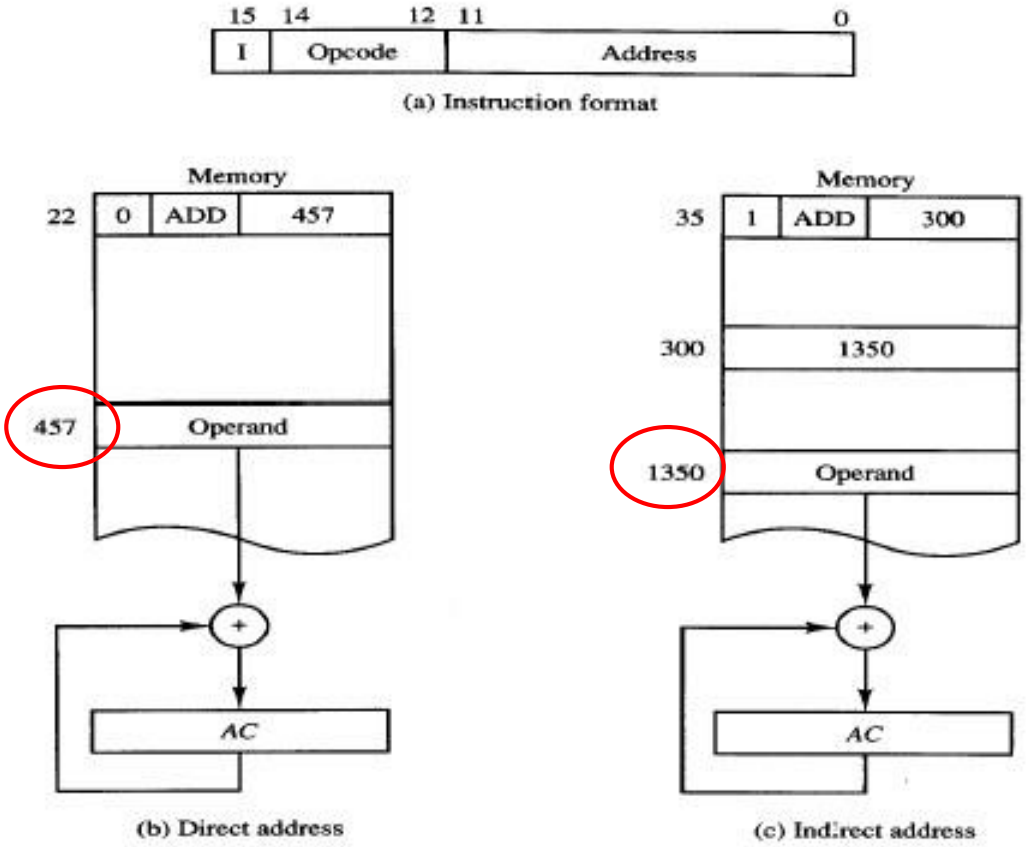



**Figure 5-2** Demonstration of direct and indirect address.

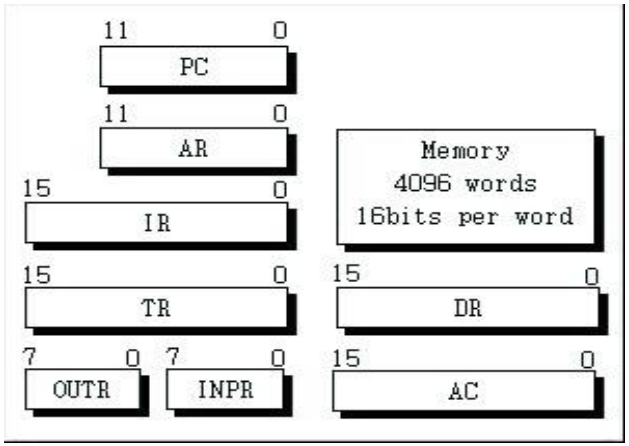
**Effective address :** 실제 사용하고자 하는 데이터가 있는 곳의 주소(유효주소)

# 5.2 컴퓨터 레지스터

- 기본 컴퓨터 레지스터와 메모리

표5-1 기본 컴퓨터를 위한 레지스터

| Register symbol | Number of bits | Register name | Function |
|---|---|---|---|
| DR | 16 | Data Register | Holds memory operand |
| AR | 12 | Address Register | Holds address for memory |
| AC | 16 | Accumulator | Processor Register |
| IR | 16 | Instruction Register | Holds Instruction Code |
| PC | 12 | Program Counter | Holds address of instruction |
| TR | 16 | Temporary Register | Holds temporary data |
| INPR | 8 | Input Register | Holds input character |
| OUTR | 8 | Output Register | Holds output character |



수고하셨습니다!