10 강

컴퓨터과학 개론

컴퓨터 구조 (2)

컴퓨터과학과 이관용 교수

학습목차

1 명령어

2 중앙처리장치

3 입출력장치 및 병렬처리

01

명령어

명령어 집합 구조

내장 프로그래밍 개념으로부터 직접적으로 도출된 개념

- ISA Instruction Set Architecture → HW와 SW의 교량 역할을 하는 개념
- 명령어 집합
  - 컴퓨터 시스템 내에 정의되어 있는 기본적인 명령어들의 집합
  - 모든 컴퓨터는 자신만의 명령어 집합을 가짐
    → 명령어 종류, 명령어 형식, 주소지정방식 등을 고려해서 결정됨
- 명령어 집합이 결정되면 그에 상응하는 하드웨어 구조가 결정됨

명령어 집합에 따른 컴퓨터 구조

CISC 복합 명령어 집합 컴퓨터, Complex Instruction Set Computer

- 복합 명령어를 포함하여 명령어와 주소지정방식의 수를 많이 사용
  - 수많은 복잡한 명령어를 탑재/사용함으로써 프로그램에서 사용되는 전체 명령어의 개수를 줄여서 프로그램의 실행 시간 단축을 위한 구조
  - 연산코드 해석 및 실행을 위한 제어장치가 복잡해지는 단점을 가짐

RISC 단축 명령어 집합 컴퓨터, Reduced Instruction Set Computer

- 명령어를 단순화하고 개수를 줄이고 하드웨어를 간단히 개선시킨 구조
  - 각 명령어의 길이를 가능한 짧게 함으로써 각 명령어의 실행 시간을 최소화

    → 많은 처리량과 빠른 속도를 지향
  - 제어장치는 비교적 간단하며, 일반적으로 하드웨어로 구성

기본적인 명령어 종류

데이터 전송 명령어

- 데이터 이동 (레지스터↔레지스터, 주기억장치↔ 레지스터, 기억장치↔기억장치 등)

데이터 처리 명령어

- 산술 명령어, 논리연산 명령어, 비트 단위 명령어, 시프트 명령어 등

프로그램 제어 명령어

- 프로그램의 제어 흐름 관리 → 무조건적 분기, 조건적 분기

입출력 명령어

- 보조기억장치 및 입출력장치 등과의 정보 교환 명령어, 인터럽트 관련 명령어
  - 인터럽트 → 프로그램의 정상 수행을 멈추고,
    CPU 이외의 다른 장치의 요구 사항을 수행하는 기능

명령어 형식

기본 형식

- 각 명령어는 실행에 필요한 모든 정보를 포함해야 함

연산자 코드 OP code	오퍼랜드 operand

- 연산자 코드 → CPU가 처리할 연산의 종류
  - 할당된 비트 수 → CPU가 수행할 수 있는 최대 명령어 개수
- 오퍼랜드("피연산자") → 명령어가 사용할 데이터
  또는 데이터가 저장되어 참조될 기억장치의 주소
  - 오퍼랜드의 크기/개수 → 명령어 집합, 명령어 등 컴퓨터 구조에 따라 달라짐

명령어 형식

오퍼랜드 개수에 따른 구분

- 연산의 대상이 되는 데이터가 어디서 추출되고,
  결과가 어디에 저장할 지에 따른 구분

3-주소 명령어 | 연산자 코드 | 오퍼랜드 | 오퍼랜드 | 오퍼랜드 |

2-주소 명령어 | 연산자 코드 | 오퍼랜드 | 오퍼랜드 | → 가장 많이 사용되는 형식

1-주소 명령어 | 연산자 코드 | 오퍼랜드 | → 누산기(AC, accumulator) 사용

0-주소 명령어 | 연산자 코드 | → 스택 사용

명령어 형식

- $W = X * (Y + Z)$

3-주소 명령어

```
ADD     Y, Z, R1     ; R1←Y+Z
MUL     X, R1, W     ; W←X*R1
```

2-주소 명령어

```
MOVE    Y, R1        ; R1←Y
ADD     Z, R1        ; R1←Z+R1
MUL     X, R1        ; R1←X*R1
MOVE    R1, W        ; W←R1
```

1-주소 명령어

```
LOAD    Y            ; AC←Y
ADD     Z            ; AC←Z+AC
MUL     X            ; AC←X*AC
STORE   W            ; W←AC
```

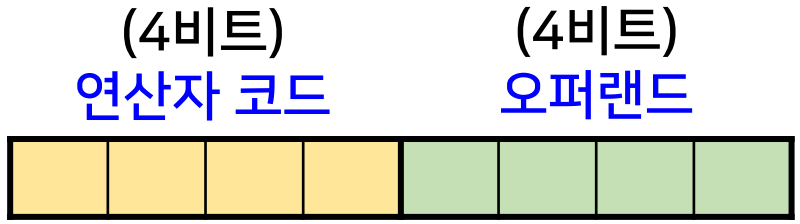
0-주소 명령어

```
PUSH  Y    ; PUSH(Y)
PUSH  Z    ; PUSH(Z)
ADD        ; PUSH(POP()+POP())
PUSH  X    ; PUSH(X)
MUL        ; PUSH(POP()*POP())
POP   W    ; W←POP()
```

명령어 형식

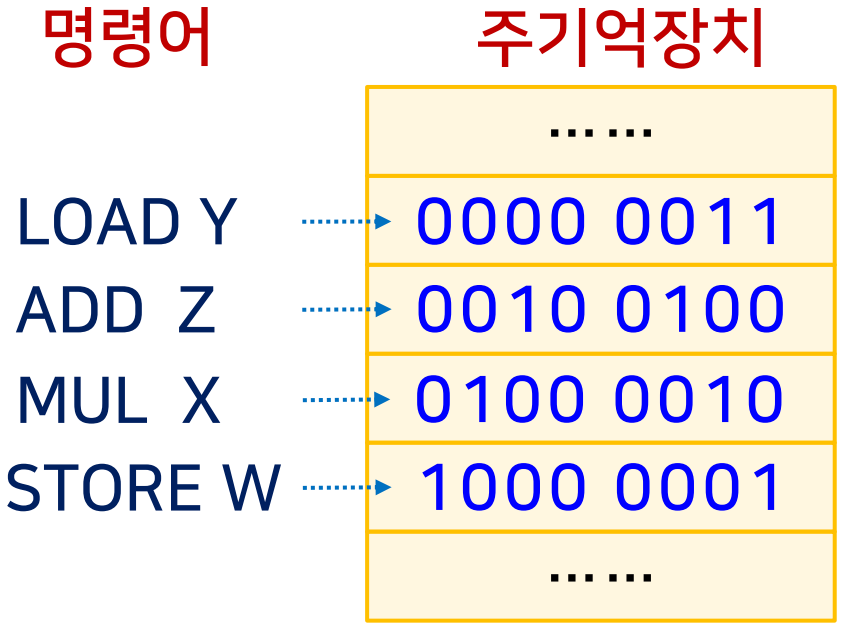
명령어의 메모리 표현

1-주소 명령어



LOAD → 0000
ADD → 0010
MUL → 0100
STORE → 1000

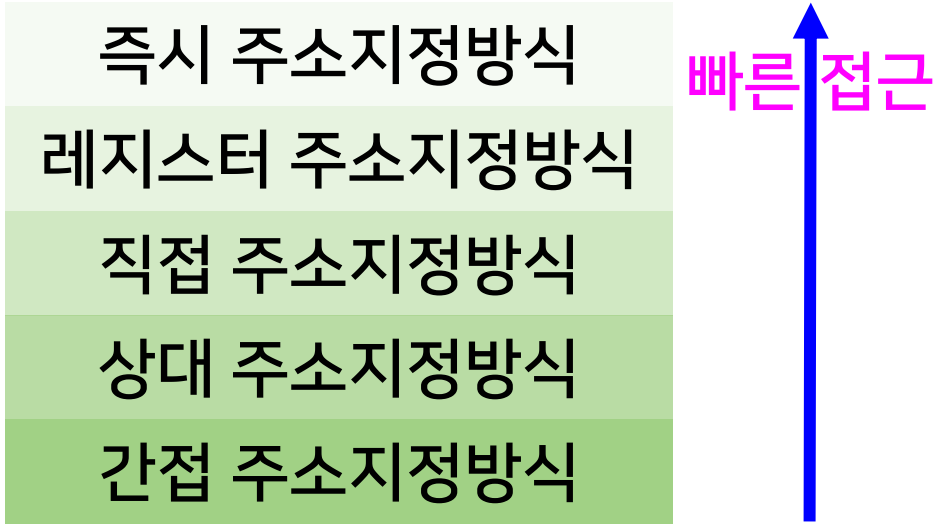
W → 0001
X → 0010
Y → 0011
Z → 0100



주소지정방식 addressing mode

연산에 사용될 데이터가 기억장치의 어디에 위치하는 지를 지정하는 방법

- 명령어 개수/길이를 줄이고, 기억장치 사용에 대한 융통성 증가시킴
- 유효주소 effective address
  - 주소지정방식에 의해 계산되어 실제 데이터가 저장된 주소



주소지정방식

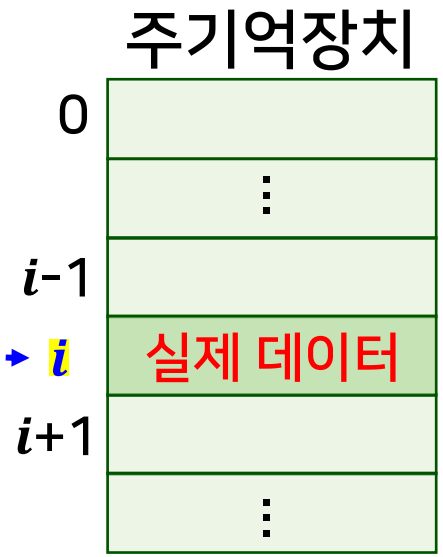
즉시 immediate 주소지정방식

연산자 코드	실제 데이터

→ 레지스터/변수의 초기화에 유용

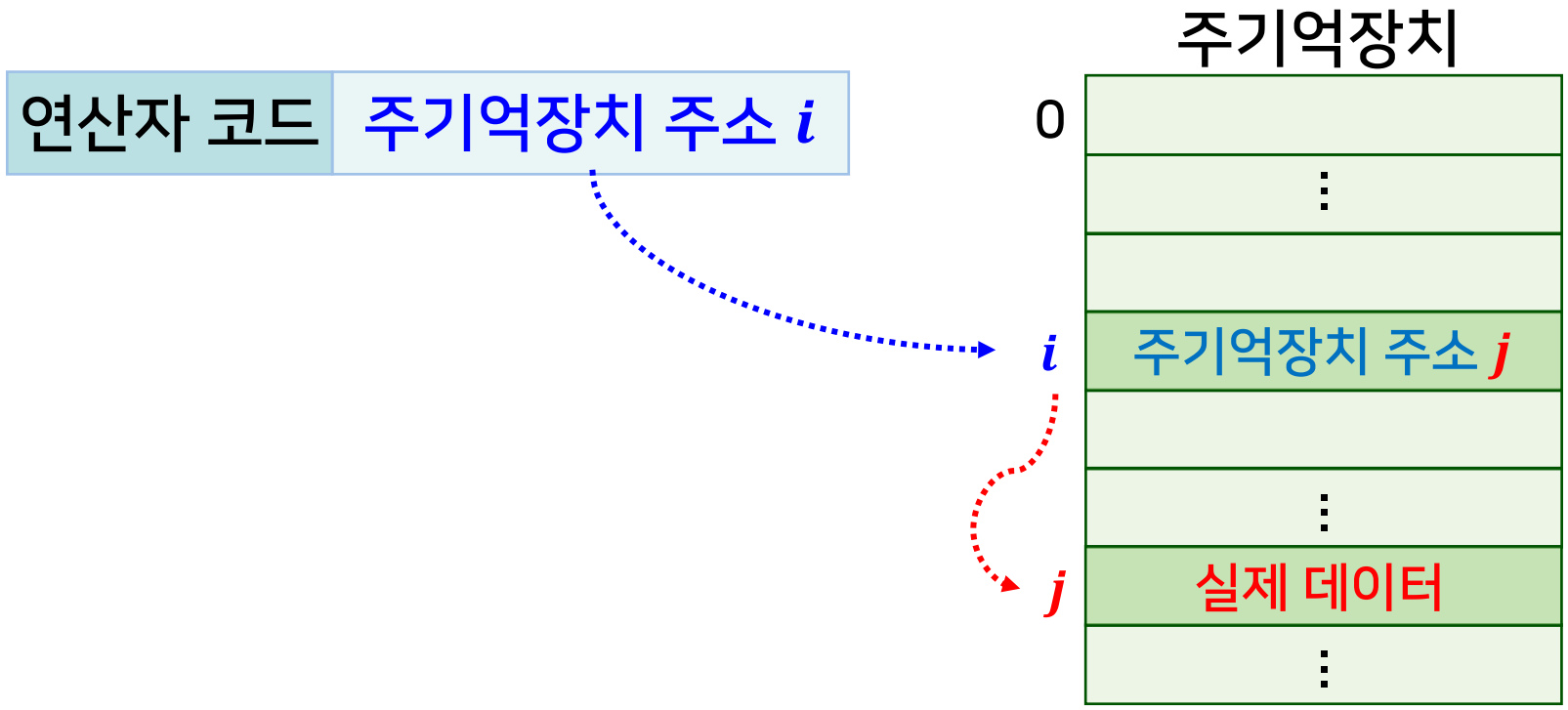
직접 direct 주소지정방식

연산자 코드	주기억장치 주소 i



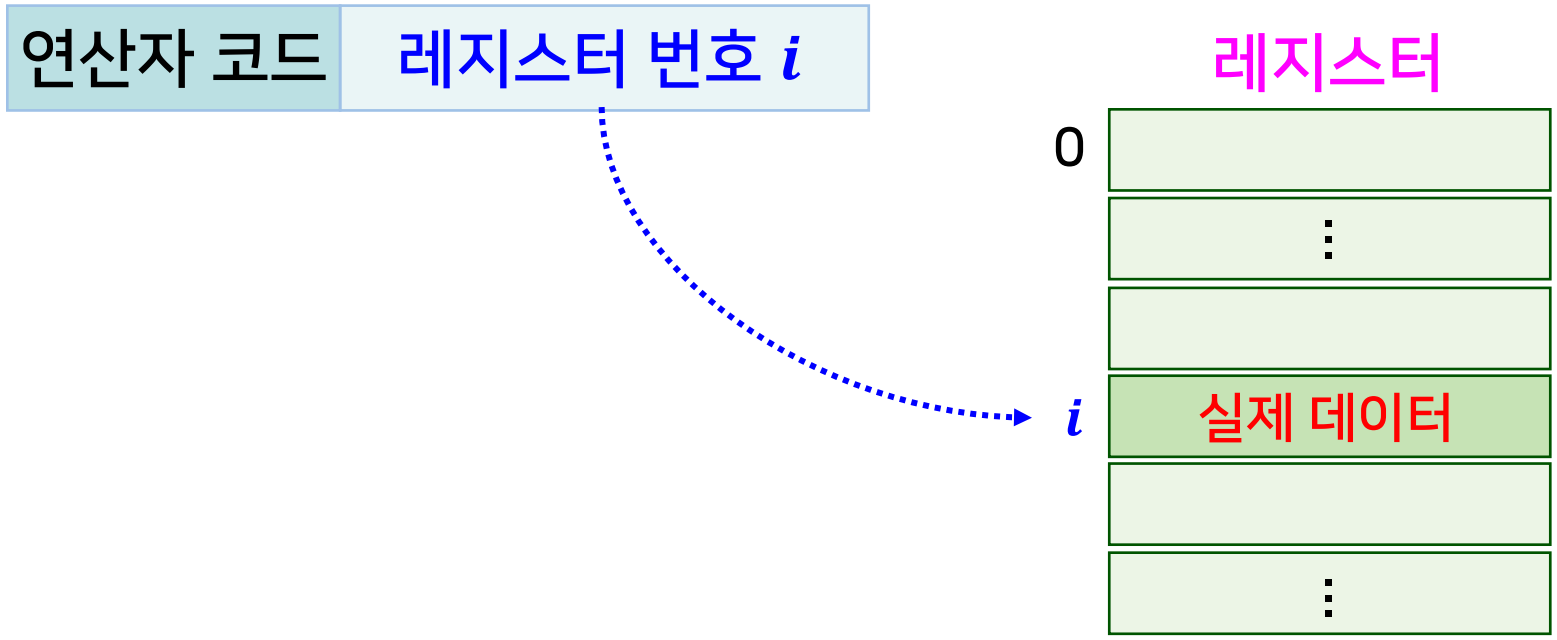
주소지정방식

간접 indirect 주소지정방식



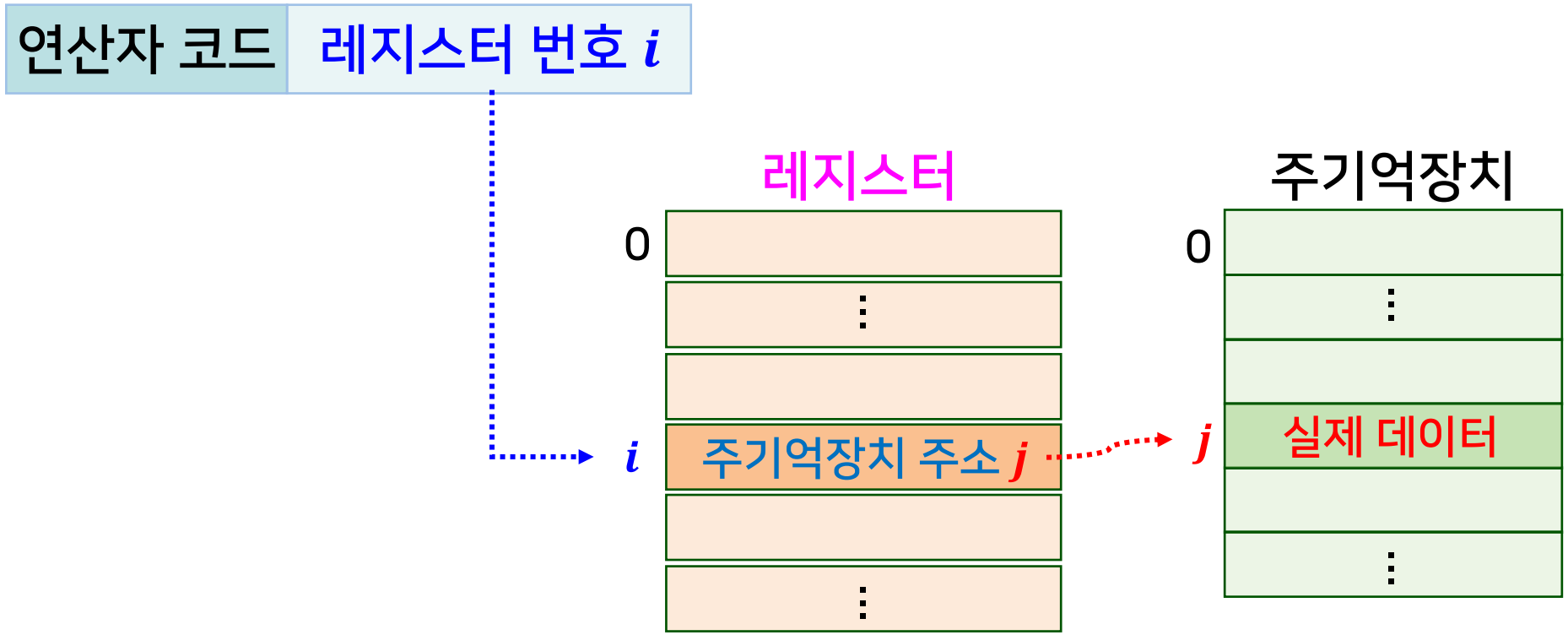
주소지정방식

레지스터 register 주소지정방식



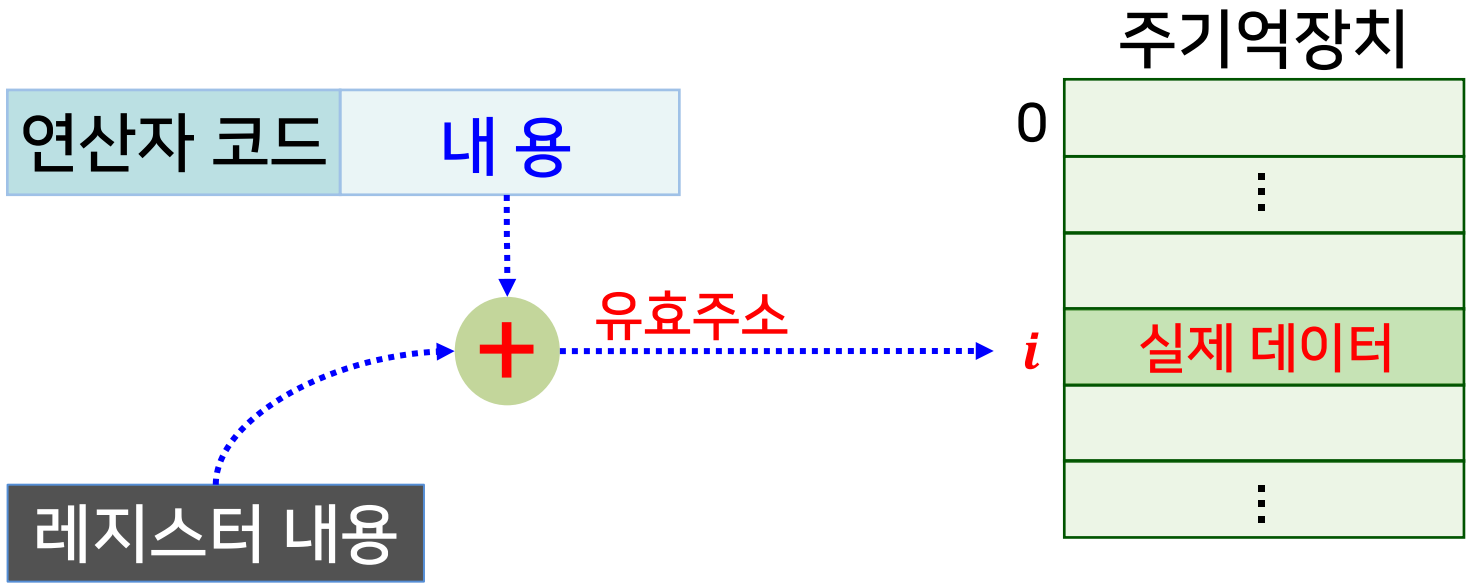
주소지정방식

레지스터 간접 register-indirect 주소지정방식



주소지정방식

상대 relative 주소지정방식



프로그램 카운터 ⇒ 분기형 명령어에서 주로 사용

인덱스 레지스터 → "인덱스된 주소지정방식" ⇒ 배열 인덱싱에 주로 사용

베이스 레지스터 → "베이스 레지스터 주소지정방식"

02

중앙처리장치

명령어를 하드웨어에 구현하는 방법

1. 마이크로 프로그램에 의한 제어장치

- Micro-programmed control device
- 산술/논리연산과 명령어 수행 순서 조작 회로가
  제어기억장치에 저장된 비트 패턴("마이크로 연산")으로 가동하는 장치
- 각 명령어는 여러 개의 마이크로 연산으로 구현
- 명령어 집합의 변경이나 명령어 추가 등이 용이
- CISC 컴퓨터 구조에서 주로 사용

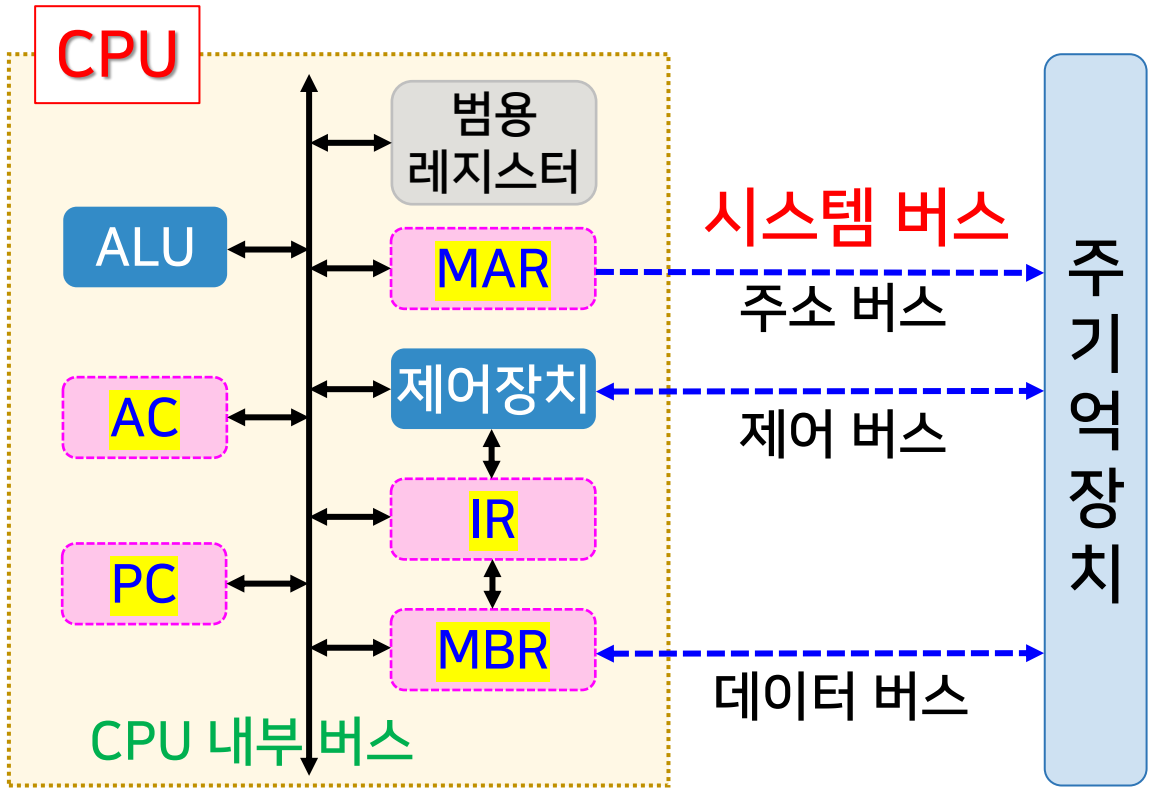
명령어를 하드웨어에 구현하는 방법

2. 직접 회로로 구성된 제어장치

- Hard-wired control device
- 연산과 명령어 수행 회로가 기억장치에 의존하지 않고 직접 구성된 제어 회로에 의해 기동하는 장치
- 빠른 수행, 명령어 집합의 변경에 쉽게 적응할 수 없음
- 명령어 수가 적은 RISC 컴퓨터 구조에서 주로 사용

레지스터

범용 레지스터, 특수 레지스터



레지스터

누산기 AC, Accumulator

- 데이터나 연산 결과를 일시적으로 저장하는 레지스터

기억장치 버퍼 레지스터 MBR, Memory Buffer Register

- 기억장치에 저장될 또는 기억장치에서 읽어온 데이터를 임시로 저장

기억장치 주소 레지스터 MAR, Memory Address Register

- 현재의 PC 내용을 시스템 버스의 주소 버스로 출력하기 전에 일시적으로 저장
  - PC 내용 → 다음에 수행될 명령어가 저장되어 있는 기억장소의 주소

처리장치

- **'연산장치 + 레지스터'를 묶어서 일컫는 표현으로 사용**
  - 모든 기능은 비트 패턴으로 구성된 마이크로 연산으로 구현

- **마이크로 연산의 부류**

R0 ← R1

레지스터 전송 마이크로 연산

R0 ← R1 + R2

R2 ← R1 - R2

R1 ← R1 - 1

산술 마이크로 연산

R1 ← R2′

R0 ← R1 ∧ R2

R0 ← R1 ∨ R2

논리 마이크로 연산

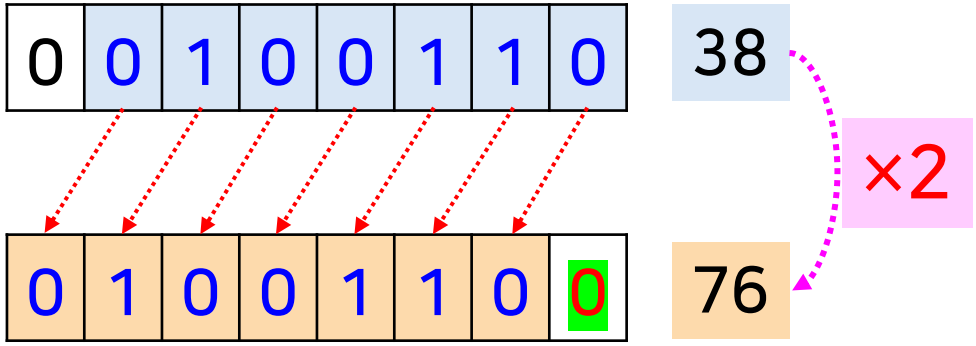
R1 ← shl R2

R1 ← shr R2

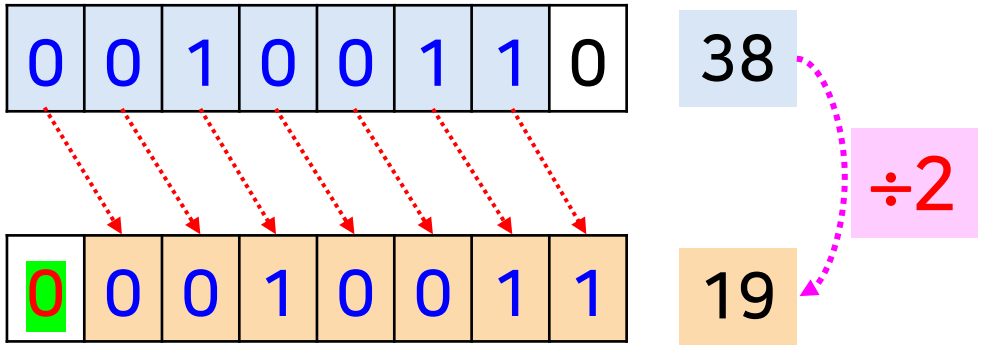
시프트 마이크로 연산

시프트 연산

왼쪽 시프트



오른쪽 시프트

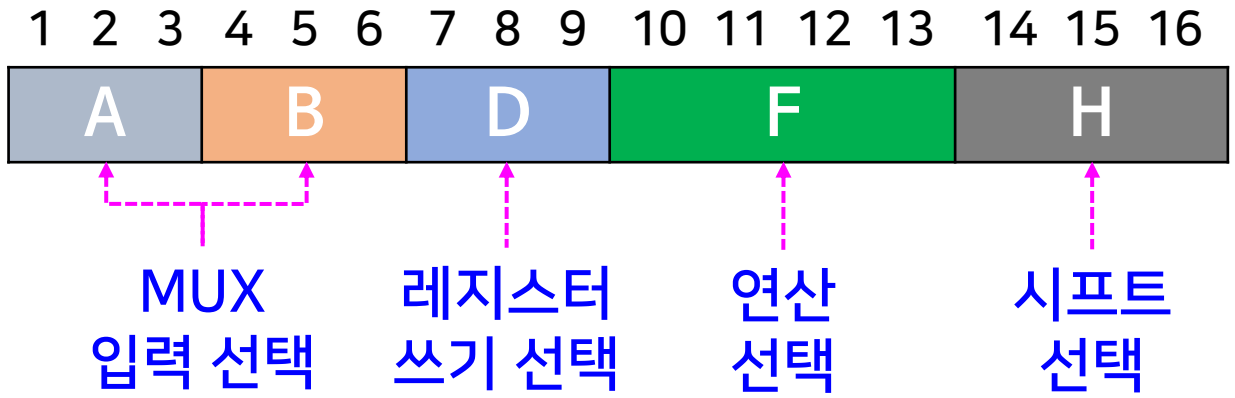


제어단어

마이크로 연산을 처리장치에 직접 다루기 위해서 필요

- 각 비트들이 처리장치의 논리회로 내의
  각종 MUX와 디코더의 선택 제어선으로 연결되어
  하드웨어를 회로 수준에서 직접적으로 통제하기 위한 것
- 각 마이크로 연산은 제어단어와 일대일 매핑

[예] 16비트 제어단어



제어단어

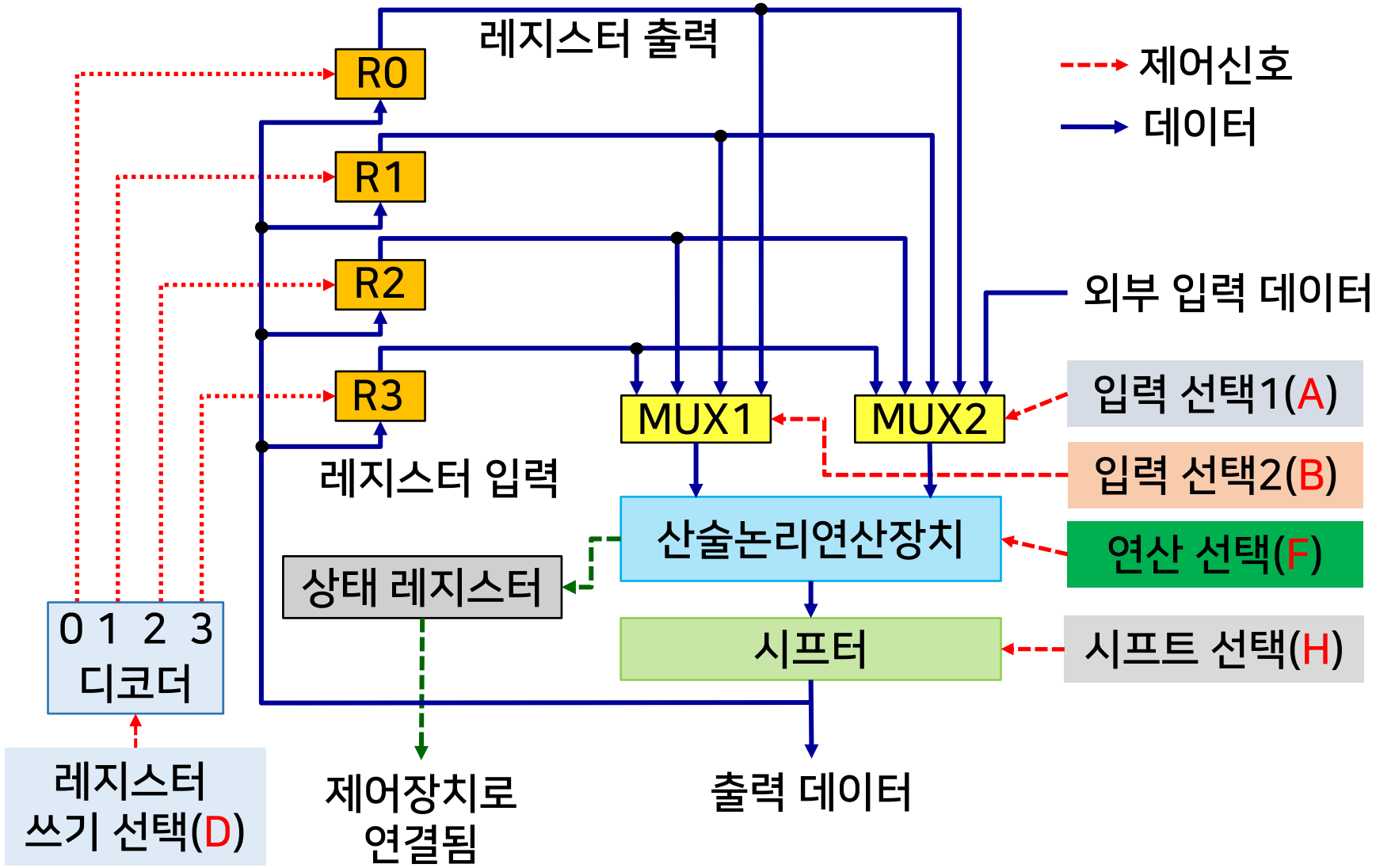
A B	000: 입력 없음 001: R0 010: R1 011: R2 100: R3 101: 외부 입력
D	000: R0 001: R1 010: R2 011: R3
F	0000: 연산 안 함. MUX2 값을 출력 0001: + 0010: - 0011: OR 0100: AND 0101: NOT 0110: XOR
H	000: 시프트 안 함. ALU 값을 그래도 출력 001: 왼쪽 시프트 010: 오른쪽 시프트

A → 010 B → 011 D → 000 F → 0001 H → 000

R0 ← R1 + R2 ⟷ 0100110000001000

마이크로 연산 제어단어

처리장치의 구성

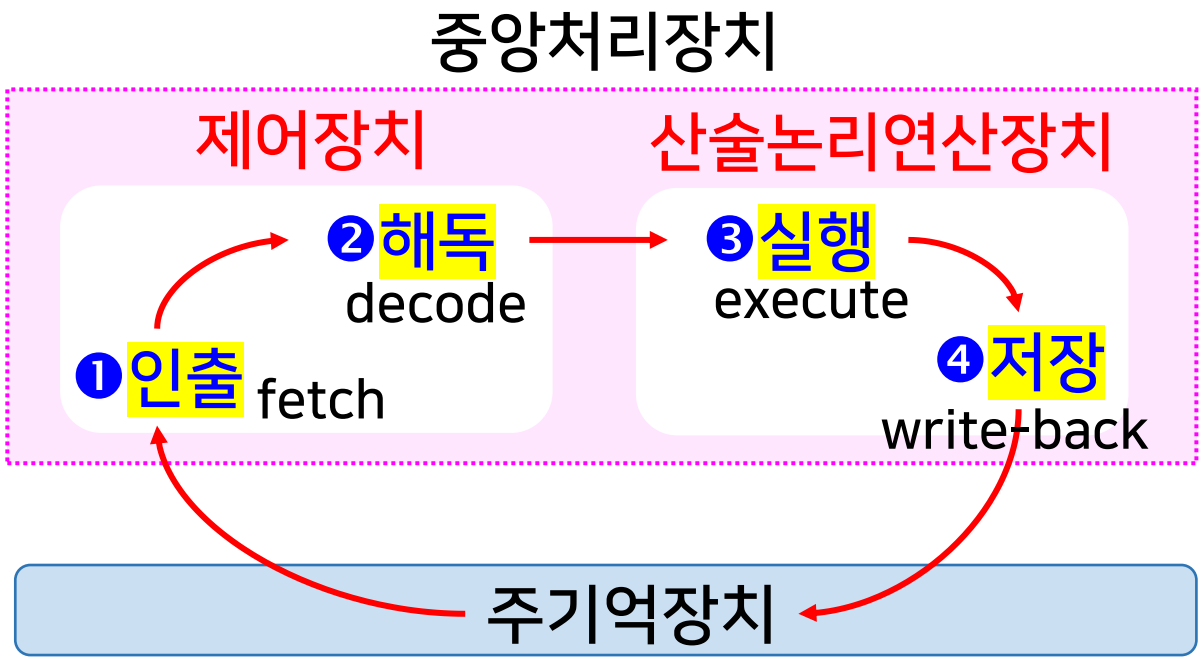

레지스터 출력
R0
R1
R2
R3
제어신호
데이터
외부 입력 데이터
MUX1
MUX2
입력 선택1(A)
입력 선택2(B)
레지스터 입력
산술논리연산장치
연산 선택(F)
상태 레지스터
0 1 2 3
디코더
시프터
시프트 선택(H)
레지스터
쓰기 선택(D)
제어장치로
연결됨
출력 데이터

제어장치

- **메모리에 저장된 명령을 어떻게 순차적으로 가져와서 수행할 것인가를 통제하는 것**
  - 두 가지의 기본적인 기능
    - 처리장치를 구동해서 특정 연산을 수행한 후 처리장치 내의 레지스터 값을 갱신하고 연산 결과를 출력
    - 현재 주어진 명령을 수행한 후 다음에 수행할 명령의 주소 정보를 생성

제어장치

명령어 사이클 instruction cycle



제어장치의 구성 요소

- **프로그램 카운터 PC, Program Counter**
  - 다음에 수행될 명령어가 저장되어 있는 주기억장치의 주소 저장
- **명령어 레지스터 IR, Instruction Register**
  - 주기억장치에서 인출되어 현재 실행 중인 명령어 저장
- **제어기억장치 control memory**
  - 마이크로 연산의 집합을 저장하고 있는 기억장치 → ROM으로 구현
- **명령어 해독기**
  - 주어진 명령어를 제어기억장치의 해당 마이크로 명령이 시작하는 주소로 매핑해 주는 것

제어장치의 구성 요소

주소 결정회로

- 명령어에 포함된 주소 정보, 제어단어와 연결된 주소 정보, 처리장치 구동 후 결과로 나오는 상태 비트 등으로부터 제어기억장치의 다음 수행할 마이크로 명령의 주소를 생성

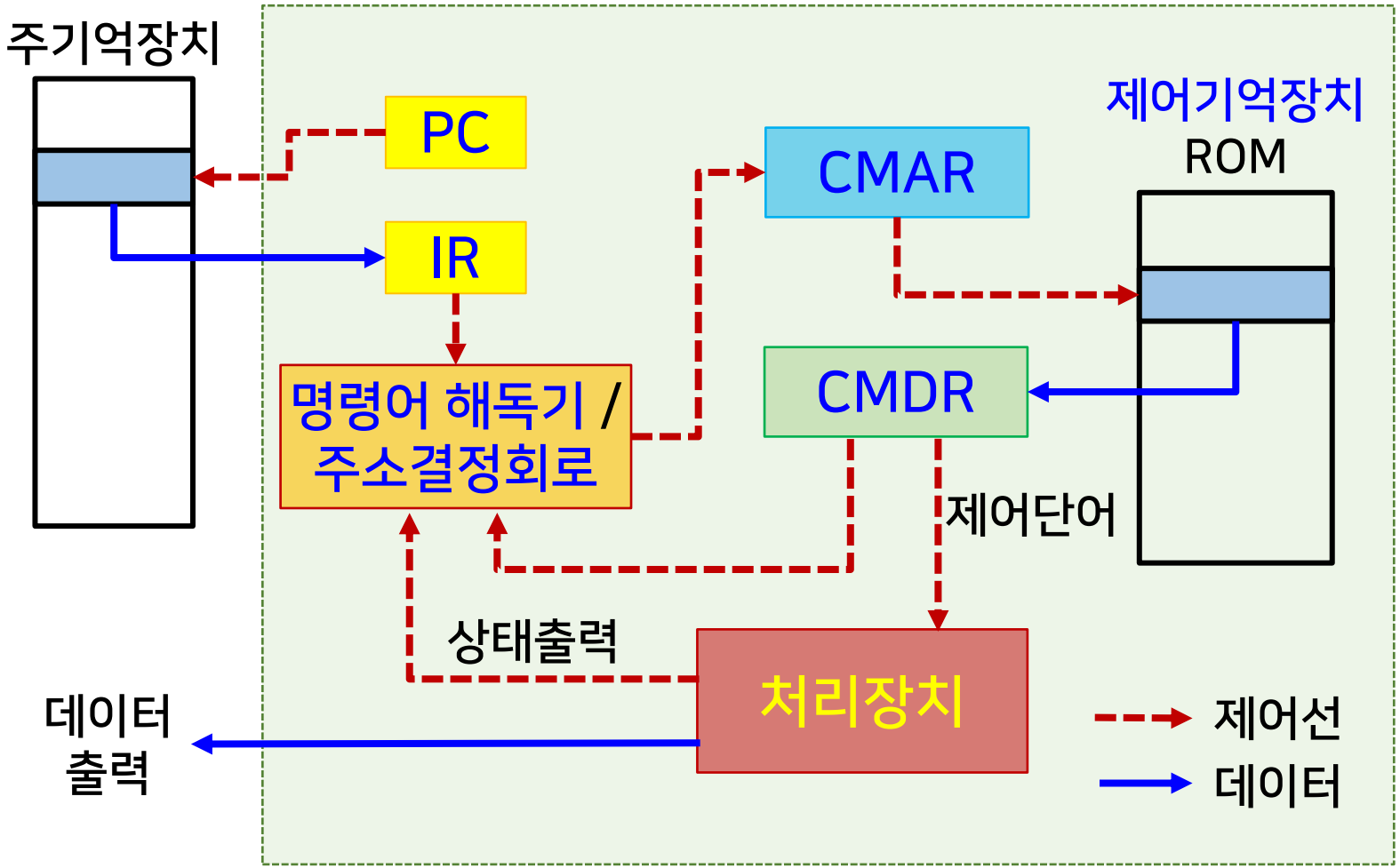
제어기억장치 주소 레지스터 CMAR

- 제어기억장치에서 다음에 수행할 마이크로 명령의 위치를 가리키는 주소 저장

제어기억장치 데이터 레지스터 CMDR

- 제어기억장치에서 가져온 다음 수행할 마이크로 연산 저장
- CMDR 없이 제어기억장치의 출력이 직접 다른 장치들로 연결 가능

제어장치의 구성

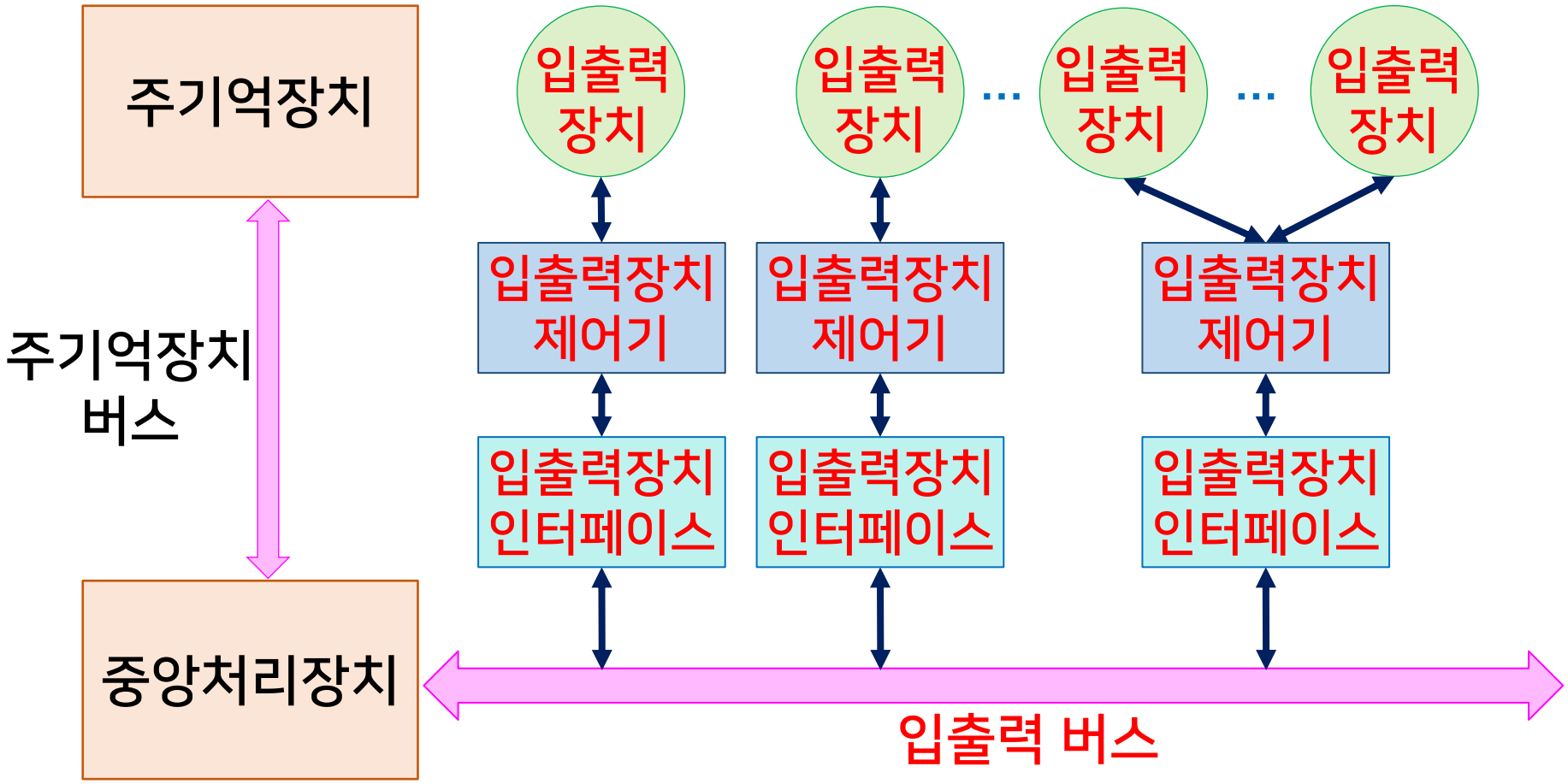

주기억장치
PC
IR
명령어 해독기 / 주소결정회로
CMAR
CMDR
제어기억장치
ROM
제어단어
상태출력
처리장치
데이터 출력
제어선
데이터
중앙처리장치

03

입출력장치 및 병렬처리

입출력 시스템

기본 입출력 시스템의 구성도



입출력 시스템의 기본 구성요소

입출력장치

- 사용자와 컴퓨터 시스템을 연결해 주는 장치
  - 키보드, 마우스, 모니터, 프린터, 디스크 등

입출력장치 제어기

- 상이한 기계적/전자적 특성을 가진 입출력장치를 물리적/전자적으로 제어해서 구동시키는 작업을 수행하는 기기
  - 모터 회전, 헤드 이동, 입출력 매체의 위치 정렬 등의 작업 수행
  - 제어기가 입출력장치에 포함된 경우 및 하나의 제어기로 여러 입출력장치를 제어하는 경우도 존재

입출력 시스템의 기본 구성요소

입출력장치 인터페이스

- 입출력장치와 중앙처리장치/주기억장치 사이의
  데이터 전송 속도, 데이터 처리 단위, 오류 확률의 차이를 상쇄해서
  올바른 전송을 위한 방법 제공

입출력 버스

- 입출력 전용으로 사용되는 정보 회선의 묶음
  - 입출력장치와 중앙처리장치 사이의 정보 교환에 사용
- 여러 장치에 의한 버스 사용 충돌을 막기 위한 중재기가 필요

입출력 제어 방식

CPU에 의한 제어

독립된 입출력 제어기 없이 입출력장치의 정보가 CPU를 통해 주기억장치에 쓰고 읽혀지는 방식

DMA 방식
Direct Memory Access

입출력장치가 주기억장치와 직접 연결. CPU는 두 장치 간의 초기 설정 및 허가에만 관여. 직접적인 정보의 이동은 장치간에 DMA 제어기가 해결

채널 방식

채널(입출력 전용의 별도 프로세서) 사용 → 정보 전송 통로 제공 및 CPU와 같은 산술/논리/분기 연산 작업도 수행 가능

CPU의 관여 시점?

프로그램에 의한 방식

CPU가 주기적으로 입출력장치에 신호를 보내 입출력 여부를 물어보는 방식

인터럽트에 의한 방식

입출력장치가 인터럽트를 통해 입출력 요청이 있을 때만 CPU가 하던 일을 중단하고 해당 장치와 연락하는 방식

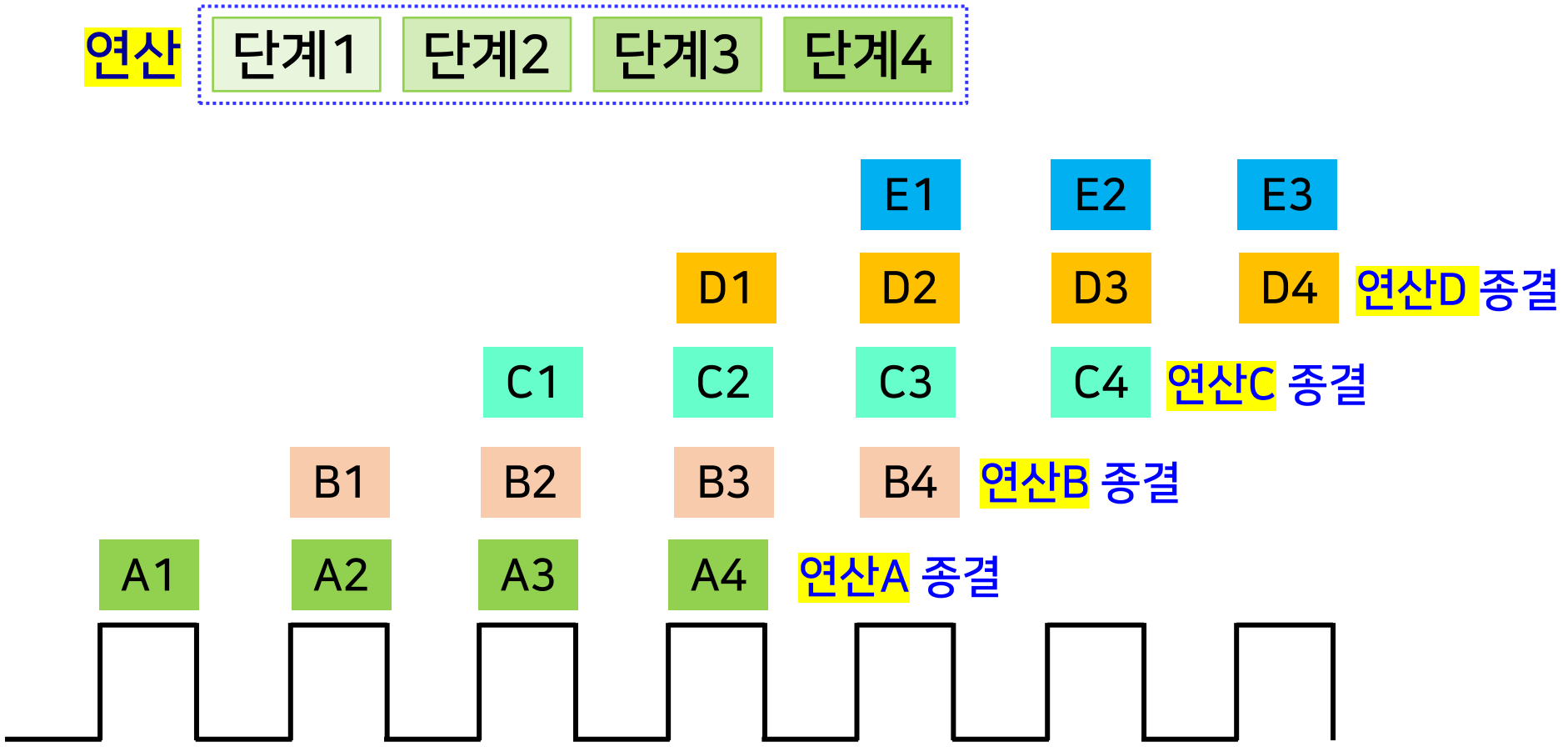
병렬처리

파이프라인 처리기 pipeline processor

- 프로그램 내에 내재하고 있는 시간적 병렬성을 활용하는 방법
- CPU 내의 하드웨어 요소의 일부를 파이프라인 형태로 구성하여
  프로그램 수행에 필요한 작업을 시간적으로 중첩하여 수행시키는 처리기
- 하나의 연산을 서로 다른 기능을 가진 여러 개의 단계(세그먼트)로 분할하여,
  각 단계가 동시에 서로 다른 데이터를 취급하도록 함
  - 현재 명령의 특정 단계가 끝나고 다음 단계로 넘어가면
    바로 다른 명령을 불러들여서 해당 단계를 동시에 처리하는 방식
  - 연달아 수행될 명령어들은 서로 간의 간섭이 없어야 함

병렬처리

파이프라인 처리기



병렬처리

멀티코어 구조

- 하나의 CPU에 2개 이상의 코어를 넣어서
  동시에 여러 개의 명령어를 처리할 수 있는 구조
  - 코어 → CPU의 일부분으로 명령을 가져와 수행하는 주회로
  - 각 코어는 수행 중인 응용 프로그램의 프로세스나 스레드를 하나씩 담당

GPGPU General Purpose computing on Graphics Processing Unit

- "그래픽스 처리장치를 사용한 범용 연산"
- 그래픽 카드의 고도의 병렬처리 능력을 연산에 사용하는 기술

정리하기

1. 명령어

- 명령어 집합 구조, CISC ↔ RISC, 명령어 종류와 형식, 주소지정방식

2. 중앙처리장치

- 명령어 구현 방법 → 마이크로 프로그램, 직접 회로
- 특수 레지스터 → AC, PC, IR, MAR, MBR
- 제어 단어, 처리장치 구성
- 제어장치 → 기능, 명령어 사이클 (인출 – 해독 – 실행 – 저장), 구성요소, 동작

3. 입출력장치 및 병렬처리

- 입출력 시스템의 구성요소 → 입출력장치, 입출력장치 제어기,
  입출력장치 인터페이스, 입출력버스
- 입출력 제어방식 → CPU (프로그램, 인터럽트), DMA, 채널
- 병렬처리 → 파이프라인 처리기, 멀티코어 구조, GPGPU

11 강

다음시간 안내

프로그래밍 언어 (1)

12강. 프로그래밍 언어 (2)

13 강 데이터베이스 (1)