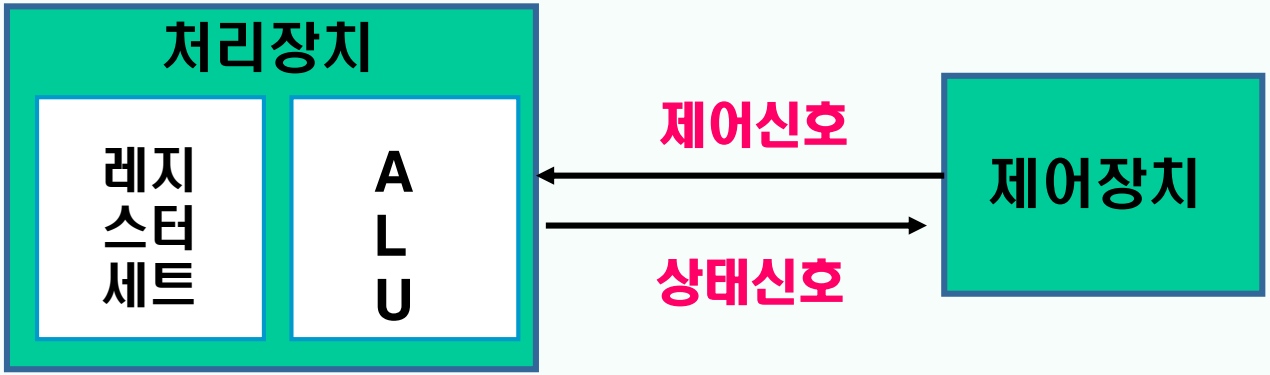
제4장 처리 장치

처리장치의 개요

- 중앙처리장치(CPU: Central Processing Unit)
  - 처리장치와 제어장치가 결합된 형태
- 처리장치(datapath) : 데이터를 처리하는 연산을 실행
- 제어장치(control unit) : 연산의 실행순서를 결정



처리장치의 구성

- 산술논리연산장치(ALU: Arithmetic and Logic Unit)와 레지스터들로 구성.
  - 산술논리연산장치: 산술, 논리, 비트연산 등의 연산을 수행
  - 레지스터: 연산에 사용되는 데이터나 연산의 결과를 저장
- 산술논리연산장치는 독립적으로 데이터를 처리하지 못하며, 반드시 레지스터들과 조합하여 데이터를 처리
  → 마이크로 연산

마이크로 연산

마이크로연산의 개요

- 레지스터에 저장되어 있는 데이터에 대해 이루어지는 기본적인 연산
  - 한 레지스터의 내용을 다른 레지스터로 옮기는 것
  - 두 레지스터의 내용을 합하는 것
  - 레지스터의 내용을 1만큼 증가시키는 것 등
- 처리장치의 동작원리를 이해하기 위해서는 마이크로연산을 이해해야 한다.

마이크로연산의 종류

- 레지스터 전송 마이크로연산
  (register transfer micro-operation)
- 산술 마이크로연산
  (arithmetic micro-operation)
- 논리 마이크로연산
  (logic micro-operation)
- 쉬프트 마이크로연산
  (shift micro-operation)

레지스터 표시

- 각각의 레지스터는 자신의 기능을 나타내는 대문자로 표시-니모닉 (박스로 표현)

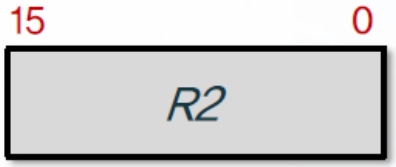
- AR(Address Register)
- PC(Program Counter)
- AC(ACcumulater)
- R1(1번째 Register)
- DR(Data Register)

레지스터 표시

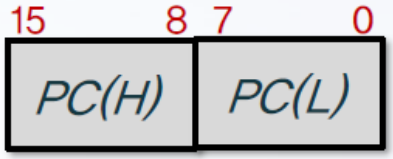
❖ 레지스터의 표현

〈레지스터 R〉

〈8비트 레지스터의 개별 비트〉



〈16비트 레지스터의 순서 표시〉



〈16비트 레지스터의 분할〉

레지스터 전송 마이크로연산 기본 기호

기 호	의 미	사 용 예
영문자(숫자와 함께)	레지스터를 표시	*AR, R2, DR, IR*
괄호	레지스터의 일부분	*R2(1), R2(7:0), AR(L)*
화살표	자료의 이동 표시	*R1 ← R2*
쉼표	동시에 실행되는 두개 이상의 마이크로연산을 구분	*R1 ← R2, R2 ← R1*
괄호	메모리에서의 어드레스	*DR ← M[AR]*

레지스터 전송 마이크로연산

- 한 레지스터에서 다른 레지스터로 2진 데이터를 전송하는 연산
- 레지스터 사이의 데이터 전송은 연산자 '←' 로 표시

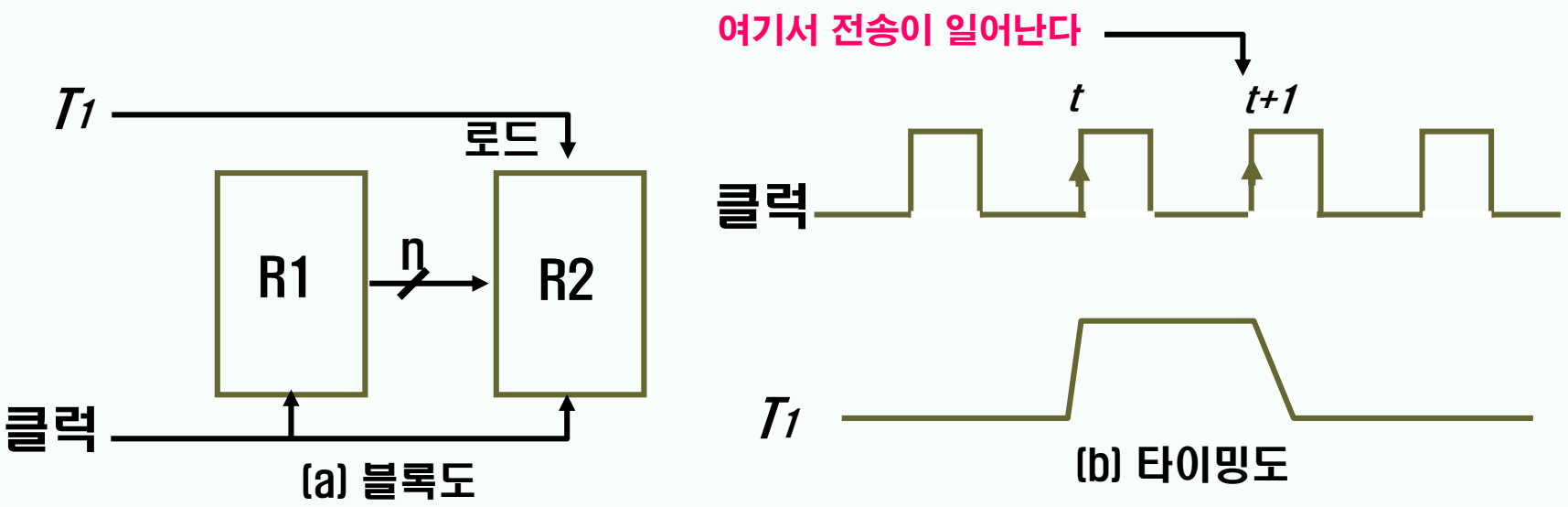
예) $R2 \leftarrow R1$

<의미> 레지스터 R1의 내용이 레지스터 R2로 전송

- 여기서 R1 : 출발 레지스터(source register)
  R2 : 도착 레지스터(destination register)

레지스터 전송 마이크로연산

- 하드웨어적인 측면에서의 레지스터 전송
- 레지스터 R1에서 R2로의 전송



〈T_1=1인 상태에서 *R1* 에서 *R2* 로의 데이터 전송〉

레지스터 전송 마이크로연산

- **레지스터 전송문**

– 앞의 그림을 조건문(conditional statement)으로 표현하면

$$If(T_1=1)\ \ then\ (R2 \leftarrow R1)$$

– 레지스터 전송문으로 표현하면

$$T_1 : R2 \leftarrow R1$$

산술 마이크로연산

- 레지스터 내의 데이터에 대해서 실행되는 산술연산
- 기본적인 산술연산으로는 덧셈, 뺄셈, 1 증가, 1 감소
  그리고 보수연산이 있다

기호 표시	의 미
$R0 \leftarrow R1 + R2$	R1과 R2의 합을 R0에 저장
$R2 \leftarrow \overline{R2}$	R2의 보수(1의 보수)를 R2에 저장
$R2 \leftarrow \overline{R2}+1$	R2에 2의 보수를 계산 후 저장
$R0 \leftarrow R1+\overline{R2}+1$	R1에 R2의 2의 보수를 더한 후 R0에 저장
$R1 \leftarrow R1+1$	R1에 1 더함 (상승 카운트)
$R1 \leftarrow R1-1$	R1에 1 뺌 (하강 카운트)

논리 마이크로연산

- 레지스터 내의 데이터에 대한 비트를 조작하는 연산.
- 기본적인 논리연산으로는 AND, OR, XOR, NOT 연산이 있다.
- 레지스터에 저장되어 있는 비트의 데이터를 조작하는데 유용하다.

기 호	의 미
$R0 \leftarrow \overline{R1}$	비트별 논리적 NOT(1의 보수)
$R0 \leftarrow R1 \wedge R2$	비트별 논리적 AND(비트 클리어)
$R0 \leftarrow R1 \vee R2$	비트별 논리적 OR(비트 세트)
$R0 \leftarrow R1 \oplus R2$	비트별 논리적 XOR(비트별 보수)

논리 마이크로연산

- $\vee$ 기호는 항상 OR임
- + 기호는 가산 및 OR 성격을 가짐

- $T_1 + T_2 : R0 \leftarrow R1 + \overline{R2} + 1,\ R3 \leftarrow R5 \vee \overline{R6}$

  $T_1 + T_2$ ↓ OR

  $\overline{R2}$ ↓ 덧셈

  $\vee$ ↓ OR

쉬프트 마이크로연산

- 레지스터 내의 데이터를 쉬프트시키는 연산
- 데이터의 측면이동에 사용

유형	기호표시	8비트 데이터의 경우	
		출발지 R2	쉬프트 후: 목적지 R1
왼쪽 쉬프트	R1 ← *shl* R2	10011110	00111100
오른쪽 쉬프트	R1 ← *shr* R2	11100101	01110010

- 쉬프트 연산을 수행하더라도 *R2* 의 값은 변하지 않는다.
- *shr* 이나 *shl* 에 대해서 입력 비트는 0으로 가정한다.
- 출력비트의 값은 버려진다.

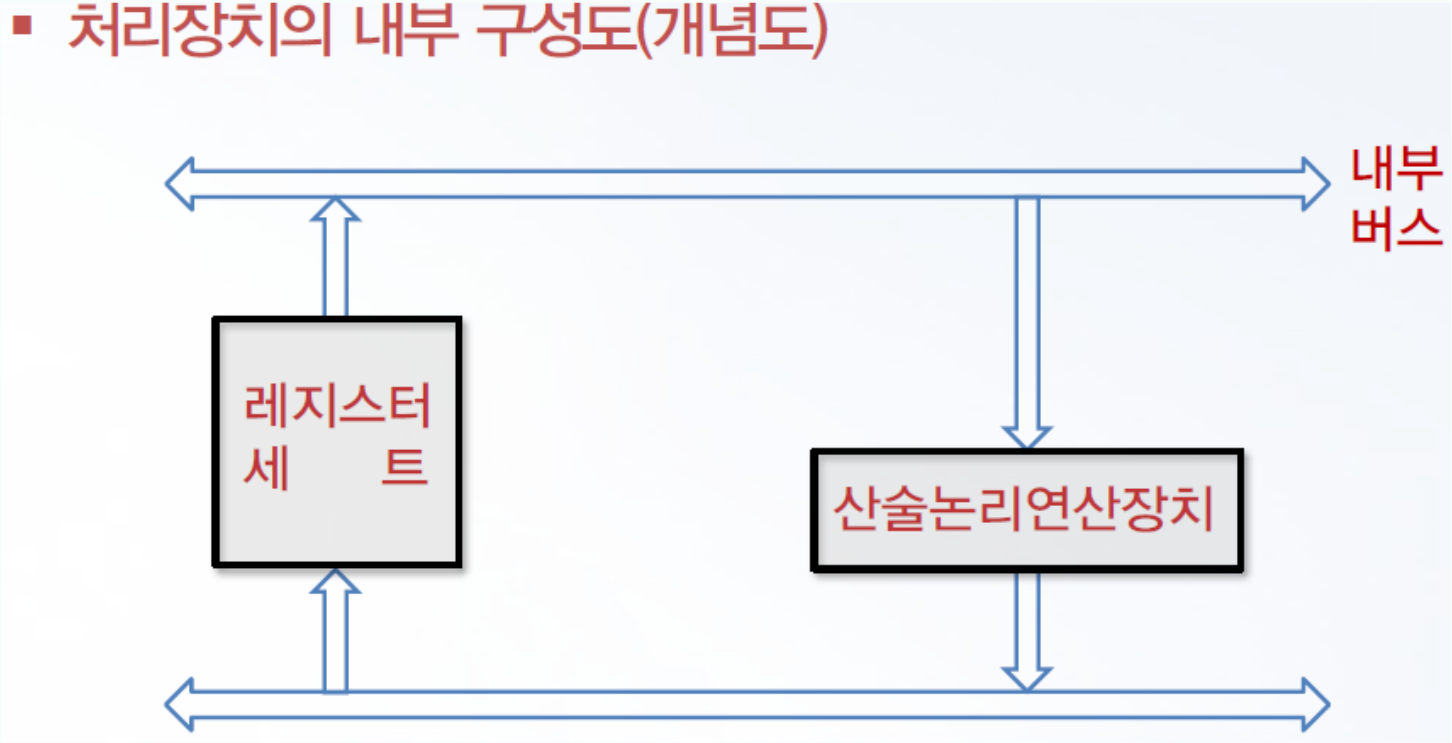
처리장치의 구성요소

처리장치의 구성요소

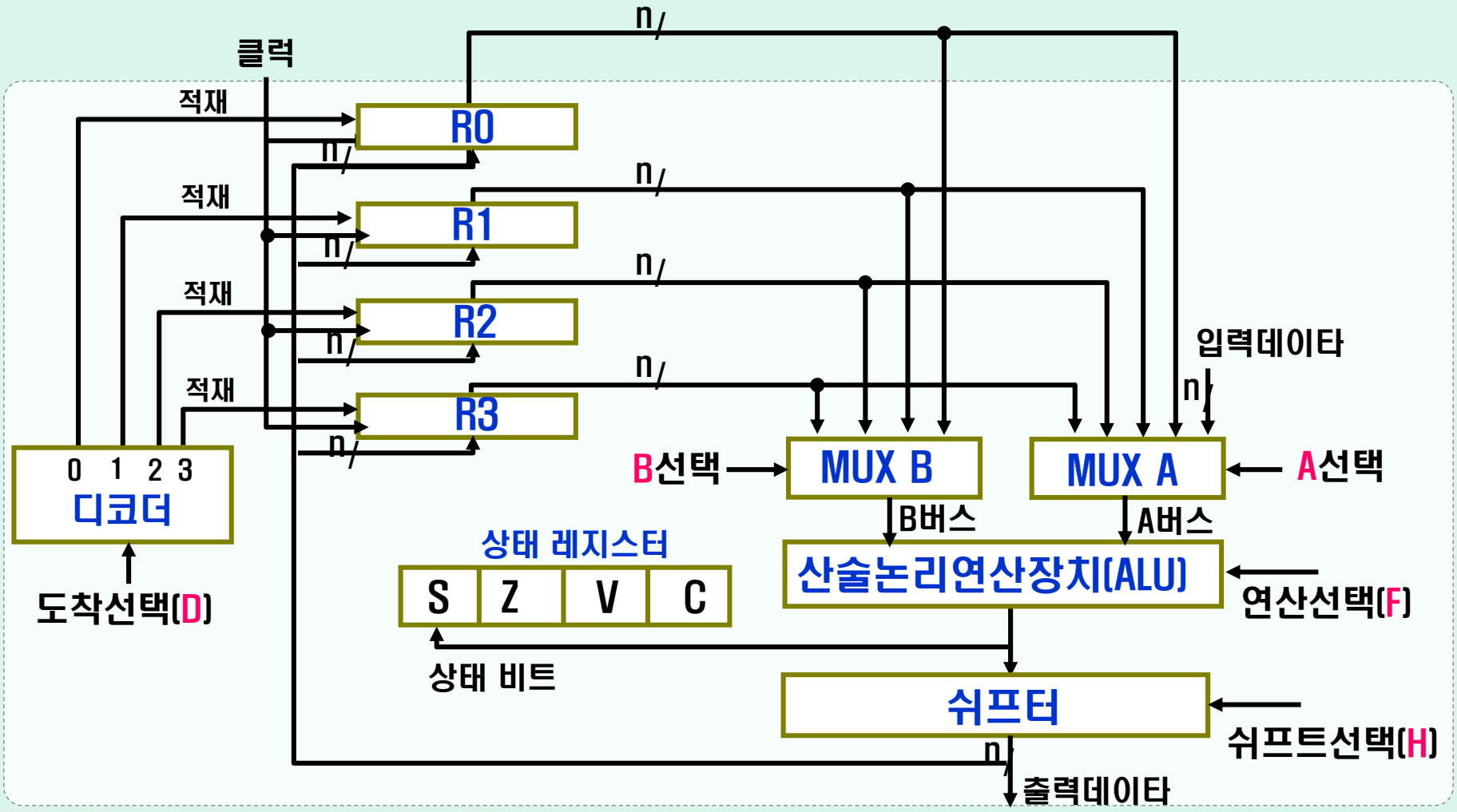
- 여러 개의 레지스터(레지스터 세트)
- 산술논리연산장치(ALU)
- 내부 버스(internal bus)

처리장치의 구성요소

- 처리장치의 내부 구성도(개념도)



처리장치의 내부구성


클럭
적재
적재
적재
적재
R0
R1
R2
R3
n
0 1 2 3
디코더
도착선택(D)
B선택
MUX B
MUX A
A선택
입력데이타
B버스
A버스
상태 레지스터
S Z V C
상태 비트
산술논리연산장치(ALU)
연산선택(F)
쉬프터
쉬프트선택(H)
출력데이타

처리장치의 동작원리

- 마이크로연산의 수행과정을 통해 처리장치의 동작을 이해
- **마이크로연산의 수행과정**

1) 지정된 출발 레지스터의 내용이 ALU의 입력으로 전달

2) ALU에서 그 연산을 실행

3) 그 결과가 도착 레지스터에 전송

처리장치의 동작원리

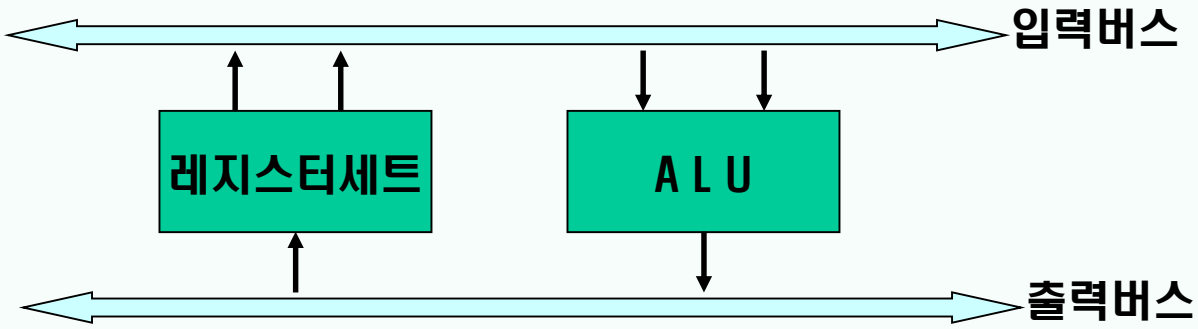
- 처리장치의 동작은 구성요소들의 선택신호에 의해 제어됨
- 마이크로연산의 예 : *R0 ← R1+ R2*

① 선택신호 A는 R1의 내용을 버스 A로 적재하기 위한 것이다
② 선택신호 B는 R2의 내용을 버스 B로 적재하기 위한 것이다
③ 선택신호 F는 ALU에서 산술연산 A+B를 수행하기 위한 것이다
④ 선택신호 H는 쉬프터에서 쉬프트 연산을 수행하기 위한 것이다
⑤ 선택신호 D는 연산결과를 R0로 적재하기 위한 것이다

내부 버스

내부버스

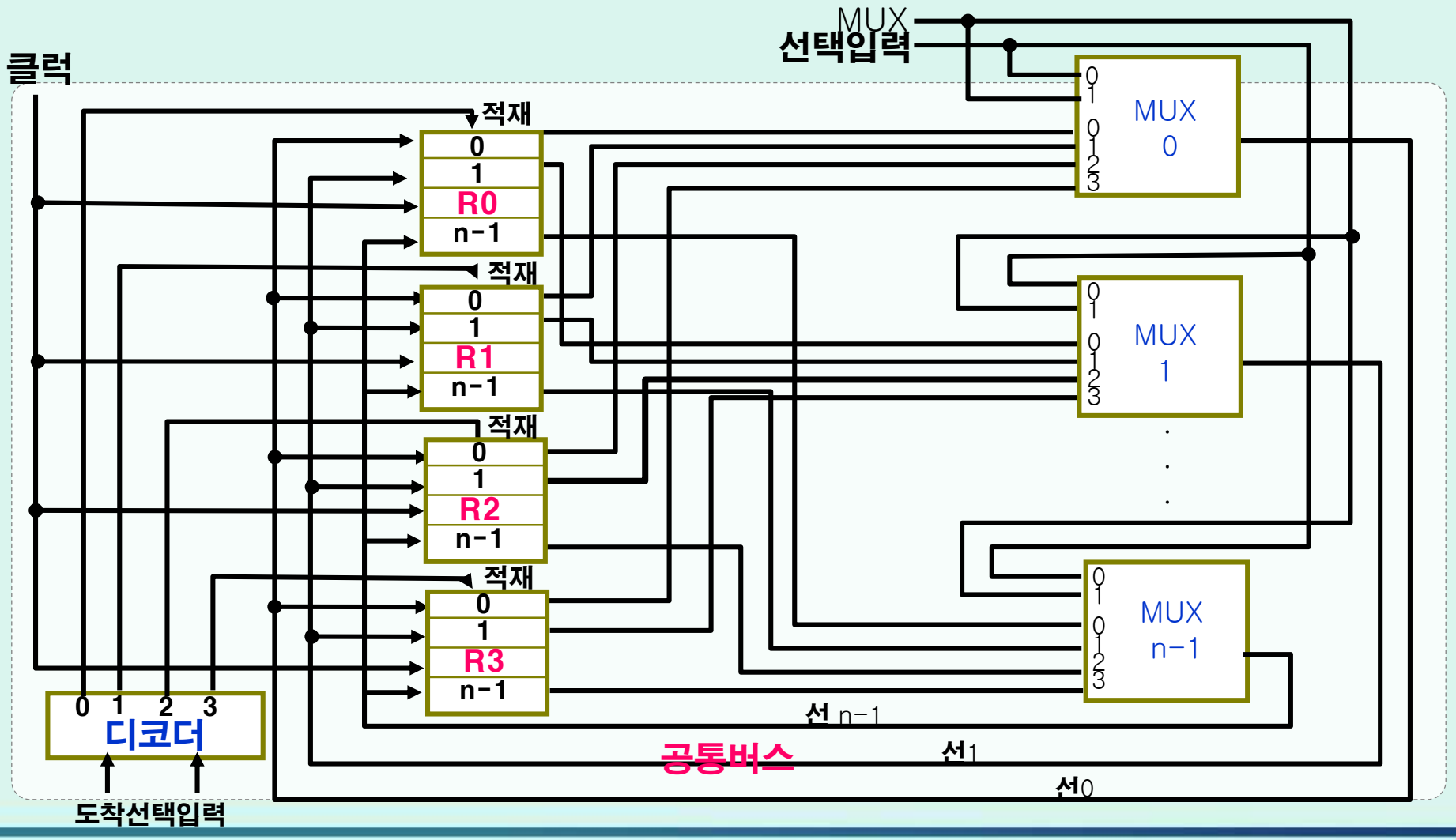
- 레지스터들 간의 데이터 전송을 위한 공통선로의 집합.
- 반대적으로 외부버스는 시스템버스로써 컴퓨터 시스템의 각 구성요소간의 통신
- 내부버스의 개념도



내부버스

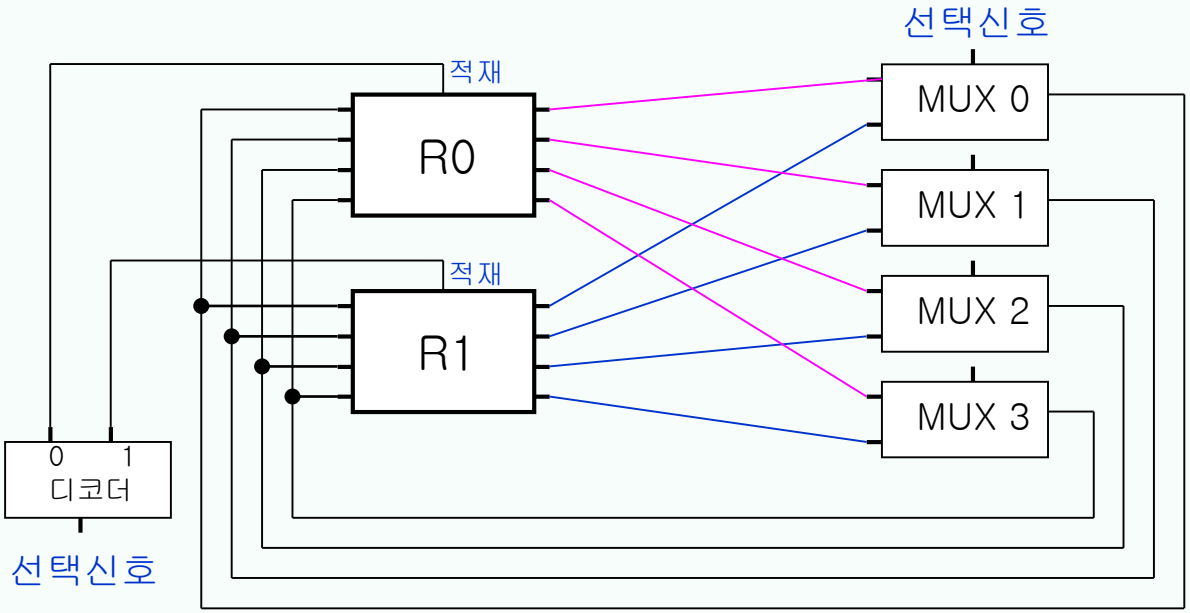
- 내부버스를 구성하는 방법
  - 멀티플렉서와 디코더를 이용
    - 멀티플렉서는 출발 레지스터 선택
    - 디코더는 도착 레지스터를 선택

내부버스(네 레지스터의 버스시스템)



내부버스

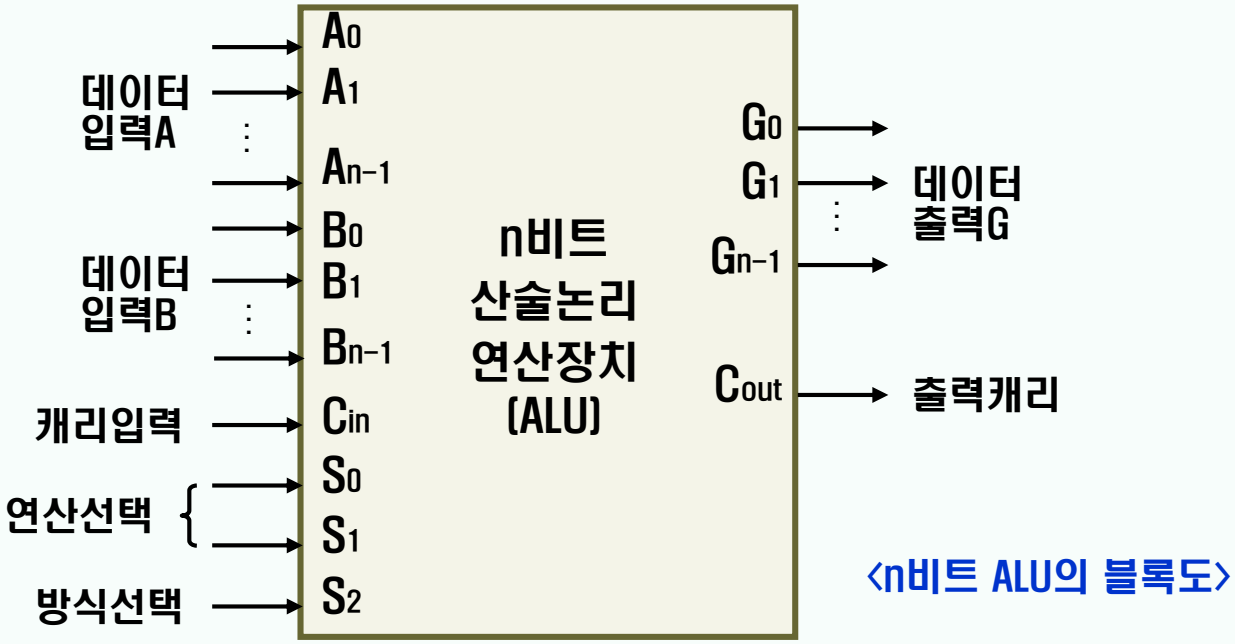
- ***R1 ← R0***



산술논리연산장치(ALU)

산술논리연산장치

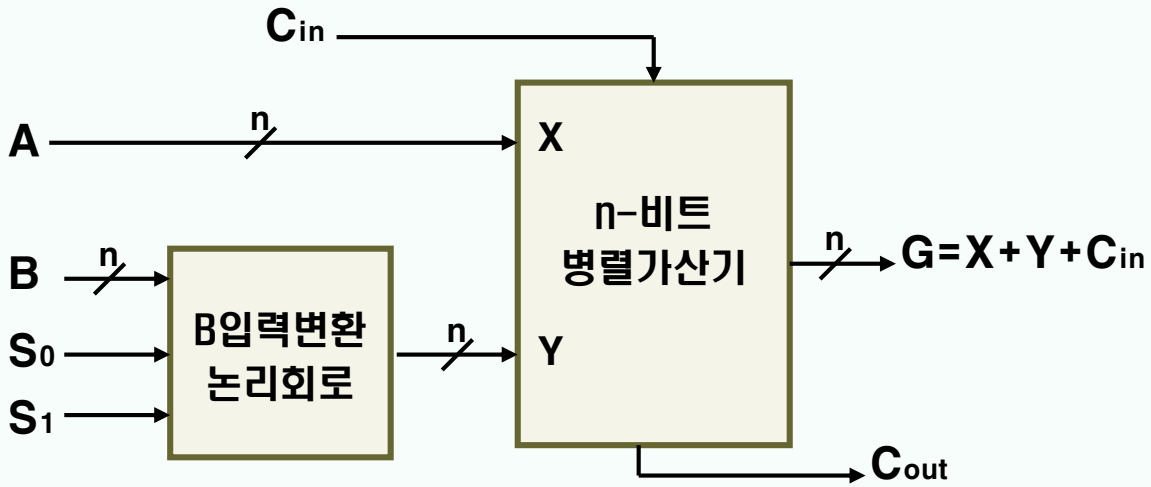
- 산술연산과 논리연산을 실행하는 조합논리회로
- 산술연산회로와 논리연산회로의 결합



<n비트 ALU의 블록도>

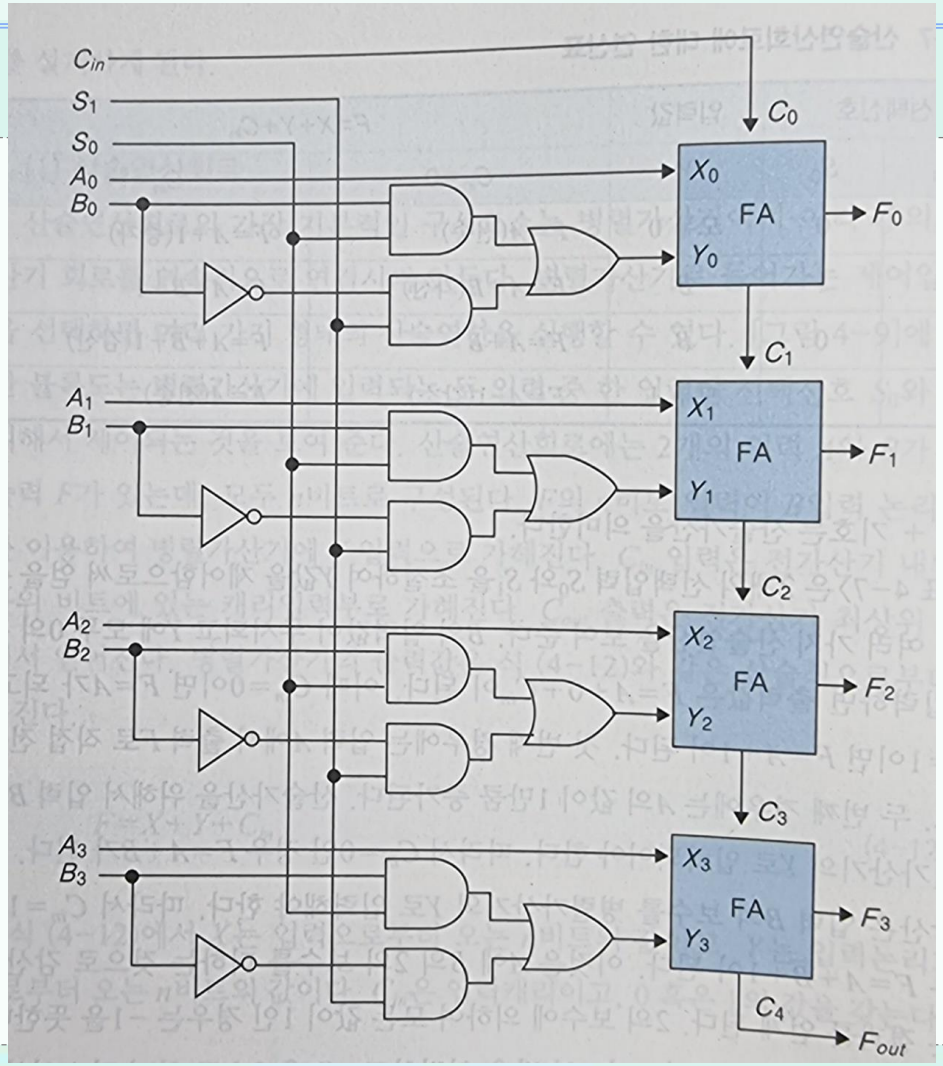
산술논리연산장치

- **산술연산회로**
  - 여러 개의 전가산기(FA)를 연속적으로 연결한 병렬가산기로 구성.
  - 병렬가산기로 들어가는 제어입력 값을 선택하여 여러 가지 형태의 산술연산을 실행.
  - **블록도**



산술논리연산장치

- 산술연산회로



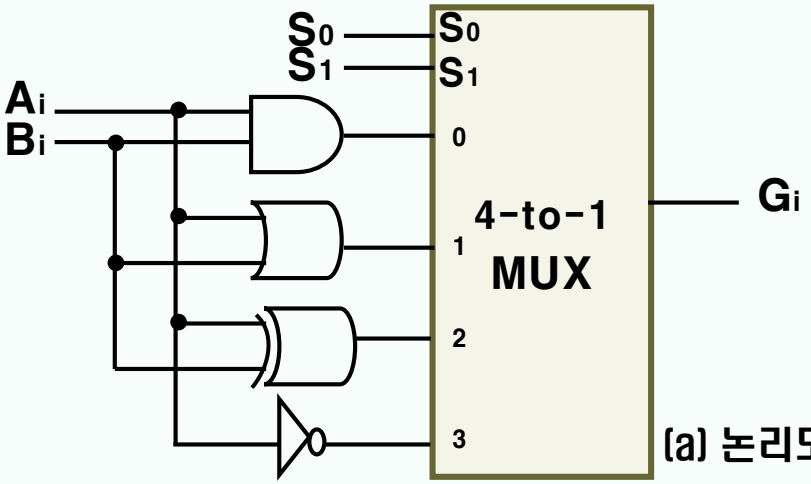
산술논리연산장치

- 산술연산의 종류

선택신호		입력값	G = A + Y + C_{in}	
S_1	S_0	Y	C_{in} = 0	C_{in} = 1
0	0	**모두 0**	G = A(전송)	G = A + 1(증가)
0	1	B	G = A + B(가산)	G = A + B + 1
1	0	$\overline{B}$	$G = A + \overline{B}$	$G = A + \overline{B} + 1$(감산)
1	1	**모두 1**	G = A −1(감소)	G = A(전송)

산술논리연산장치

- 논리연산회로
  - 레지스터에 있는 각 비트를 독립된 2진 변수로 간주하여 비트별 연산을 실행.
  - AND, OR, XOR, NOT연산 등이 있으며, 이를 이용하여 복잡한 연산을 유도.
  - 논리연산회로와 함수표



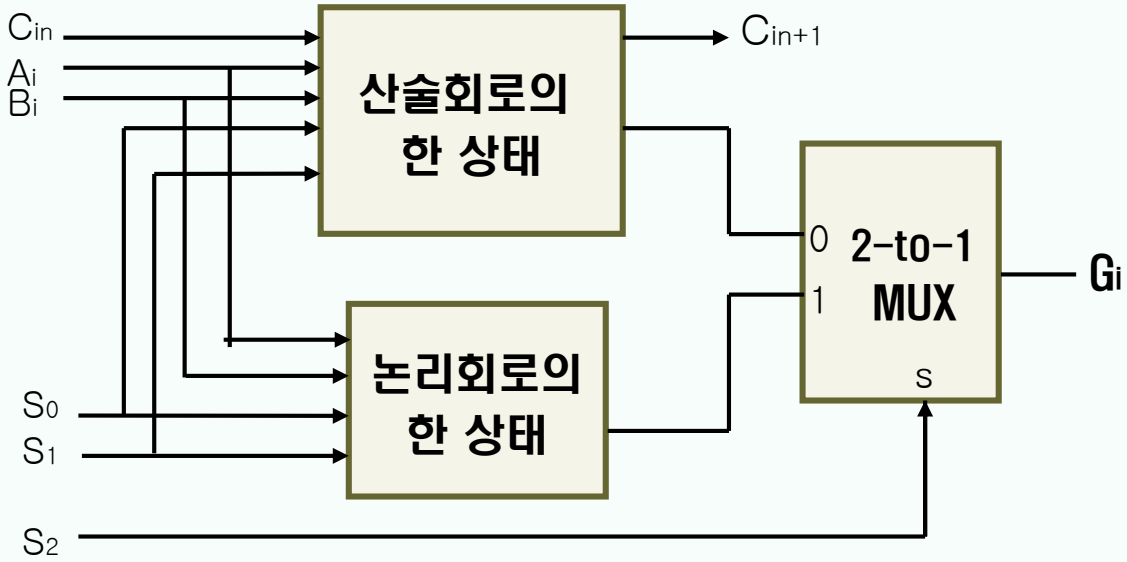
(a) 논리도

S_0 S_1	출 력	연 산
0 0	$G = A \wedge B$	AND
0 1	$G = A \vee B$	OR
1 0	$G = A \oplus B$	XOR
1 1	$G = \bar{A}$	NOT

(b) 함수표

산술논리연산장치

- 산술논리 연산장치
  - 산술연산장치와 논리연산장치를 결합
  - 한 단계의 ALU 구성도

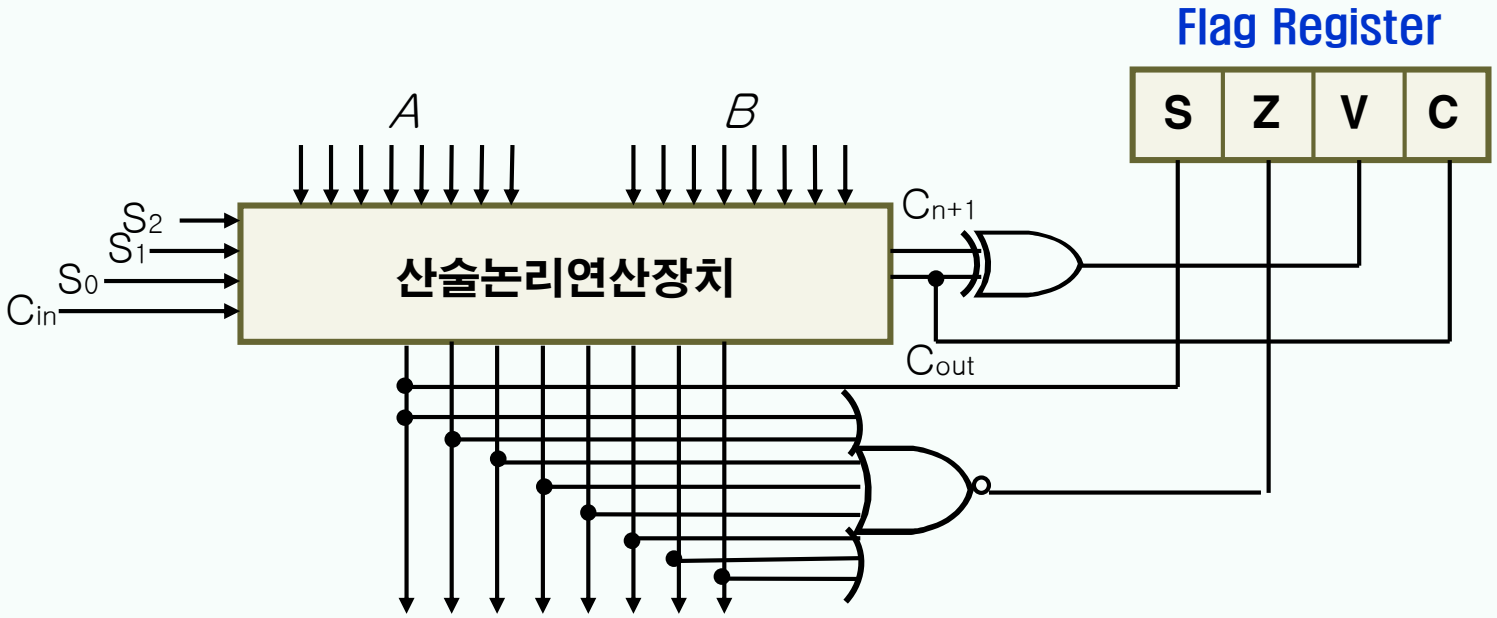


산술논리연산장치

8비트 데이터의 경우				연 산	기 능
S_2	S_1	S_0	C_{in}		
0	0	0	0	G = A	A의 전송
0	0	0	1	G = A +1	A에 1 더하기
0	0	1	0	G = A + B	덧셈
0	0	1	1	G = A+ B +1	캐리값 1과 더하기
0	1	0	0	G = A+ $\bar{B}$	A에 B의 1의 보수 더하기
0	1	0	1	G = A+ $\bar{B}$ +1	뺄셈
0	1	1	0	G = A – 1	A에서 1빼기
0	1	1	1	G = A	A의 전송
1	0	0	X	G = A∧B	AND
1	0	1	X	G = A∨B	OR
1	1	0	X	G = A⊕B	XOR
1	1	1	X	G = $\bar{A}$	A의 보수

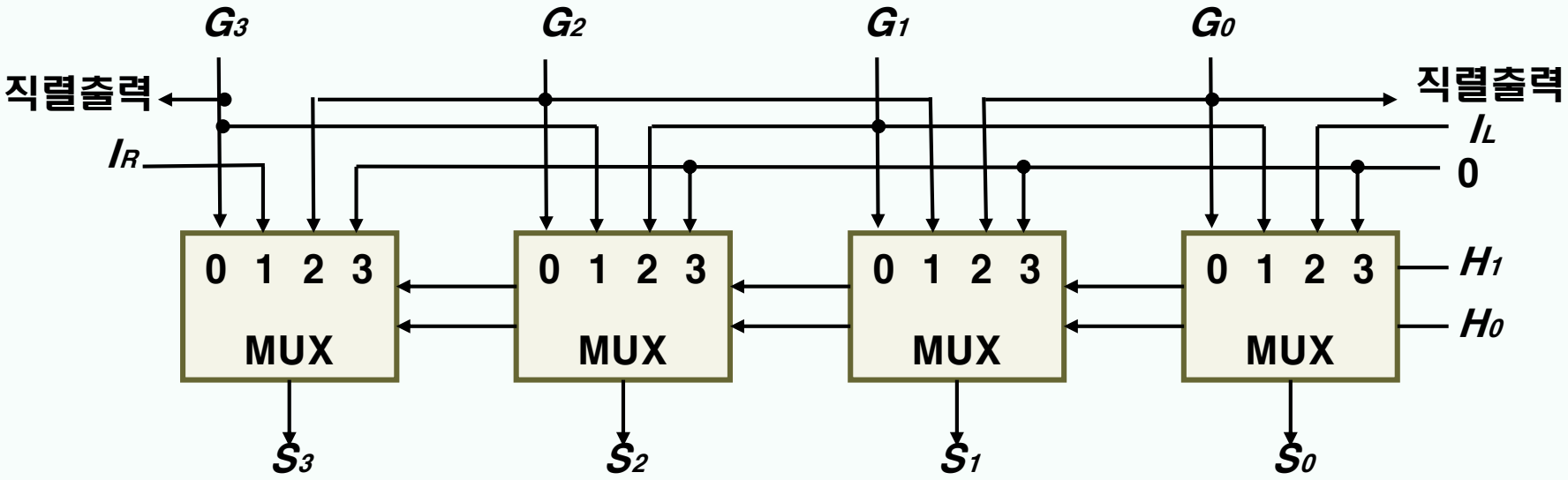
상태 레지스터(flag register)

- ALU에서 산술연산이 수행된 후 연산결과에 의해 나타나는 상태 값을 저장.
- 상태 레지스터들은 C(carry bit), S(sign bit), Z(zero bit), V(overflow bit)로 구성.



쉬프터(shifter)

- 입력데이터의 모든 비트들을 각각 서로 이웃한 비트로 자리를 옮기는 쉬프트연산을 수행



쉬프터(shifter)

- 쉬프트 연산

H_1	H_0	연 산	기 능
0	0	*S←G*	쉬프트 없이 전송
0	1	*S←shr G*	우측 쉬프트하여 전송
1	0	*S←shl G*	좌측 쉬프트하여 전송
1	1	*S←0*	모든 출력비트에 0을 전송

제어단어

제어단어

- 제어단어(control word)

  : 제어변수(선택신호)들의 묶음

  - 선택신호

    -처리장치내에서 수행되는 마이크로연산을 선택하는 변수

    -처리장치의 버스, ALU, 쉬프터, 도착 레지스터 등을 제어

    -선택신호 즉, 제어변수가 특정한 마이크로연산을 선택

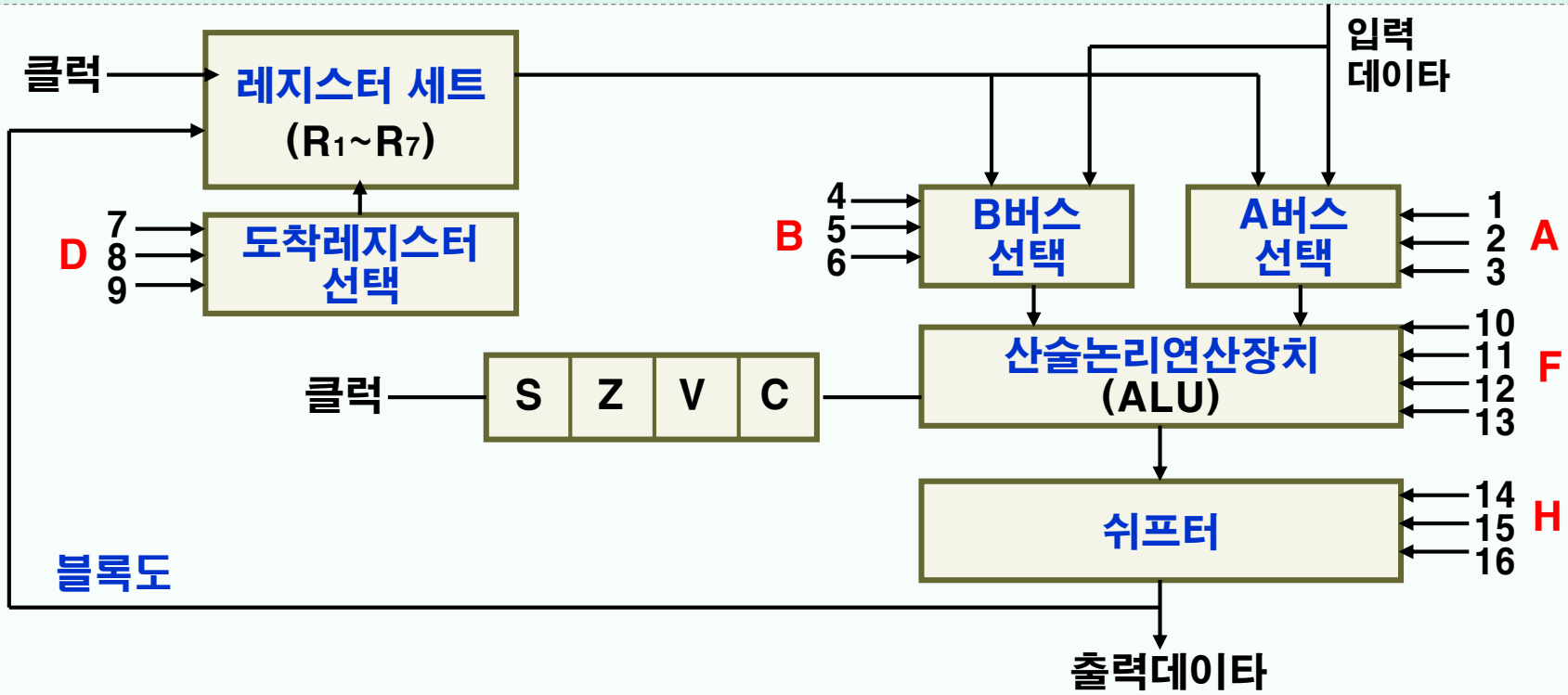
    -이러한 제어변수들의 묶음을 제어단어(control word)라고 한다.

제어단어

- 처리장치의 예
  - 레지스터 세트 : 7개의 레지스터(R1 ~ R7)
  - 산술논리연산장치 : 12가지 연산
  - 쉬프터 : 6가지 연산

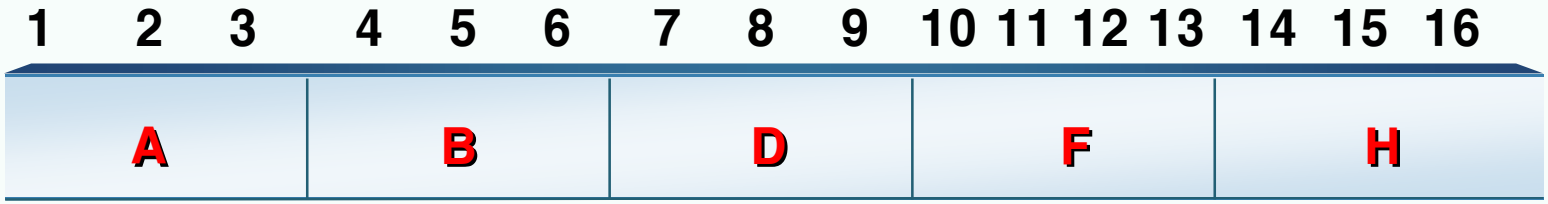
제어단어

- 처리장치의 구조



제어단어

- 제어단어의 내역



[제어단어의 각 필드]

제어단어

- 제어단어의 필드

  A 필드 : ALU로 입력되는 A 버스 선택(3 비트)

  B 필드 : ALU로 입력되는 B 버스 선택(3 비트)

  F 필드 : ALU의 연산 선택(4 비트)

  H 필드 : 쉬프터의 연산 선택(3 비트)

  D 필드 : 도착 레지스터 선택(3 비트)

제어단어

- 제어단어의 내역표

2진코드	A	B	D	F		H
				$C_{in}=0$	$C_{in}=1$	
000	외부입력	외부입력	없음	F=A	F=A+1	쉬프트 없음
001	R1	R1	R1	F=A+B	F =A+B+1	SHR
010	R2	R2	R2	$F=A+\overline{B}$	F=A-B	SHL
011	R3	R3	R3	F=A-1	F=A	bus=0
100	R4	R4	R4	$F=A \wedge B$	-	-
101	R5	R5	R5	$F=A \vee B$	-	ROR
110	R6	R6	R6	$F=A \oplus B$	-	ROL
111	R7	R7	R7	$F=\overline{A}$	-	-

제어단어

- 작성 예

R1←R2 - R3

① A 필드 : ALU의 A 버스 입력으로 R2의 내용을 보낸다.

② B 필드 : ALU의 B 버스 입력으로 R3의 내용을 보낸다.

③ D 필드 : 연산 결과를 도착 레지스터 R1으로 보낸다.

④ F 필드 : ALU에서 감산 연산(F=A-B)을 수행한다.

⑤ H 필드 : 쉬프터에서 연산을 수행하지 않는다.(쉬프트 없음)

제어단어

- 작성 방법 *(R1←R2 - R3)*

2진코드	A	B	D	F		H
				$C_{in}=0$	$C_{in}=1$	
000	외부입력	외부입력	없음	F=A	F=A+1	쉬프트 없음
001	R1	R1	R1	F=A+B	F =A+B+1	SHR
010	R2	R2	R2	F=A+$\overline{B}$	F=A-B	SHL
011	R3	R3	R3	F=A-1	F=A	bus=0
100	R4	R4	R4	F=A∧B	-	-
101	R5	R5	R5	F=A∨B	-	ROR
110	R6	R6	R6	F=A⊕B	-	ROL
111	R7	R7	R7	F=$\overline{A}$	-	-

제어단어

- 작성 결과

필드	A	B	D	F	H
기호	*R2*	*R3*	*R1*	*F=A-B*	쉬프트없음
2진코드	010	011	001	0101	000

제어단어

- 여러 가지 마이크로 연산에 대한 제어단어의 예

마이크로연산	기호표시					2진 제어단어				
	A	B	D	F	H	A	B	D	F	H
R1←R2-R3	R2	R3	R1	F=A-B	쉬프트없음	010	011	001	0101	000
R4←shr(R5+R6)	R5	R6	R4	F=A+B	SHR	101	110	100	0010	001
R7←R7+1	R7	-	R7	F=A+1	쉬프트없음	111	000	111	0001	000
R1←R2	R2	-	R1	F=A	쉬프트없음	010	000	001	0000	000
Output←R3	R3	-	NONE	F=A	쉬프트없음	011	000	000	0000	000
R4←rol R4	R4	-	R4	F=A	ROL	100	000	100	0000	110
R5←0	-	-	R5	-	bus=0	000	000	101	0000	011

제어단어

- 제어단어 생성을 위한 효과적인 방법
  - 작성된 제어단어를 기억장치에 저장하고, 기억장치의 출력을 처리장치의 각 구성요소의 선택신호로 연결.
  - 이렇게 하면 기억장치로부터 연속적인 제어단어를 읽음으로써 처리장치에서의 마이크로 연산이 정해진 순서대로, 연속적으로 수행된다.