조합논리회로

디코더, 인코더
멀티플렉서, 코드변환기

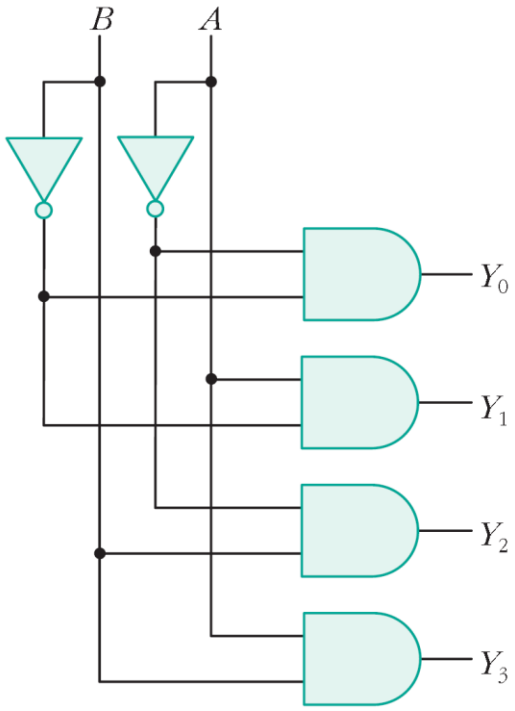
❑ 디코더

- **디코더**(decoder) : 입력선에 나타나는 n비트의 2진 코드를 최대 2^n개의 서로 다른 정보로 바꿔주는 조합논리회로

입력		출력			
B	A	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

$Y_0 = \overline{B}\overline{A},\ Y_1 = \overline{B}A,\ Y_2 = B\overline{A},\ Y_3 = BA$

(a) 진리표와 논리식



(b) 논리 회로

그림 3-41 2×4 디코더

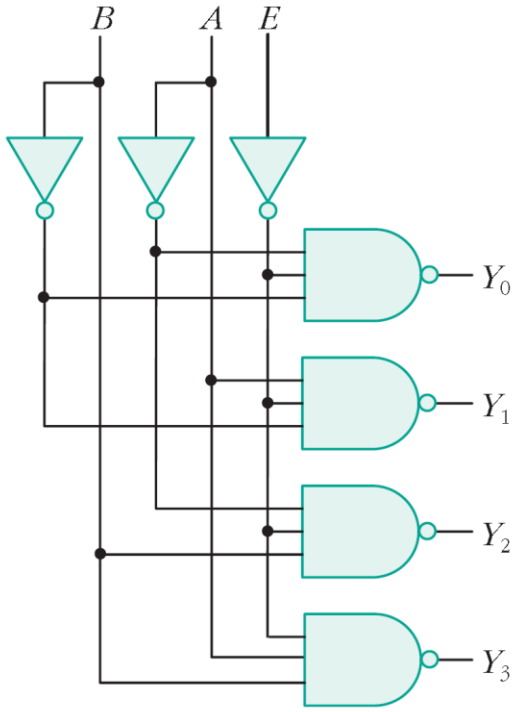
❖ 실제 디코더 회로

- 실제 IC들은 AND 게이트가 아닌 NAND 게이트로 구성되어 출력은 그림과 같이 반대로 된다.
- 또 대부분의 디코더 IC는 인에이블(enable) 입력이 있어 회로를 제어한다.

입력			출력			
$\overline{E}$	B	A	Y_3	Y_2	Y_1	Y_0
1	x	x	1	1	1	1
0	0	0	1	1	1	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	1

$$Y_0 = \overline{\overline{E}BA},\ Y_1 = \overline{\overline{E}B\overline{A}},\ Y_2 = \overline{\overline{E}\overline{B}A},\ Y_3 = \overline{\overline{E}\overline{B}\overline{A}}$$

(a) 진리표와 논리식



(b) 논리 회로

그림 3-42 인에이블이 있는 2×4 NAND 디코더

❖ 3×8 디코더

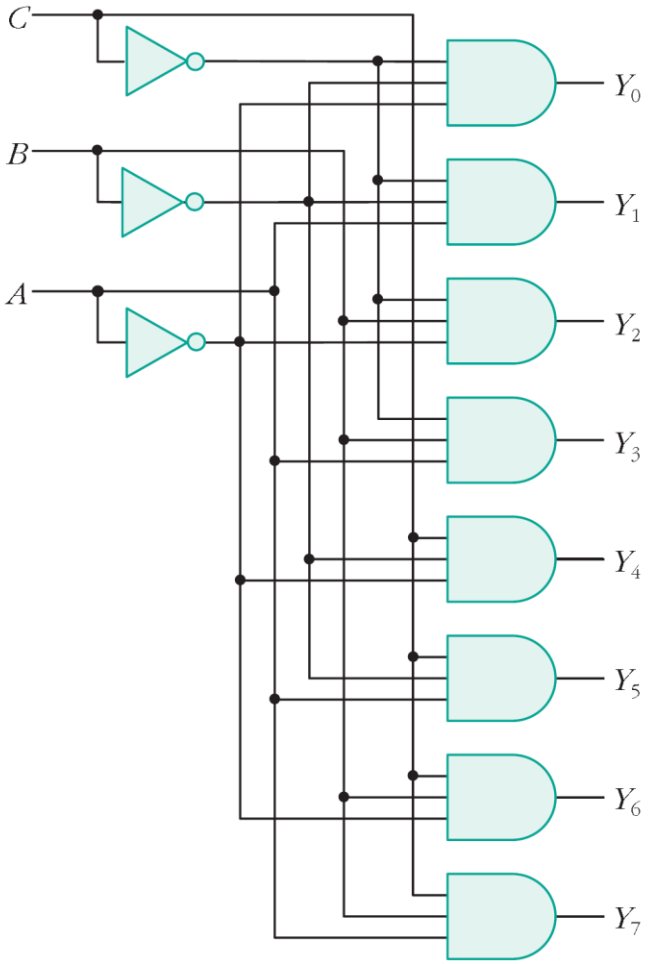
- 3개의 입력에 따라서 8개의 출력 중 하나가 선택

입력			출력							
C	B	A	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

$$Y_0 = \overline{C}\overline{B}\overline{A},\ Y_1 = \overline{C}\overline{B}A,\ Y_2 = \overline{C}B\overline{A},\ Y_3 = \overline{C}BA$$

$$Y_4 = C\overline{B}\overline{A},\ Y_5 = C\overline{B}A,\ Y_6 = CB\overline{A},\ Y_7 = CBA$$

(a) 진리표와 논리식



(b) 논리 회로

그림 3-43 3×8 디코더 회로

❖ 2개의 3×8 디코더로 4×16 디코더를 구성

D=0	상위 디코더만 enable되어 출력은 Y_0~ Y_7 중의 하나가 1로 되고, 하위 디코더 출력들은 모두 0이 된다.
D=1	하위 디코더만 enable 되어 출력은 Y_8~ Y_{15} 중의 하나가 1로 되고, 상위 디코더 출력들은 모두 0이 된다.

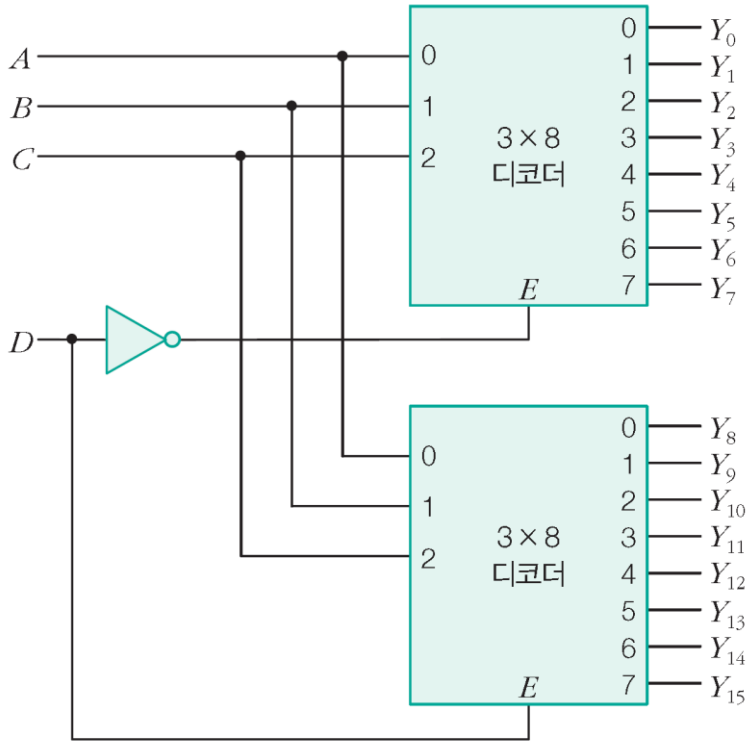


그림 3-44 3×8 디코더 2개를 이용한 4×16 디코더

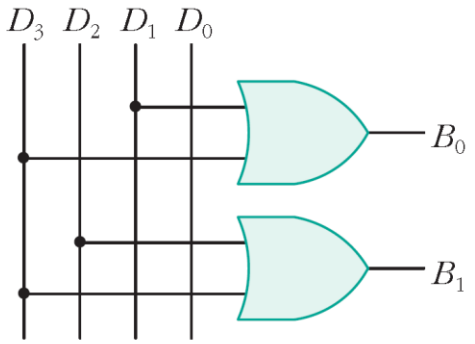
❑ 인코더

- 부호기라고도 하는 인코더(encoder)는 디코더의 반대 기능을 수행하는 조합 논리 회로로, 2^n개를 입력받아 n개를 출력한다.
- 인코더는 2^n개 중 활성화된 1비트 입력 신호를 받아 그 숫자에 해당하는 n비트 2진 정보를 출력한다.

입력				출력	
D_3	D_2	D_1	D_0	B_1	B_0
0	0	0	1	0	0
0	0	1	0	0	1
1	1	0	0	1	0
1	0	0	0	1	1

$$B_1 = D_2 + D_3,\ B_0 = D_1 + D_3$$

(a) 진리표와 논리식



(b) 논리 회로

그림 3-45 4×2 인코더

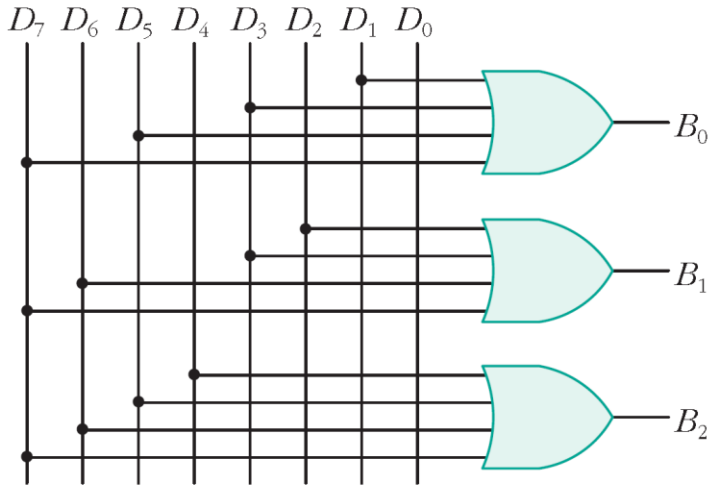
❖ 8×3 인코더

- 8(=2^3)개의 입력과 3개의 출력을 가지며, 입력의 신호에 따라 3개의 2진 조합으로 출력한다.

입력								출력		
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	B_2	B_1	B_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

$$B_2 = D_4 + D_5 + D_6 + D_7,\ B_1 = D_2 + D_3 + D_6 + D_7$$
$$B_0 = D_1 + D_3 + D_5 + D_7$$

(a) 진리표와 논리식



(b) 논리 회로

그림 3-46 8×3 인코더

❑ 멀티플렉서

- **멀티플렉서**(multiplexer, MUX)는 여러 개의 입력선들 중에서 하나를 선택하여 출력선에 연결하는 조합논리회로이다. 선택선들의 값에 따라서 특별한 입력선이 선택된다.
- 멀티플렉서는 많은 입력들 중 하나를 선택하여 선택된 입력선의 2진 정보를 출력선에 넘겨주기 때문에 **데이터 선택기**(data selector)라 부르기도 한다.
- **디멀티플렉서**(demultiplexer, DEMUX) 는 정보를 한 선으로 받아 2^n 개의 가능한 출력 선들 중 하나를 선택하여, 받은 정보를 전송하는 회로다.

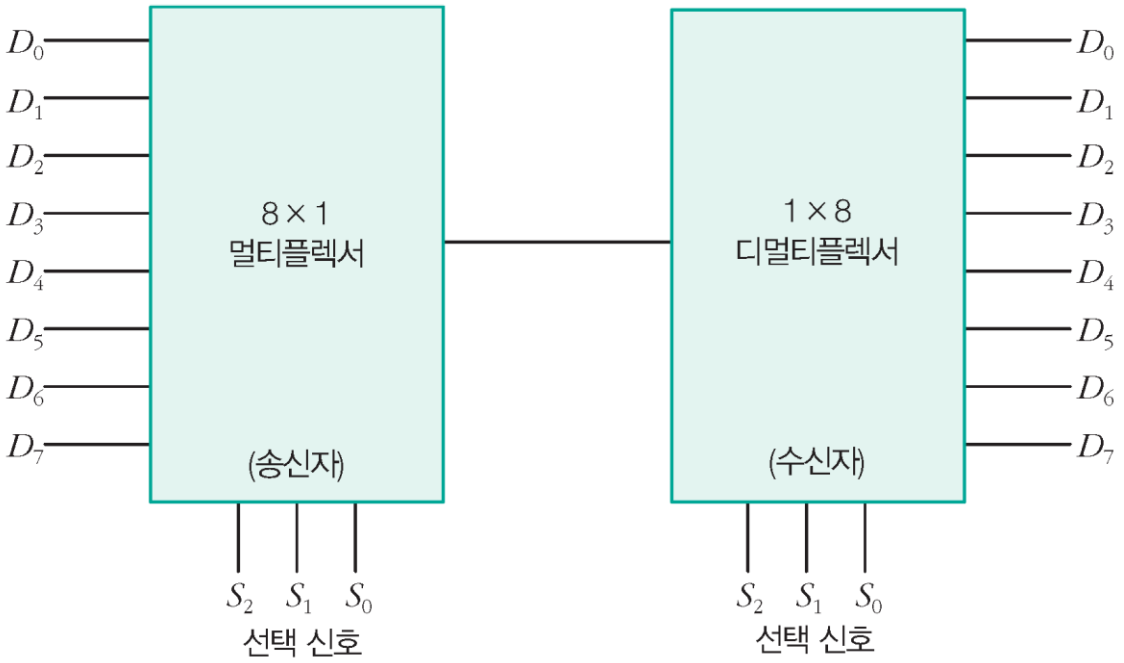


그림 3-47 멀티플렉서와 디멀티플렉서의 역할

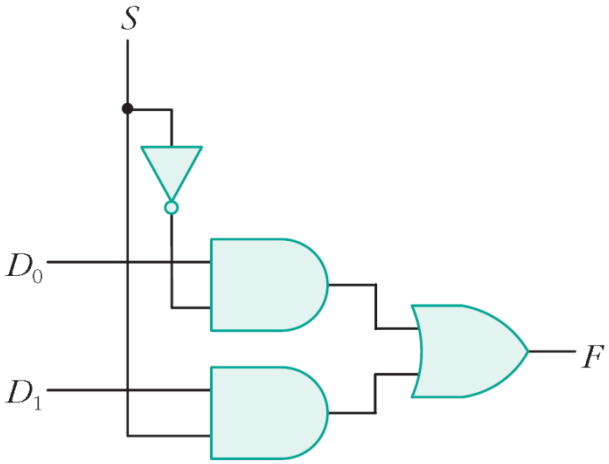
❖ 2×1 멀티플렉서

- 2(=2^1)개의 입력중의 하나를 선택선 S에 입력된 값에 따라서 출력으로 보내주는 조합회로

선택선 S	출력 F
0	D_0
1	D_1

$$F = \overline{S}D_0 + SD_1$$

(a) 진리표와 논리식



(b) 논리 회로

그림 3-48 2×1 멀티플렉서

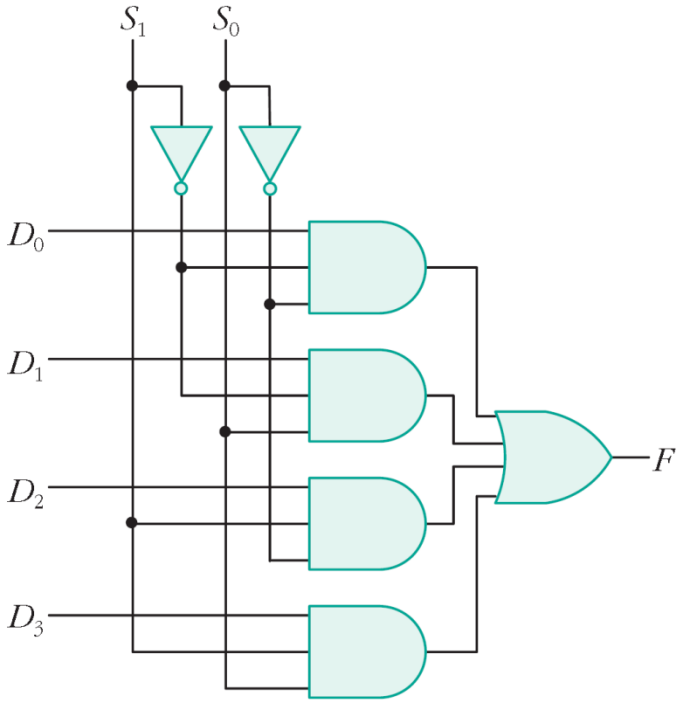
❖ 4×1 멀티플렉서

- 4(=2^2)개의 입력중의 하나를 선택선 S_1과 S_0에 입력된 값에 따라서 출력으로 보내주는 조합회로

선택선		출력
S_1	S_0	F
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

$$F = \overline{S}_1\overline{S}_0D_0 + \overline{S}_1S_0D_1 + S_1\overline{S}_0D_2 + S_1S_0D_3$$

(a) 진리표와 논리식

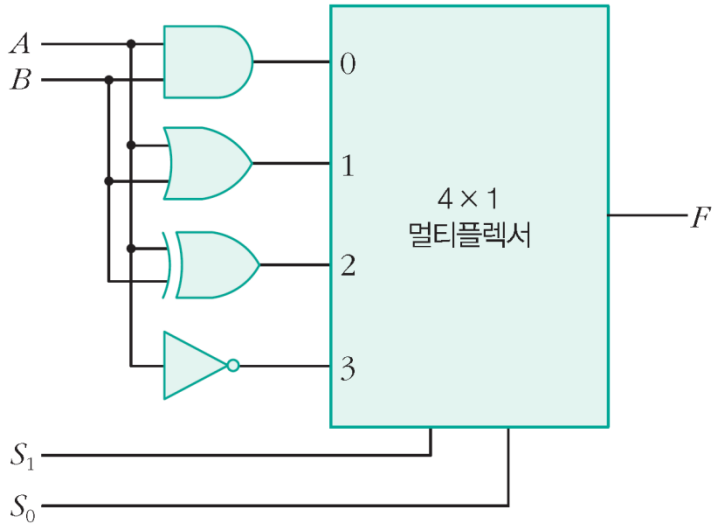


(b) 논리 회로

그림 3-49 4×1 멀티플렉서

❖ 4×1 멀티플렉서 응용

- 4×1 멀티플렉서를 이용하여 두 입력 A, B에 대해 AND, OR, XOR, NOT 논리 연산을 수행하는 하드웨어 모듈



S_1	S_0	출력	연산
0	0	$F=AB$	AND
0	1	$F=A+B$	OR
1	0	$F=A \oplus B$	XOR
1	1	$F=\overline{A}$	NOT

그림 3-50 4×1 멀티플렉서를 이용한 논리 연산 하드웨어 모듈

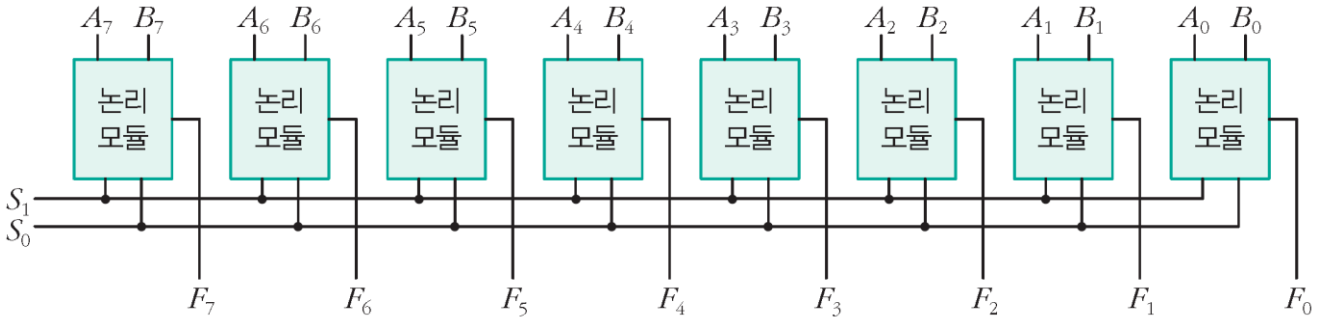


그림 3-51 8비트 논리 연산 장치 구성 예

❖ 멀티플렉서를 이용한 조합회로 구현

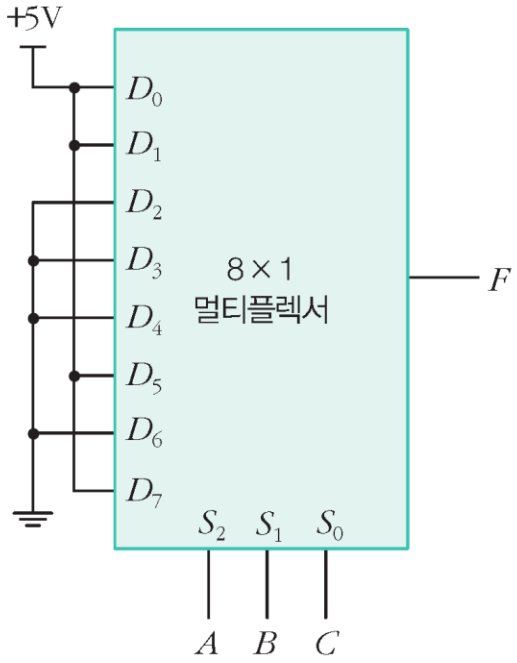
- $F(A,B,C) = \sum m(0,1,5,7)$를 8×1 멀티플렉서로 구현하는 경우

  ☞ 3개의 선택선을 입력 A, B, C 로 사용

입력			출력
A	B	C	F
0	0	0	$1(D_0)$
0	0	1	$1(D_1)$
0	1	0	$0(D_2)$
0	1	1	$0(D_3)$
1	0	0	$0(D_4)$
1	0	1	$1(D_5)$
1	1	0	$0(D_6)$
1	1	1	$1(D_7)$

$F(A, B, C) = \Sigma m(0, 1, 5, 7)$

(a) 진리표와 논리식



(b) 회로도

그림 3-52 8×1 멀티플렉서를 이용한 회로

❖ 멀티플렉서를 이용한 조합회로 구현(계속)

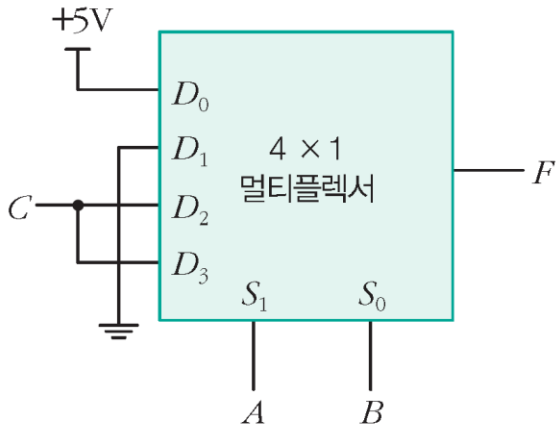
- $F(A,B,C)=\sum m(0,1,5,7)$ 를 4×1 멀티플렉서로 구현하는 경우

  ☞ A, B는 선택선으로 C는 D_0, D_1, D_2, D_3을 조합하여 사용

입력			출력	
A	B	C	F	
0	0	0	$D_0=1$	1
		1		1
0	1	0	$D_1=0$	0
		1		0
1	0	0	$D_2=C$	0
		1		1
1	1	0	$D_3=C$	0
		1		1

$F(A, B, C)=\Sigma m(0, 1, 5, 7)$

(a) 진리표와 논리식



(b) 회로도

그림 3-53 4×1 멀티플렉서를 이용한 회로

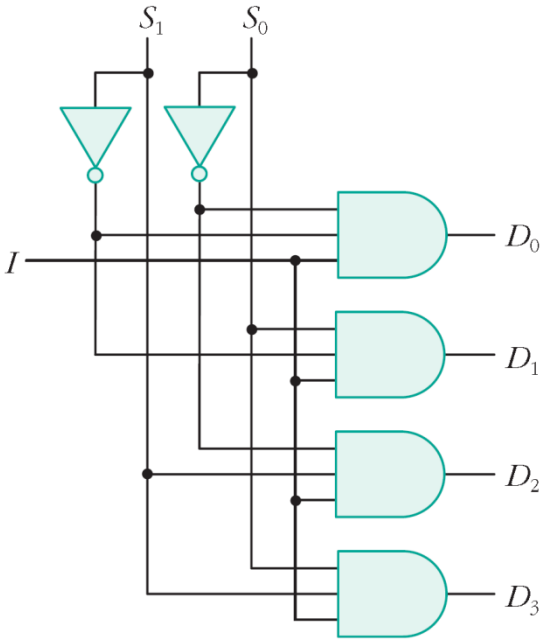
❑ 디멀티플렉서

- **디멀티플렉서**(demultiplexer)는 하나의 입력선에 데이터를 입력하면 선택선 n개로 $2n$개 중 하나를 출력하는 조합 논리 회로다. **데이터 분배기**라고도 한다.
- 1×4 디멀티플렉서는 선택선(S_1, S_0) 2개로 출력(D_3, D_2, D_1, D_0) 4개 중 하나를 선택해 입력(I)을 연결한다.

선택선		출력			
S_1	S_0	D_3	D_2	D_1	D_0
0	0	0	0	0	I
0	1	0	0	I	0
1	0	0	I	0	0
1	1	I	0	0	0

$$D_0 = \overline{S}_1\overline{S}_0 I,\ D_1 = \overline{S}_1 S_0 I,\ D_2 = S_1\overline{S}_0 I,\ D_3 = S_1 S_0 I$$

(a) 진리표와 논리식



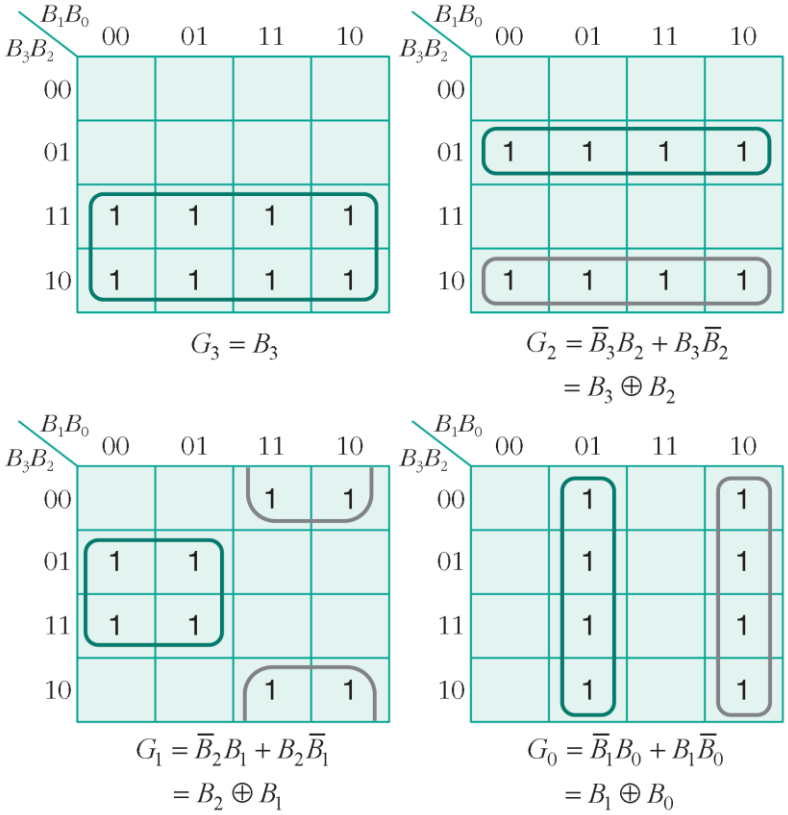
(b) 논리 회로

그림 3-54 1×4 디멀티플렉서

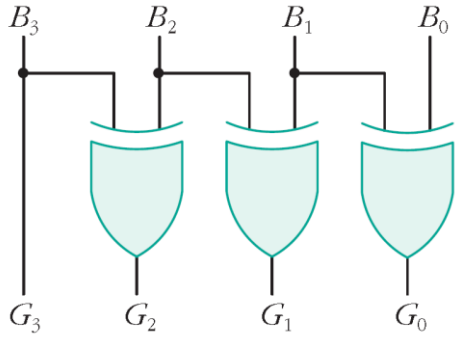
❑ 코드 변환기 (2진 코드 → 그레이 코드 변환)

2진 코드(입력) $B_3\ B_2\ B_1\ B_0$	그레이 코드(출력) $G_3\ G_2\ G_1\ G_0$
0 0 0 0	0 0 0 0
0 0 0 1	0 0 0 1
0 0 1 0	0 0 1 1
0 0 1 1	0 0 1 0
0 1 0 0	0 1 1 0
0 1 0 1	0 1 1 1
0 1 1 0	0 1 0 1
0 1 1 1	0 1 0 0
1 0 0 0	1 1 0 0
1 0 0 1	1 1 0 1
1 0 1 0	1 1 1 1
1 0 1 1	1 1 1 0
1 1 0 0	1 0 1 0
1 1 0 1	1 0 1 1
1 1 1 0	1 0 0 1
1 1 1 1	1 0 0 0

(a) 진리표



(b) 논리식의 간소화



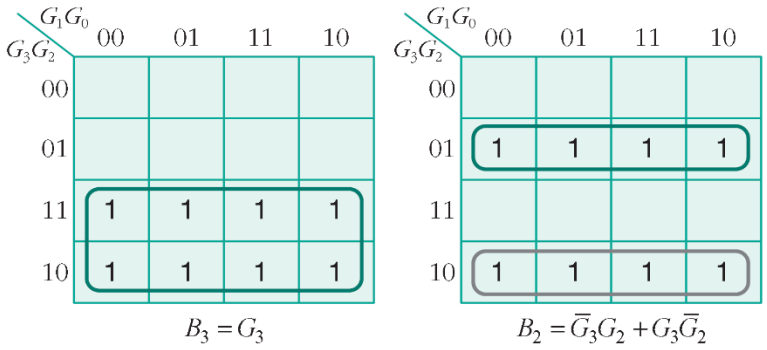
(c) 논리 회로

그림 3-55 2진 코드를 그레이 코드로 변환하는 회로

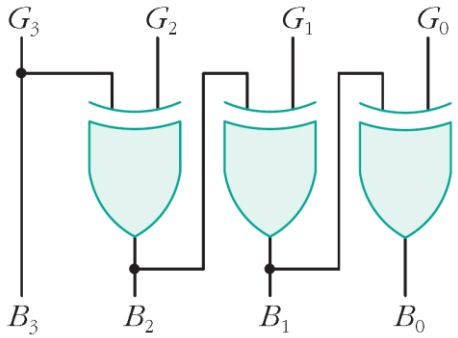
❑ 코드 변환기 (그레이 코드 → 2진 코드 변환)

그레이 코드(입력) $G_3\ G_2\ G_1\ G_0$	2진 코드(출력) $B_3\ B_2\ B_1\ B_0$
0 0 0 0	0 0 0 0
0 0 0 1	0 0 0 1
0 0 1 0	0 0 1 1
0 0 1 1	0 0 1 0
0 1 0 0	0 1 1 1
0 1 0 1	0 1 1 0
0 1 1 0	0 1 0 0
0 1 1 1	0 1 0 1
1 0 0 0	1 1 1 1
1 0 0 1	1 1 1 0
1 0 1 0	1 1 0 0
1 0 1 1	1 1 0 1
1 1 0 0	1 0 0 0
1 1 0 1	1 0 0 1
1 1 1 0	1 0 1 1
1 1 1 1	1 0 1 0

(a) 진리표



(b) 논리식의 간소화



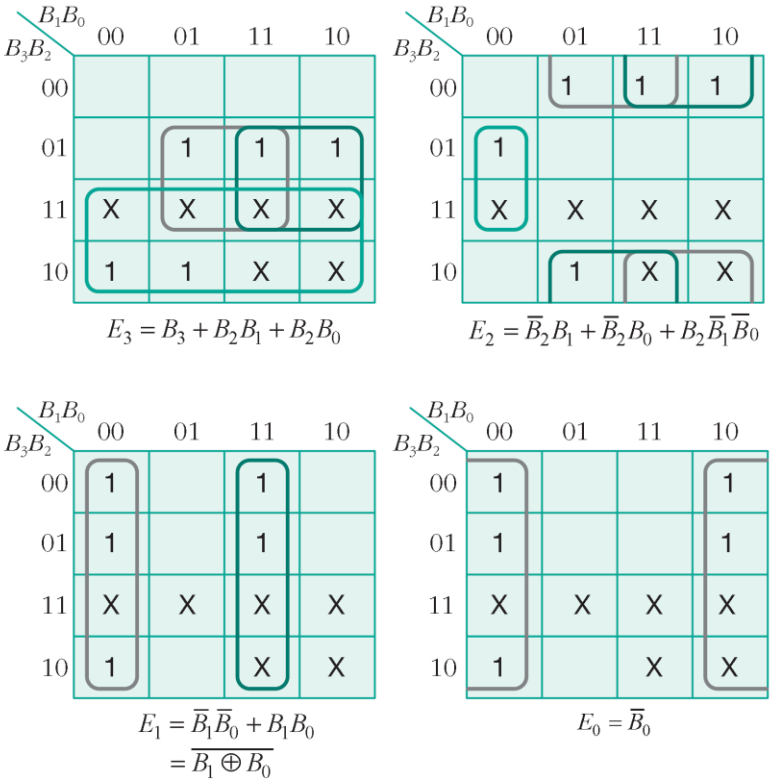
(c) 논리 회로

그림 3-56 그레이 코드를 2진 코드로 변환하는 회로

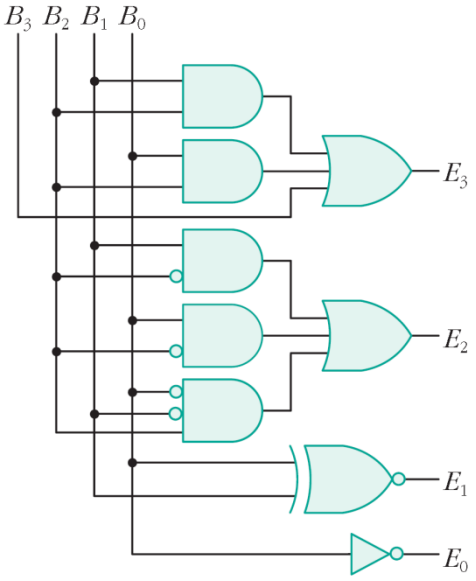
❑ 코드 변환기 (BCD 코드 → 3초과 코드 변환)

BCD 코드(입력) $B_3\ B_2\ B_1\ B_0$	3초과 코드(출력) $E_3\ E_2\ E_1\ E_0$
0 0 0 0	0 0 1 1
0 0 0 1	0 1 0 0
0 0 1 0	0 1 0 1
0 0 1 1	0 1 1 0
0 1 0 0	0 1 1 1
0 1 0 1	1 0 0 0
0 1 1 0	1 0 0 1
0 1 1 1	1 0 1 0
1 0 0 0	1 0 1 1
1 0 0 1	1 1 0 0
1 0 1 0	× × × ×
1 0 1 1	× × × ×
1 1 0 0	× × × ×
1 1 0 1	× × × ×
1 1 1 0	× × × ×
1 1 1 1	× × × ×

(a) 진리표



(b) 논리식의 간소화



(c) 논리 회로

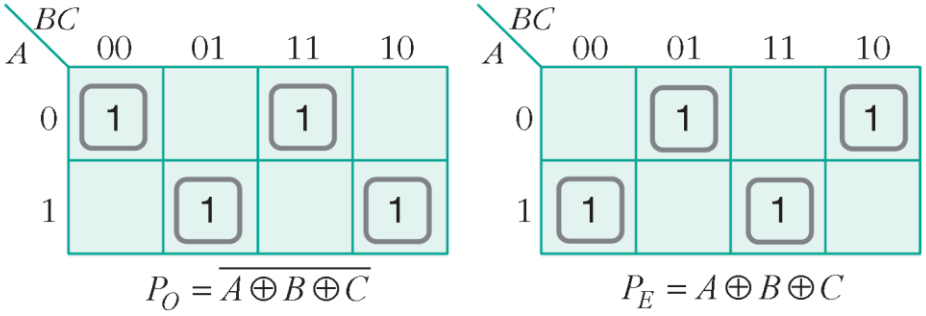
그림 3-57 BCD 코드를 3초과 코드로 변환하는 회로

❑ 패리티 발생기와 검사기

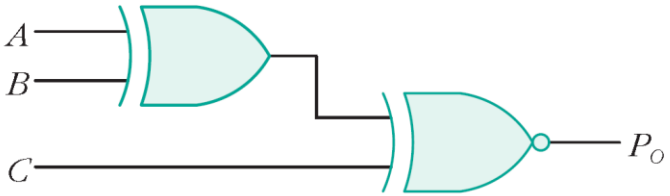
❖ 패리티 발생기

입력			출력	
A	B	C	P_O(홀수)	P_E(짝수)
0	0	0	1	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	0	1

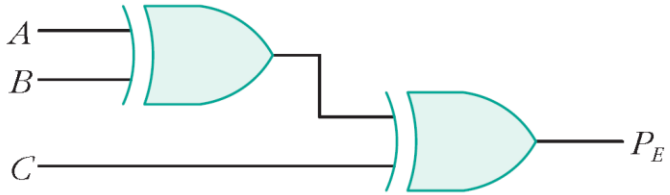
(a) 진리표



(b) 논리식의 간소화



(c) 홀수 패리티 발생기 논리 회로



(d) 짝수 패리티 발생기 논리 회로

그림 3-58 패리티 발생기

❖ 패리티 검사기

입력				출력
A	B	C	P_O(홀수)	Y_O(홀수)
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

입력				출력
A	B	C	P_E(짝수)	Y_E(짝수)
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

(a) 진리표

❖ 패리티 검사기(계속)

- 패리티 검사기 출력이 $Y = 0$이면 에러가 발생하지 않았다고 판단하고, $Y = 1$이면 에러가 발생했다고 판단한다.

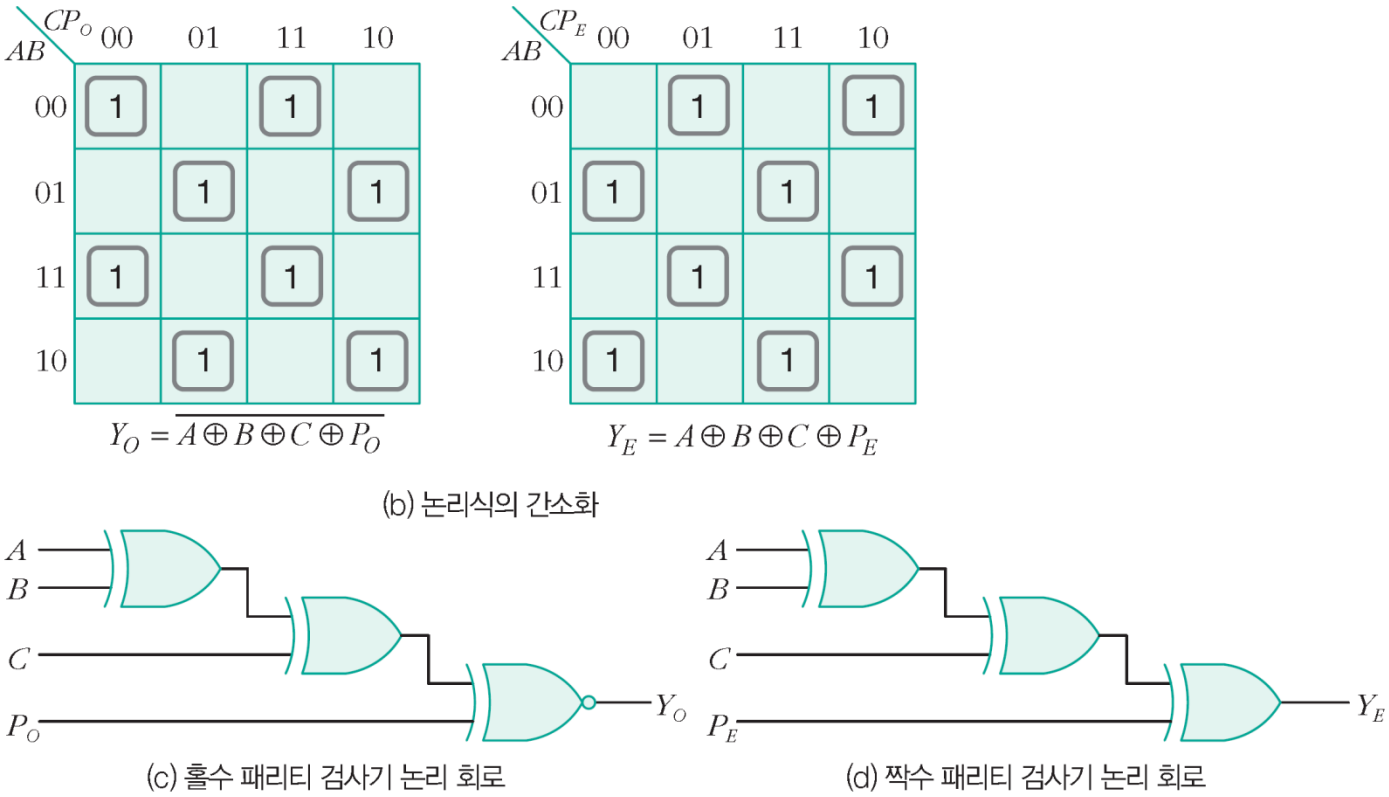


그림 3-59 패리티 검사기

❖ 데이터 전송 시스템

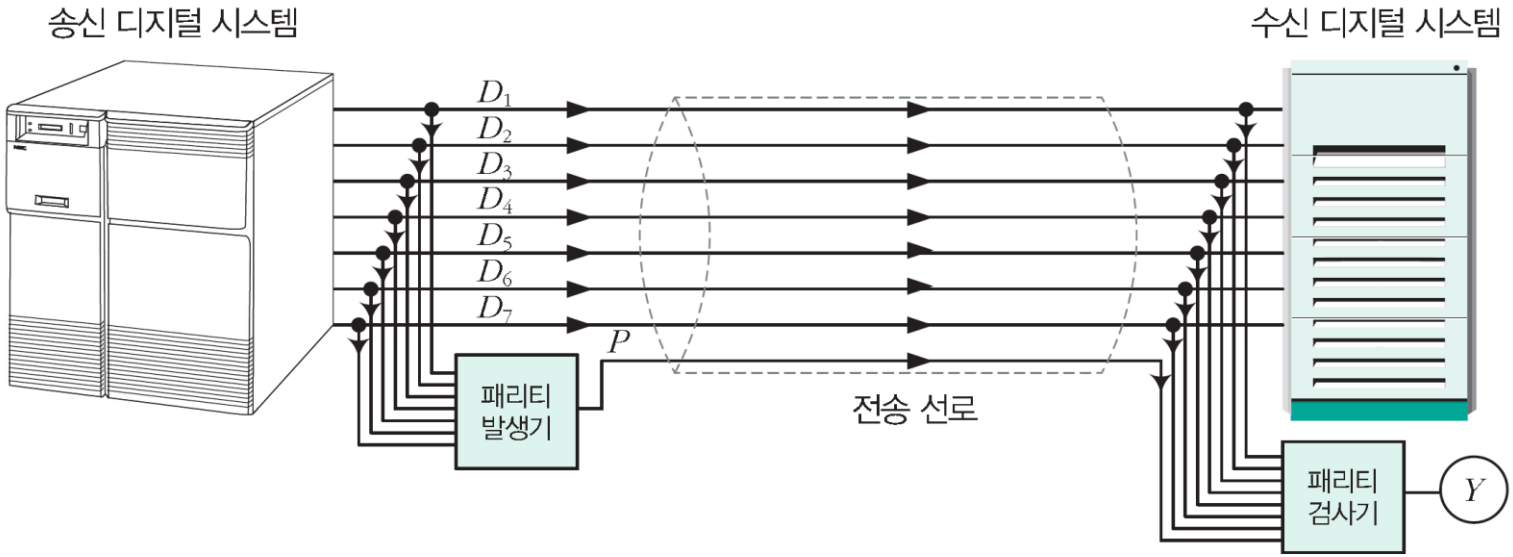


그림 3-60 데이터 전송 시스템에서 패리티 비트를 이용한 에러 검출

3 ROM을 사용한 조합 논리 회로의 설계

- 다음 불 함수를 ROM을 사용해 구현하는 예

$$F_1(A,B) = \sum m(1, 2, 3) \qquad F_2(A,B) = \sum m(0, 2)$$

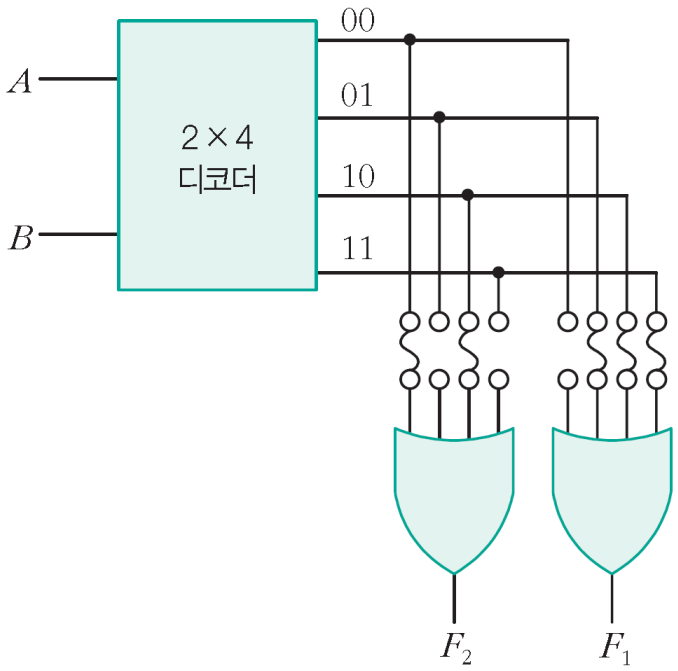


그림 3-61 ROM을 사용해 구현한 조합 논리 회로

수고하셨습니다!