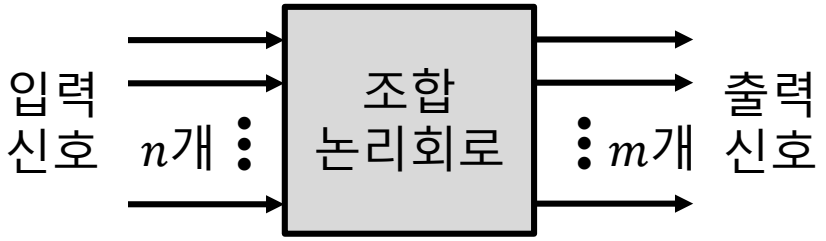
Lecture 06

조합논리회로

조합논리회로

- 조합논리회로란?
  - 논리곱(AND), 논리합(OR), 논리 부정(NOT)의 3가지 기본 논리회로의 조합으로 만들어짐
  - 구성 요소
    ① 입력 신호
    ② 논리 게이트
    ③ 출력 신호
  - 예,
    - 가산기(adder)
    - 비교기(comparator)
    - 디코더(decoder)
    - 멀티플렉서(multiplexer)
    - …



가산기(Adder)

- 가산기란?
  - 2진 입력 데이터를 더하는 조합논리회로임
  - 반가산기(HA: **half adder**)와 전가산기(FA: **full adder**)의 2개가 있음
- 반가산기(half-adder)
  - 한 자리 2진수 2개를 입력하여 합(sum: S)과 캐리(carry: C)를 계산하는 덧셈회로임

A	0	0	1	1
$+B$	$+0$	$+1$	$+0$	$+1$
$C\ S$	0 0	0 1	0 1	1 1

가산기(Adder)

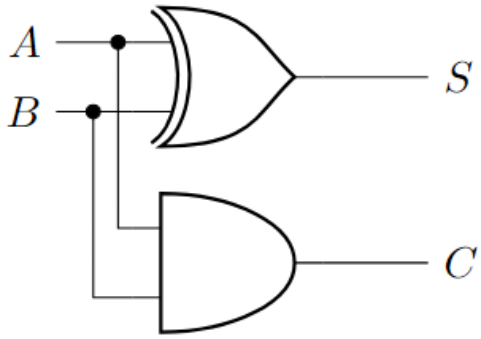
- 반가산기(half adder)

진리표

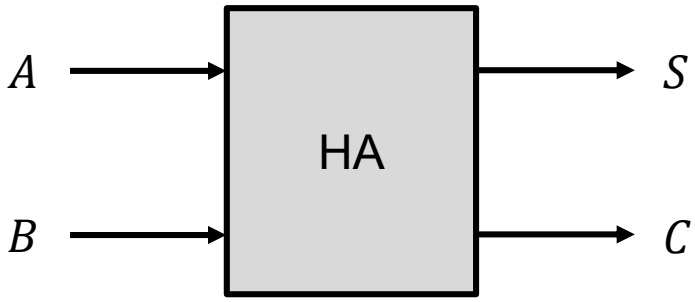
입력		출력	
A	***B***	***S***	***C***
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$S = \bar{A}B + A\bar{B} = A \oplus B$$
$$C = AB$$

논리회로



논리 기호



가산기(Adder)

- 전가산기(full adder)
  - 반가산기는 아랫자리에서 발생한 캐리를 고려하지 못하므로 2비트 이상의 2진수 덧셈은 할 수 없음
  - 전가산기는 2개의 2진수 입력 A, B와 아랫자리부터 올라온 캐리 C_i를 포함하여 한 자리 2진수 3개를 더하는 조합회로임

C_i	0	1	0	1	0	1	0	1
A	0	0	1	1	0	0	1	1
$+B$	$+0$	$+0$	$+0$	$+0$	$+1$	$+1$	$+1$	$+1$
$C_o\ S$	0 0	0 1	0 1	1 0	0 1	1 0	1 0	1 1

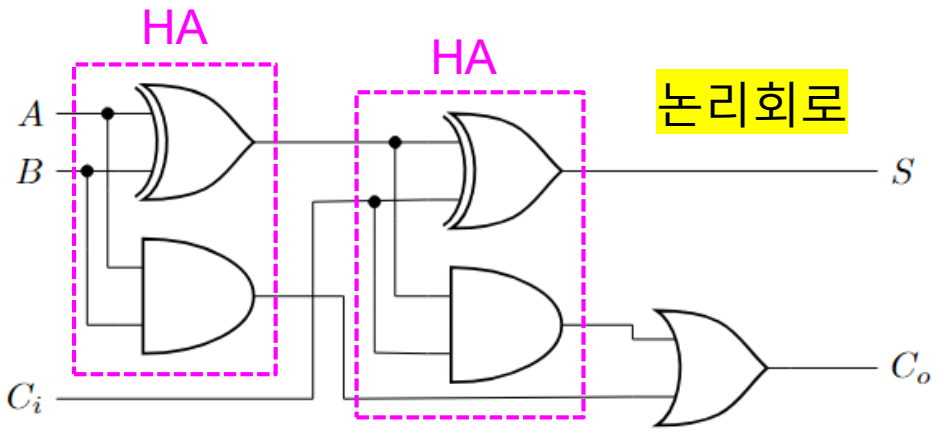
가산기(Adder)

- 전가산기(full adder)

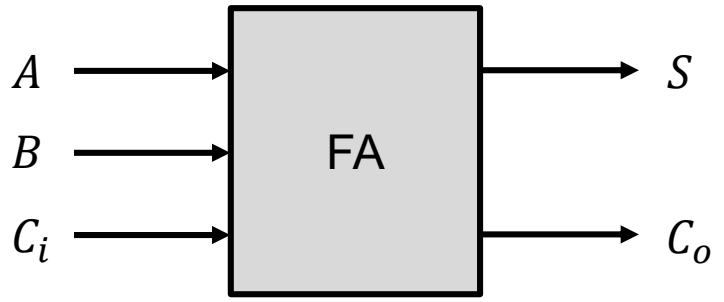
진리표

입력			출력	
A	B	C_i	S	C_o
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$S = A \oplus B \oplus C_i$$
$$C_o = C_i(A \oplus B) + AB$$

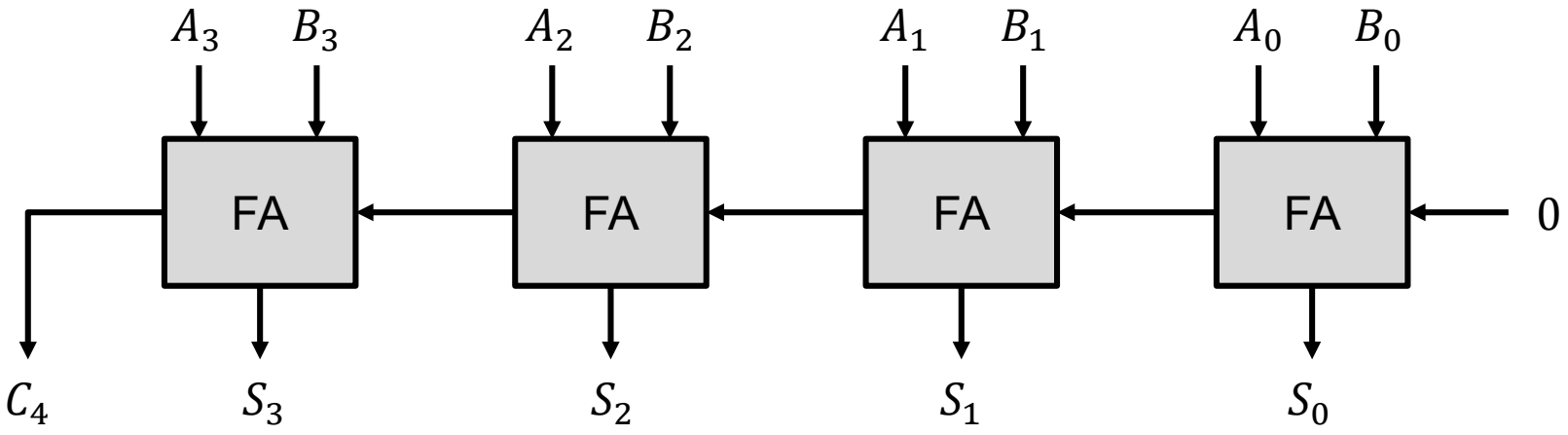


논리 기호



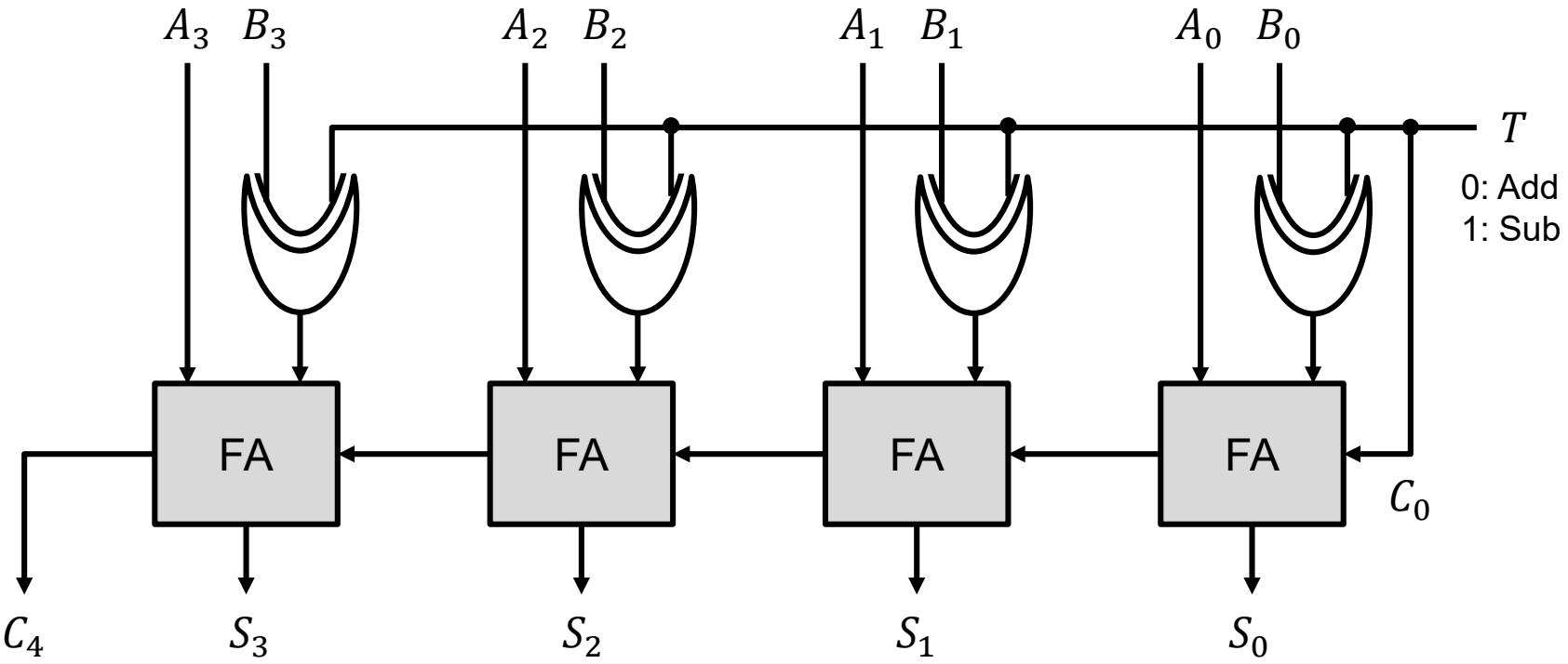
가산기(Adder)

- 병렬 가산기(parallel adder)
  - 전가산기 여러 개를 병렬로 연결하여 2비트 이상인 가산기를 만들 수 있으며, 이를 병렬 가산기라고 함
  - 예, $A = A_3A_2A_1A_0$와 $B = B_3B_2B_1B_0$가 각각 4비트이며, 전가산기의 최하위 비트의 캐리는 0을 입력하며, 계산 결과 합은 $S = S_3S_2S_1S_0$이며, 마지막 캐리는 C_4임



가산기(Adder)

- 병렬 가감산기(parallel adder/subtractor)
  - $T = 0 \Rightarrow S = A + (B \oplus 0) + 0 = A + B$
  - $T = 1 \Rightarrow S = A + (B \oplus 1) + 1 = A + \bar{B} + 1 = A - B$



가산기(Adder)

- 병렬 가산기의 속도가 느림
  - 아랫단 가산기에서 캐리가 나와야만 바로 윗단을 계산 할 수 있기 때문임
  - 리플-캐리(ripple carry)라고 함
- 고속 가산기
  - 캐리 예측 가산기(**CLA: carry look-ahead adder**)를 사용함
  - i단에서 발생하는 캐리의 논리식은 다음과 같음

$$C_{i+1} = A_iB_i + (A_i + B_i)C_i \quad \text{또는} \quad C_{i+1} = A_iB_i + (A_i \oplus B_i)C_i$$

C_i	0	1	0	1	0	1	0	1
A_i	0	0	1	1	0	0	1	1
$+ B_i$	$+ 0$	$+ 0$	$+ 0$	$+ 0$	$+ 1$	$+ 1$	$+ 1$	$+ 1$
$C_{i+1}\ S_i$	0 0	0 1	0 1	1 0	0 1	1 0	1 0	1 1

가산기(Adder)

- 고속 가산기
  - i단에서 발생하는 캐리의 논리식은 다음과 같음

$$C_{i+1} = A_iB_i + (A_i + B_i)C_i \quad \text{또는} \quad C_{i+1} = A_iB_i + (A_i \oplus B_i)C_i$$

  - Propagate : $P_i = A_i \oplus B_i$ 또는 $P_i = A_i + B_i$
  - Generate : $G_i = A_iB_i$
  - 예, 4비트 가산기

$$C_1 = G_0 + P_0C_0$$

$$C_2 = G_1 + P_1C_1 = G_1 + P_1(G_0 + P_0C_0) = G_1 + P_1G_0 + P_1P_0C_0$$

$$C_3 = G_2 + P_2C_2 = G_2 + P_2(G_1 + P_1G_0 + P_1P_0C_0) = G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C_0$$

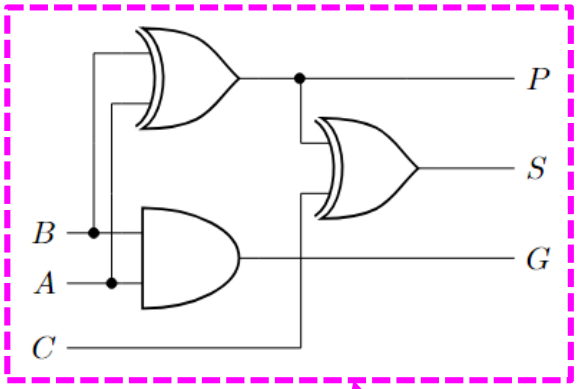
$$C_4 = G_3 + P_3C_3 = G_3 + P_3(G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C_0) = G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0 + P_3P_2P_1P_0C_0$$

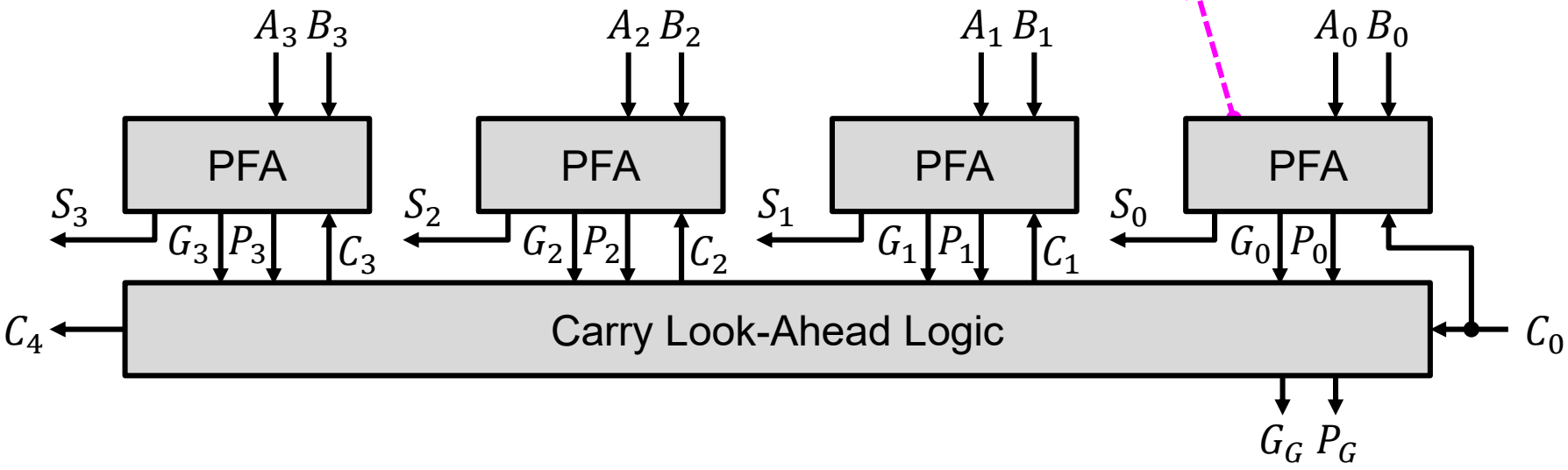
$$S_i = A_i \oplus B_i \oplus C_i = P_i \oplus C_i$$

가산기(Adder)

- 고속 가산기
  - 예, 4비트 가산기

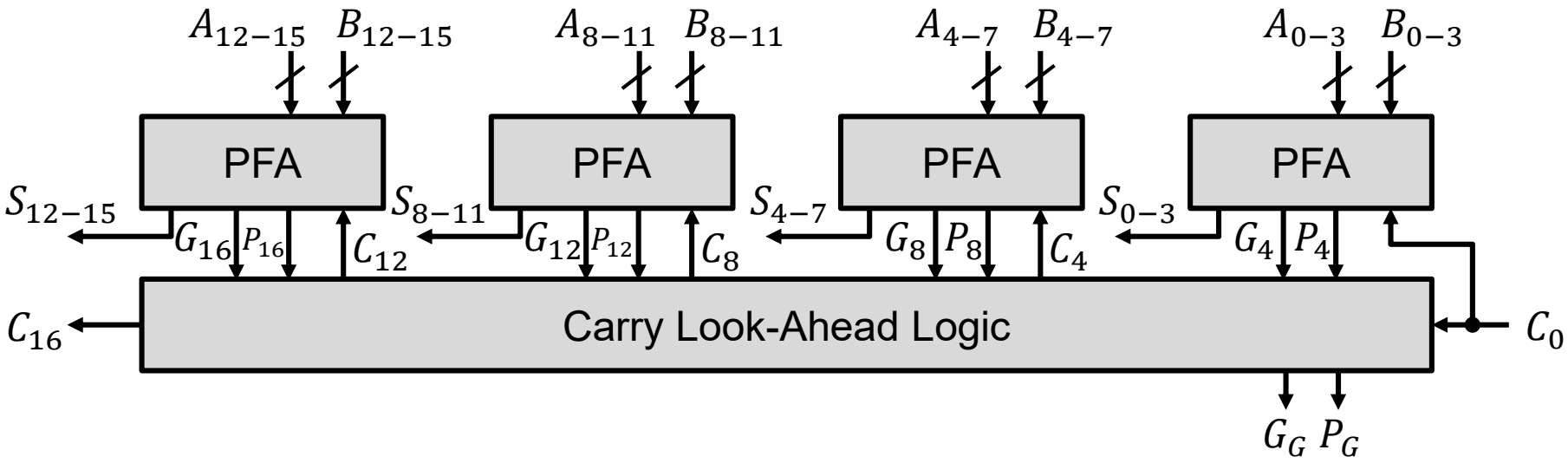
Partial Full Adder (PFA)





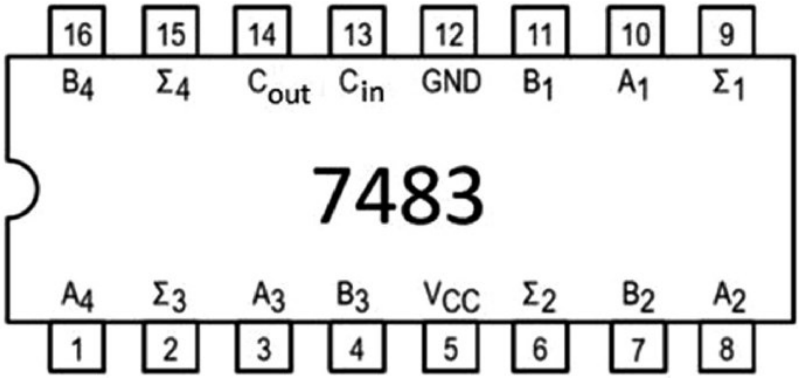
가산기(Adder)

- 고속 가산기
  - 예, 4비트 캐리 예측 가산기로부터 16비트 캐리 예측 가신기를 말들 수 있음



가산기(Adder)

- 고속 가산기
  - IC 7483 : 4비트 캐리 예측 가산기



Source: STMicroelectronics, *Datasheet*, https://www.alldatasheet.com/datasheet-pdf/view/23067/ (accessed on 2024.08.23).

가산기(Adder)

- BCD(**Binary Coded Decimal**) 가산기
  - BCD는 2진수를 10진수로 변환할 때 많이 쓰임
  - RTC(Real-Time Clock) 칩에서 많이 활용됨
  - BCD 코드 표현 범위가 0에서 9까지

BCD코드				10진값
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9

가산기(Adder)

- BCD(**Binary Coded Decimal**) 가산기

2진합					BCD합					10진값
K	Z_3	Z_2	Z_1	Z_0	C	S_3	S_2	S_1	S_0	
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	2
0	0	0	1	1	0	0	0	1	1	3
0	0	1	0	0	0	0	1	0	0	4
0	0	1	0	1	0	0	1	0	1	5
0	0	1	1	0	0	0	1	1	0	6
0	0	1	1	1	0	0	1	1	1	7
0	1	0	0	0	0	1	0	0	0	8
0	1	0	0	1	0	1	0	0	1	9

2진합					BCD합					10진값
K	Z_3	Z_2	Z_1	Z_0	C	S_3	S_2	S_1	S_0	
0	1	0	1	0	1	0	0	0	0	10
0	1	0	1	1	1	0	0	0	1	11
0	1	1	0	0	1	0	0	1	0	12
0	1	1	0	1	1	0	0	1	1	13
0	1	1	1	0	1	0	1	0	0	14
0	1	1	1	1	1	0	1	0	1	15
1	0	0	0	0	1	0	1	1	0	16
1	0	0	0	1	1	0	1	1	1	17
1	0	0	1	0	1	1	0	0	0	18
1	0	0	1	1	1	1	0	0	1	19

가산기(Adder)

- BCD(**Binary Coded Decimal**) 가산기
  - BCD합의 캐리 비트 C는 K가 1일 때 항상 1이 되며, 2진합 (Z_3, Z_2, Z_1, Z_0)에도 기반함

$$C = K + F(Z_3, Z_2, Z_1, Z_0) = K + Z_3Z_2 + Z_3Z_1$$

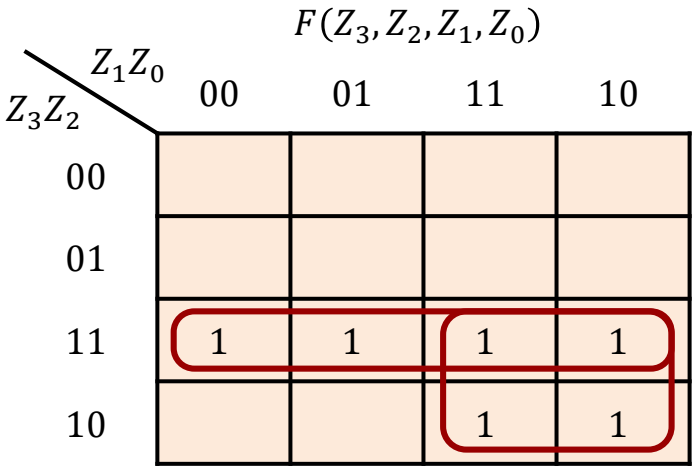
  - BCD 연산 offset
    - 예,

$6 + 7 = 0110_{(2)} + 0111_{(2)} = 0\ 1101_{(2)}$

$0\ 1101_{(2)}$결과는 BCD가 아니므로 BCD로 만들어야 함

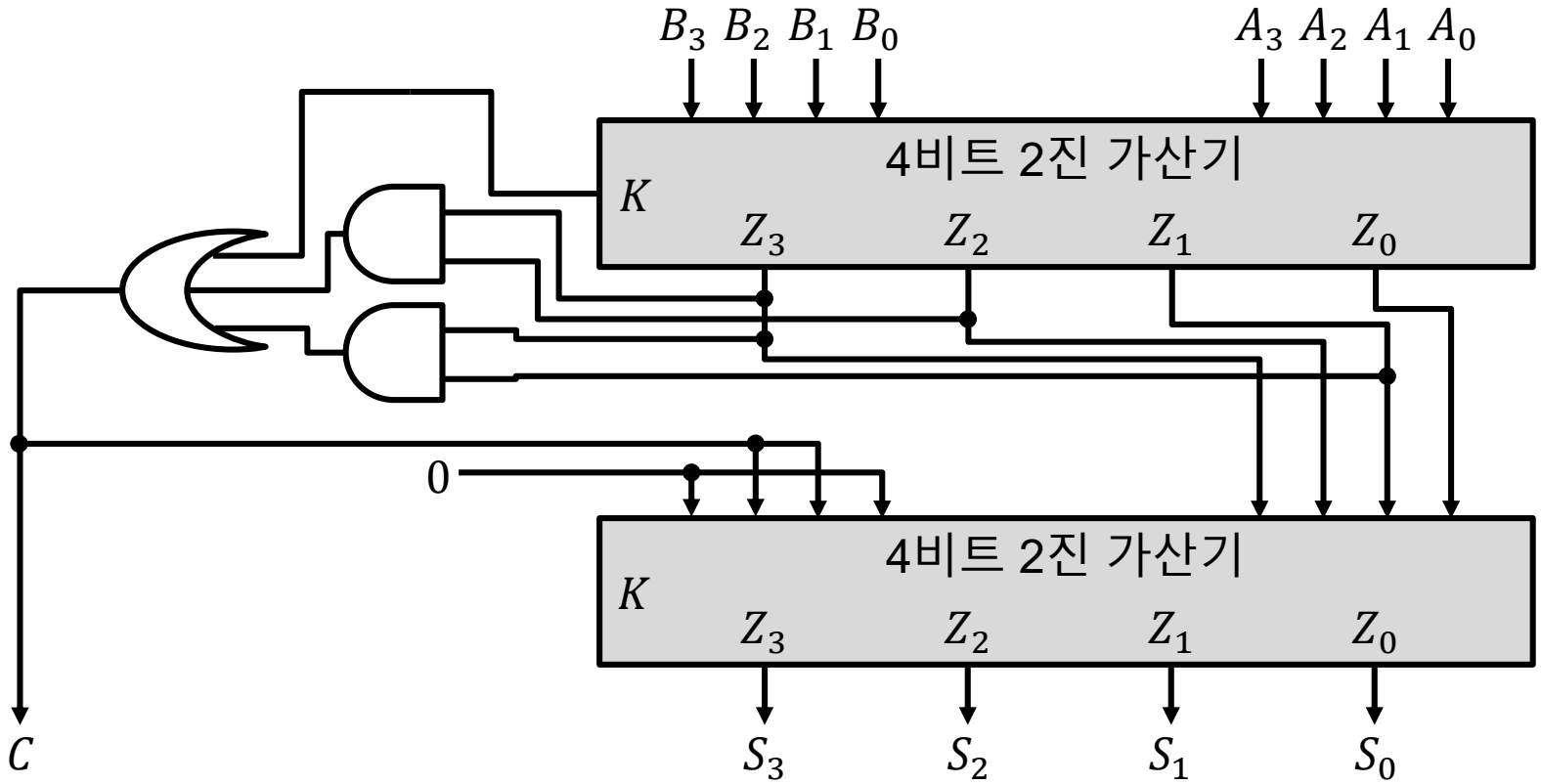
BCD 표현을 위해 4비트를 사용하지만 BCD 범위가 0부터 9까지 때문에 $0110_{(2)} = 6$인 offset를 더해주면 됨

따라서 $0\ 1101_{(2)} + 0110_{(2)} = 1\ 0011_{(2)}$이 됨



가산기(Adder)

- BCD(**Binary Coded Decimal**) 가산기



가산기(Adder)

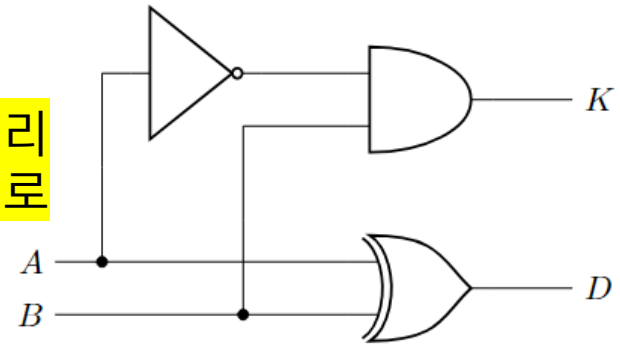
- 반감산기(half subtractor)
  - 한 자리 2진수 2개를 입력하여 차(difference, D)와 빌림 수(borrow, K)를 계산하는 뺄셈 회로임

$$\begin{array}{r} A \\ -\,B \\ \hline K\ D \end{array} \qquad \begin{array}{r} 0 \\ -\,0 \\ \hline 0\ 0 \end{array} \qquad \begin{array}{r} 0 \\ -\,1 \\ \hline 1\ 1 \end{array} \qquad \begin{array}{r} 1 \\ -\,0 \\ \hline 0\ 1 \end{array} \qquad \begin{array}{r} 1 \\ -\,1 \\ \hline 0\ 0 \end{array}$$

입력		출력	
A	***B***	***D***	***K***
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$$D = \bar{A}B + A\bar{B} = A \oplus B$$
$$K = \bar{A}B$$

논리 회로

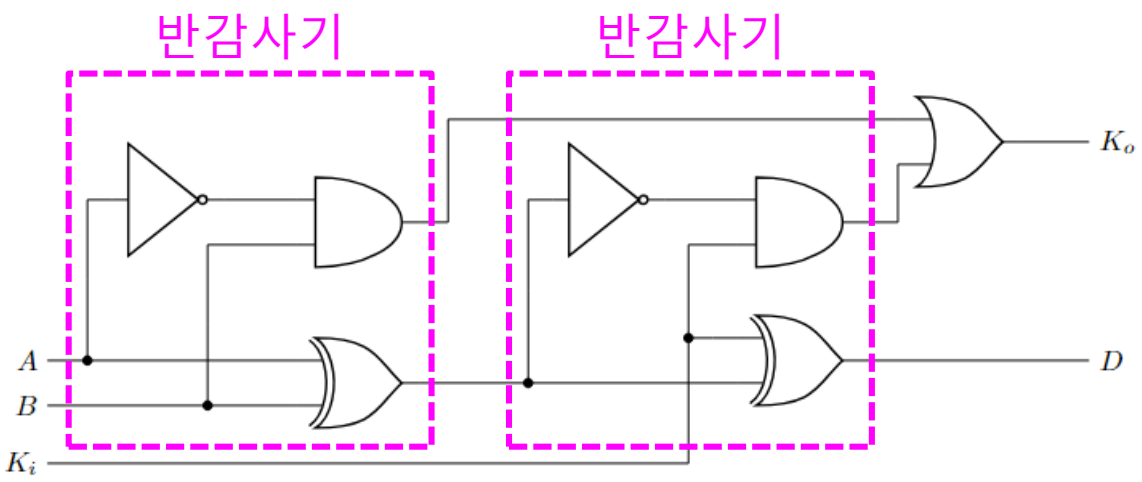


가산기(Adder)

- 전감산기(full subtractor)
  - 2비트 이상의 2진수 입력 A, B와 아래 단으로 빌려주는 수 K_i를 포함하여 $A - B - K_i$를 계산하는 조합회로임

입력			출력	
A	B	K_i	D	K_o
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

$$D = A \oplus B \oplus K_i$$
$$K_o = (\overline{A \oplus B})K_i + \bar{A}B$$



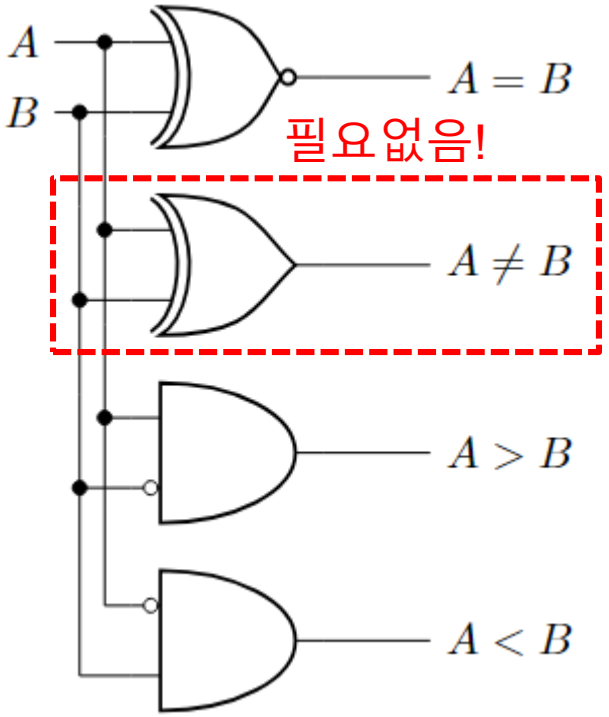
비교기(Comparator)

- 비교기는 두 2진수 값의 크기를 비교하는 회로임

입력		출력			
A	B	$A = B$ F_1	$A \neq B$ F_2	$A > B$ F_3	$A < B$ F_4
0	0	1	0	0	0
0	1	0	1	0	1
1	0	0	1	1	0
1	1	1	0	0	0

$F_1 = \overline{A \oplus B}$ $\quad F_3 = A\bar{B}$

$F_2 = A \oplus B$ $\quad F_4 = \bar{A}B$

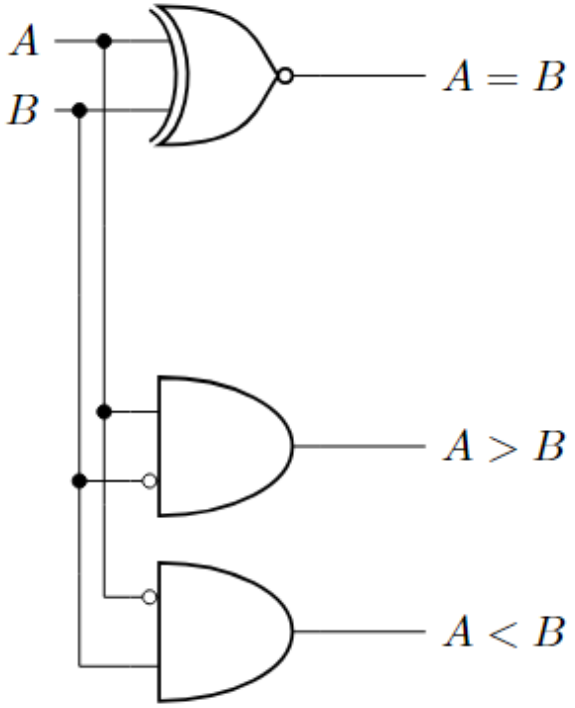


비교기(Comparator)

- 비교기는 두 2진수 값의 크기를 비교하는 회로임

입력		출력		
A	B	$A = B$ F_1	$A > B$ F_3	$A < B$ F_4
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

$F_1 = \overline{A \oplus B}$ $F_2 = A\bar{B}$ $F_3 = \bar{A}B$



비교기(Comparator)

- 2비트 비교기

입력				출력		
A_1, B_1	$AGBI$ $A_0 > B_0$	$ALBI$ $A_0 < B_0$	$AEBI$ $A_0 = B_0$	$AGBO$ $A > B$	$ALBO$ $A < B$	$AEBO$ $A = B$
$A_1 > B_1$	×	×	×	1	0	0
$A_1 < B_1$	×	×	×	0	1	0
$A_1 = B_1$	1	0	0	1	0	0
$A_1 = B_1$	0	1	0	0	1	0
$A_1 = B_1$	0	0	1	0	0	1
$A_1 = B_1$	0	0	0	1	1	0
$A_1 = B_1$	0	1	1	0	0	1
$A_1 = B_1$	1	0	1	0	0	1
$A_1 = B_1$	1	1	0	0	0	0
$A_1 = B_1$	1	1	1	0	0	1

- 회로가 정상 동작하면 발생하지 않은 경우
- 비정상적으로 동작하면 사용자에게 알려주기 위한 출력 패턴을 제조사가 정의함
- 여기서 이러한 경우들을 고려하지 않음

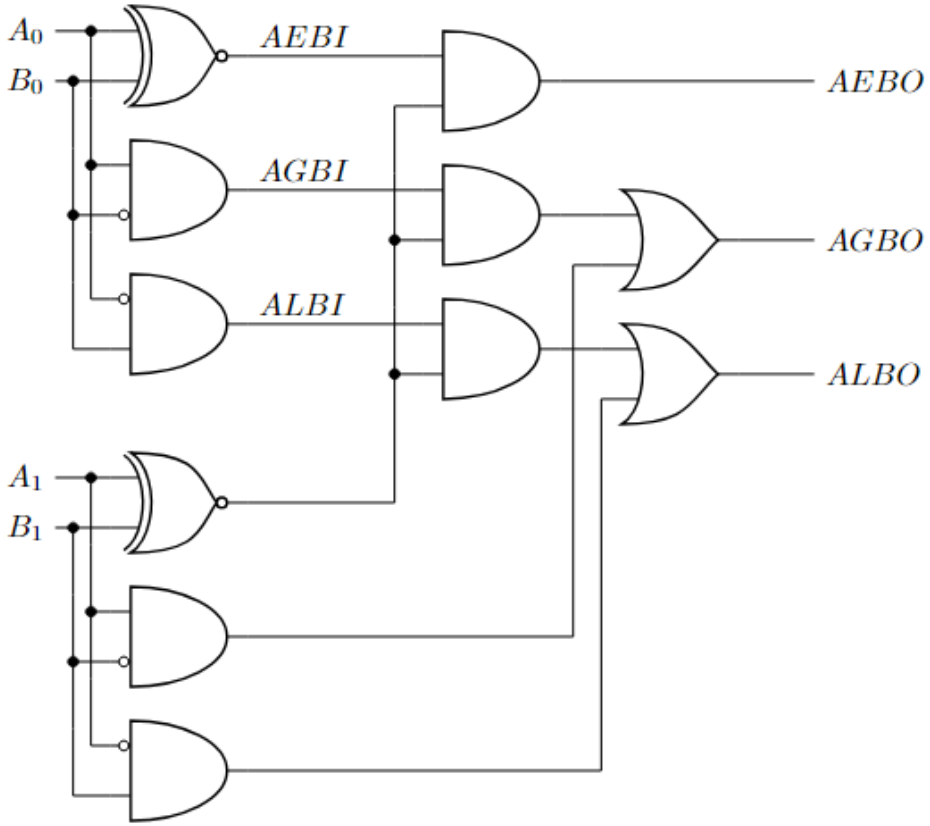
비교기(Comparator)

- 2비트 비교기

$$AEBO = AEBI \cdot (A_1 = B_1) = AEBI \cdot (\overline{A_1 \oplus B_1})$$

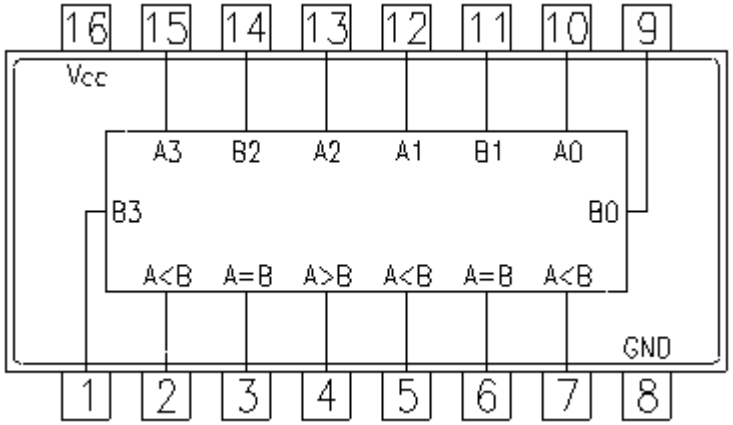
$$AGBO = AGBI \cdot (A_1 = B_1) + (A_1 > B_1)$$
$$= AGBI \cdot (\overline{A_1 \oplus B_1}) + A_1\overline{B_1}$$

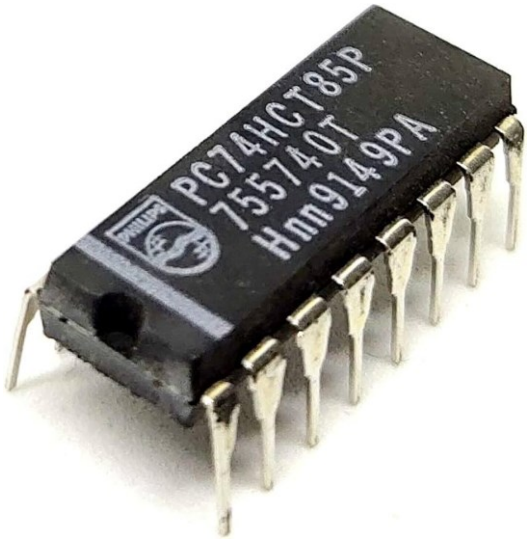
$$ALBO = ALBI \cdot (A_1 = B_1) + (A_1 < B_1)$$
$$= ALBI \cdot (\overline{A_1 \oplus B_1}) + \overline{A_1}B_1$$



비교기(Comparator)

- 4비트 비교기 : IC 7485

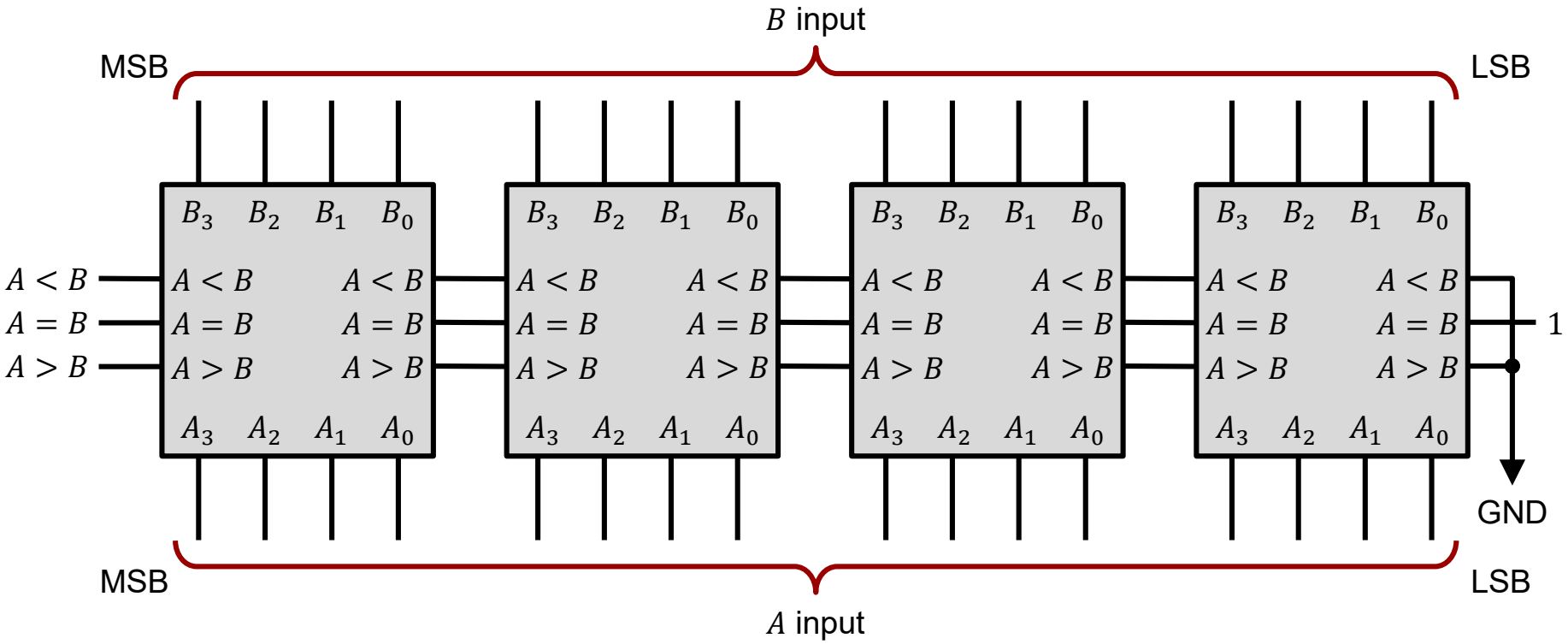




Source: Texas Instruments, *Datasheet*, https://www.alldatasheet.com/datasheet-pdf/view/833263/ (accessed on 2024.08.23).

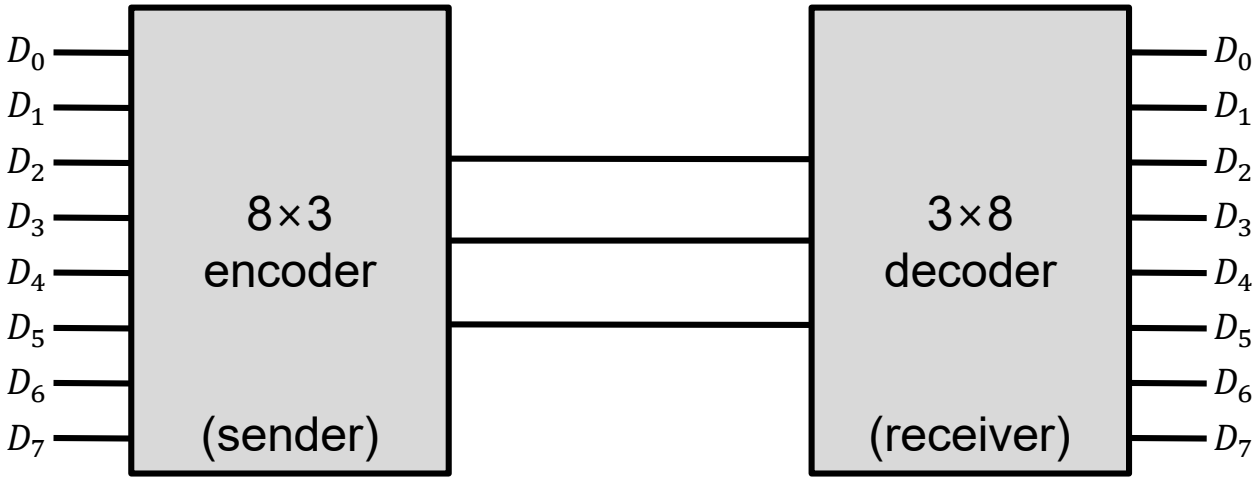
비교기(Comparator)

- 4비트 비교기로부터 16비트 비교기 만들기



디코더(Decoder)

- n비트로 된 2진 코드는 서로 다른 2^n개의 정보를 표현할 수 있음
- 디코더는 입력선에 나타나는 n비트 2진 코드를 최대 2^n가지 정보로 바꿔주는 조합논리회로임



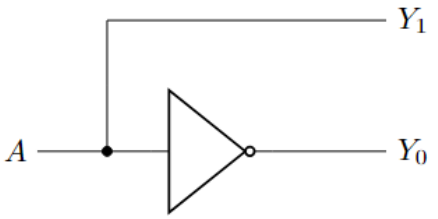
디코더(Decoder)

- 1×2 디코더
  - 입력 1개와 출력 2개로 구성됨
  - 인에이블(enable) 신호를 포함할 수 있음

입력	출력	
A	Y_1	Y_o
0	0	1
0	1	0

$$Y_0 = \bar{A}$$

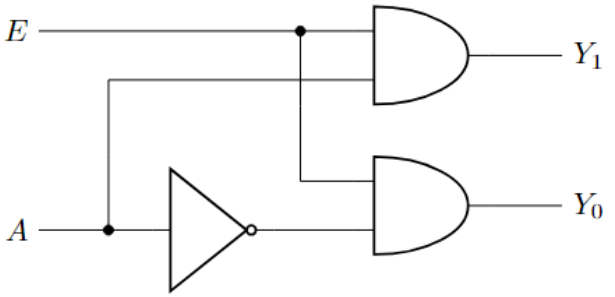
$$Y_1 = A$$



입력		출력	
E	A	Y_1	Y_o
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	0

$$Y_0 = E\bar{A}$$

$$Y_1 = EA$$



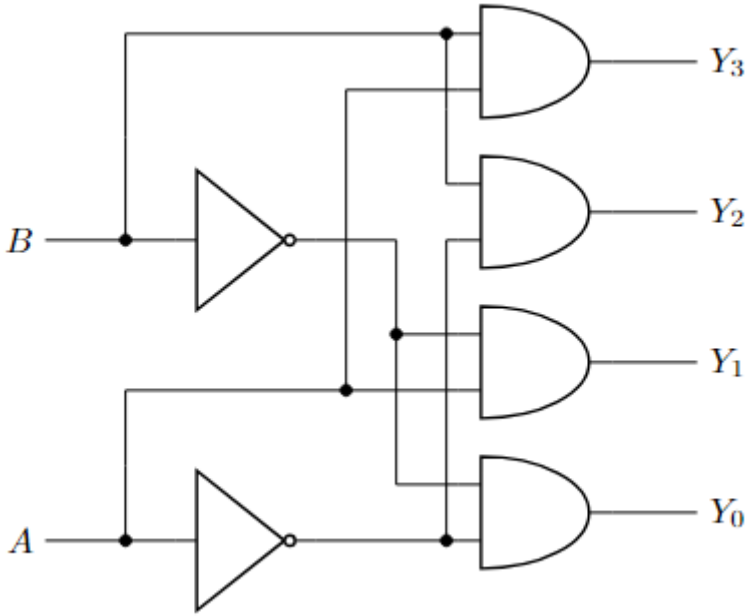
디코더(Decoder)

- 2×4 디코더
  - 입력 2개와 출력 4개로 구성됨
  - 인에이블(enable) 신호를 포함할 수 있음

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

$$Y_3 = BA \qquad Y_2 = B\bar{A}$$

$$Y_1 = \bar{B}A \qquad Y_0 = \bar{B}\bar{A}$$



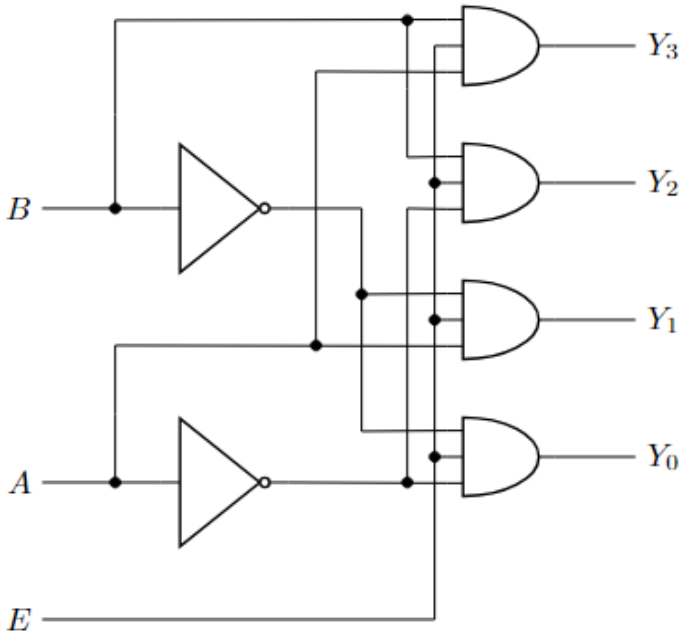
디코더(Decoder)

- 2×4 디코더
  - 입력 2개와 출력 4개로 구성됨
  - 인에이블(enable) 신호를 포함할 수 있음

입력			출력			
E	B	A	Y_3	Y_2	Y_1	Y_o
0	×	×	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

$$Y_3 = EBA \qquad Y_2 = EB\bar{A}$$

$$Y_1 = E\bar{B}A \qquad Y_0 = E\bar{B}\bar{A}$$



디코더(Decoder)

- 3 × 8 디코더
  - 입력 3개와 출력 8개로 구성됨
  - 인에이블(enable) 신호를 포함할 수 있음

입력			출력							
C	B	A	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_o
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

디코더(Decoder)

- 3×8 디코더
  - 입력 3개와 출력 8개로 구성됨
  - 인에이블(enable) 신호를 포함할 수 있음

2×4 디코더
입력 A, B와 출력 Y_{3-0}
인에이블 신호 $\bar{C}$

입력			출력							
C	B	A	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_o
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

디코더(Decoder)

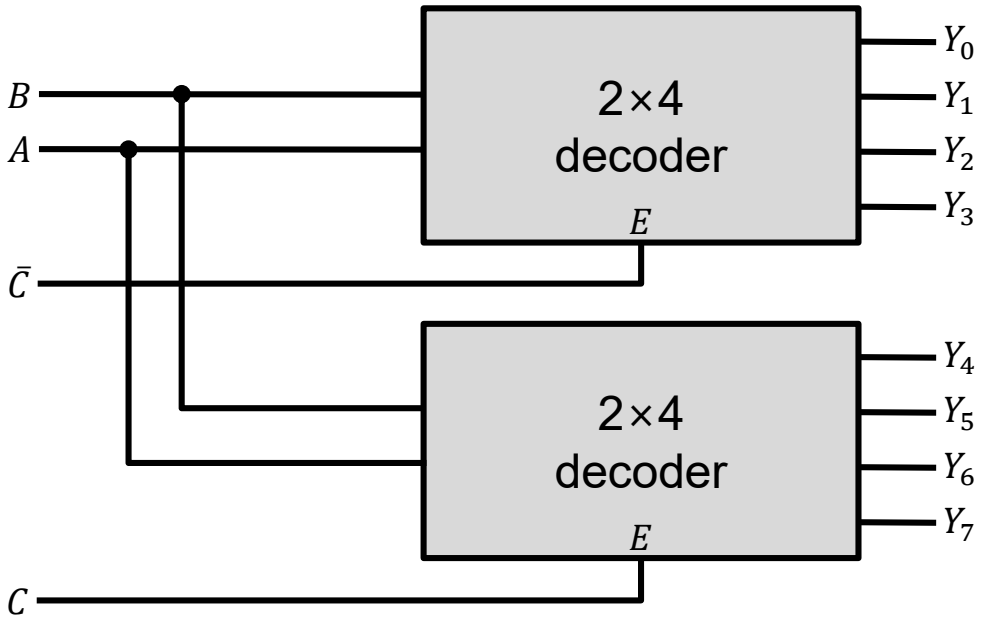
- 3×8 디코더
  - 입력 3개와 출력 8개로 구성됨
  - 인에이블(enable) 신호를 포함할 수 있음

2×4 디코더
입력 A, B와 출력 Y_{7-4}
인에이블 신호 C

입력			출력							
C	B	A	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_o
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

디코더(Decoder)

- 3 × 8 디코더
  - 입력 3개와 출력 8개로 구성됨
  - 인에이블(enable) 신호를 포함할 수 있음



디코더(Decoder)

- 디코더를 이용한 조합논리회로
  - 디코더의 출력을 최소항으로 볼 수 있으므로 원하는 조합논리회로를 쉽게 구성할 수 있음

입력		출력				최소항/ 최고항
B	A	Y_3	Y_2	Y_1	Y_o	
0	0	0	0	0	1	$Y_0 = \bar{B}\bar{A} = m_0 = \overline{M_0}$
0	1	0	0	1	0	$Y_1 = \bar{B}A = m_1 = \overline{M_1}$
1	0	0	1	0	0	$Y_2 = B\bar{A} = m_2 = \overline{M_2}$
1	1	1	0	0	0	$Y_3 = BA = m_3 = \overline{M_3}$

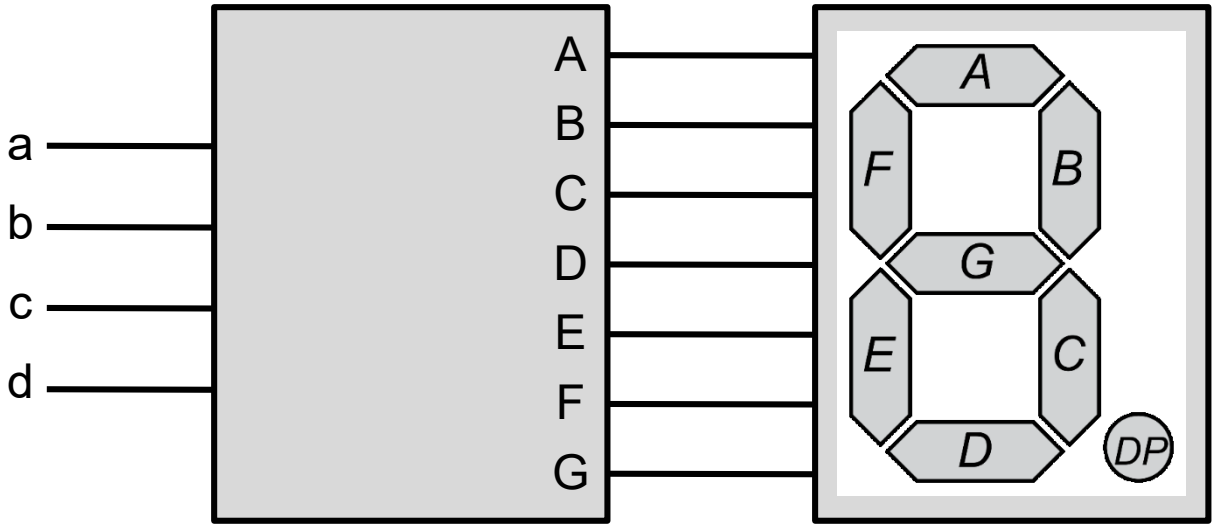
  - 예,

$$F_1 = \sum m(0,2,3) = m_0 + m_2 + m_3 = Y_0 + Y_2 + Y_3$$

$$F_2 = \prod M(1,3) = M_1 M_3 = \overline{\overline{M_1 M_3}} = \overline{\overline{M_1} + \overline{M_3}} = \overline{m_1 + m_3}$$

디코더(Decoder)

- BCD-7-세그먼트 디코더



디코더(Decoder)

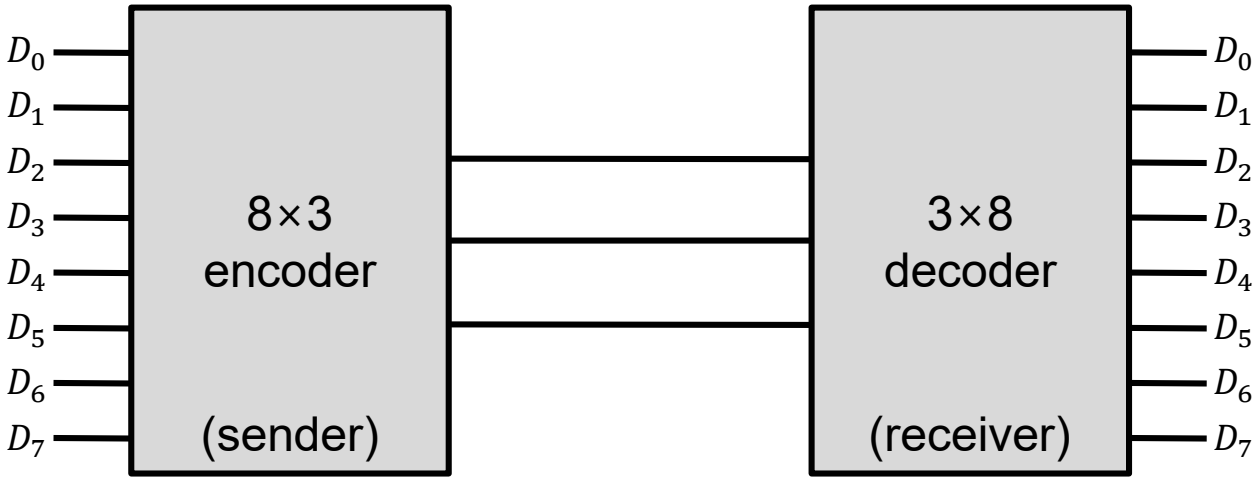
- BCD-7-세그먼트 디코더

입력				출력						
a	b	c	d	A	B	C	D	E	F	G
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	0	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0

입력				출력						
a	b	c	d	A	B	C	D	E	F	G
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	0	0	1	1
1	0	1	0	X	X	X	X	X	X	X
1	0	1	1	X	X	X	X	X	X	X
1	1	0	0	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X

인코더(Encoder)

- 인코더는 신호 2^n개를 입력 받아 출력 신호 n개를 만들어 내는 조합논리회로임
- 2^n개 중 활성화된 1비트 입력 신호를 받아서 그 숫자에 해당하는 n비트 2진 정보를 출력함

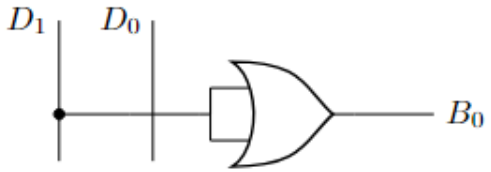


인코더(Encoder)

- 2×1 인코더 : 입력 2개와 출력 1개로 구성됨
- 4×2 인코더 : 입력 4개와 출력 2개로 구성됨

입력		출력
D_1	D_0	B_o
0	1	0
1	0	1

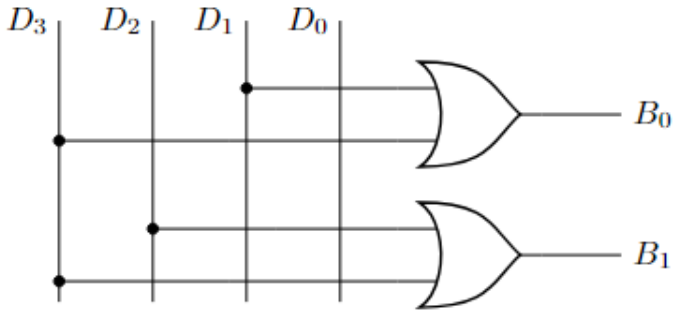
$$B_0 = D_1$$



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

$$B_1 = D_2 + D_3$$

$$B_0 = D_1 + D_3$$



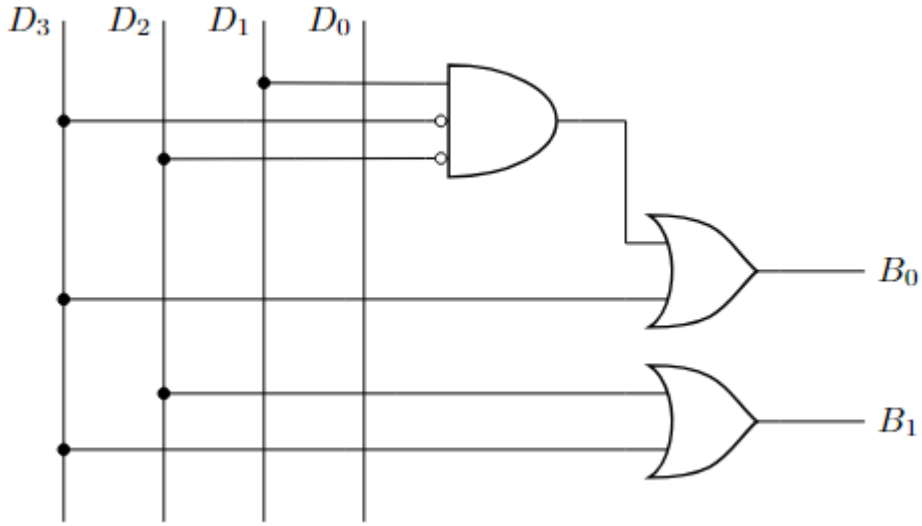
인코더(Encoder)

- 우선순위 인코더(**priority encoder**)
  - 입력에 우선순위를 정하여 입력이 여러 개 있을 때 우선순위가 높은 입력 값에 해당하는 출력 신호를 만들어내는 조합논리회로임

입력				출력	
D_3	D_2	D_1	D_0	B_1	B_o
0	0	0	1	0	0
0	0	1	X	0	1
0	1	X	X	1	0
1	X	X	X	1	1

$$B_1 = D_3 + D_2$$

$$B_0 = D_3 + \overline{D}_3\overline{D}_2D_1$$

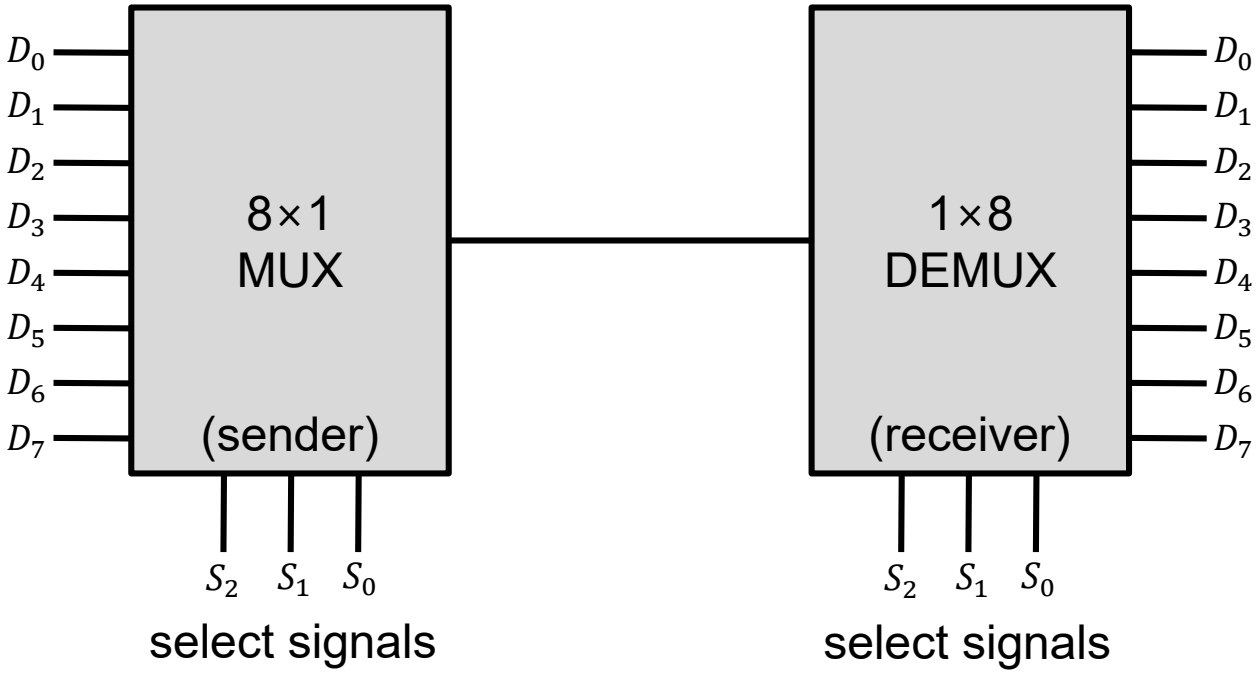


멀티플렉서(Multiplexer)

- 멀티플렉서
  - 여러 개의 입력선 중에서 하나를 선택하여 출력선에 연결하는 조합논리회로이며, 선택선의 값에 따라 입력선을 선택함
  - 데이터 선택기(data selector)라고도 함
- 디멀티플렉서(demultiplexer)
  - 멀티플렉서와 반대로 정보를 한 선으로 받아서 2^n개의 가능한 출력선들 중 하나를 선택하여 받은 정보를 전송하는 조합논리회로임
  - 데이터 분배기(data distributor)라고도 함

멀티플렉서(Multiplexer)

- 멀티플렉서 ⇔ MUX(먹스)
- 디멀티플렉서 ⇔ DEMUX(디먹스)

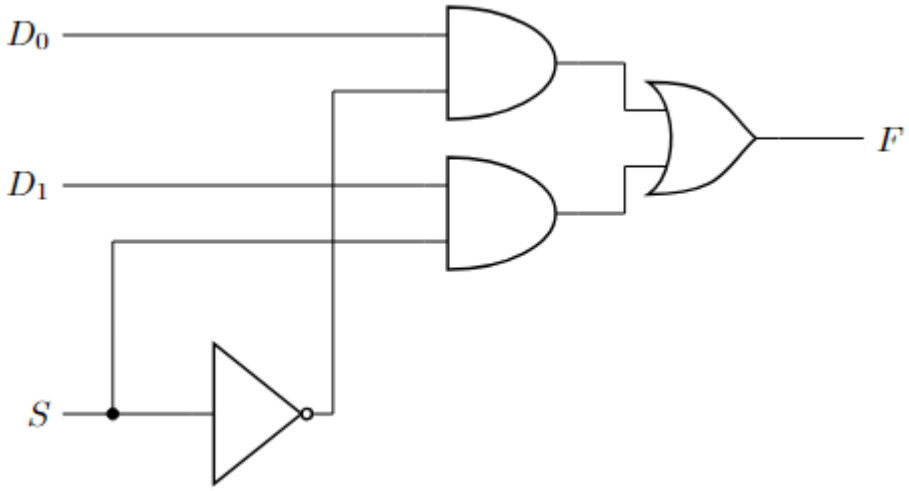


멀티플렉서(Multiplexer)

- 2×1 멀티플렉서
  - 입력 2개 중 하나를 선택선 S에 입력된 값에 따라 출력으로 보내주는 조합 논리회로임

선택선	출력
S	F
0	D_0
1	D_1

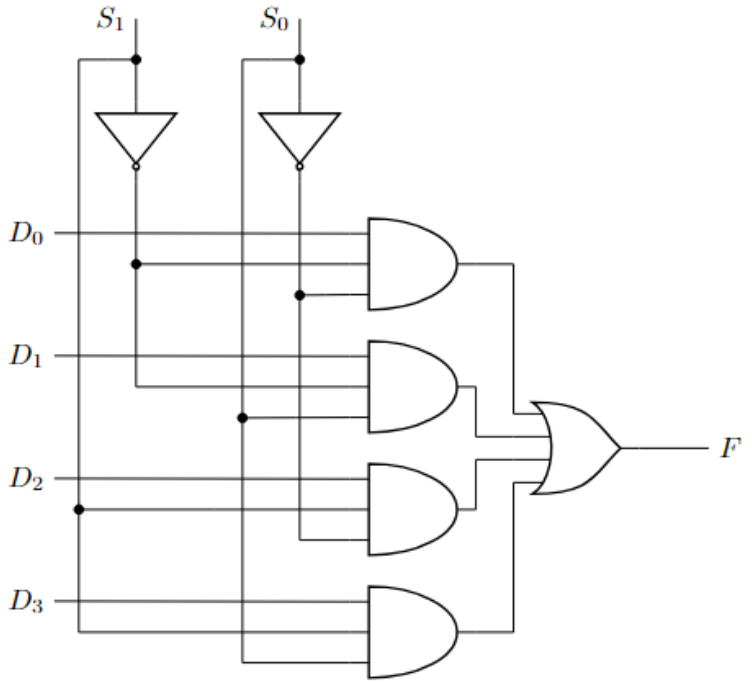
$$F = \bar{S}D_0 + SD_1$$



멀티플렉서(Multiplexer)

- 4×1 멀티플렉서
  - 입력 4개 중 하나를 선택선 S_1과 S_0에 입력된 값에 따라 출력으로 보내주는 조합논리회로임

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



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

멀티플렉서(Multiplexer)

- 8×1 멀티플렉서
  - 입력 8개 중 하나를 3개의 선택선 (S_2, S_1, S_0)에 입력된 값에 따라 출력으로 보내주는 조합논리회로임

선택선			출력
S_2	S_1	S_0	F
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3
1	0	0	D_4
1	0	1	D_5
1	1	0	D_6
1	1	1	D_7

멀티플렉서(Multiplexer)

- 8×1 멀티플렉서
  - 입력 8개 중 하나를 3개의 선택선 (S_2, S_1, S_0)에 입력된 값에 따라 출력으로 보내주는 조합논리회로임

선택선			출력
S_2	S_1	S_0	F
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3
1	0	0	D_4
1	0	1	D_5
1	1	0	D_6
1	1	1	D_7

4×1 멀티플렉서
입력 (D_3, D_2, D_1, D_0)
선택선 (S_1, S_0)
출력 F_0

멀티플렉서(Multiplexer)

- 8×1 멀티플렉서
  - 입력 8개 중 하나를 3개의 선택선 (S_2, S_1, S_0)에 입력된 값에 따라 출력으로 보내주는 조합논리회로임

선택선			출력
S_2	S_1	S_0	F
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3
1	0	0	D_4
1	0	1	D_5
1	1	0	D_6
1	1	1	D_7

4×1 멀티플렉서
입력 (D_7, D_6, D_5, D_4)
선택선 (S_1, S_0)
출력 F_1

멀티플렉서(Multiplexer)

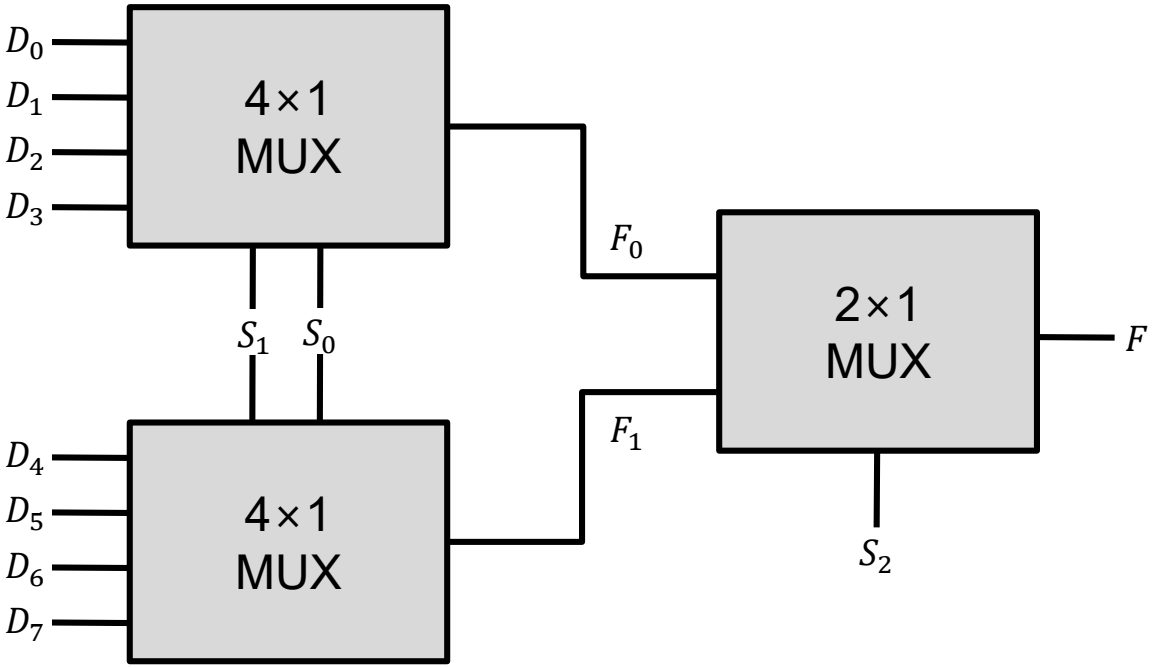
- 8×1 멀티플렉서
  - 입력 8개 중 하나를 3개의 선택선 (S_2, S_1, S_0)에 입력된 값에 따라 출력으로 보내주는 조합논리회로임

선택선			출력
S_2	S_1	S_0	F
0	0	0	F_0
0	0	1	
0	1	0	
0	1	1	
1	0	0	F_1
1	0	1	
1	1	0	
1	1	1	

2×1 멀티플렉서
입력 (F_1, F_0)
선택선 S_2
출력 F

멀티플렉서(Multiplexer)

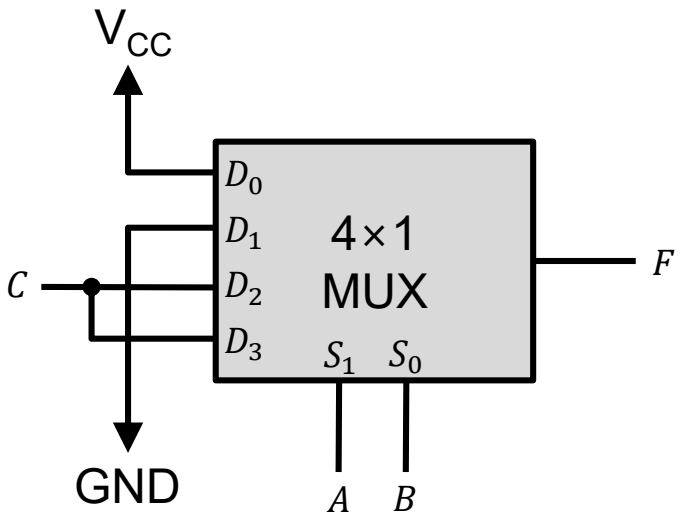
- 8×1 멀티플렉서
  - 입력 8개 중 하나를 3개의 선택선 (S_2, S_1, S_0)에 입력된 값에 따라 출력으로 보내주는 조합논리회로임



멀티플렉서(Multiplexer)

- 멀티플렉서를 이용한 조합논리회로 구현
  - 선택선과 입력선의 일부를 입력으로 사용하고 입력선을 V_{CC}(5V 또는 3.3V)나 GND(0V)와 연결하면 조합논리회로를 구현할 수 있음
  - 예, $F(A,B,C) = \sum m(0,1,5,7)$

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



코드 변환기

- 2진 코드 – 그레이 코드 변환

2진 코드(입력)				그레이 코드(출력)			
B_3	B_2	B_1	B_0	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

2진 코드(입력)				그레이 코드(출력)			
B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_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

코드 변환기

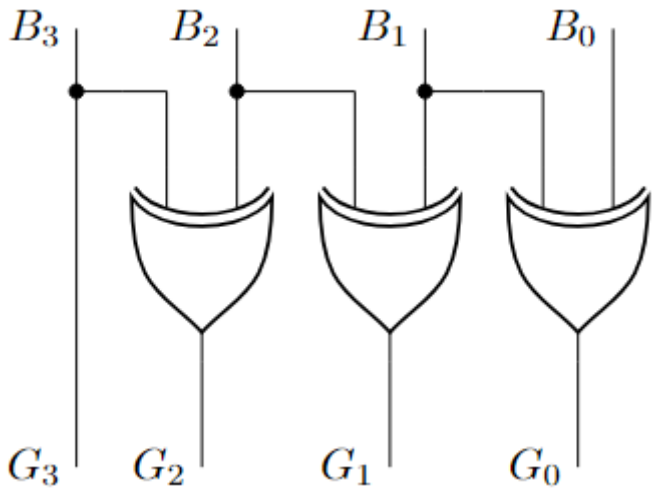
- 2진 코드 – 그레이 코드 변환

$$G_3 = B_3$$

$$G_2 = B_3 \oplus B_2$$

$$G_1 = B_2 \oplus B_1$$

$$G_0 = B_1 \oplus B_0$$



코드 변환기

- 그레이 코드 – 2진 코드 변환

그레이 코드(입력)				2진 코드(출력)			
G_3	G_2	G_1	G_0	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

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

코드 변환기

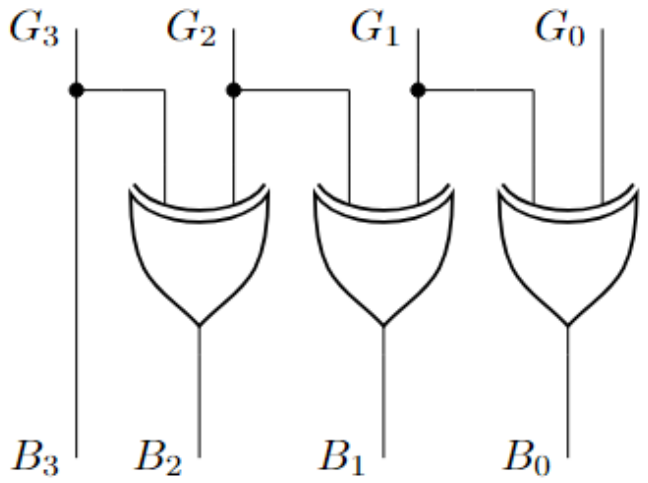
- 그레이 코드 – 2진 코드 변환

$$B_3 = G_3$$

$$B_2 = G_3 \oplus G_2$$

$$B_1 = G_2 \oplus G_1$$

$$B_0 = G_1 \oplus G_0$$



패리티 발생기/검출기

- 패리티(parity) 발행
  - 원래의 데이터에 1비트 패리티를 추가하여 1의 개수를 짝수 또는 홀수로 맞춤
  - 1의 개수를 짝수로 맞추면 **짝수 패리티**, 홀수로 맞추면 **홀수 패리티**가 됨
- 패리티 검출
  - 패리티가 추가된 데이터에서 1의 개수가 짝수인지 홀수인지를 검사함
  - XOR/XNOR를 사용하여 짝수/홀수 패리티를 발생할 수 있음