COMBINATIONAL LOGIC

INEL 4205 - CHAPTER 4

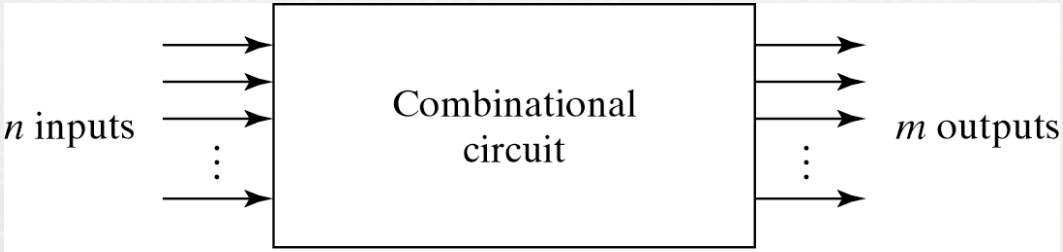


Fig. 4-1 Block Diagram of Combinational Circuit

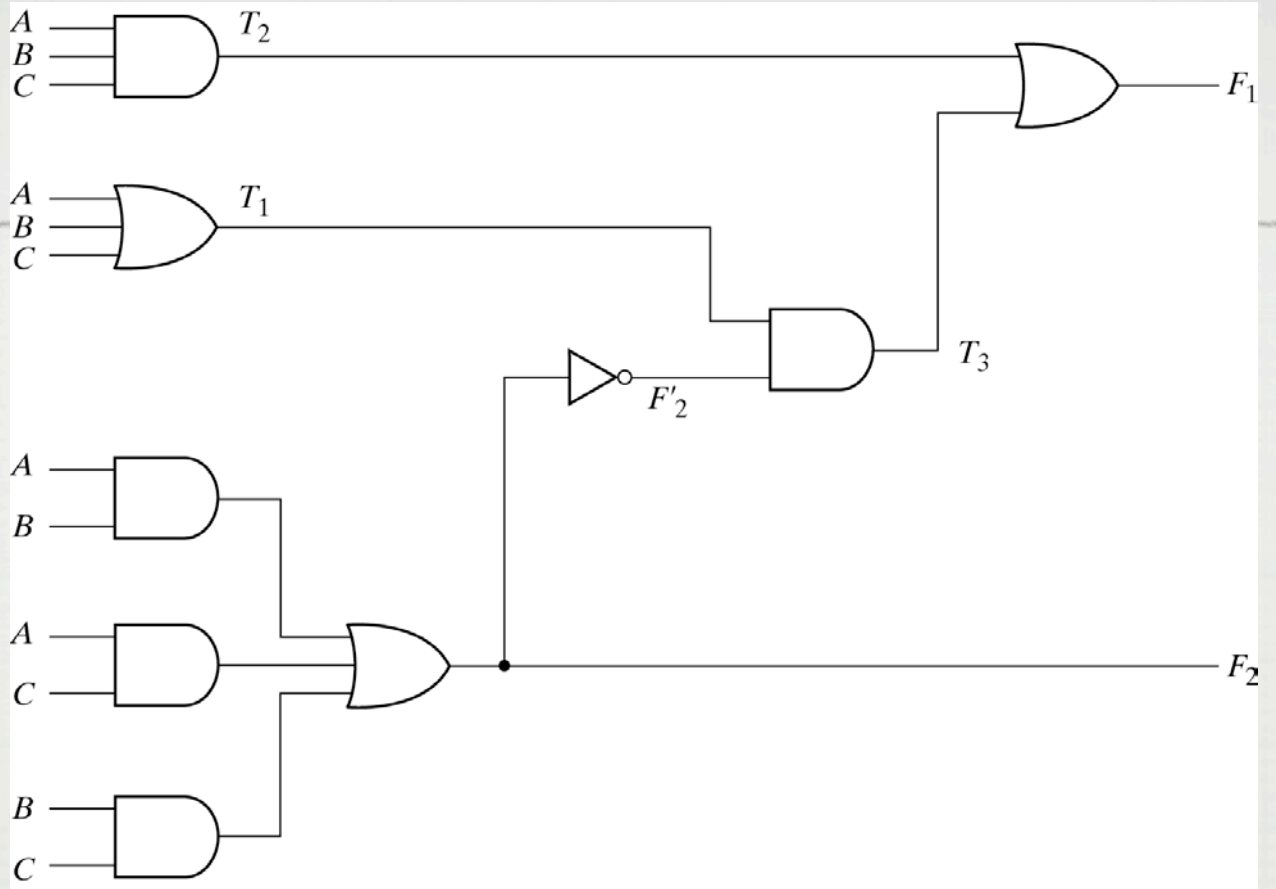


Fig. 4-2 Logic Diagram for Analysis Example

YOU CAN USE INTERMEDIATE VARIABLES TO CONSTRUCT THE BOOLEAN FUNCTIONS FROM A COMPLEX LOGIC DIAGRAM.

Table 4-2
Truth Table for Code-Conversion Example

Input BCD				Output Excess-3 Code			
A	*B*	*C*	*D*	*w*	*x*	*y*	*z*
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

TABLE 4-2 BCD TO EXCESS-3 CODE

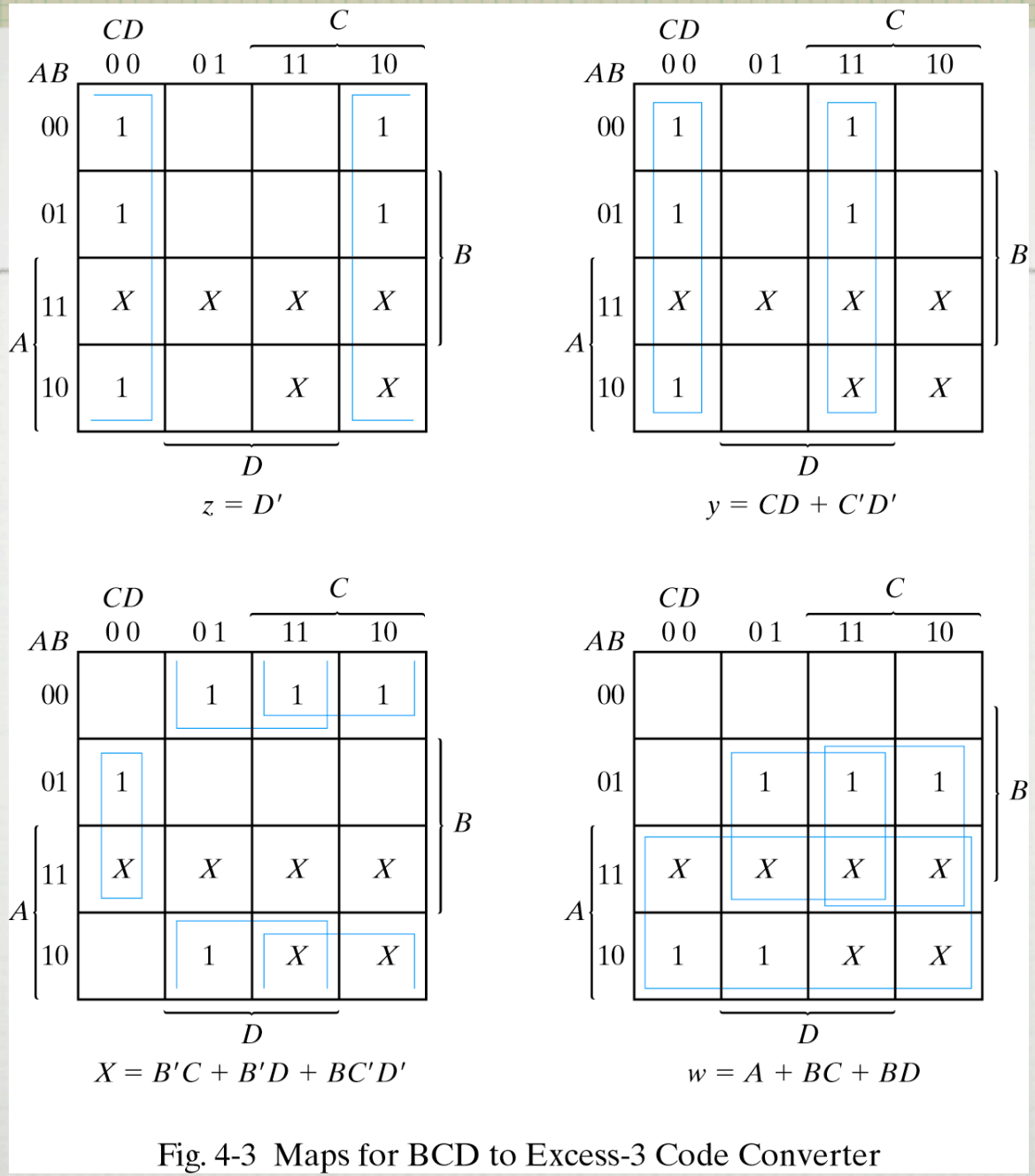


Fig. 4-3 Maps for BCD to Excess-3 Code Converter

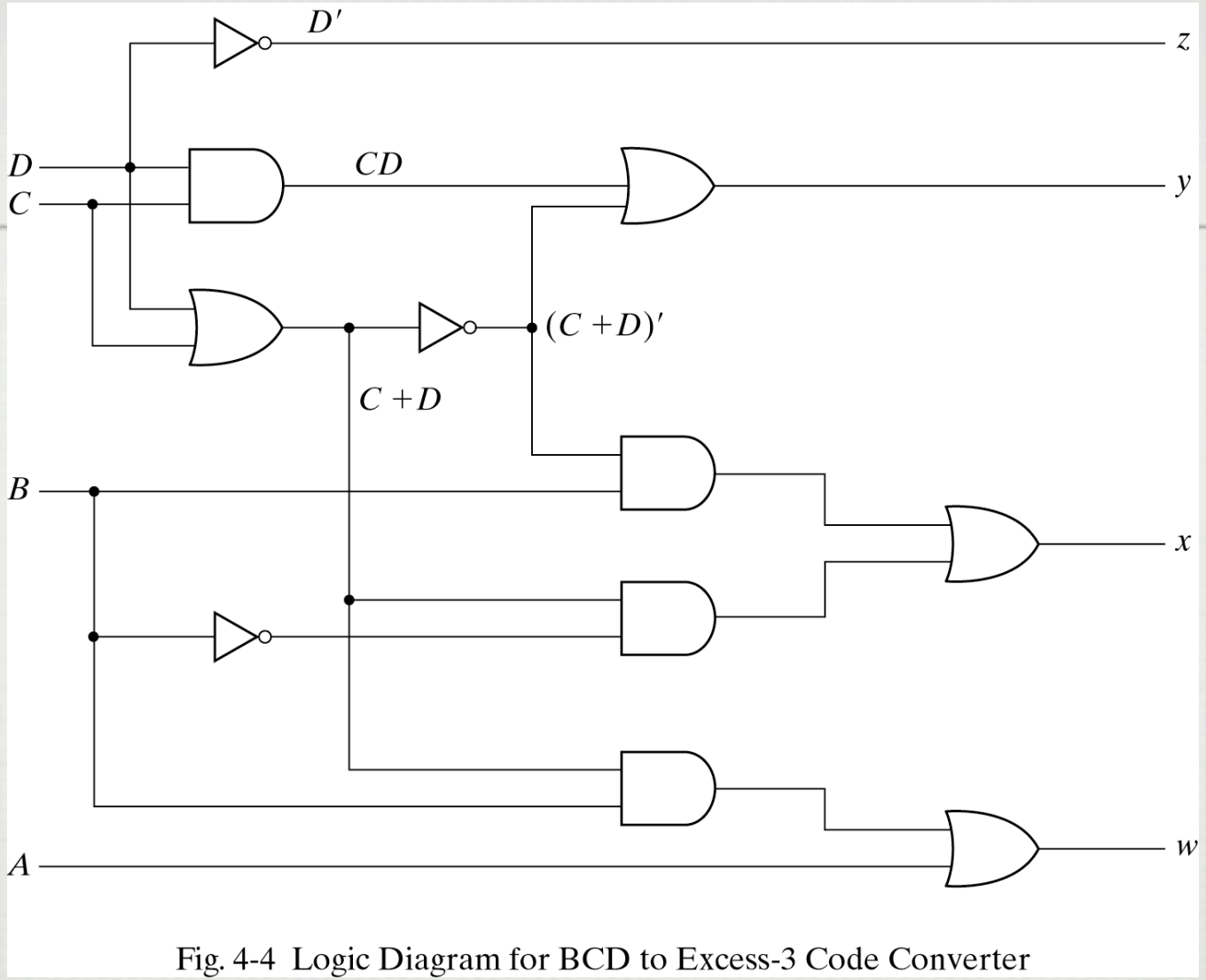
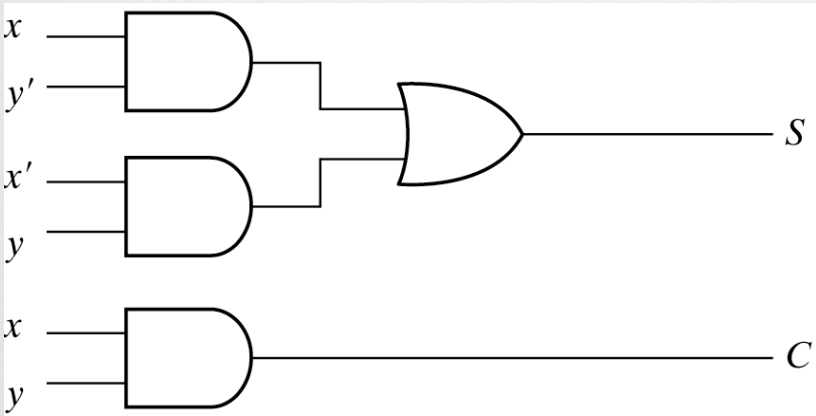


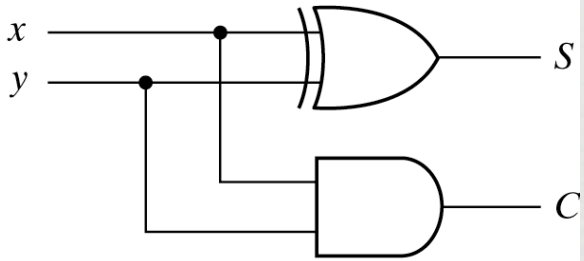
Fig. 4-4 Logic Diagram for BCD to Excess-3 Code Converter

Table 4-3
Half Adder

x	y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



(a) $S = xy' + x'y$
$C = xy$

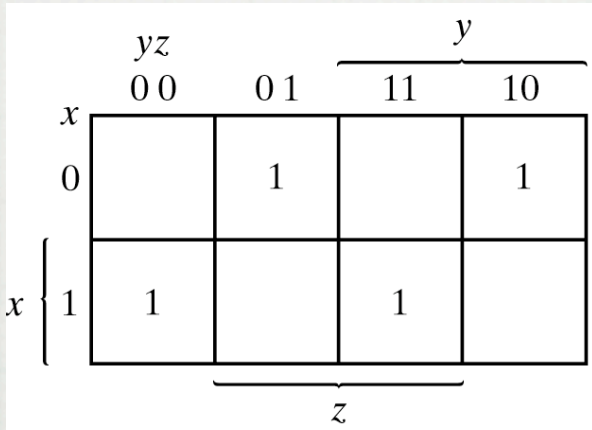


(b) $S = x \oplus y$
$C = xy$

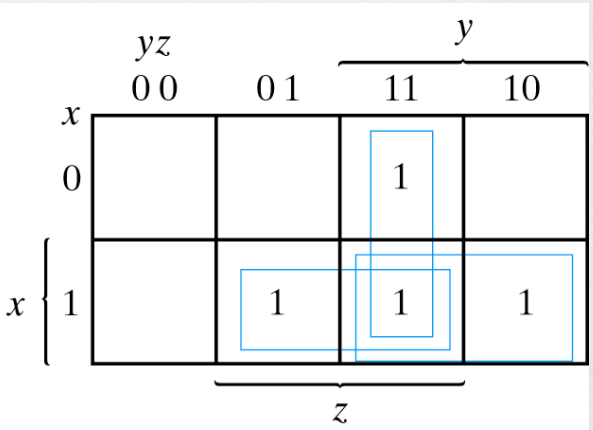
Fig. 4-5 Implementation of Half-Adder

Table 4-4
Full Adder

x	y	z	C	S
0	0	0	0	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	1	1



$S = x'y'z + x'yz' + xy'z' + xyz$



$S = xy + xz + yz$
$= xy + xy'z + x'yz$

Fig. 4-6 Maps for Full Adder

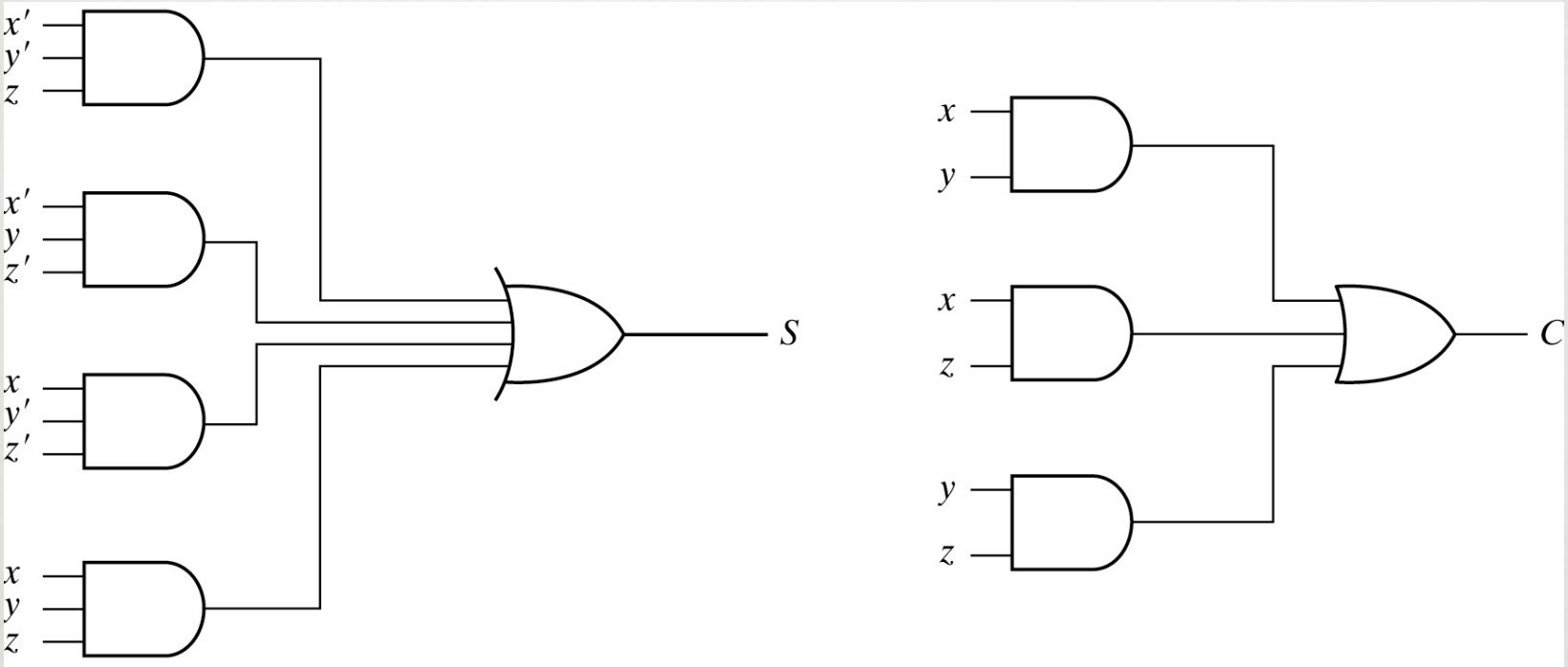


Fig. 4-7 Implementation of Full Adder in Sum of Products

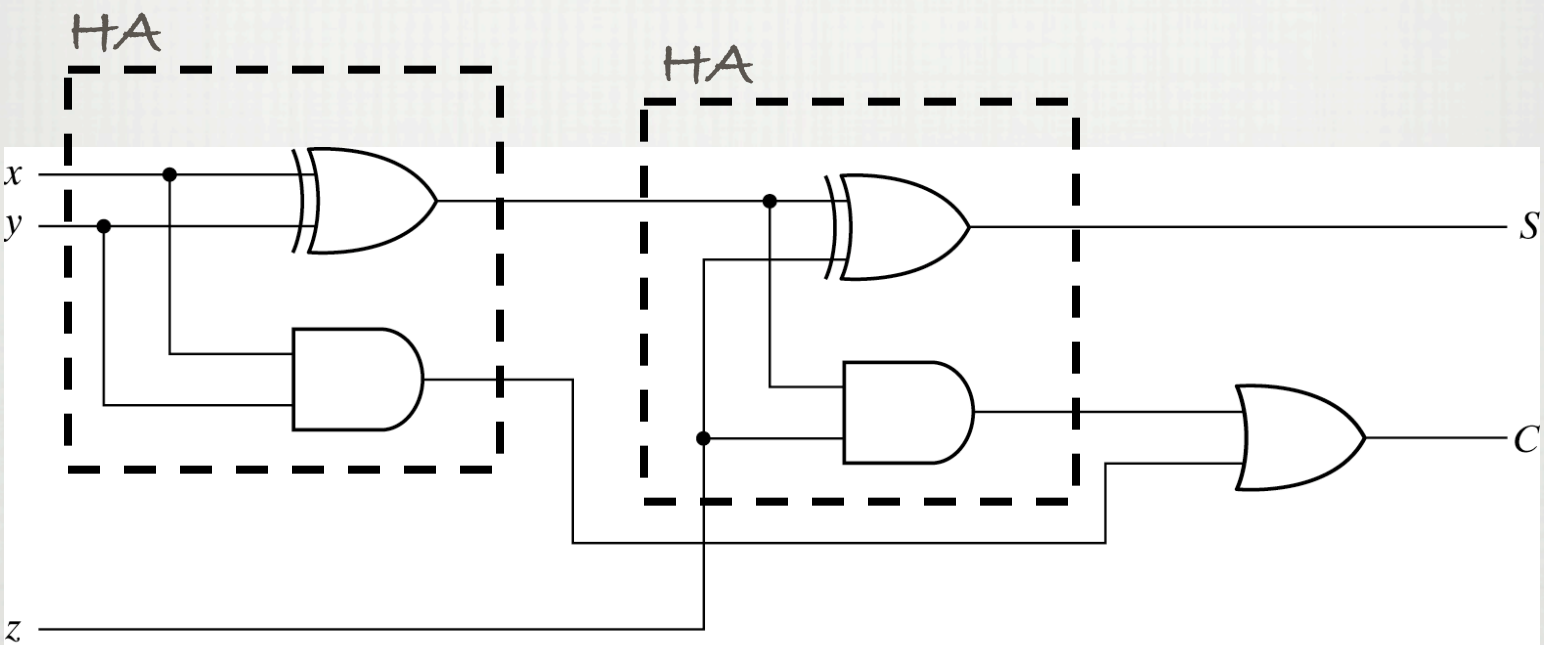


Fig. 4-8 Implementation of Full Adder with Two Half Adders and an OR Gate

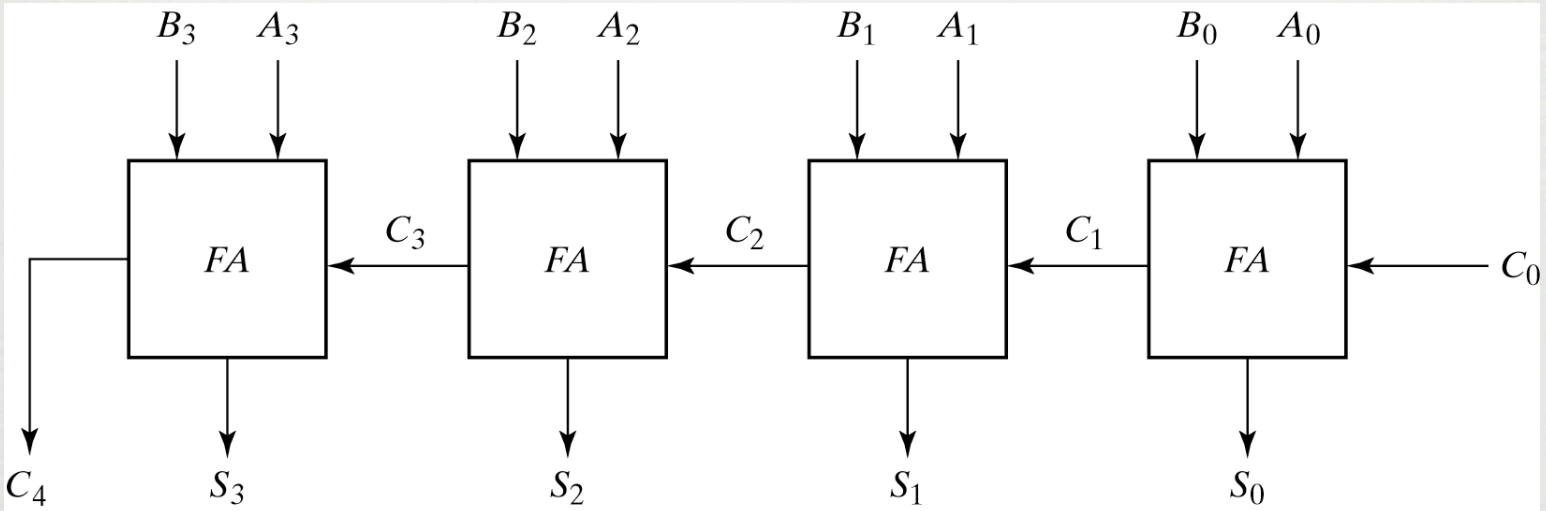


Fig. 4-9 4-Bit Adder

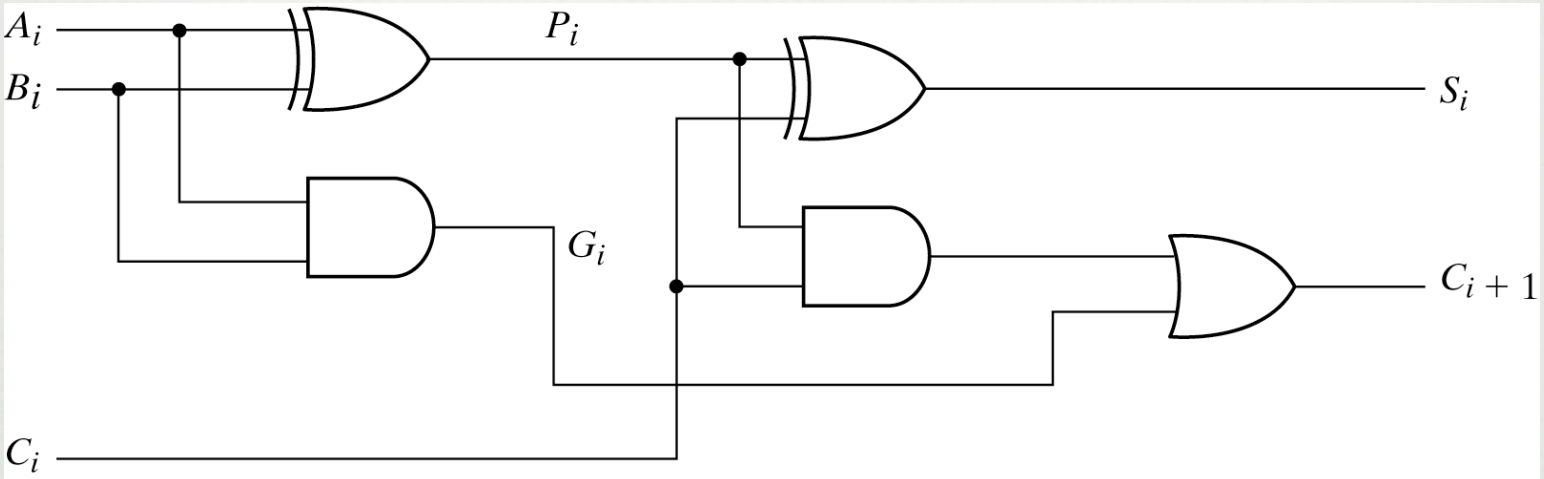


Fig. 4-10 Full Adder with P and G Shown

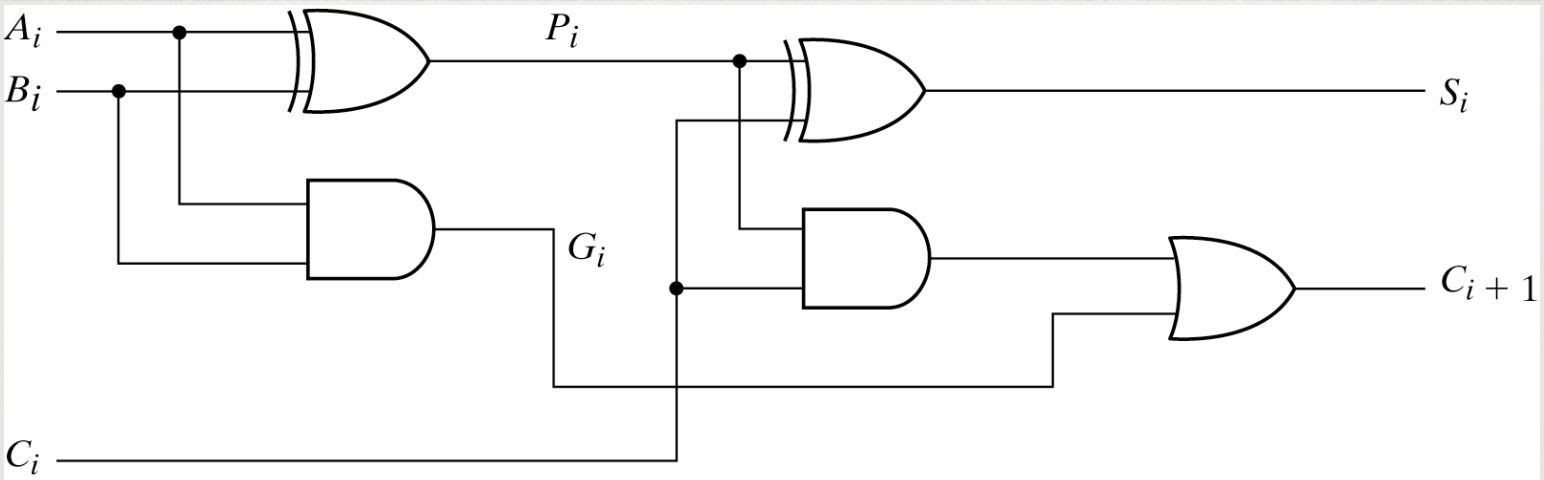


Fig. 4-10 Full Adder with P and G Shown

$$P_i = A_i \oplus B_i$$

$$G_i = A_i B_i$$

$$S_i = P_i \oplus C_i$$

$$C_{i+1} = G_i + P_i C_i$$

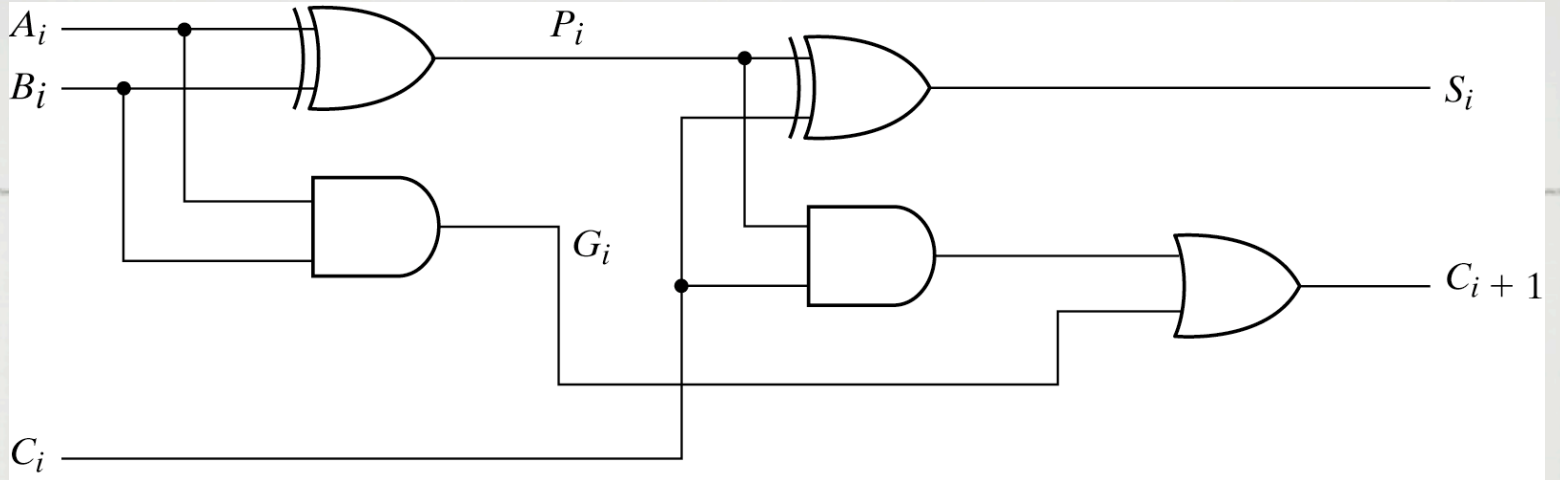


Fig. 4-10 Full Adder with P and G Shown

$$P_i = A_i \oplus B_i$$

$$G_i = A_i B_i$$

$$S_i = P_i \oplus C_i$$

$$C_{i+1} = G_i + P_i C_i$$

C_0 = input carry

$$C_1 = G_0 + P_0 C_0$$

$$C_2 = G_1 + P_1 C_1 = G_1 + P_1(G_0 + P_0 C_0) = G_1 + P_1 G_0 + P_1 P_0 C_0$$

$$C_3 = G_2 + P_2 C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$$

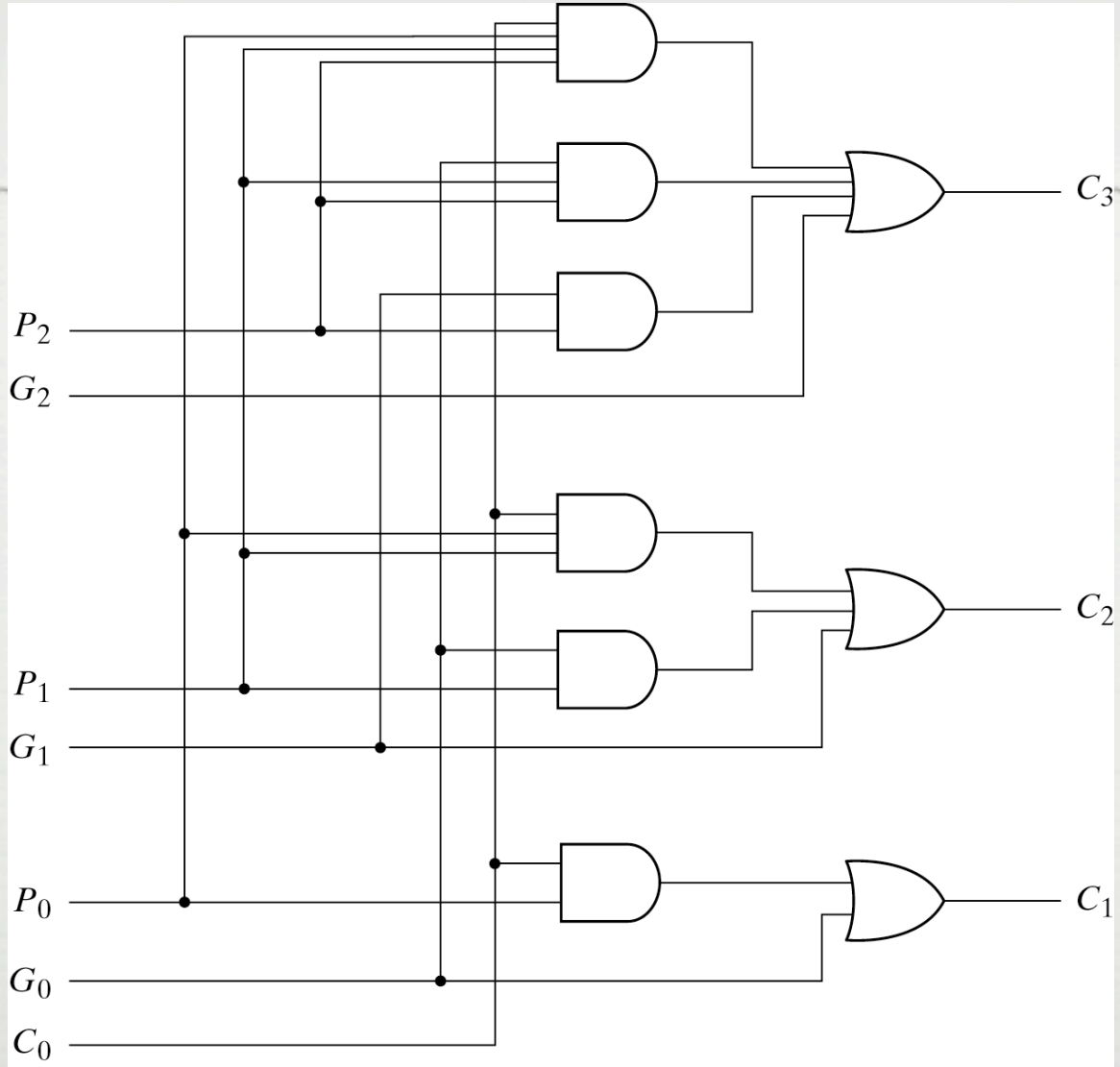


Fig. 4-11 Logic Diagram of Carry Lookahead Generator

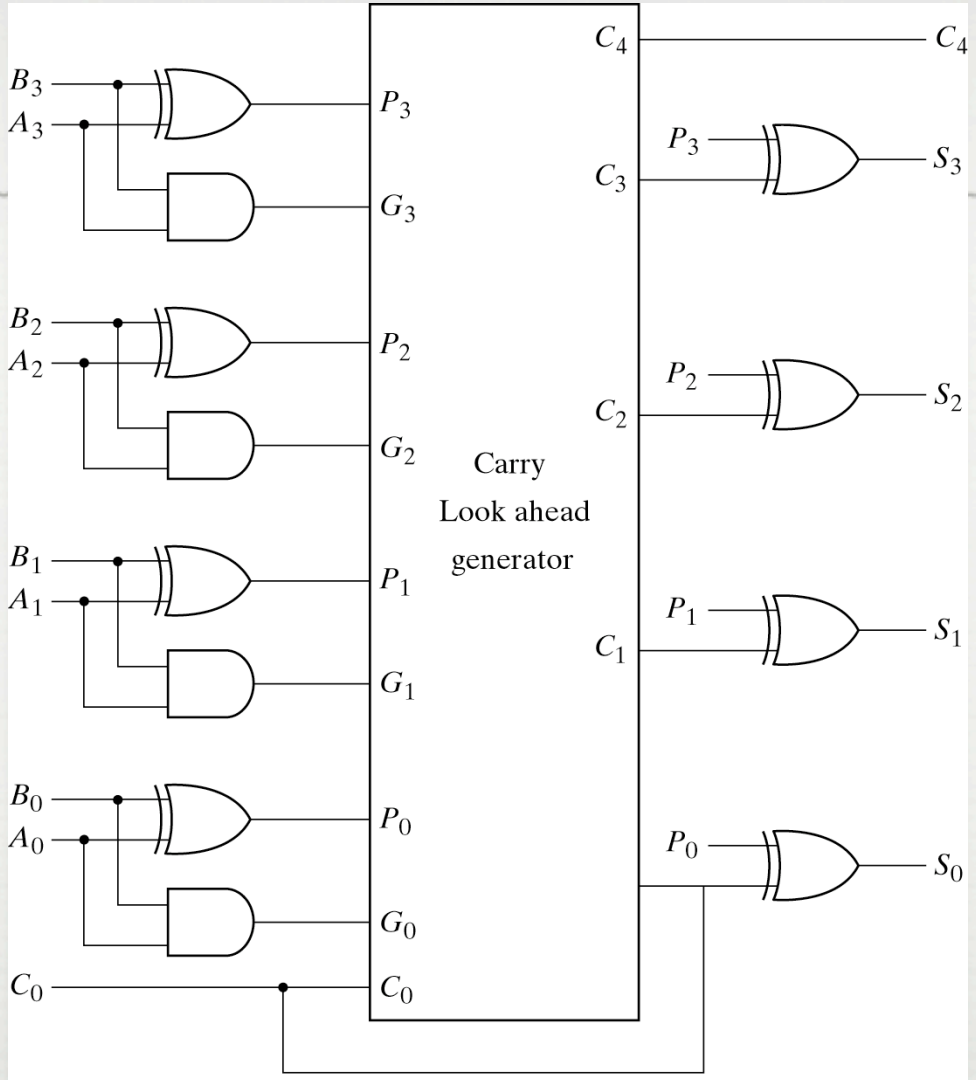


Fig. 4-12 4-Bit Adder with Carry Lookahead

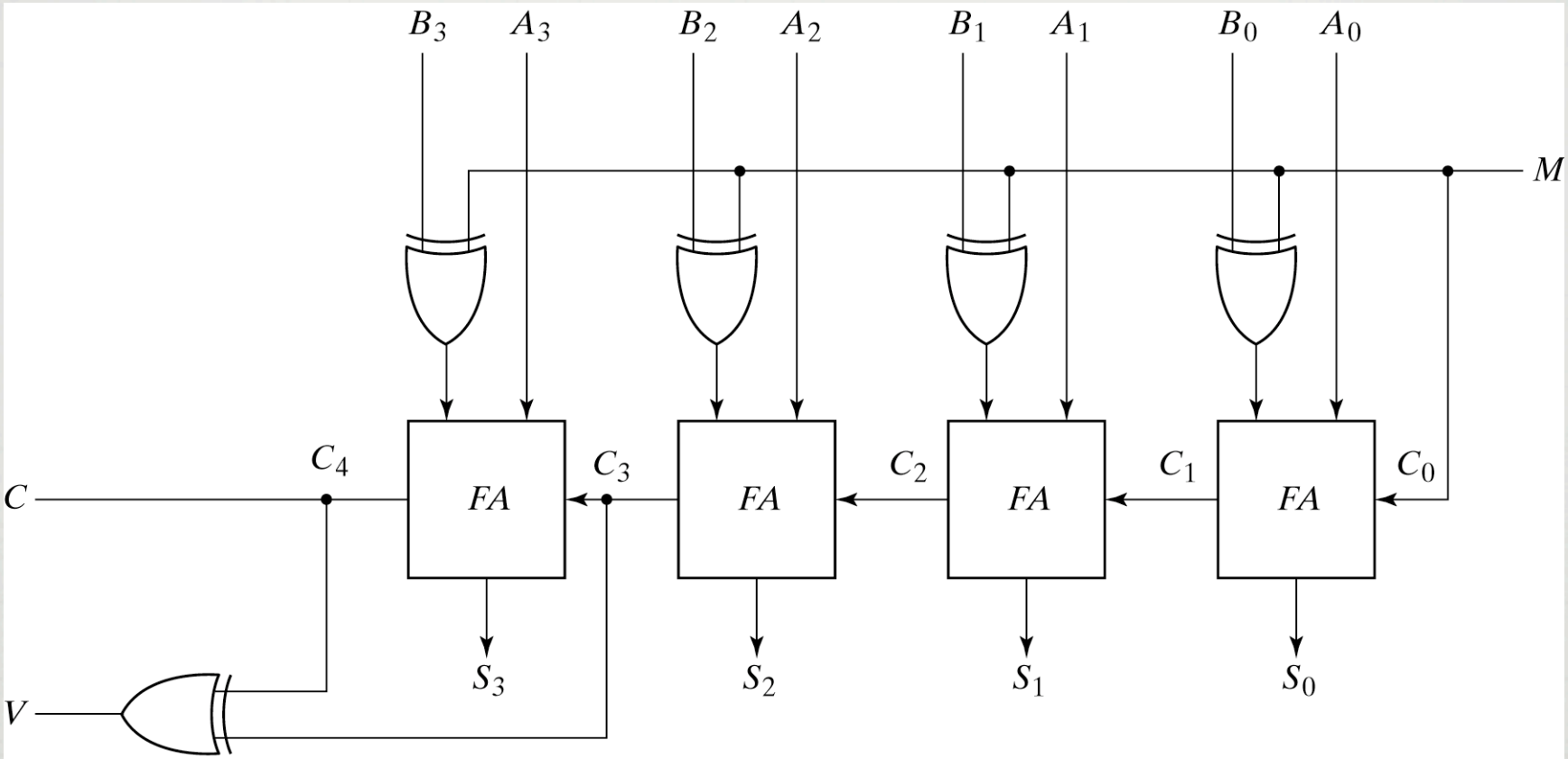


Fig. 4-13 4-Bit Adder Subtractor

M=1 COMPLEMENTS THE B-BITS AND ADDS 1, THUS FORMING THE 2'S COMPLEMENT OF B

carries:	0	1		carries:	1	0	
+70		0	1000110	−70		1	0111010
+80		0	1010000	−80		1	0110000
+150		1	0010110	−150		0	1101010

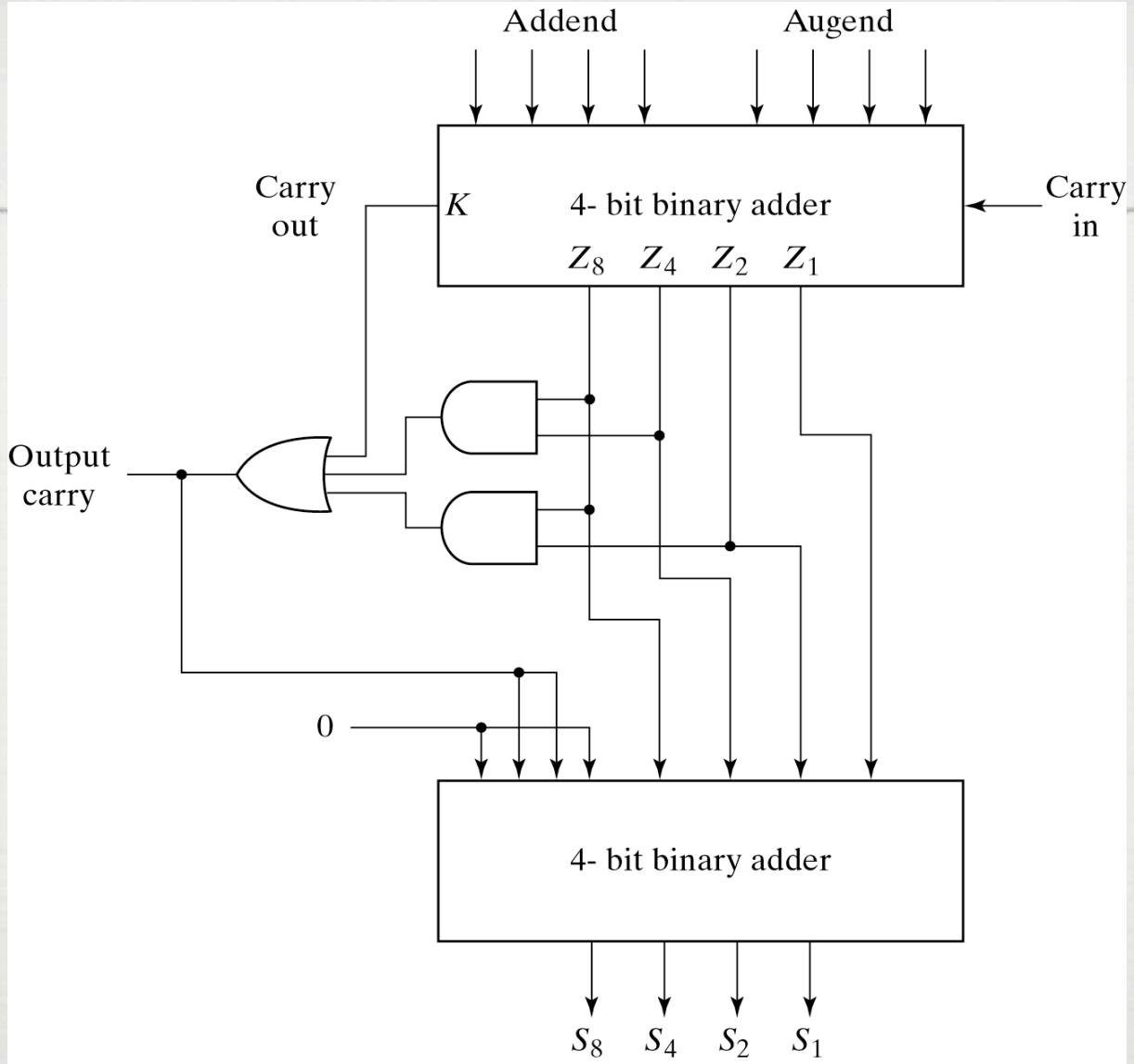


Fig. 4-14 Block Diagram of a BCD Adder

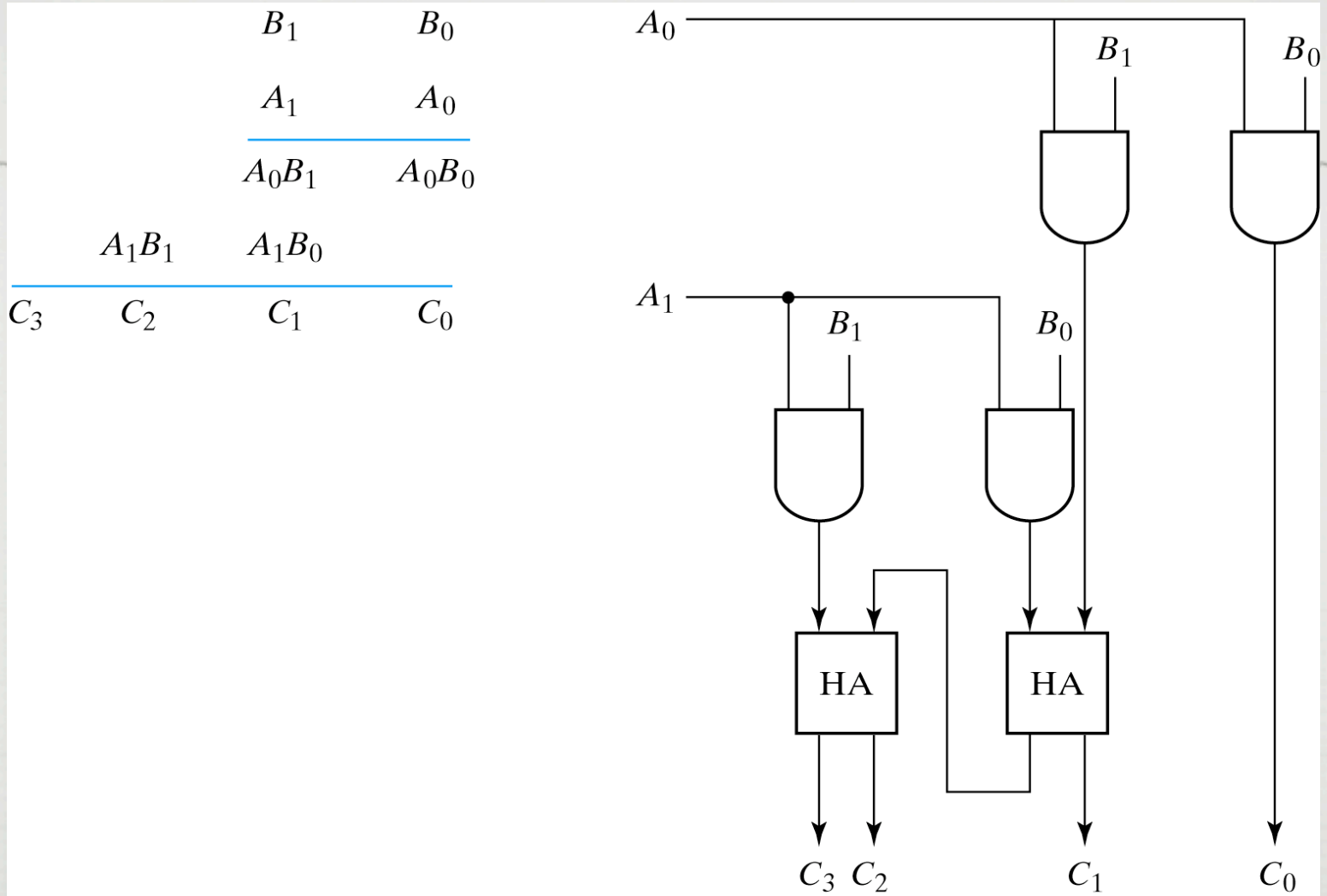


Fig. 4-15 2-Bit by 2-Bit Binary Multiplier

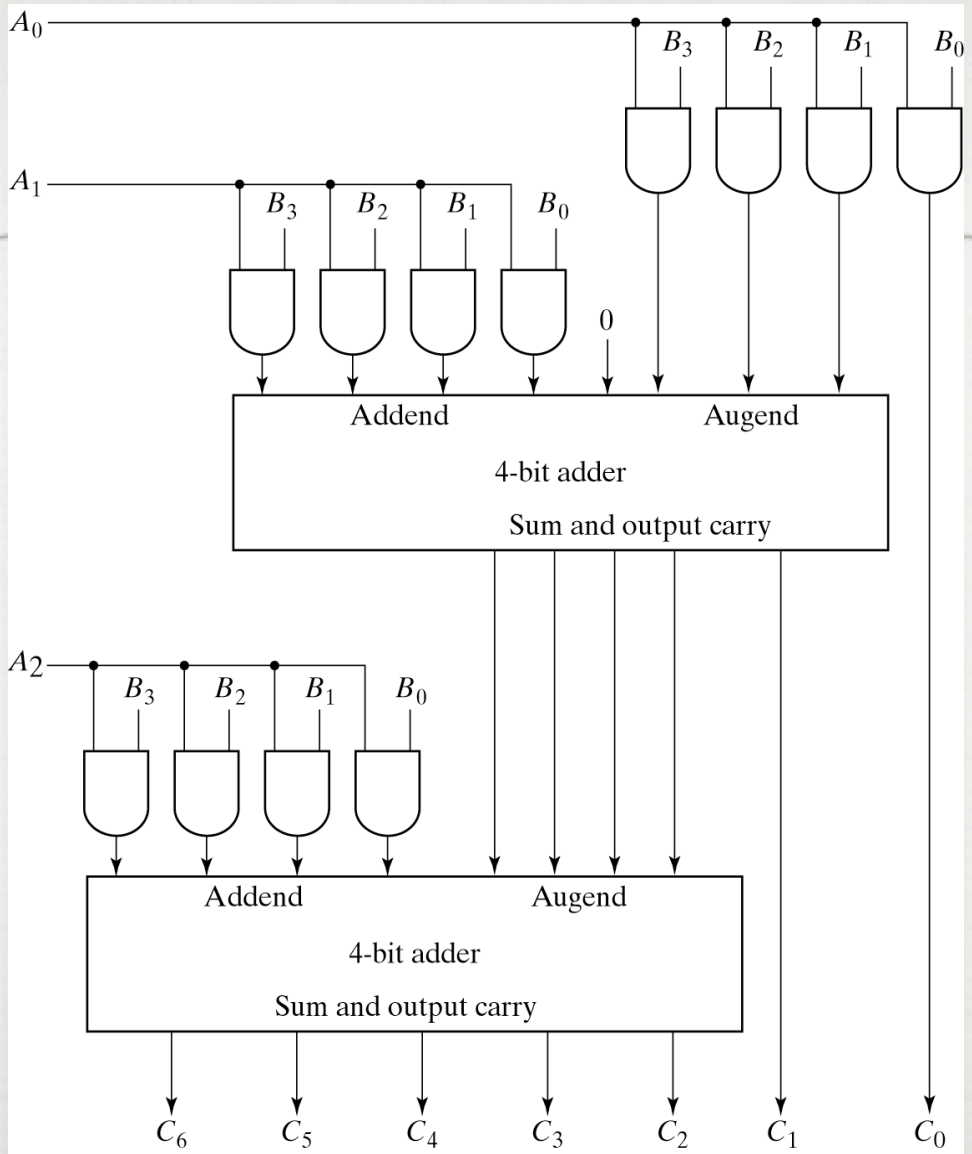


Fig. 4-16 4-Bit by 3-Bit Binary Multiplier

MAGNITUDE COMPARATOR

$$A = A_3 A_2 A_1 A_0$$

$$B = B_3 B_2 B_1 B_0$$

$$x_i = A_i B_i + A_i' B_i' \qquad \text{for } i = 0, 1, 2, 3$$

$$(A = B) = x_3 x_2 x_1 x_0$$

$$(A > B) = A_3 B_3' + x_3 A_2 B_2' + x_3 x_2 A_1 B_1' + x_3 x_2 x_1 A_0 B_0'$$

$$(A < B) = A_3' B_3 + x_3 A_2' B_2 + x_3 x_2 A_1' B_1 + x_3 x_2 x_1 A_0' B_0$$

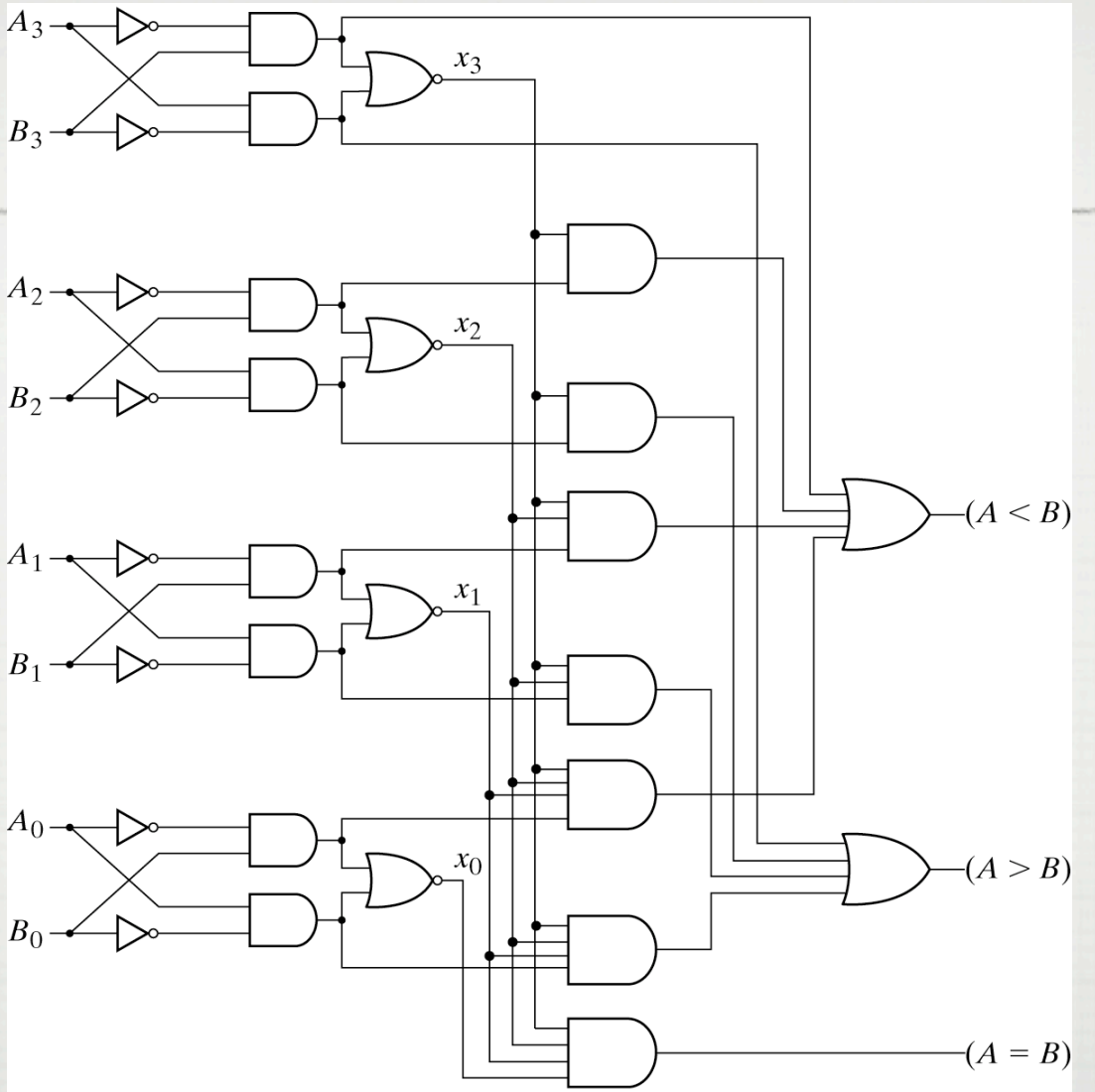


Fig. 4-17 4-Bit Magnitude Comparator

Table 4-6
Truth Table of a 3-to-8-Line Decoder

Inputs			Outputs							
x	y	z	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

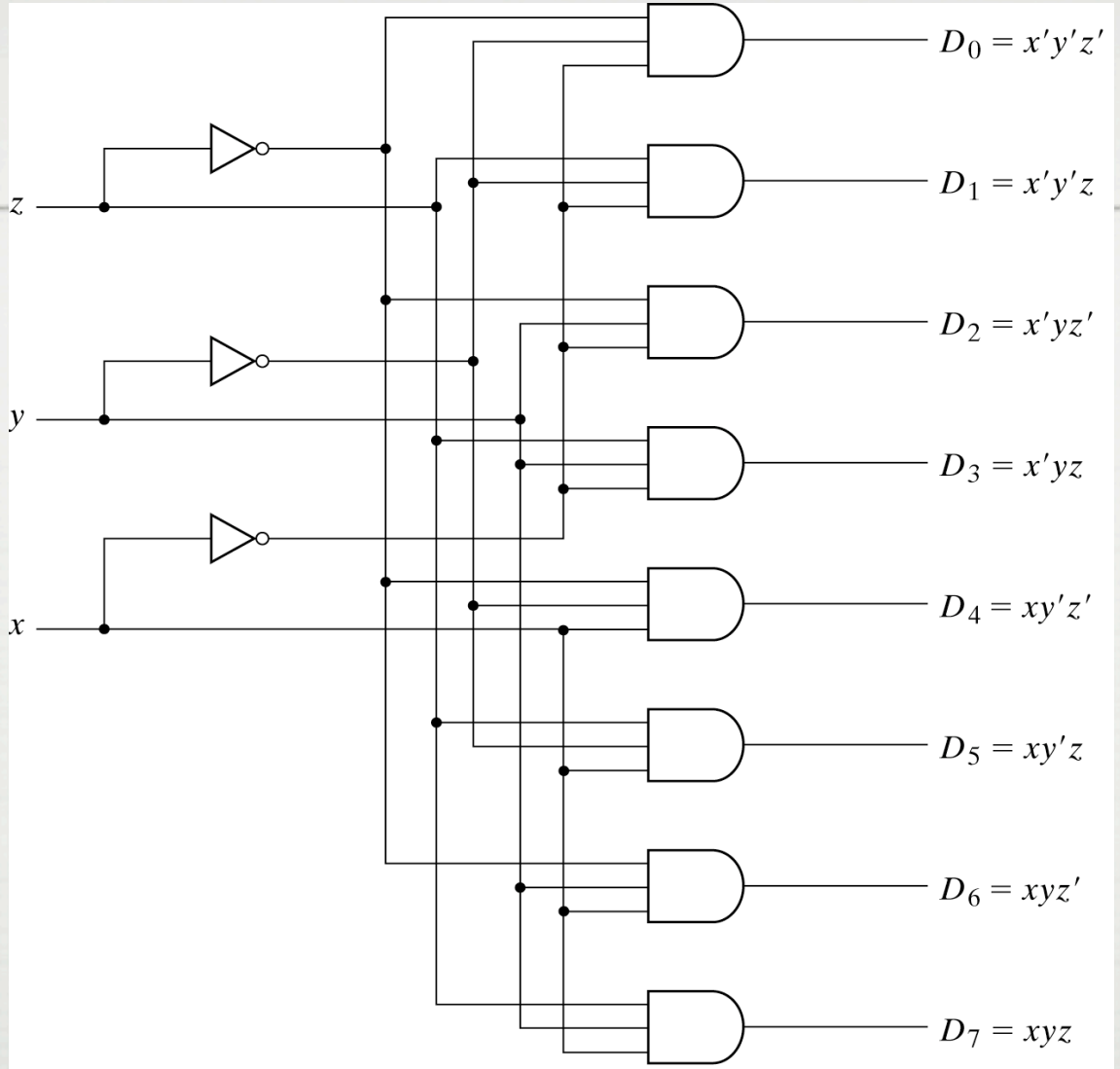
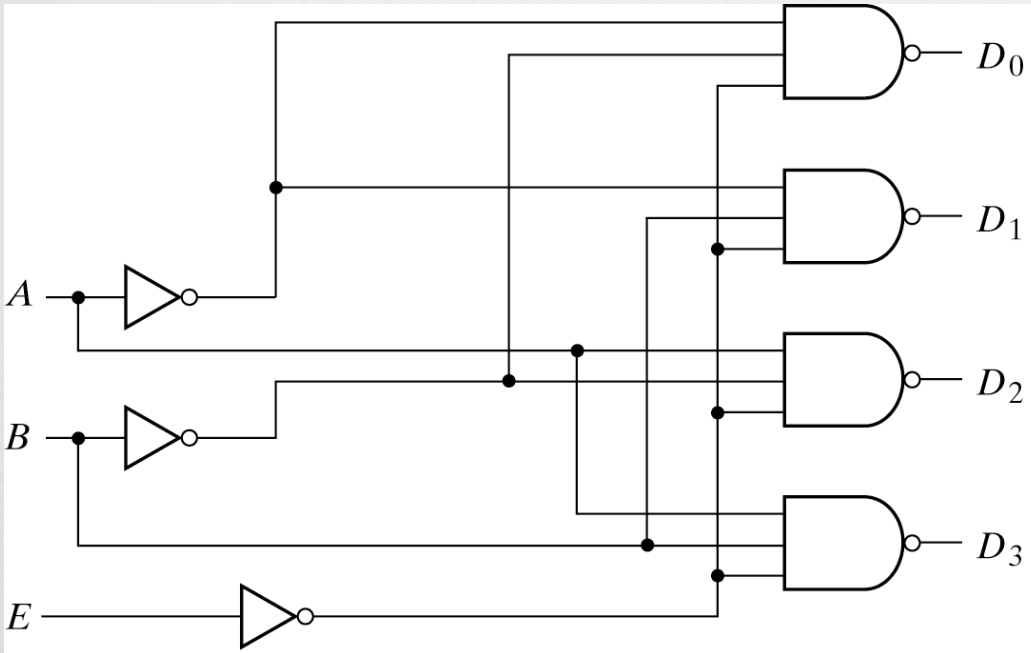


Fig. 4-18 3-to-8-Line Decoder



(a) Logic diagram

E	A	B	D_0	D_1	D_2	D_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

(b) Truth table

Fig. 4-19 2-to-4-Line Decoder with Enable Input

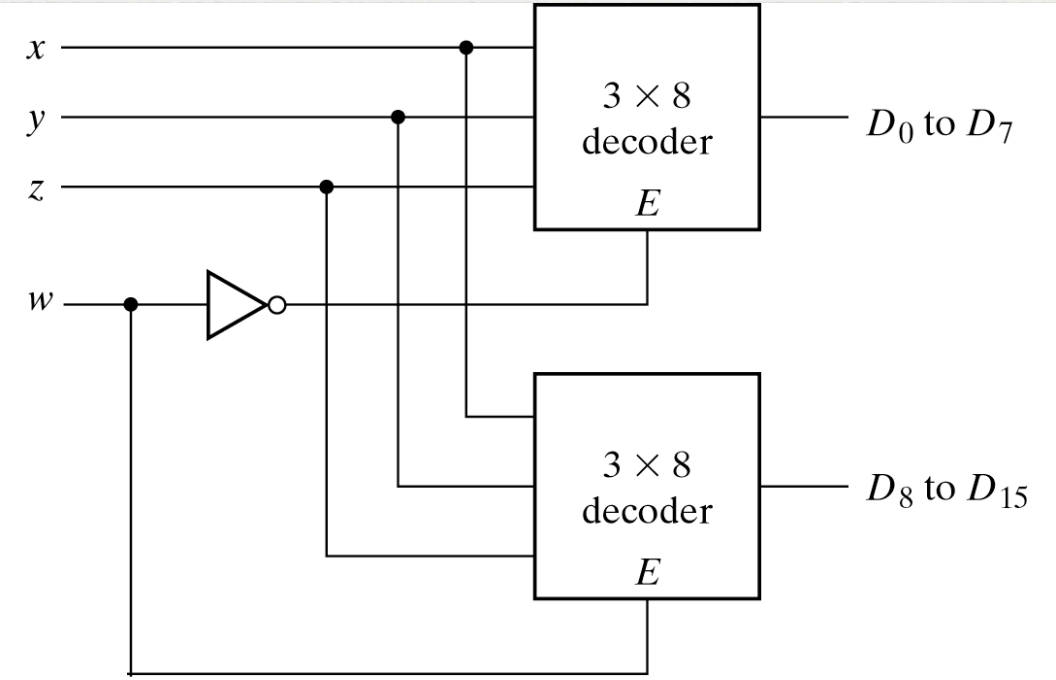


Fig. 4-20 4 × 16 Decoder Constructed with Two 3 × 8 Decoders

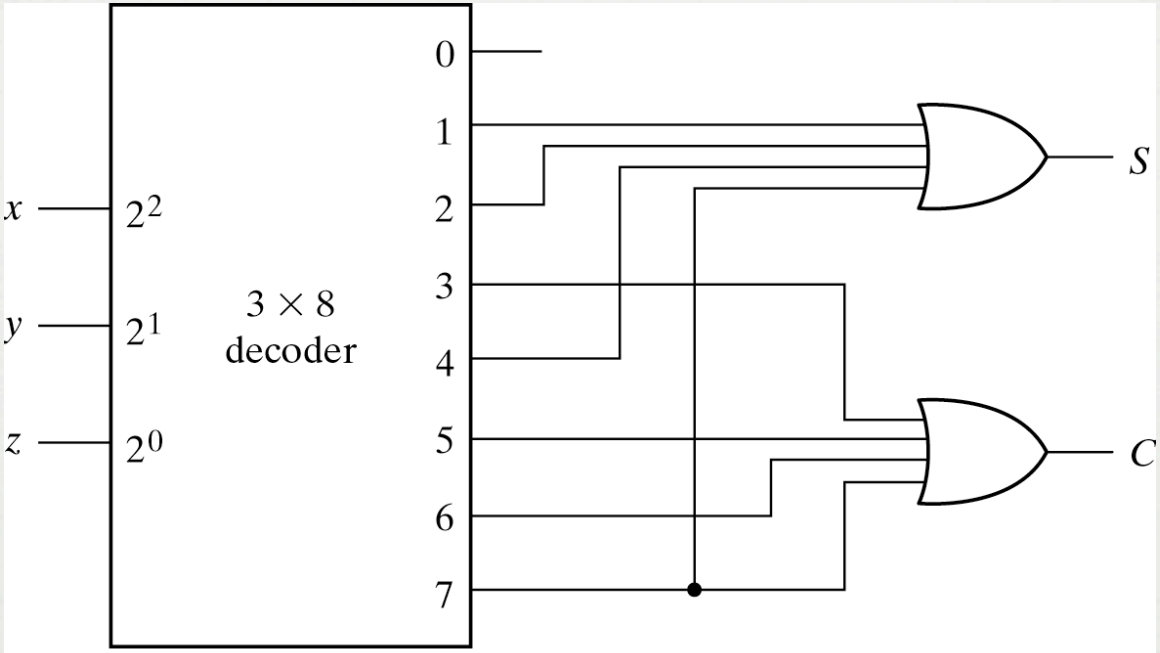


Fig. 4-21 Implementation of a Full Adder with a Decoder

Table 4-7

Truth Table of Octal-to-Binary Encoder

Inputs								Outputs		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x	y	z
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

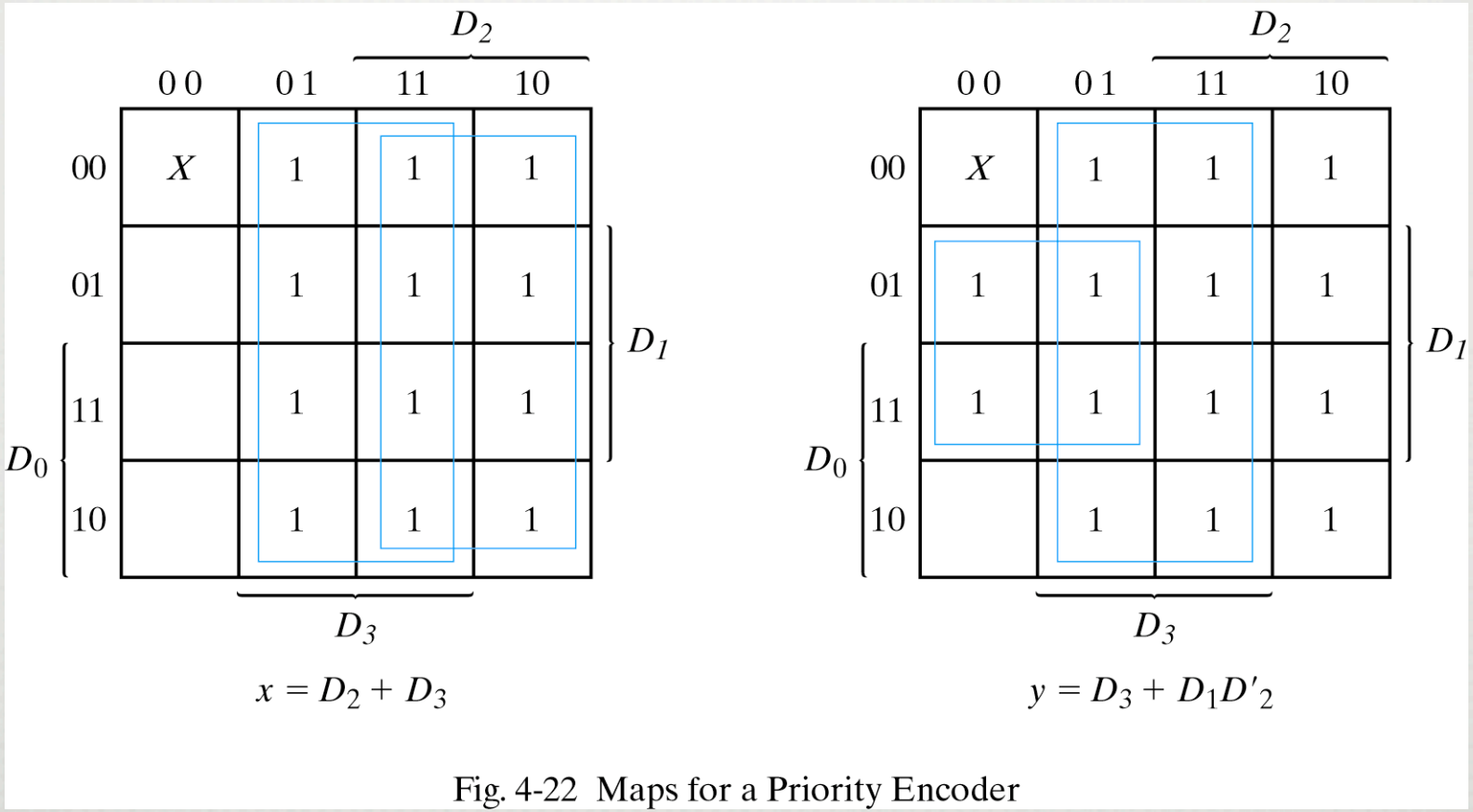


Fig. 4-22 Maps for a Priority Encoder

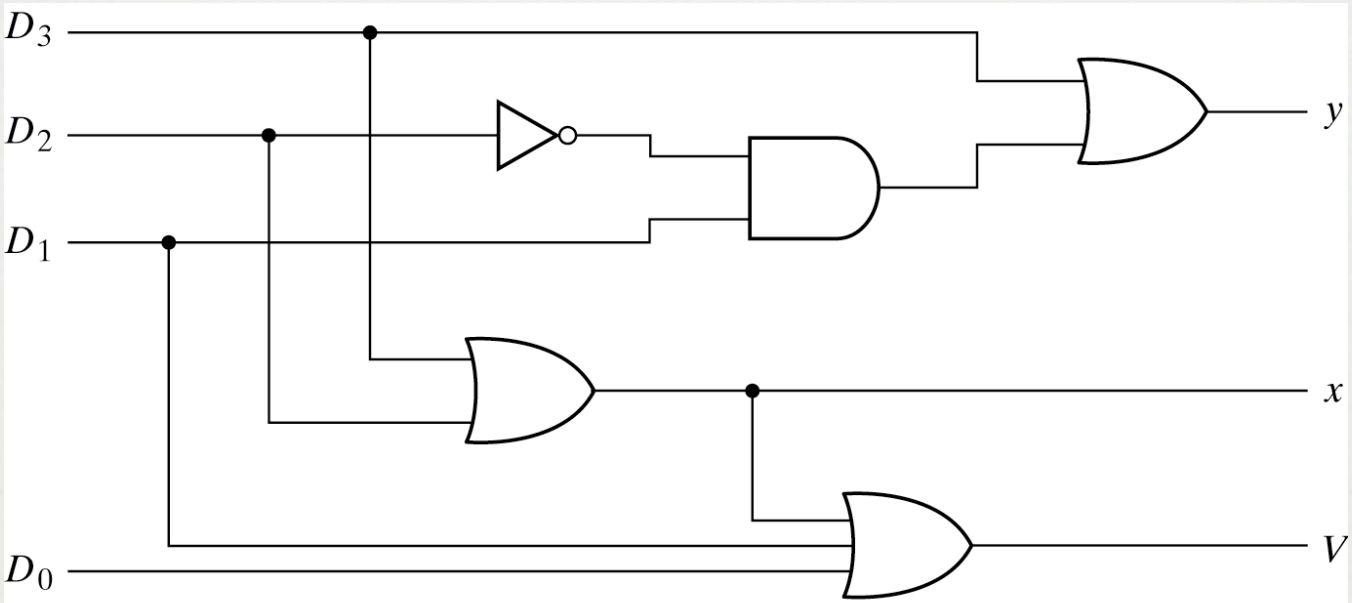
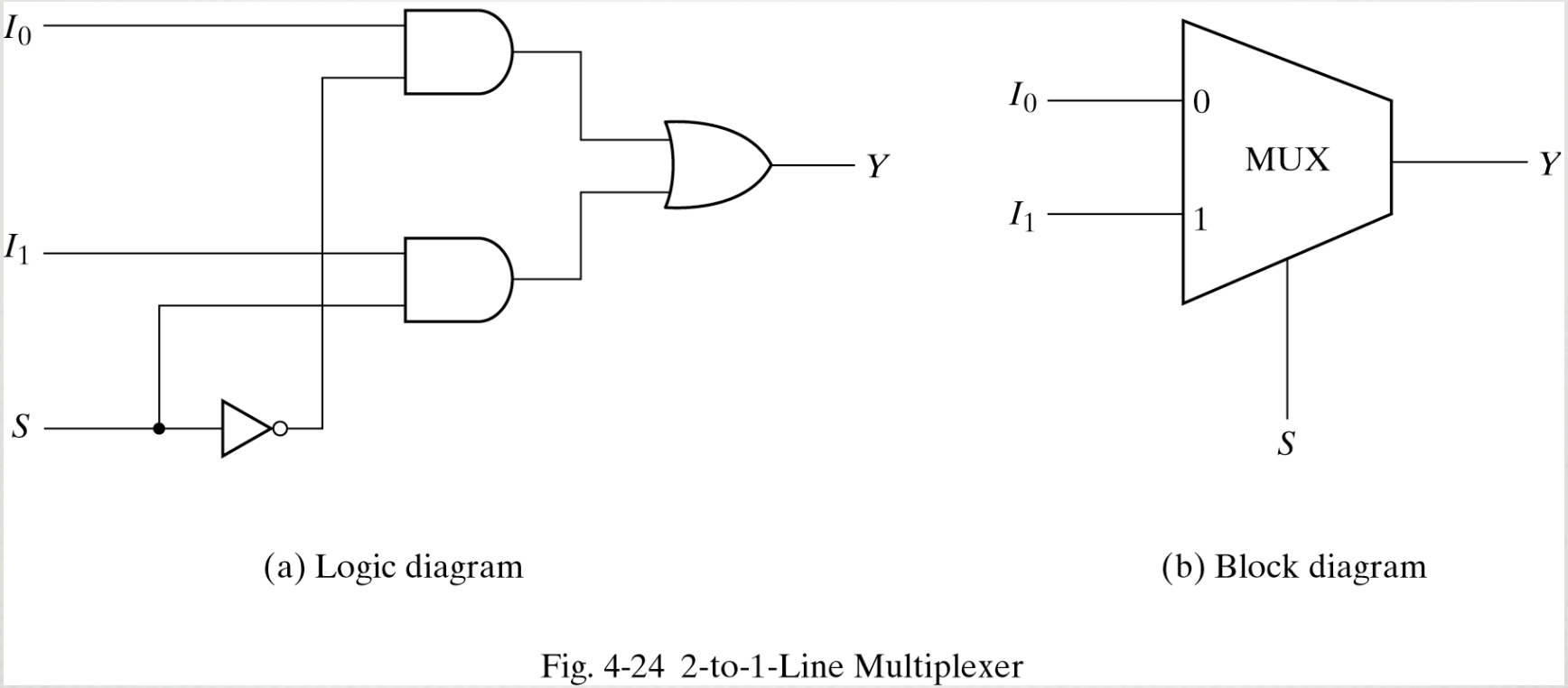


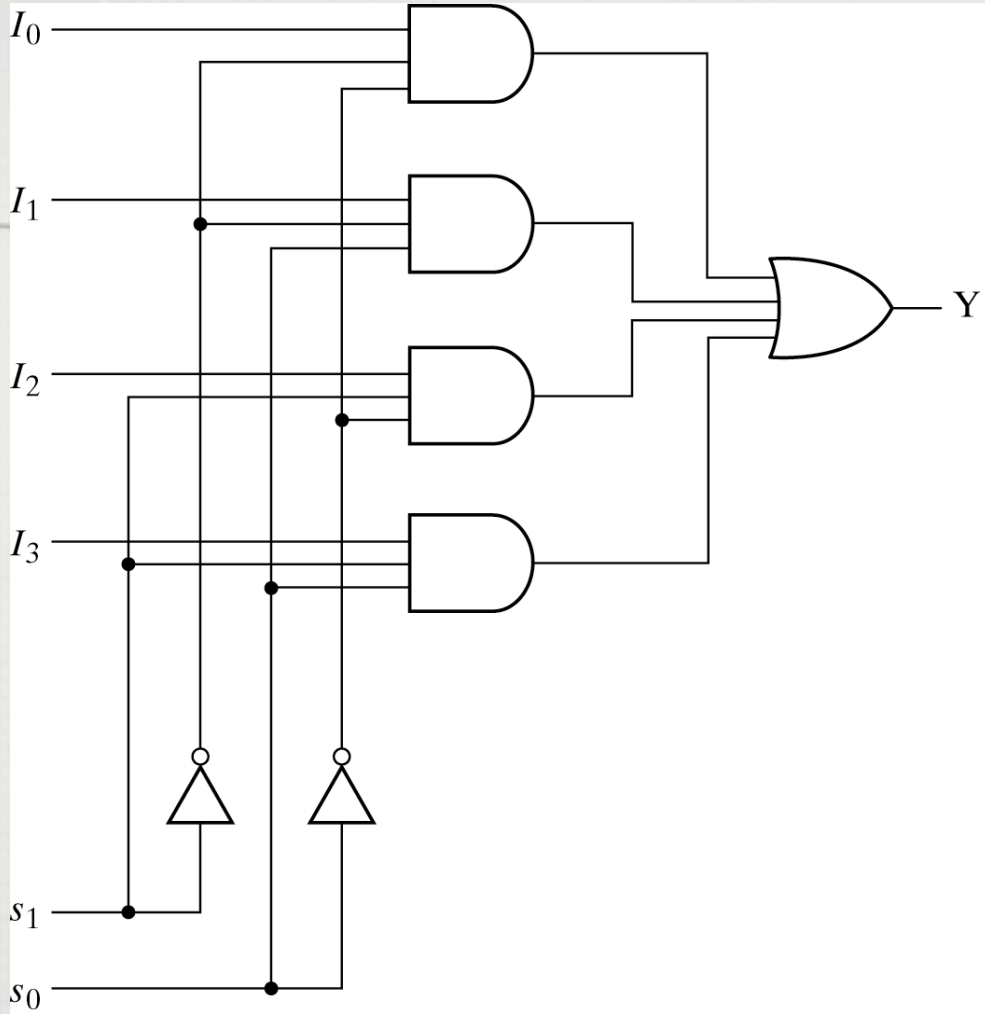
Fig. 4-23 4-Input Priority Encoder



(a) Logic diagram

(b) Block diagram

Fig. 4-24 2-to-1-Line Multiplexer

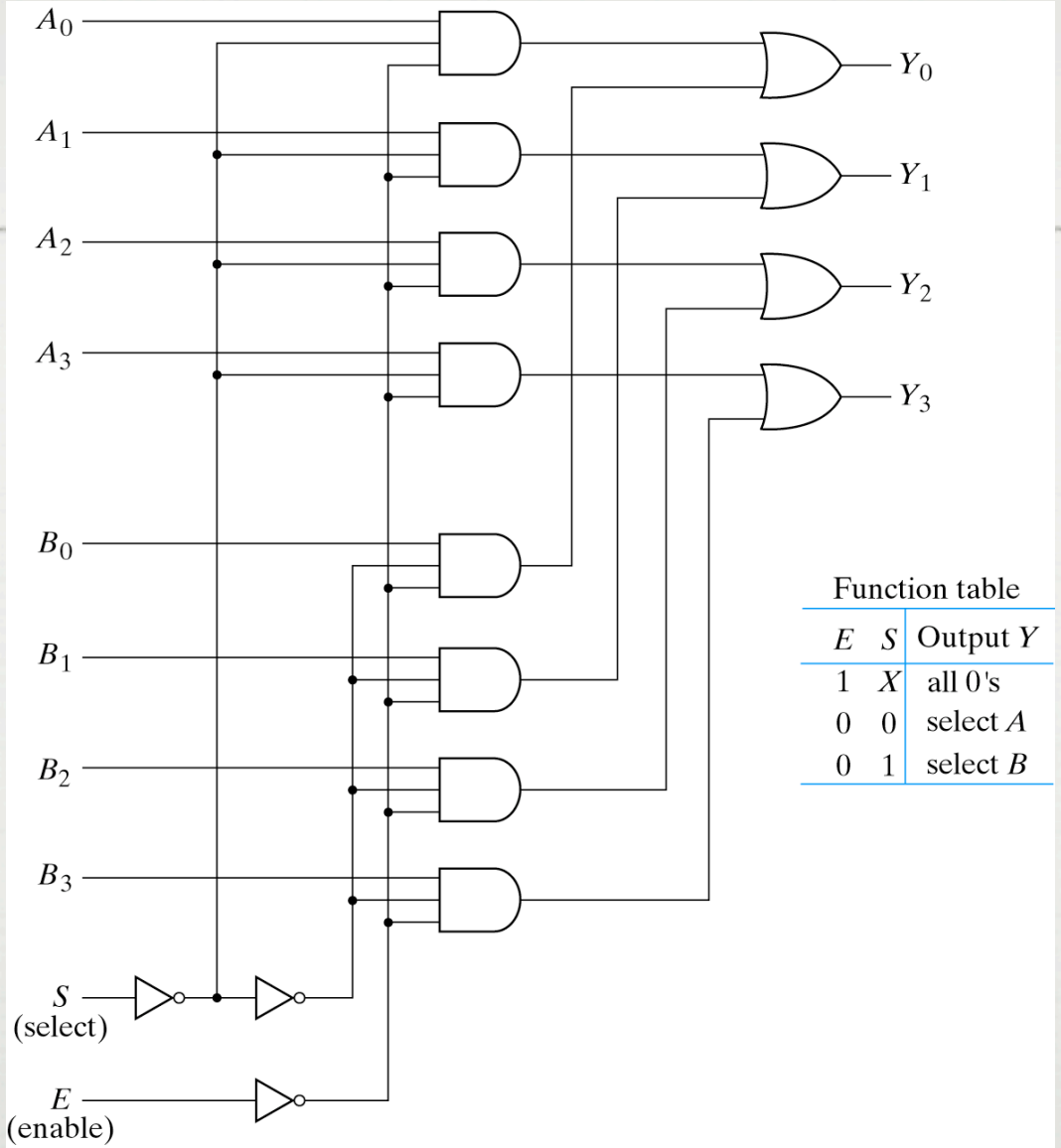


(a) Logic diagram

s_1	s_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

(b) Function table

Fig. 4-25 4-to-1-Line Multiplexer



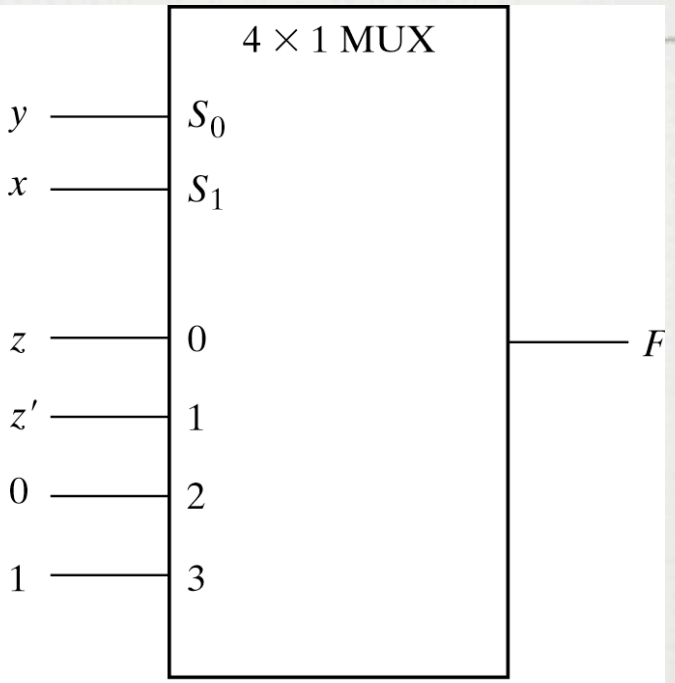
Function table

E	S	Output Y
1	X	all 0's
0	0	select A
0	1	select B

Fig. 4-26 Quadruple 2-to-1-Line Multiplexer

x	y	z	F	
0	0	0	0	$F = z$
0	0	1	1	
0	1	0	1	$F = z'$
0	1	1	0	
1	0	0	0	$F = 0$
1	0	1	0	
1	1	0	1	$F = 1$
1	1	1	1	

(a) Truth table



(b) Multiplexer implementation

Fig. 4-27 Implementing a Boolean Function with a Multiplexer

A	B	C	D	F	
0	0	0	0	0	$F = D$
0	0	0	1	1	
0	0	1	0	0	$F = D$
0	0	1	1	1	
0	1	0	0	1	$F = D'$
0	1	0	1	0	
0	1	1	0	0	$F = 0$
0	1	1	1	0	
1	0	0	0	0	$F = 0$
1	0	0	1	0	
1	0	1	0	0	$F = D$
1	0	1	1	1	
1	1	0	0	1	$F = 1$
1	1	0	1	1	
1	1	1	0	1	$F = 1$
1	1	1	1	1	

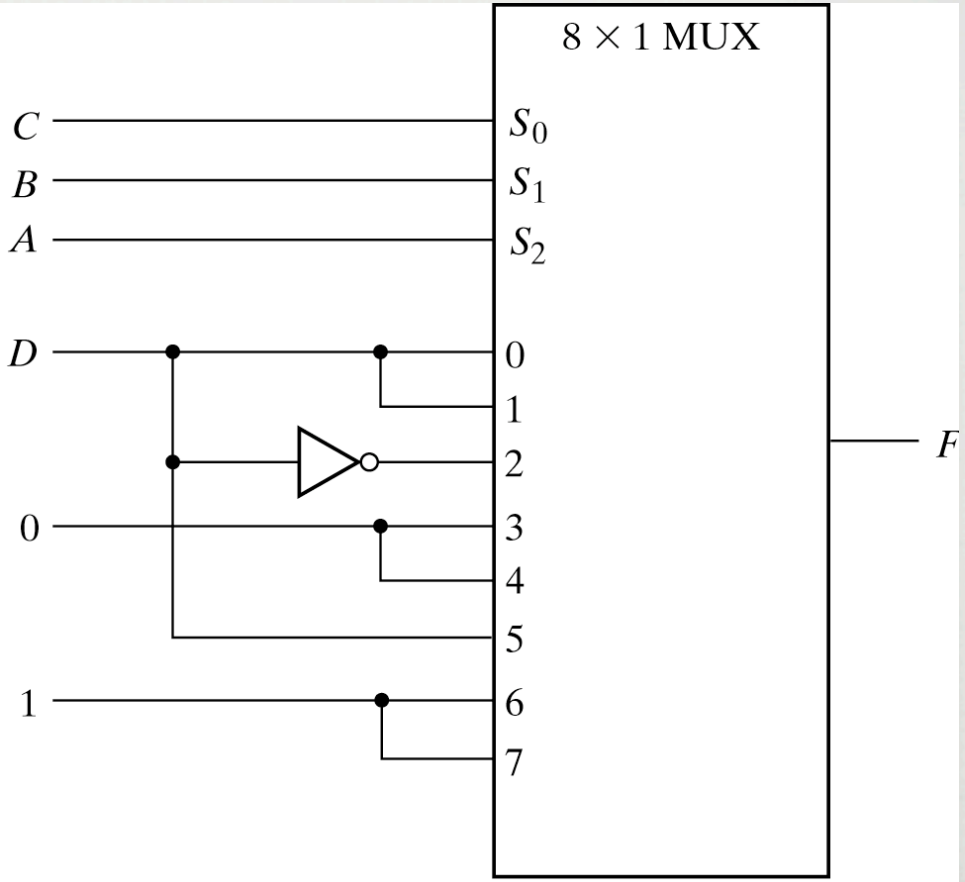


Fig. 4-28 Implementing a 4-Input Function with a Multiplexer

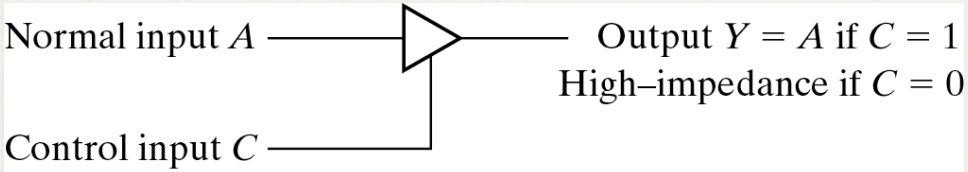
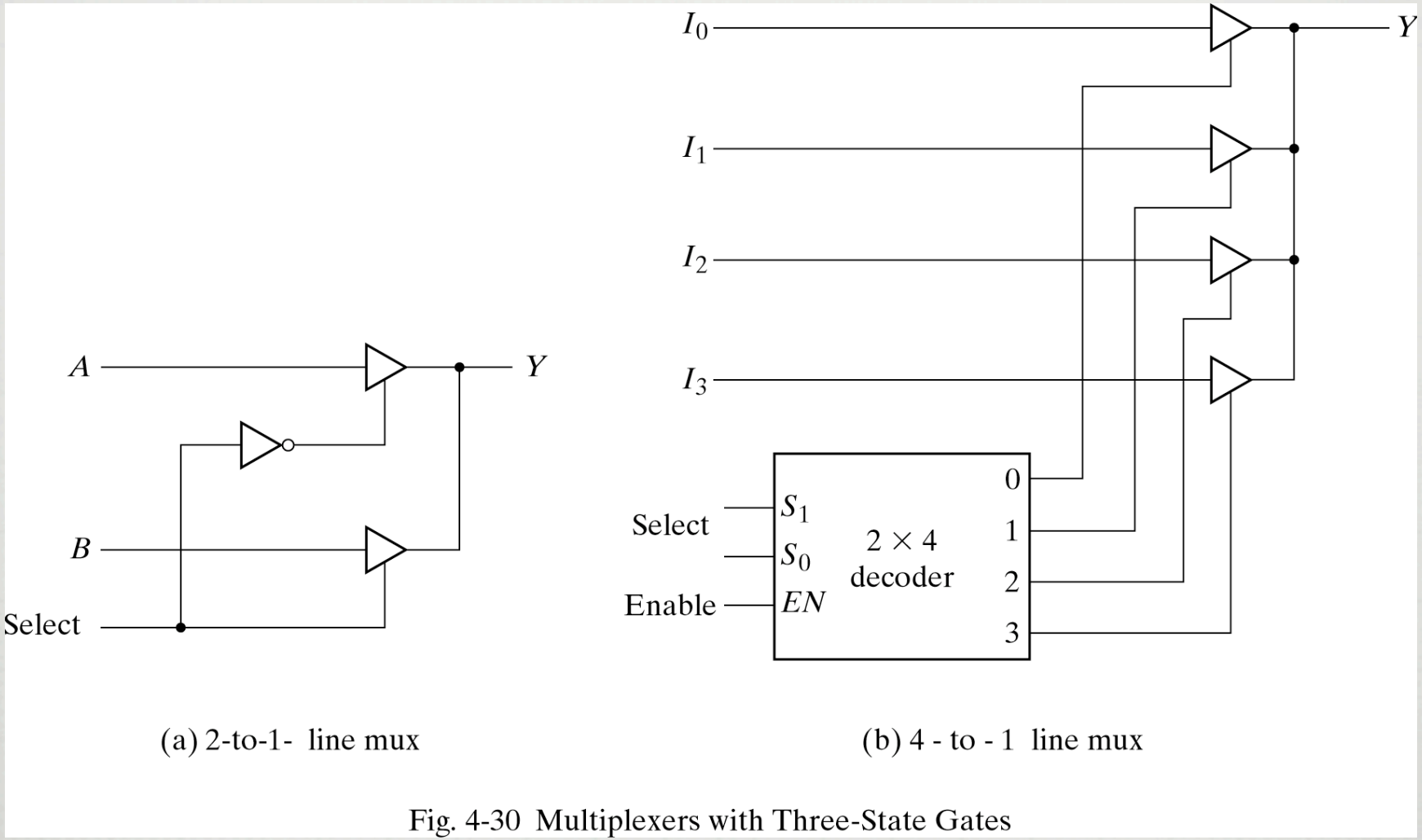


Fig. 4-29 Graphic Symbol for a Three-State Buffer



(a) 2-to-1- line mux

(b) 4 - to - 1 line mux

Fig. 4-30 Multiplexers with Three-State Gates

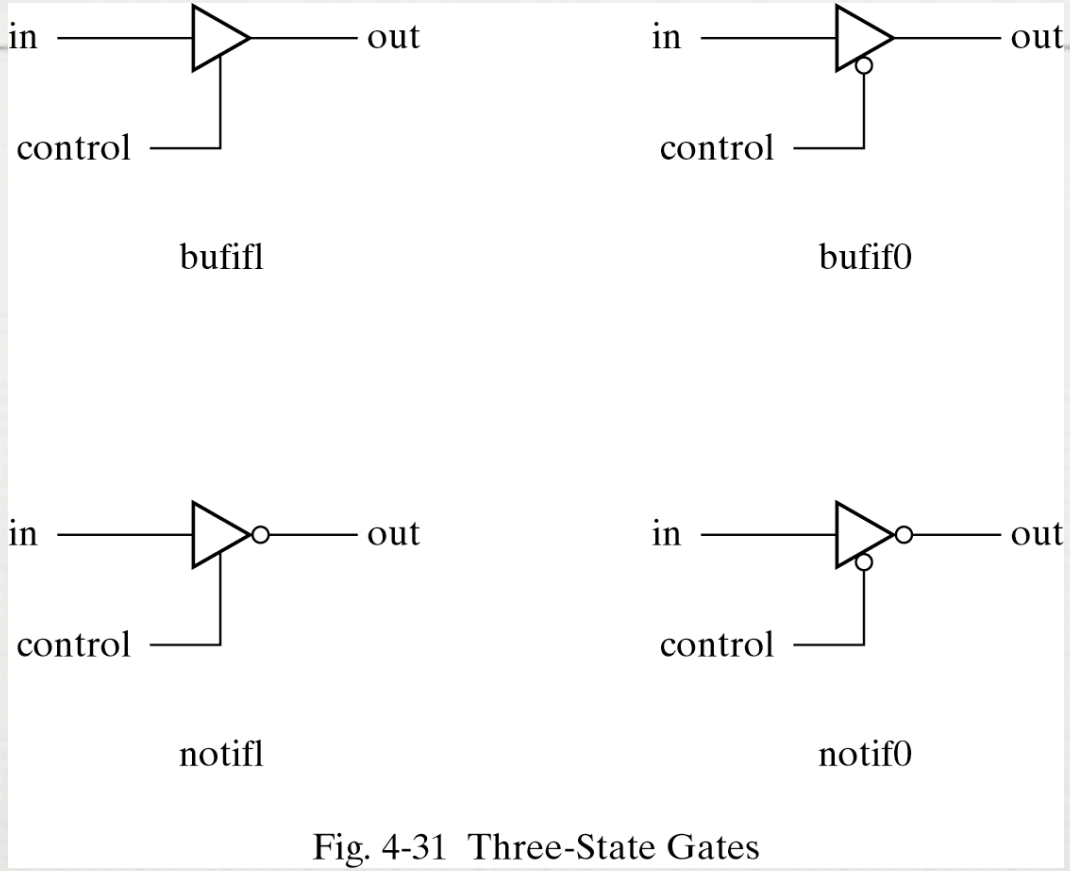


Fig. 4-31 Three-State Gates

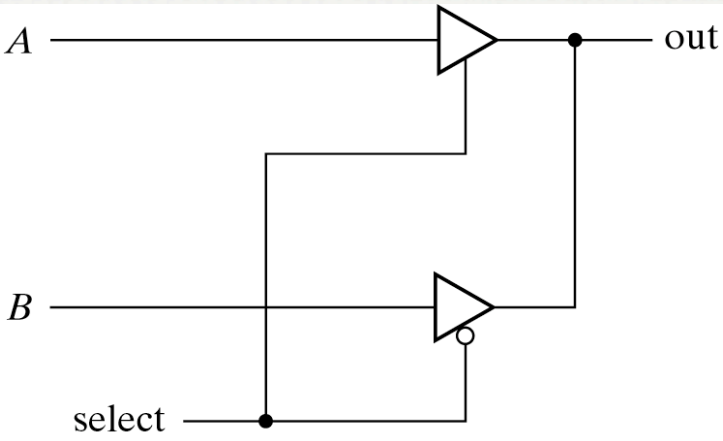


Fig. 4-32 2-to-1-Line Multiplexer with Three-State Buffers

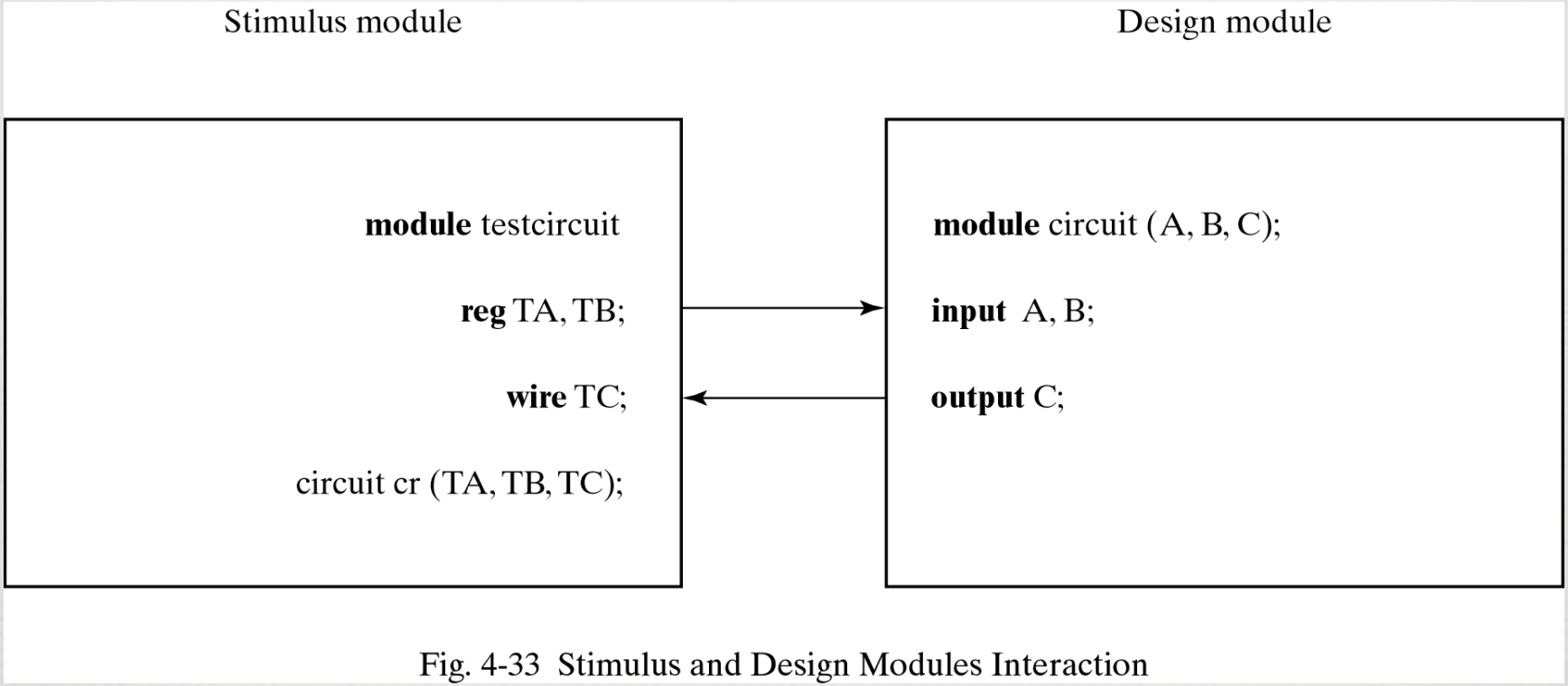


Fig. 4-33 Stimulus and Design Modules Interaction

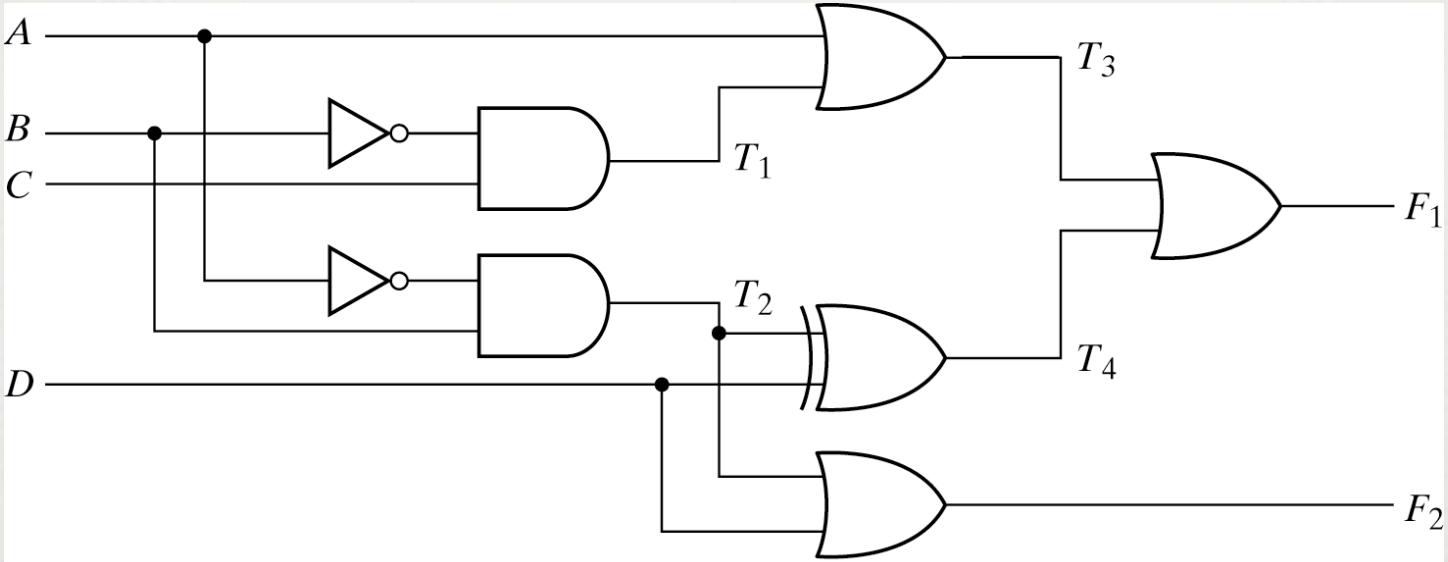


Fig. P4-1

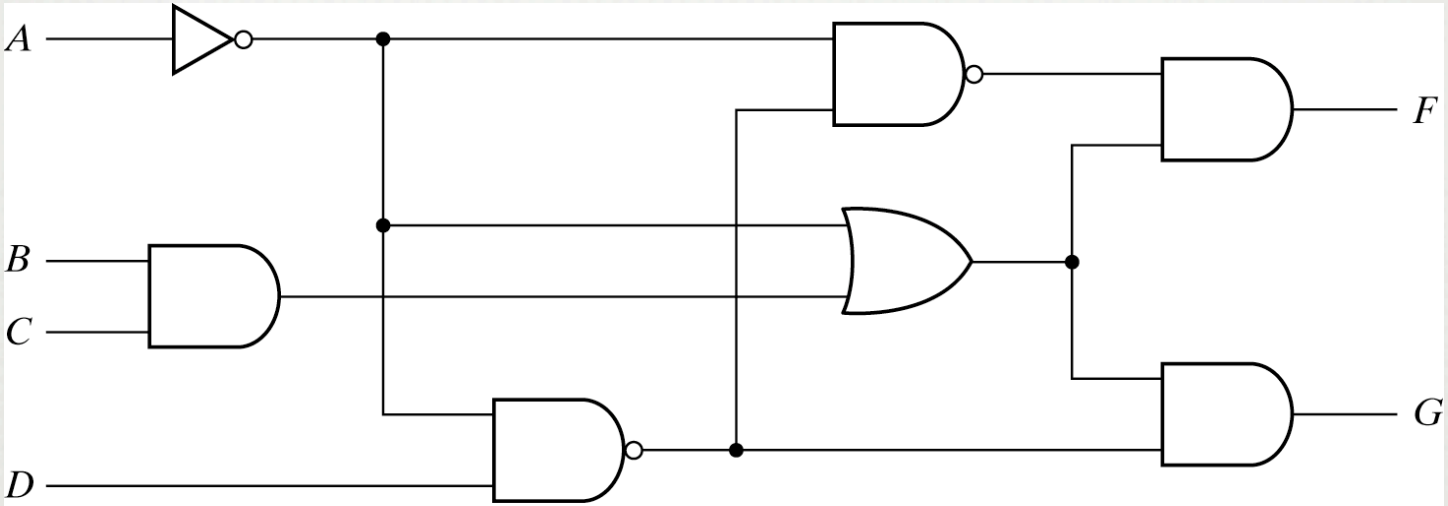
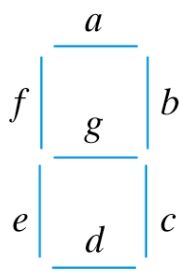


Fig. P4-2



(a) Segment designation

(b) Numerical designation for display

Fig. P4-9

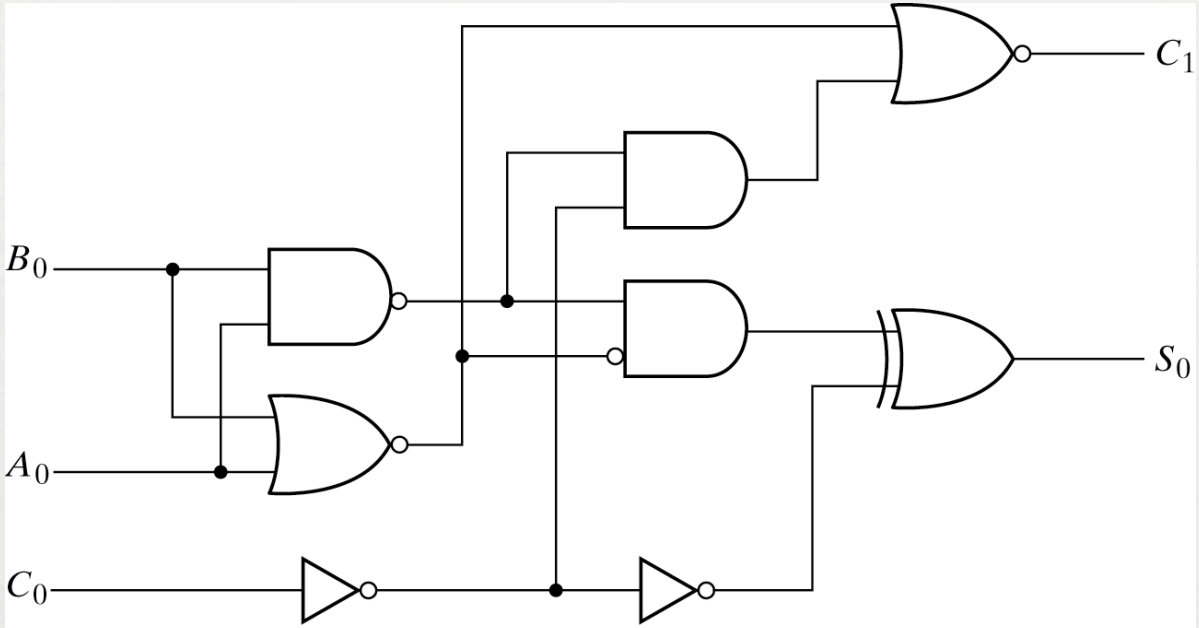


Fig. P4-17 First Stage of a Parallel Adder