Chapter 4

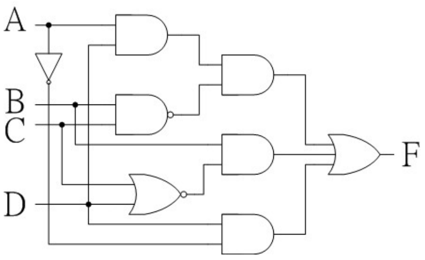
Combinational Logic

J.J. Shann

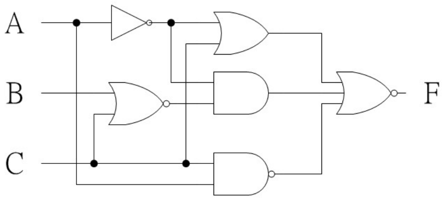
§4-3

1. Analyze the following circuit to obtain the simplified Boolean expression in sum-of-products form for output *F* in terms of the input variables:

(a)



(b)



§4-4

2. Design a combinational circuit by the following steps:
   i. Develop the truth table of the circuit.
   ii. Apply two-level optimization. Derive both the sum-of products (SoP) and product-of-sums (PoS) forms and calculate the gate input count for each form.
   iii. Repeat ii by applying multiple-level optimization.

(a) The combinational circuit accepts a 4-bit binary number (*w x y z*) and generate an output (*f*) which is equal to 1 only if the input is greater than twelve or less than three.

(b) The combinational circuit accepts a two 2-bit binary numbers A_1A_0 and B_1B_0 and generate the product ($P_3P_2P_1P_0$) of them.

EX-3

(a)

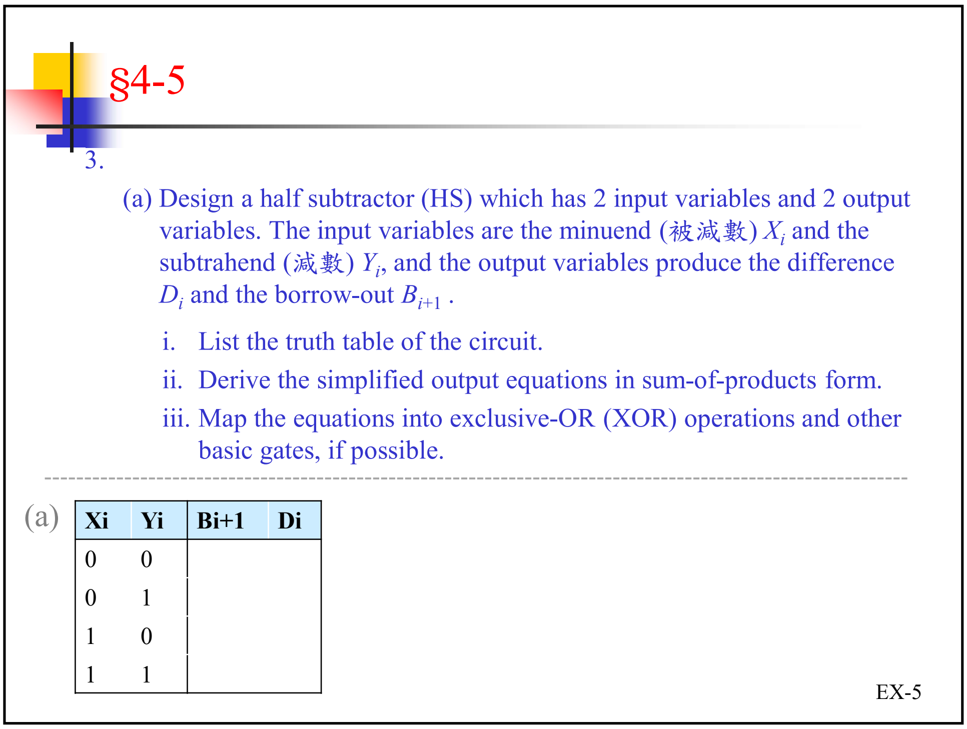
w	***x***	***y***	***z***	***f***
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

The combinational circuit accepts a 4-bit binary number (w x y z) and generate an output (f) which is equal to 1 only if the input is greater than twelve or less than three.

wx \ *yz*	00	01	11	10
00				
01				
11				
10				

wx \ *yz*	00	01	11	10
00				
01				
11				
10				

EX-4



§4-5

3.

(a) Design a half subtractor (HS) which has 2 input variables and 2 output variables. The input variables are the minuend (被減數) X_i and the subtrahend (減數) Y_i, and the output variables produce the difference D_i and the borrow-out B_{i+1} .

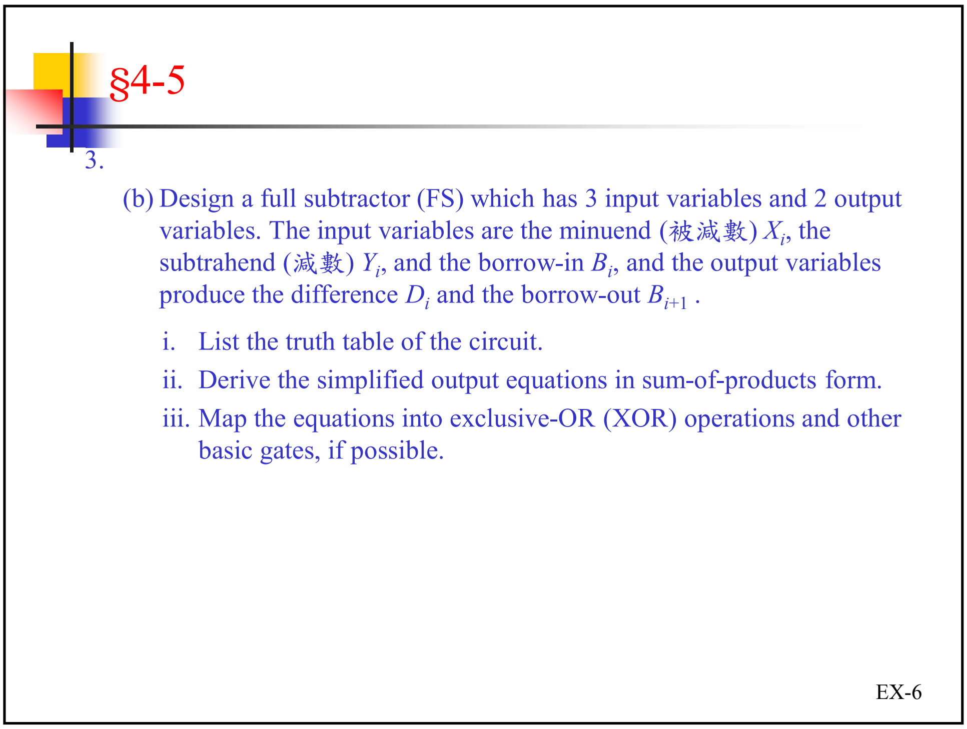
i. List the truth table of the circuit.

ii. Derive the simplified output equations in sum-of-products form.

iii. Map the equations into exclusive-OR (XOR) operations and other basic gates, if possible.

(a)

Xi	Yi	Bi+1	Di
0	0		
0	1		
1	0		
1	1		



§4-5

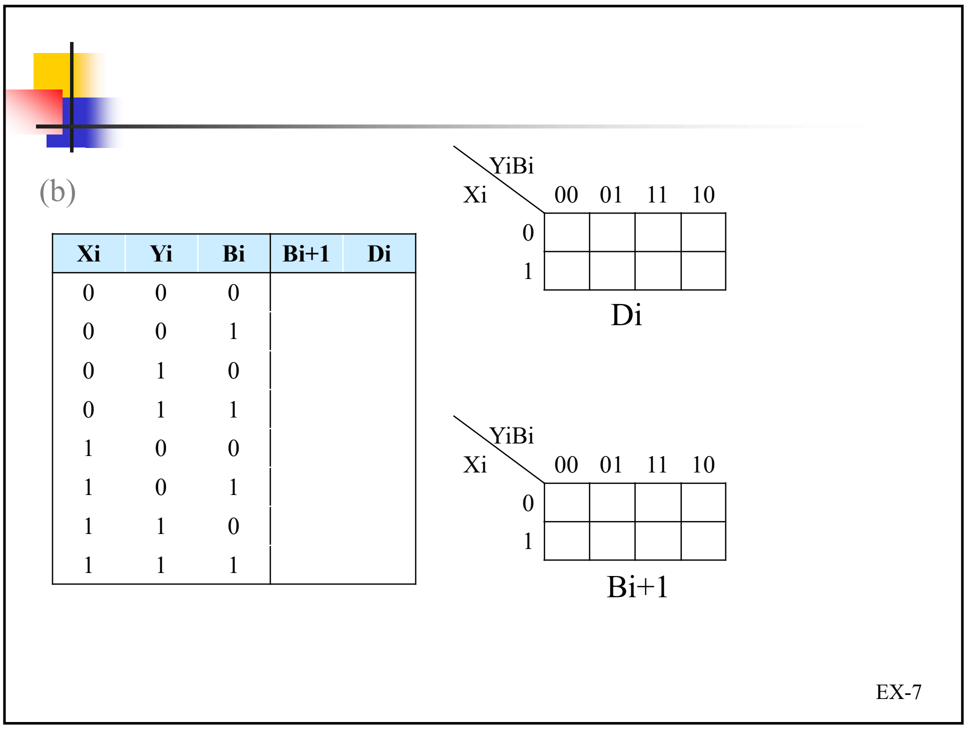
3.

(b) Design a full subtractor (FS) which has 3 input variables and 2 output variables. The input variables are the minuend (被減數) X_i, the subtrahend (減數) Y_i, and the borrow-in B_i, and the output variables produce the difference D_i and the borrow-out B_{i+1} .

i. List the truth table of the circuit.

ii. Derive the simplified output equations in sum-of-products form.

iii. Map the equations into exclusive-OR (XOR) operations and other basic gates, if possible.



(b)

Xi	Yi	Bi	Bi+1	Di
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

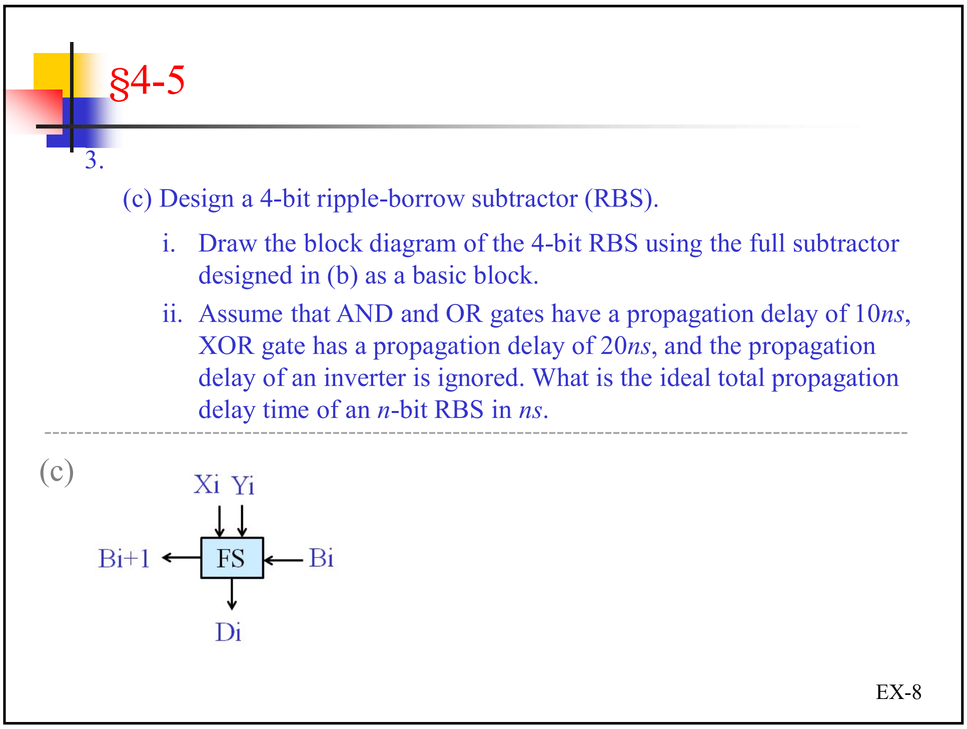
Xi \ YiBi	00	01	11	10
0				
1				

Di

Xi \ YiBi	00	01	11	10
0				
1				

Bi+1

EX-7



§4-5

3.

(c) Design a 4-bit ripple-borrow subtractor (RBS).

i. Draw the block diagram of the 4-bit RBS using the full subtractor designed in (b) as a basic block.

ii. Assume that AND and OR gates have a propagation delay of 10*ns*, XOR gate has a propagation delay of 20*ns*, and the propagation delay of an inverter is ignored. What is the ideal total propagation delay time of an *n*-bit RBS in *ns*.

(c)

EX-8

§4-5

3.

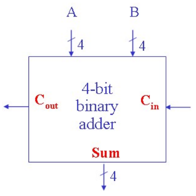
(d) Design a 4-bit borrow-look-ahead (BLA) subtractor.

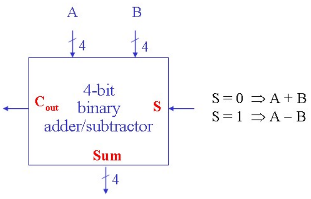
i. Derive the equation for each borrow generate G_i, borrow propagate P_i, output borrow B_{i+1}, and difference bit D_i , where $i = 0, 1, 2, 3$.

ii. Assume that the propagation delays of XOR, AND, OR, and NOT gates are the same as that describe in (c). What is the ideal total propagation delay time of an *n*-bit single-level BLA subtractor in *ns*? Why?

§4-6

4. Given the excess-3 code as follows, design an excess-3 adder that adds two decimal digits in excess-3 code and a carry-in bit, and generates a carry out bit C_{out} and excess-3 coded sum bits $S_3S_2S_1S_0$. First, add these two digits and carry in bit as binary numbers. If its carry out K_{out} is equal to 1, then add 3 to its sum $Z_3Z_2Z_1Z_0$; otherwise, subtract 3 from the sum for correct results. Explain why the algorithm described above is correct and draw the block diagram of the excess-3 adder. Use binary adder and binary adder-subtractor as basic building blocks.





Decimal digits	Excess-3 code
0	0011
1	0100
2	0101
3	0110
4	0111
5	1000
6	1001
7	1010
8	1011
9	1100

<Ans>

K	Z3	Z2	Z1	Z0	C	S3	S2	S1	S0	Sum
										0
										1
										2
										3
										4
										5
										6
										7
										8
										9
										10
										11
										12
										13
										14
										15
										16
										17
										18
										19

§4-9

5. (a) A combinational circuit is defined by the following Boolean functions:

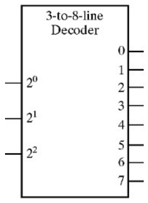
$$f1(x, y, z) = \Sigma m(1, 4, 5)$$
$$f2(x, y, z) = \Sigma m(0, 2, 3, 4, 6, 7)$$

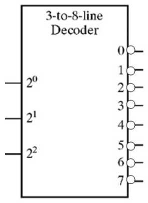
i. Design the circuit using a 3-to-8-line decoder with ***active HIGH*** outputs and external gates. Minimize the number of inputs of each gate.

ii. Design the circuit using a 3-to-8-line decoder with ***active LOW*** outputs and external gates. Minimize the number of inputs of each gate.

(b) Repeat (a) for the following functions:

$$f1(x, y, z) = \Sigma m(0, 2, 3, 5, 7)$$
$$f2(x, y, z) = \Sigma m(1, 2, 4)$$





§4-10

6. (a) Design a 4-input priority encoder with inputs D3, D2, D1, and D0, and outputs A1, A0, and V. Assume that input D0 has the highest priority and input D3 has the lowest priority.

   i. Derive the truth table of the priority encoder.

   ii. Derive the Boolean expression for each output.

   (b) Repeat (a) for a hepta (7)-binary priority encoder. Assume that 6 has the highest priority and 0 has the lowest priority.

(a)

D0	D1	D2	D3	A1	A0	V
0	0	0	0			
1	x	x	x			
0	1	x	x			
0	0	1	x			
0	0	0	1			

§4-11

7. (a) Given the following function, Implement the function *f* and draw the block diagram using

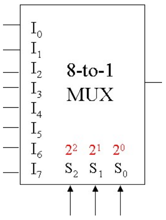
   $f(\mathrm{w}, x, y, z) = \Sigma m(3, 4, 6, 7, 8, 9, 10, 11, 13, 14)$

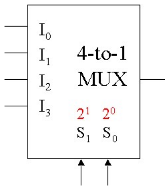
   i. a 8×1 multiplexer and external gates, if necessary. Connect inputs *w*, *x*, and *y* to the MUX selection lines. Show the truth table for deriving the inputs of the MUX.

   ii. a 4×1 multiplexer and external gates, if necessary. Connect *w* and *x* to the MUX selection line. Show the truth table for deriving the inputs of the MUX.

   (b) Repeat (a) for the following function:

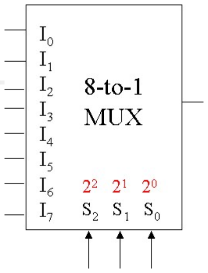
   $f(\mathrm{w}, x, y, z) = \Sigma m(2, 5, 6, 8, 9, 12, 13, 14)$

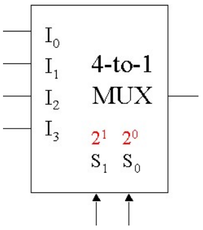




(a) $f(\mathrm{w}, x, y, z) = \Sigma m(3, 4, 6, 7, 8, 9, 10, 11, 13, 14)$

w	*x*	*y*	*z*	*f*
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

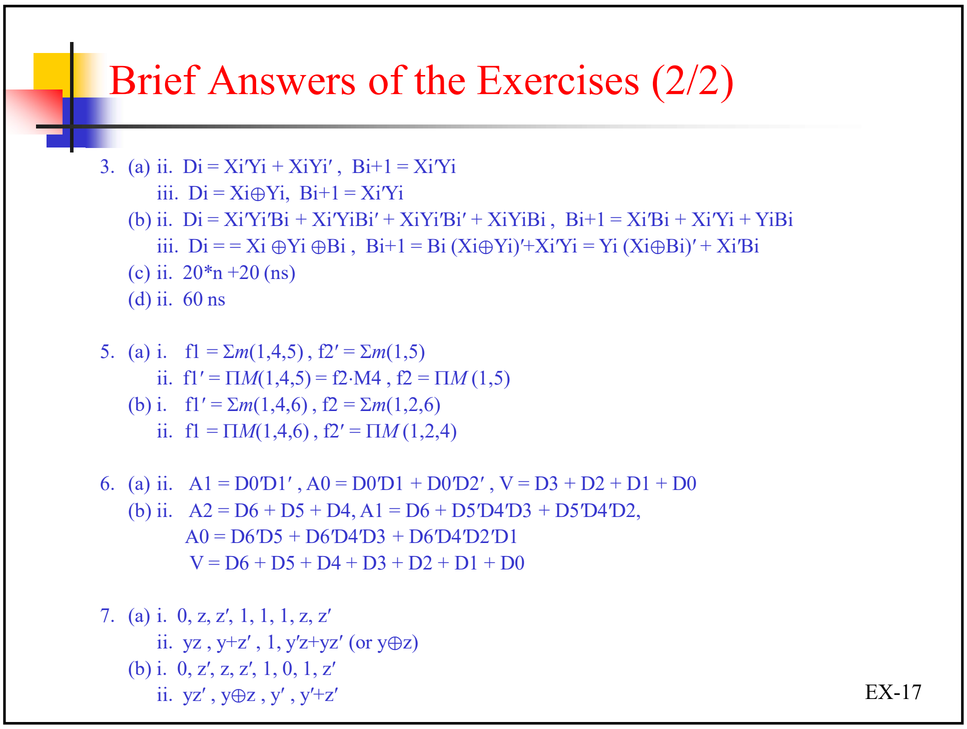




Brief Answers of the Exercises (1/2)

1. (a) F = A′D + BC′ + B′D
   (b) F = 0

2. (a) ii. SOP: F = w′x′y′ + w′x′z′ + wxz + wxy , GIC = 20
   POS: F = (w + x′) (w′ + x) (w′ + y + z) (w + y′ + z′) , GIC = 18
   iii. SOP: F = w′x′ (y′ + z′) + wx (z + y), GIC = 16
   POS: No multiple-level optimization.
   (b) ii. SOP: P3 = A1A0B1B0, P2 = A1A0′B1 + A1B1B0′
   P1 = A1′A0B1 + A0B1B0′ + A1B1′B0 + A1A0′B0, P0 = A0B0
   GIC = 34
   POS: P3 = A1A0B1B0, P2 = A1B1 (A0′ + B0′)
   P1 = (B1 + B0) (A1 + A0) (A0 + B0) (A1 + B1) (A1′ + A0′ + B1′ + B0′)
   P0 = A0B0
   GIC = 32
   iii. SOP: w = A0B0 , x = A1B1 , y = A0B1, z = A1B0
   P3 = wx , P2 = xw′ , P1 = yz′ + zy′ , P0 = w
   GIC = 21
   POS: w = A0B0 , x = A1B1
   P3 = wx, P2 = xw′, P1 = (B1 + B0) (A1 + A0) w x (w′ + x′) , P0 = w
   GIC = 21



Brief Answers of the Exercises (2/2)

3. (a) ii. Di = Xi′Yi + XiYi′ , Bi+1 = Xi′Yi
   iii. Di = Xi⊕Yi, Bi+1 = Xi′Yi
   (b) ii. Di = Xi′Yi′Bi + Xi′YiBi′ + XiYi′Bi′ + XiYiBi , Bi+1 = Xi′Bi + Xi′Yi + YiBi
   iii. Di = = Xi ⊕Yi ⊕Bi , Bi+1 = Bi (Xi⊕Yi)′+Xi′Yi = Yi (Xi⊕Bi)′ + Xi′Bi
   (c) ii. 20*n +20 (ns)
   (d) ii. 60 ns

5. (a) i. f1 = Σ*m*(1,4,5) , f2′ = Σ*m*(1,5)
   ii. f1′ = Π*M*(1,4,5) = f2·M4 , f2 = Π*M* (1,5)
   (b) i. f1′ = Σ*m*(1,4,6) , f2 = Σ*m*(1,2,6)
   ii. f1 = Π*M*(1,4,6) , f2′ = Π*M* (1,2,4)

6. (a) ii. A1 = D0′D1′ , A0 = D0′D1 + D0′D2′ , V = D3 + D2 + D1 + D0
   (b) ii. A2 = D6 + D5 + D4, A1 = D6 + D5′D4′D3 + D5′D4′D2,
   A0 = D6′D5 + D6′D4′D3 + D6′D4′D2′D1
   V = D6 + D5 + D4 + D3 + D2 + D1 + D0

7. (a) i. 0, z, z′, 1, 1, 1, z, z′
   ii. yz , y+z′ , 1, y′z+yz′ (or y⊕z)
   (b) i. 0, z′, z, z′, 1, 0, 1, z′
   ii. yz′ , y⊕z , y′ , y′+z′