

Chapter 3

Combinational Logic Circuits

3.1 Problems

Problem 3.1 (Implementing an OR gate using AND and NOT) Implement a 2-input OR gate using only AND and NOT gates. (Hint: Start with the truth table that implements \overline{Y} .)

(ans:

The truth table for an OR gate follows.

A	B	$Y = A + B$	\overline{Y}
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

$$\overline{Y} = \overline{A} \cdot \overline{B}$$

Hence, connect A to a NOT gate and then one input of the AND, connect B to a NOT gate and then to the other input of the AND, and connect the AND output to an NOT gate to form

$$Y = \overline{(\overline{Y})} = \overline{(\overline{A} \cdot \overline{B})} = A + B$$

)

Problem 3.2 (Using only NAND gates) This problem demonstrates that only a NAND gate is needed to implement the basic gates and, hence, any combinatorial logic circuit. Using only two-input NAND gates, implement a NOT, an OR, and an AND gate.

(ans:

The truth table for a NAND gate follows.

A	B	$Y = \overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

To form NOT gate, connect A and B together to form new A. Hence, only $AB = 00$ and $AB = 11$ occur.

For AND gate connect NOT gate to NAND output.

The truth table for an OR gate follows.

A	B	$Y = A + B$	\overline{Y}
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Implement

$$\overline{Y} = \overline{A} \cdot \overline{B}$$

Hence, connect A to NOT (implemented with a NAND) and then to AND (implemented with a NAND), connect B to NOT (implemented with a NAND) and then to AND (implemented with a NAND), and connect AND output to NOT (implemented with a NAND) to form

$$Y = \overline{(\overline{Y})} = \overline{(\overline{A} \cdot \overline{B})} = A + B$$

The careful student will note that the output NOT can be eliminated if the NOT at the output of the NAND that implements that AND is eliminated.

)

Problem 3.3 (From logic equation to truth table) Generate the truth table that corresponds to the logic equation given by

$$Y = A \cdot \overline{B} + B \cdot \overline{C} + C \cdot \overline{A}$$

Implement a logic circuit that has a small number of gates.

(ans:

A	B	C	Y	Reason
0	0	0	0	Remaining 0
0	0	1	1	$C \cdot \overline{A} = 1$
0	1	0	1	$B \cdot \overline{C} = 1$
0	1	1	0	Remaining 0
1	0	0	1	$A \cdot \overline{B} = 1$
1	0	1	1	$A \cdot \overline{B} = 1$
1	1	0	1	$B \cdot \overline{C} = 1$
1	1	1	0	Remaining 0

Logic circuit 1: Implement

$$\overline{Y} = \overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot C$$

$$Y = \overline{(\overline{Y})} = \overline{\overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot C}$$

Logic circuit 2: Last two terms can be combined to implement

$$\begin{aligned}\bar{Y} &= \bar{A} \cdot \bar{B} \cdot \bar{C} + B \cdot C \\ Y &= \overline{(\bar{Y})} = \overline{\bar{A} \cdot \bar{B} \cdot \bar{C} + B \cdot C}\end{aligned}$$

)

Problem 3.4 (Distributive law) Use the distributive law to generate the truth table that corresponds to the logic equation given by

$$Y = A \cdot (\bar{B} + C)$$

Implement one logic circuit directly from the logic equation. Implement a second logic circuit from the truth table using the sum-of-products approach. Advanced courses in logic design teach how to implement logic circuits having the minimum number of gates.

(ans:

$$Y = A \cdot \bar{B} + A \cdot C$$

A	B	C	Y	Reason
0	0	0	0	Remaining 0
0	0	1	0	Remaining 0
0	1	0	0	Remaining 0
0	1	1	0	Remaining 0
1	0	0	1	$A \cdot \bar{B} = 1$
1	0	1	1	$A \cdot \bar{B} = 1$
1	1	0	0	Remaining 0
1	1	1	1	$A \cdot C = 1$

Logic circuit 1: Implement

$$Y = A \cdot \bar{B} \cdot \bar{C} + A \cdot \bar{B} \cdot C + A \cdot B \cdot C$$

)

Problem 3.5 (Logic circuit analysis) Generate the truth table that corresponds to the logic circuit shown in Figure 5.36. Draw a different logic circuit that implements this truth table.

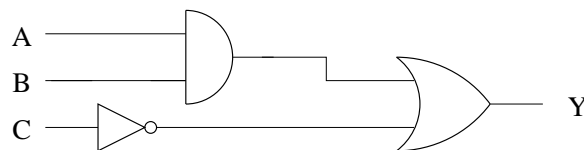


Figure 5.35 Logic circuit for Problem 5.5.

(ans:

Logic equation

$$Y = A \cdot B + \bar{C}$$

A	B	C	Y	Reason
0	0	0	1	$\overline{C} = 1$
0	0	1	0	Remaining 0
0	1	0	1	$\overline{C} = 1$
0	1	1	0	Remaining 0
1	0	0	1	$\overline{C} = 1$
1	0	1	0	Remaining 0
1	1	0	1	$A \cdot B = 1$
1	1	1	1	$A \cdot B = 1$

$$\overline{Y} = \overline{A} \cdot \overline{B} \cdot C + \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot C$$

$$Y = \overline{(\overline{Y})} = \overline{\overline{A} \cdot \overline{B} \cdot C + \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot C}$$

)

Problem 3.6 (Proving de Morgan's second law) Starting with the truth table that implements $\overline{A + B}$, implement an alternative logic circuit that proves de Morgan's second law given in Eq. (5.10).

(ans:

de Morgan's second law is

$$\overline{A + B} = \overline{A} \cdot \overline{B}$$

Entering the Y values from the logic equation $Y = \overline{A + B}$ produces the following truth table.

A	B	Y = $\overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	0

We observe that $Y = \overline{A} \cdot \overline{B}$, proving de Morgan's second law.

)

Problem 3.7 (7-Segment display: Lighting the b segment) Implement the logic circuit that recognizes the BCD codes that light the b segment in a 7-segment display.

(ans:

From the truth table in Figure 5.29, we make all don't cares $\times = 1$, resulting in two $b = 0$ values - for digits 5 and 6. We implement

$$\overline{b} = (\overline{b_3} \cdot b_2 \cdot \overline{b_1} \cdot b_0) + (\overline{b_3} \cdot b_2 \cdot b_1 \cdot \overline{b_0})$$

$$b = \overline{(\overline{b})} = \overline{(\overline{b_3} \cdot b_2 \cdot \overline{b_1} \cdot b_0) + (\overline{b_3} \cdot b_2 \cdot b_1 \cdot \overline{b_0})}$$

)

Problem 3.8 (Digital adder: Carry bit logic circuit) Extending Figure 5.32, implement the logic circuit that produces the carry bit C_n from the inputs A_n , B_n , and C_{n-1} .

(ans:

$$C_n = (\overline{A_n} \cdot B_n \cdot C_{n-1}) + (A_n \cdot \overline{B_n} \cdot C_{n-1}) + (A_n \cdot B_n \cdot \overline{C_{n-1}}) + (A_n \cdot B_n \cdot C_{n-1})$$

)

Problem 3.9 (Direct implementation of binary multiplication) Using the truth table in Figure 5.34, implement the four logic circuits that produce each term in the product $P_3P_2P_1P_0$.

(ans: Enclose each product in parentheses to help the interpretation of the logic equation.

$$P_3 = A_1 \cdot A_0 \cdot B_1 \cdot B_0$$

$$P_2 = (A_1 \cdot \overline{A_0} \cdot B_1 \cdot \overline{B_0}) + (A_1 \cdot \overline{A_0} \cdot B_1 \cdot B_0) + (A_1 \cdot A_0 \cdot B_1 \cdot \overline{B_0})$$

This can be simplified by combining the last two terms to produce

$$P_2 = (A_1 \cdot \overline{A_0} \cdot B_1 \cdot \overline{B_0}) + (A_1 \cdot \overline{A_0} \cdot B_1)$$

$$\begin{aligned} P_1 = & (\overline{A_1} \cdot A_0 \cdot B_1 \cdot \overline{B_0}) + (\overline{A_1} \cdot A_0 \cdot B_1 \cdot B_0) + (A_1 \cdot \overline{A_0} \cdot \overline{B_1} \cdot B_0) \\ & + (A_1 \cdot \overline{A_0} \cdot B_1 \cdot B_0 + A_1 \cdot A_0 \cdot \overline{B_1} \cdot B_0) + (A_1 \cdot A_0 \cdot B_1 \cdot \overline{B_0}) \end{aligned}$$

This can be simplified by combining the first two terms to produce

$$\begin{aligned} P_1 = & (\overline{A_1} \cdot A_0 \cdot B_1 + A_1 \cdot \overline{A_0} \cdot \overline{B_1} \cdot B_0) \\ & + (A_1 \cdot \overline{A_0} \cdot B_1 \cdot B_0) + (A_1 \cdot A_0 \cdot \overline{B_1} \cdot B_0) + (A_1 \cdot A_0 \cdot B_1 \cdot \overline{B_0}) \end{aligned}$$

$$P_0 = (\overline{A_1} \cdot A_0 \cdot \overline{B_1} \cdot B_0) + (\overline{A_1} \cdot A_0 \cdot B_1 \cdot B_0) + (A_1 \cdot A_0 \cdot \overline{B_1} \cdot B_0) + (A_1 \cdot A_0 \cdot B_1 \cdot B_0)$$

)

Problem 3.10 (Direct implementation of binary division) Using the truth table in Figure 5.35, design the logic circuit to produce the quotient Q_1Q_0 and divide-by-zero error E .

(ans:

Q_1 : Making all $\times = 0$ produces only two terms that equal 1.

$$Q_1 = (A_1 \cdot \overline{A_0} \cdot \overline{B_1} \cdot B_0) + (A_1 \cdot A_0 \cdot \overline{B_1} \cdot B_0)$$

Q_0 : Making all $\times = 1$ produces 7 terms that equal 0.

$$\begin{aligned} \overline{Q_0} = & (\overline{A_1} \cdot \overline{A_0} \cdot \overline{B_1} \cdot B_0) + (\overline{A_1} \cdot \overline{A_0} \cdot B_1 \cdot \overline{B_0}) + (\overline{A_1} \cdot \overline{A_0} \cdot B_1 \cdot B_0) \\ & + (\overline{A_1} \cdot A_0 \cdot B_1 \cdot \overline{B_0}) + (\overline{A_1} \cdot A_0 \cdot B_1 \cdot B_0) + (A_1 \cdot \overline{A_0} \cdot \overline{B_1} \cdot B_0) \\ & + (A_1 \cdot \overline{A_0} \cdot B_1 \cdot B_0) \end{aligned}$$

Combining terms produces 5 terms:

$$\begin{aligned}\overline{Q_0} &= (\overline{A_1} \cdot \overline{A_0} \cdot \overline{B_1} \cdot B_0) + (\overline{A_1} \cdot \overline{A_0} \cdot B_1) \\ &\quad + (\overline{A_1} \cdot A_0 \cdot B_1) + (A_1 \cdot \overline{A_0} \cdot \overline{B_1} \cdot B_0) \\ &\quad + (A_1 \cdot \overline{A_0} \cdot B_1 \cdot B_0)\end{aligned}$$

$$Q_0 = \overline{(\overline{Q_0})}$$

$E = 1$ only when $B_1B_0 = 00$:

$$E = \overline{B_1} \cdot \overline{B_0}$$

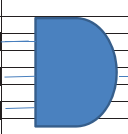
)

3.2 Excel Projects

Project 3.1 (3-Input AND gate) *Extend Example 13.13 to implement a 3-input AND gate.*

(ans:

1	A	B	C	D	E	F	G	H	I	J	K
2	A	B	C		Y = A-B-C						
3	0	0	0		0						
4	0	0	1		0						
5	0	1	0		0						
6	0	1	1		0						
7	1	0	0		0						
8	1	0	1		0						
9	1	1	0		0						
10	1	1	1		1						



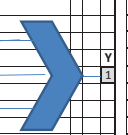
1	A	B	C	D	E	F	G	H	I	J	K
2	A	B	C		Y = A-B-C						
3	0	0	0		=IF(AND(A3,B3,C3),1,0)						
4	0	0	1		=IF(AND(A4,B4,C4),1,0)						
5	0	1	0		=IF(AND(A5,B5,C5),1,0)						
6	0	1	1		=IF(AND(A6,B6,C6),1,0)						
7	1	0	0		=IF(AND(A7,B7,C7),1,0)						
8	1	0	1		=IF(AND(A8,B8,C8),1,0)						
9	1	1	0		=IF(AND(A9,B9,C9),1,0)						
10	1	1	1		=IF(AND(A10,B10,C10),1,0)						

)

Project 3.2 (3-Input OR gate) *Extend Example 13.14 to implement a 3-input OR gate.*

(ans:

1	A	B	C	D	E	F	G	H	I	J	K	L
2	A	B	C		Y = A+B+C							
3	0	0	0		0							
4	0	0	1		1							
5	0	1	0		1							
6	0	1	1		1							
7	1	0	0		1							
8	1	0	1		1							
9	1	1	0		1							
10	1	1	1		1							



1	A	B	C	D	E	F	G	H	I	J	K	L
2	A	B	C		Y = A+B+C							
3	0	0	0		=IF(OR(A3,B3,C3),1,0)							
4	0	0	1		=IF(OR(A4,B4,C4),1,0)							
5	0	1	0		=IF(OR(A5,B5,C5),1,0)							
6	0	1	1		=IF(OR(A6,B6,C6),1,0)							
7	1	0	0		=IF(OR(A7,B7,C7),1,0)							
8	1	0	1		=IF(OR(A8,B8,C8),1,0)							
9	1	1	0		=IF(OR(A9,B9,C9),1,0)							
10	1	1	1		=IF(OR(A10,B10,C10),1,0)							

)

Project 3.3 (Decimal to binary conversion) *Extend Example 13.16 to convert a decimal number into its 4-bit representation, with each bit in a separate column.*

(ans:

	A	B	C	D	E	F
1	Decimal		b ₃	b ₂	b ₁	b ₀
2	13		1	1	0	1

	A	B	C	D	E	F
1	Decimal		b ₃	b ₂	b ₁	b ₀
2	13		=MOD(INT(A2/8),2)	=MOD(INT(A2/4),2)	=MOD(INT(A2/2),2)	=MOD(A2,2)

)

Project 3.4 (Verifying logic equations) Using Example 13.17 as a guide, compose a worksheet that implements the truth table for the following logic equations.

$$1. Y = (A \cdot B) + (\bar{A} \cdot B)$$

$$2. Y = (A + \bar{B}) \cdot C$$

$$3. Y = A + B \cdot \bar{C} + \bar{A} \cdot \bar{B} \cdot C$$

(ans: The following show the VBA Macros in this problem. Sub up3 is assigned to up-arrows in 2-input worksheets and Sub up7 is assigned to up-arrows in 3-input worksheets.

```
Sub down()
If Range("A2").Value > 0 Then
    Range("A2").Value = Range("A2").Value - 1
End If
End Sub



-----



Sub up7()
If Range("A2").Value < 7 Then
    Range("A2").Value = Range("A2").Value + 1
End If
End Sub

-----



End Sub
Sub up3()
If Range("A2").Value < 3 Then
    Range("A2").Value = Range("A2").Value + 1
End If
End Sub
```

1.

	A	B	C	D	E	F
1	row		A	B		$Y=(A \cdot B) + (\bar{A} \cdot B)$
2	3		1	1		1
3						
4						

	A	B	C	D	E	F
1	row		A	B		$Y=(A \cdot B) + (\bar{A} \cdot B)$
2	3		=MOD(INT(A2/2),2)	=MOD(INT(A2),2)		=IF(OR(AND(C2,D2),AND(NOT(C2),D2)),1,0)
3						
4						

2.

	A	B	C	D	E	F	G	H
1	row		A	B	C		Y	$Y=(A+\bar{B}) \cdot C$
2	6		1	1	0		0	$Y=(A \cdot C) + (\bar{B} \cdot C)$
3								
4								

	A	B	C	D	E	F	G
1	row		A	B	C		Y
2	7		=MOD(INT(A2/4),2)	=MOD(INT(A2/2),2)	=MOD(A2,2)		=IF(OR(AND(C2,E2),AND(NOT(D2),E2)),1,0)

3.

	A	B	C	D	E	F	G
1	row		A	B	C		$Y=A + (B \cdot \sim C) + (\sim A \cdot \sim B \cdot C)$
2	4		1	0	0		1

	A	B	C	D	E	F	G
1	row		A	B	C		$Y=A + (B \cdot \sim C) + (\sim A \cdot \sim B \cdot C)$
2	4		=MOD(INT(A2/4),2)	=MOD(INT(A2/2),2)	=MOD(A2,2)		=IF(OR(C2, AND(D2,NOT(E2))),AND(NOT(C2),NOT(D2),E2)),1,0)

)

Project 3.5 (Implementing a 7-segment display) Using Example 13.18 as a guide, compose a work-sheet that forms a 7-segment display based on the binary value of the digit, rather than the digit value.

(ans:

	A	B	C	D	E	F	G	H	I
1	TYPE digit						1		
2	7					0		1	
3						0			
4						0		1	
5	binary code					0			
6	b ₃	0		a=	1				
7	b ₂	1		b=	1				
8	b ₁	1		c=	1				
9	b ₀	1		d=	0				
10				e=	0				
11				f=	0				
12				g=	0				
13									

	A	B	C	D	E
6	b ₃	=MOD(INT(A2/8),2)		a=	=IF(NOT(OR(AND(NOT(B6),NOT(B7),NOT(B8),B9),AND(NOT(B6),B7,NOT(B8),NOT(B9)))),1,0)
7	b ₂	=MOD(INT(A2/4),2)		b=	=IF(NOT(OR(AND(NOT(B6),B7,NOT(B8),B9),AND(NOT(B6),B7,B8,NOT(B9)))),1,0)
8	b ₁	=MOD(INT(A2/2),2)		c=	=IF(NOT(AND(NOT(B6), NOT(B7),B8,NOT(B9))),1,0)
9	b ₀	=MOD(A2,2)		d=	=IF(NOT(OR(AND(NOT(B6),NOT(B7),NOT(B8),B9),AND(NOT(B6),B7,NOT(B8),NOT(B9))),AND(NOT(B6),B7,B8,B9),AND(B6,NOT(B7),NOT(B8),B9))),1,0)
10				e=	=IF(OR(AND(NOT(B6),NOT(B7),NOT(B8),NOT(B9)),AND(NOT(B6),NOT(B7),B8,NOT(B9)),AND(NOT(B6),B7,B8,NOT(B9))),AND(B6,NOT(B7),NOT(B8),NOT(B9))),1,0)
11				f=	=IF(NOT(OR(AND(NOT(B6),NOT(B7),NOT(B8),B9),AND(NOT(B6),NOT(B7),B8,NOT(B9))),AND(NOT(B6),NOT(B7),B8,B9),AND(NOT(B6),B7,B8,B9))),1,0)
12				g=	=IF(NOT(OR(AND(NOT(B6),NOT(B7),NOT(B8),NOT(B9)),AND(NOT(B6),NOT(B7),NOT(B8),B9),AND(NOT(B6),B7,B8,B9))),1,0)

	F	G	H
1		=E6	
2	=E11		=E7
3		=E12	
4	=E10		=E8
5		=E9	

)

