Solution Manual of Digital Logic And Computer Design 2nd Edition Morris Mano

Digital Control System (Tribhuvan Vishwavidalaya)

Solution Manual of Digital Logic & Computer Design (2th Ed.)

Morris Mano

Ch#1 – Ch#10

Published By: Muhammad Hassan Riaz Yousufi

To Read Online & Download: WWW.ISSUU.COM/SHEIKHUHASSAN

Problem Solutions to Problems Marked With a * in

Logic Computer Design Fundamentals, Ed. 2

C H A P T E R 1

1-1.

Decimal, Binary, Octal and Hexadecimal Numbers from $(16)_{10}$ to $(31)_{10}$

Dec	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Bin	1 0000	1 0001	1 0010	1 0011	1 0100	1 0101	1 0110	1 0111	1 1000	1 1001	1 1010	1 1011	1 1100	1 1101	1 1110	1 1111
Oct	20	21	22	23	24	25	26	27	30	31	32	33	34	35	36	37
Hex	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F

1-4.

$(1101001)_2 = 2^6 + 2^5 + 2^3 + 2^0 = 105$

$(10001011.011)_2 = 2^7 + 2^3 + 2^1 + 2^0 + 2^{-2} + 2^{-3} = 139.375$

$(10011010)_2 = 2^7 + 2^4 + 2^3 + 2^1 = 154$

1-7.

Decimal	Binary	Octal	Hexadecimal
369.3125	101110001.0101	561.24	171.5
189.625	10111101.101	275.5	BD.A
214.625	11010110.101	326.5	D6.A
62407.625	1111001111000111.101	171707.5	F3C7.A

1-9.

a)

7562/8	=	945 + 2/8	→	2
945/8	=	118 +1/8	→	1
118/8	=	14 + 6/8	→	6
14/8	=	1 + 6/8	→	6
1/8	=	1/8	→	1
0.45×8	=	3.6	→	3
0.60×8	=	4.8	→	4
0.80×8	=	6.4	→	6
0.20x8	=	3.2	→	3

$(7562.45)_{10} = (16612.3463)_8$

b) $(1938.257)_{10} = (792.41CA)_{16}$

c) $(175.175)_{10} = (10101111.001011)_2$

1-11.

a) $(673.6)_8 = (110\ 111\ 011.110)_2$

$= (1BB.C)_{16}$

b) $(E7C.B)_{16} = (1110\ 0111\ 1100.1011)_2$

$= (7174.54)_8$

c) $(310.2)_4 = (11\ 01\ 00.10)_2$

$= (64.4)_8$

1-15.

a) $(BEE)_r = (2699)_{10}$

$11 \times r^2 + 14 \times r^1 + 14 \times r^0 = 2699$

$11 \times r^2 + 14 \times r - 2685 = 0$

By the quadratic equation: $r = 15$ or $r \approx -16.27$

ANSWER: $r = 15$

b) $(365)_r = (194)_{10}$

$3 \times r^2 + 6 \times r^1 + 5 \times r^0 = 194$

$3 \times r^2 + 6 \times r - 189 = 0$

By the quadratic equation: $r = -9$ or 7

ANSWER: r = 7

1-17.

$(694)_{10} = (0110\ 1001\ 0100)_{BCD}$

$(835)_{10} = (1000\ 0011\ 0101)_{BCD}$

1		
0110	1001	0100
+1000	+0011	+0101
1111	1100	1001
+0110	+0110	+0000
0001 0101	1 0010	1001

1-20.

a) $(0100\ 1000\ 0110\ 0111)_{BCD} = (4867)_{10}$

$= (1001100000011)_2$

b) $(0011\ 0111\ 1000.0111\ 0101)_{BCD} = (378.75)_{10}$

$= (101111010.11)_2$

1-23.

a) $(101101101)_2$

b) $(0011\ 0110\ 0101)_{BCD}$

c) $0011\ \ 0011 \quad 0011\ \ 0110 \quad 0011\ \ 0101_{ASCII}$

1-25.

BCD Digits with Odd and Even Parity

	0	1	2	3	4	5	6	7	8	9
Odd	1 0000	0 0001	0 0010	1 0011	0 0100	1 0101	1 0110	0 0111	0 1000	1 1001
Even	0 0000	1 0001	1 0010	0 0011	1 0100	0 0101	0 0110	1 0111	1 1000	0 1001

Problem Solutions to Problems Marked With a * in

Logic Computer Design Fundamentals, Ed. 2

C H A P T E R 2

2-1.

a) $\overline{XYZ} = \overline{X} + \overline{Y} + \overline{Z}$

Verification of DeMorgan's Theorem

X	Y	Z	XYZ	$\overline{XYZ}$	$\overline{X}+\overline{Y}+\overline{Z}$
0	0	0	0	1	1
0	0	1	0	1	1
0	1	0	0	1	1
0	1	1	0	1	1
1	0	0	0	1	1
1	0	1	0	1	1
1	1	0	0	1	1
1	1	1	1	0	0

b) $X + YZ = (X + Y) \cdot (X + Z)$

The Second Distributive Law

X	Y	Z	YZ	$X+YZ$	$X+Y$	$X+Z$	$(X+Y)(X+Z)$
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

c) $\overline{X}Y + \overline{Y}Z + X\overline{Z} = X\overline{Y} + Y\overline{Z} + \overline{X}Z$

X	Y	Z	$\overline{X}Y$	$\overline{Y}Z$	$X\overline{Z}$	$\overline{X}Y+\overline{Y}Z+X\overline{Z}$	$X\overline{Y}$	$Y\overline{Z}$	$\overline{X}Z$	$X\overline{Y}+Y\overline{Z}+\overline{X}Z$
0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	0	0	1	1
0	1	0	1	0	0	1	0	1	0	1
0	1	1	1	0	0	1	0	0	1	1
1	0	0	0	0	1	1	1	0	0	1
1	0	1	0	1	0	1	1	0	0	1
1	1	0	0	0	1	1	0	1	0	1
1	1	1	0	0	0	0	0	0	0	0

2-2.

a) $\overline{X}\,\overline{Y} + \overline{X}Y + XY = \overline{X} + Y$

$= (\overline{X}Y + \overline{X}\,\overline{Y}) + (\overline{X}\,Y + XY)$

$= \overline{X}(Y + \overline{Y}) + Y(X + \overline{X}) +$

$= \overline{X} + Y$

b) $\overline{A}\,B + \overline{B}\,\overline{C} + AB + \overline{B}\,C = 1$

$= (\overline{A}\,B + AB) + (\overline{B}\,\overline{C} + \overline{B}\,C)$

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

$= B + \overline{B}$

$= 1$

c) $Y + \overline{X}\,Z + X\,\overline{Y} = X + Y + Z$

$= Y + X\,\overline{Y} + \overline{X}\,Z$

$= (Y + X)(Y + \overline{Y}) + \overline{X}\,Z$

$= Y + X + \overline{X}\,Z$

$= Y + (X + \overline{X})(X + Z)$

$= X + Y + Z$

d) $\overline{X}\,\overline{Y} + \overline{Y}\,Z + XZ + XY + Y\,\overline{Z} = \overline{X}\,\overline{Y} + XZ + Y\,\overline{Z}$

$= \overline{X}\,\overline{Y} + \overline{Y}\,Z(X + \overline{X}) + XZ + XY + Y\,\overline{Z}$

$= \overline{X}\,\overline{Y} + X\,\overline{Y}\,Z + \overline{X}\,\overline{Y}\,Z + XZ + XY + Y\,\overline{Z}$

$= \overline{X}\,\overline{Y}\,(1 + Z) + X\,\overline{Y}\,Z + XZ + XY + Y\,\overline{Z}$

$= \overline{X}\,\overline{Y} + XZ(1 + \overline{Y}) + XY + Y\,\overline{Z}$

$= \overline{X}\,\overline{Y} + XZ + XY\,(Z + \overline{Z}) + Y\,\overline{Z}$

$= \overline{X}\,\overline{Y} + XZ + XY\,Z + Y\,\overline{Z}\,(1 + X)$

$= \overline{X}\,\overline{Y} + XZ(1 + Y) + Y\,\overline{Z}$

$= \overline{X}\,\overline{Y} + XZ + Y\,\overline{Z}$

2-7.

a) $\overline{X}\,\overline{Y} + XYZ + \overline{X}Y = \overline{X} + XYZ = (\overline{X} + XY)(\overline{X} + Z)$

$= (\overline{X} + X)(\overline{X} + Y)(\overline{X} + Z) = (\overline{X} + Y)(\overline{X} + Z) = \overline{X} + YZ$

b) $X + Y(Z + \overline{X}\,\overline{Z}) = X + YZ + \overline{X}\,Y\,\overline{Z} = X + (YZ + \overline{X})(YZ + Y\overline{Z}) = \mathrm{X} + \mathrm{Y}(\overline{\mathrm{X}} + \mathrm{YZ})$

$= X + \overline{X}Y + YZ = (X + \overline{X})(X + Y) + YZ = X + Y + YZ = X + Y$

c) $\overline{W}X(\overline{Z} + \overline{Y}Z) + X(W + \overline{W}\,YZ) = \overline{W}X\overline{Z} + \overline{W}X\overline{Y}Z + WX + \overline{W}XYZ$

$= WX + \overline{W}X\overline{Z} + \overline{W}XZ = \mathrm{WX} + \overline{\mathrm{W}}\mathrm{X} = \mathrm{X}$

d) $(AB + \overline{A}\,\overline{B})(CD + \overline{C}\,\overline{D}) + \overline{AC}$

$= AB\overline{C}\,\overline{D} + ABCD + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}CD + \overline{A} + \overline{C}$

$= \overline{A} + \overline{C} + ABCD$

$= \overline{A} + \overline{C} + A(BCD)$

$= \overline{A} + \overline{C} + BCD$

$= \overline{A} + \overline{C} + C(BD)$

$= \overline{A} + \overline{C} + BD$

2-9.

a) $\overline{F} = (\overline{A} + B)(A + \overline{B})$

b) $\overline{F} = ((V + \overline{W})\overline{X} + \overline{Y})Z$

c) $\overline{F} = [\overline{W} + \overline{X} + (Y + \overline{Z})(\overline{Y} + Z)][W + X + Y\overline{Z} + \overline{Y}Z]$

d) $\overline{F} = \overline{A}B\overline{C} + (A + B)\overline{C} + \overline{A}(B + C)$

2-10.

Truth Tables a, b, c

X	Y	Z	a
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

A	B	C	b
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

W	X	Y	Z	c
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

a) Sum of Minterms: $\overline{X}YZ + X\overline{Y}Z + XY\overline{Z} + XYZ$

Product of Maxterms: $(X + Y + Z)(X + Y + \overline{Z})(X + \overline{Y} + Z)(\overline{X} + Y + Z)$

b) Sum of Minterms: $\overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{B}\,C + \overline{A}\,B\,C + A\,B\,C$

Product of Maxterms: $(A + \overline{B} + C)(\overline{A} + B + C)(\overline{A} + B + \overline{C})(\overline{A} + \overline{B} + C)$

c) Sum of Minterms: $\overline{W}\,\overline{X}\,Y\,\overline{Z} + \overline{W}\,X\,Y\,\overline{Z} + W\,\overline{X}\,Y\,\overline{Z} + W\,X\,\overline{Y}\,\overline{Z} + W\,X\,\overline{Y}\,Z + W\,X\,Y\,\overline{Z} + W\,X\,Y\,Z$

Product of Maxterms: $(W + X + Y + Z)(W + X + Y + \overline{Z})(W + X + \overline{Y} + \overline{Z})$
$(W + \overline{X} + Y + Z)(W + \overline{X} + Y + \overline{Z})(W + \overline{X} + \overline{Y} + \overline{Z})$
$(\overline{W} + X + Y + Z)(\overline{W} + X + Y + \overline{Z})(\overline{W} + X + \overline{Y} + \overline{Z})$

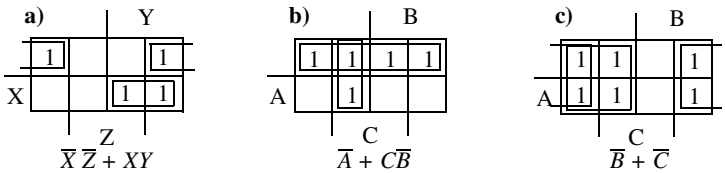
2-12.

a) $(AB + C)(B + \overline{C}D) = AB + BC + AB\overline{C}D = AB + BC$ s.o.p.
$= B(A + C)$ p.o.s.

b) $\overline{X} + X\,((X + \overline{Y})(Y + \overline{Z})) = (\overline{X} + X)(\overline{X} + (X + \overline{Y})(Y + \overline{Z}))$
$= (\overline{X} + X + \overline{Y})(\overline{X} + Y + \overline{Z}) = \overline{X} + Y + \overline{Z}$ s.o.p. and p.o.s.

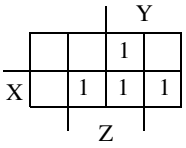
c) $(A + B\overline{C} + CD)(\overline{B} + EF) = (A + B + C)(A + \mathrm{B} + \mathrm{D})(A + \overline{C} + D)(\overline{B} + E)(\overline{B} + F)$ p.o.s.
$(A + B\overline{C} + CD)(\overline{B} + EF) = A(\overline{B} + EF) + B\overline{C}(\overline{B} + EF) + CD(\overline{B} + EF)$
$= A\overline{B} + AEF + B\overline{C}EF + \overline{B}CD + CDEF$ s.o.p.

2-15.



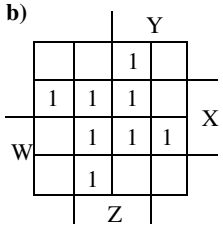
2-18.

a)



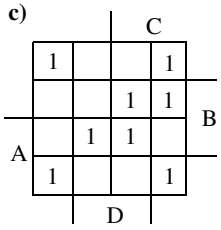
$\Sigma m(3, 5, 6, 7)$

b)



$\Sigma m(3, 4, 5, 7, 9, 13, 14, 15)$

c)



$\Sigma m(0, 2, 6, 7, 8, 10, 13, 15)$

2-19.

Using K-maps:

a) Prime = XZ, WX, $\overline{X}\,\overline{Z}$, $W\overline{Z}$
Essential = XZ, $\overline{X}\,\overline{Z}$

b) Prime = CD, AC, $\overline{B}\,\overline{D}$, $\overline{A}BD$, $\overline{B}C$
Essential = AC, $\overline{B}\,\overline{D}$, $\overline{A}BD$

c) Prime = AB, AC, AD, $B\overline{C}$, $\overline{B}D$, $\overline{C}D$
Essential = AC, $B\overline{C}$, $\overline{B}D$

2-22.

Using K-maps:

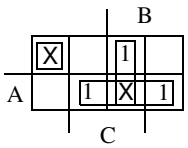
a) s.o.p. $CD + A\overline{C} + \overline{B}\,D$
p.o.s.$(\overline{C} + D)(A + D)(A + \overline{B} + C)$

b) s.o.p.$\overline{A}\,\overline{C} + \overline{B}\,\overline{D} + A\,\overline{D}$
p.o.s.$(\overline{C} + \overline{D})(\overline{A} + \overline{D})(A + \overline{B} + \overline{C})$

c) s.o.p.$\overline{B}\,\overline{D} + \overline{A}BD + (\overline{A}BC$ or $\overline{A}C\overline{D})$
p.o.s.$(\overline{A} + \overline{B})(B + \overline{D})(\overline{B} + C + D)$

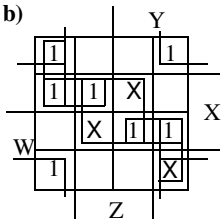
2-25.

a)



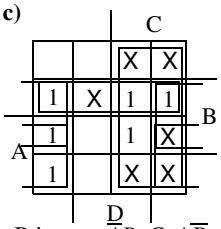
Primes = AB, AC, BC, $\overline{A}\,\overline{B}\,\overline{C}$
Essential = AB, AC, BC
$F = AB + AC + BC$

b)



Primes = $\overline{X}\,\overline{Z}$, XZ, $\overline{W}X\overline{Y}$, WXY, $\overline{W}\,\overline{Y}\,\overline{Z}$, $WY\overline{Z}$
Essential = $\overline{X}\,\overline{Z}$
$F = \overline{X}\,\overline{Z} + \overline{W}X\overline{Y} + WXY$

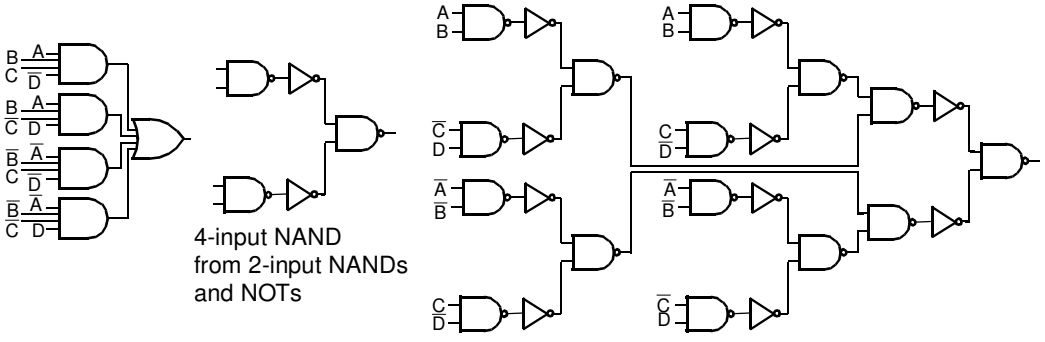
c)



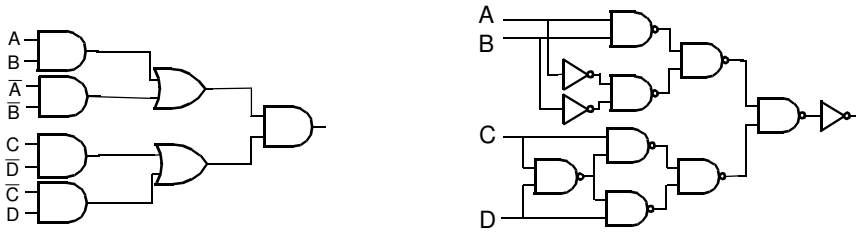
Primes = $\overline{A}B$, C, $A\overline{D}$, $B\overline{D}$
Essential = C, $A\overline{D}$
$F = C + A\overline{D} + (B\overline{D}$ or $\overline{A}B)$

2-28.

a)

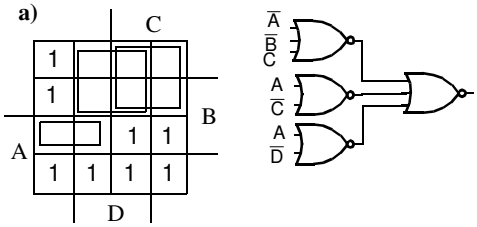


b)



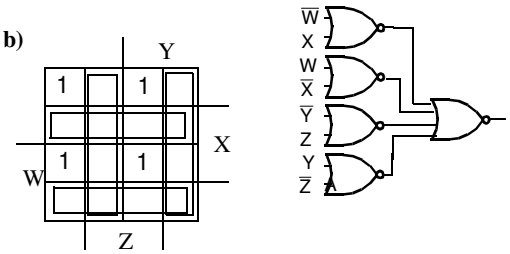
2-30.

a)



$$F = (\overline{A} + \overline{B} + C)(A + \overline{C})(A + \overline{D})$$

b)



$$F = (\overline{W} + X)(W + \overline{X})(\overline{Y} + Z)(Y + \overline{Z})$$

2-34.

$$X \oplus Y = X\overline{Y} + \overline{X}Y$$

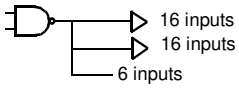
$$\text{Dual } (X \oplus Y) = \text{Dual } (X\overline{Y} + \overline{X}Y)$$

$$= (X + \overline{Y})(\overline{X} + Y)$$

$$\overline{X\overline{Y} + \overline{X}Y} = (\overline{X} + Y)(X + \overline{Y})$$

$$= (X + \overline{Y})(\overline{X} + Y)$$

2-37.



2-39.

$$4 \times 0.5 = 2 \text{ ns}$$

2-44.

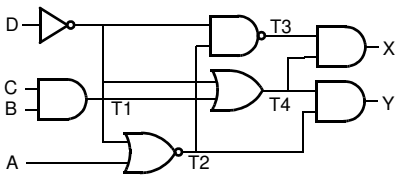
				P-Logic				N-Logic			
X	Y	NAND	NOR	X	Y	NAND	NOR	X	Y	NAND	NOR
L	L	H	H	0	0	1	1	1	1	0	0
L	H	H	L	0	1	1	0	1	0	0	1
H	L	H	L	1	0	1	0	0	1	0	1
H	H	L	L	1	1	0	0	0	0	1	1

Problem Solutions to Problems Marked With a * in

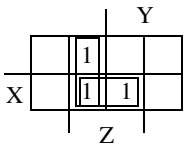
Logic Computer Design Fundamentals, Ed. 2

CHAPTER 3

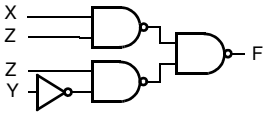
3-2.



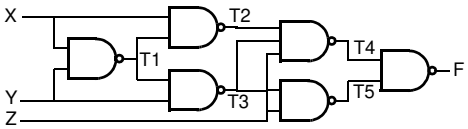
$$T1 = BC, \quad T2 = \overline{A}D$$
$$T3 = 1, \quad T4 = \overline{D} + BC$$
$$X = T3T4 = \overline{D} + BC$$
$$Y = T2T4 = \overline{A}D(\overline{D} + BC) = \overline{A}\,BCD$$



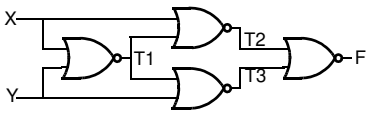
$XZ + Z\overline{Y}$



X	Y	Z	T1	T2	T3	T4	T5	F
0	0	0	1	1	1	1	1	0
0	0	1	1	1	1	0	0	1
0	1	0	1	1	0	1	1	0
0	1	1	1	1	0	1	1	0
1	0	0	1	0	1	1	1	0
1	0	1	1	0	1	1	0	1
1	1	0	0	1	1	1	1	0
1	1	1	0	1	1	0	0	1

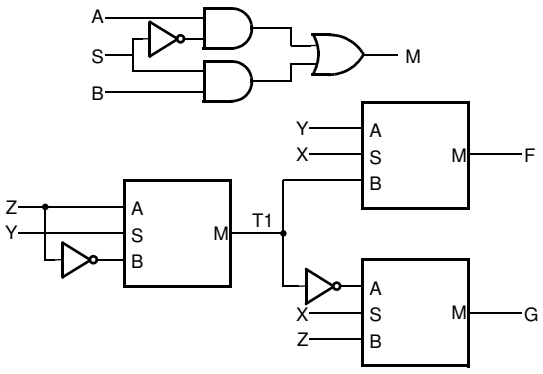


3-3.



X	Y	T1	T2	T3	F
0	0	1	0	0	1
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

3-6.



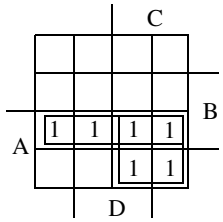
$$M = A\overline{S} + BS$$

$$T_1 = Z\overline{Y} + \overline{Z}Y$$

$$F = Y\overline{X} + T_1X = Y\overline{X} + X(Z\overline{Y} + \overline{Z}Y)$$
$$= \overline{X}Y + X\overline{Y}Z + XY\overline{Z}$$

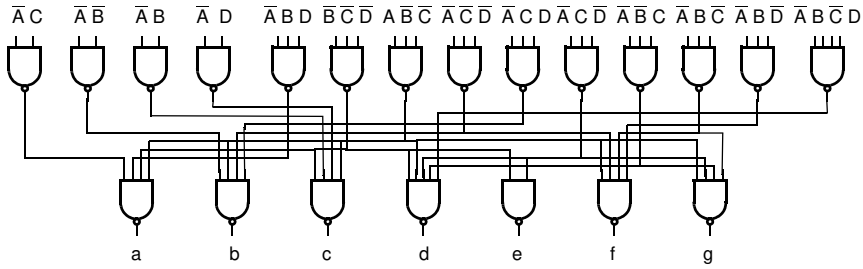
$$G = \overline{T}_1\overline{X} + ZX = XZ + \overline{X}(\overline{Z} + Y)(Z + \overline{Y})$$
$$= XZ + \overline{X}(YZ + \overline{Y}\,\overline{Z}) = XZ + \overline{X}YZ + \overline{X}\,\overline{Y}\,\overline{Z}$$

3-11.

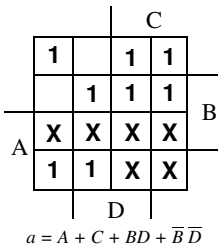


$F = AB + AC$

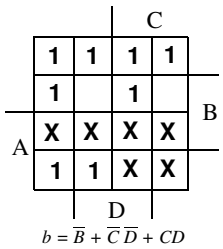
3-13.



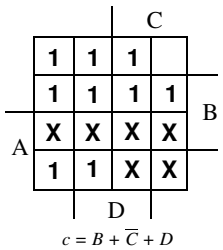
3-15.



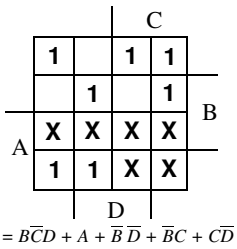
$a = A + C + BD + \overline{B}\,\overline{D}$



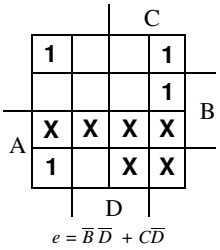
$b = \overline{B} + \overline{C}\,\overline{D} + CD$



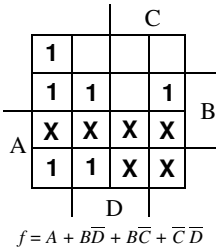
$c = B + \overline{C} + D$



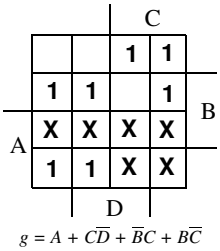
$d = B\overline{C}D + A + \overline{B}\,\overline{D} + \overline{B}C + C\overline{D}$



$e = \overline{B}\,\overline{D} + C\overline{D}$



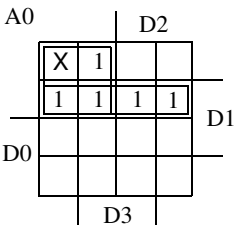
$f = A + B\overline{D} + B\overline{C} + \overline{C}\,\overline{D}$

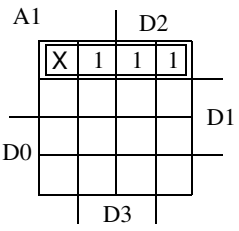


$g = A + C\overline{D} + \overline{B}C + B\overline{C}$

3-20.

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



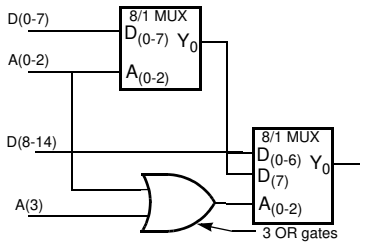


$V = D0 + D1 + D2 + D3$

$A0 = D1 + \overline{D0}\,\overline{D2}$

$A1 = \overline{D0}\,\overline{D1}$

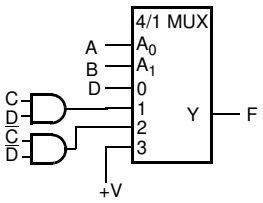
3-25.



3-29.

A	B	C	D	F	
0	0	0	0	0	
0	0	0	1	1	
0	0	1	0	0	D
0	0	1	1	1	
0	1	0	0	1	$\overline{C}\,\overline{D}$
0	1	0	1	0	
0	1	1	0	0	
0	1	1	1	0	

A	B	C	D	F	
1	0	0	0	0	CD
1	0	0	1	0	
1	0	1	0	0	
1	0	1	1	1	
1	1	0	0	1	+V
1	1	0	1	1	
1	1	1	0	1	
1	1	1	1	1	



3-35.

$$C_1 = \overline{T_3 + T_2} = \overline{T_1\overline{C}_0 + T_2} = \overline{\overline{A_0B_0}\,\overline{C_0} + \overline{A_0 + B_0}} = \overline{(\overline{A}_0 + \overline{B}_0)\overline{C}_0 + \overline{A}_0\overline{B}_0} = (A_0B_0 + C_0)(A_0 + B_0)$$

$$C_1 = A_0B_0 + A_0C_0 + B_0C_0$$

$$S_0 = C_0 \oplus T_4 = C_0 \oplus T_1\overline{T}_2 = C_0 \oplus \overline{A_0B_0}(A_0 + B_0) = C_0 \oplus (\overline{A}_0 + \overline{B}_0)(A_0 + B_0) = C_0 \oplus A_0\overline{B}_0 + \overline{A}_0B_0$$

$$S_0 = A_0 \oplus B_0 \oplus C_0$$

3-38.

1	0	0	1	1	0	0	0	1	0	0	1	1	0	0	1	1	0	1	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	1	1	0	0	1	1	1	0	1	1	0	0	1	1	0	0	1	0	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	1	1	0	1	0	0	0	0	1	1	0	0	1	1	1	0	1	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

3-41.

+43	=	0101011	43		0101011
-17	=	1101111	+(–17)	+	1101111
-43	=	1010101			10011010
+17	=	0010001	= 26	=	0011010
			–43		1010101
			+ 17	+	0010001
			= –26	=	1100110

3-45.

	S	A	B	C4	S3	S2	S1	S0
a)	0	0111	0110	0	1	1	0	1
b)	0	0100	0101	0	1	0	0	1
c)	1	1100	1010	1	0	0	1	0
d)	1	0101	1010	0	1	0	1	1
e)	1	0000	0010	0	1	1	1	0

3-49.

78430258 98989899 09580089 99999999

3-52.

BCD

	A	B	C	D	E	F	G	H
0	0	0	0	0	1	0	0	1
1	0	0	0	1	1	0	0	0
2	0	0	1	0	0	1	1	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	0	1
5	0	1	0	1	0	1	0	0
6	0	1	1	0	0	0	1	1
7	0	1	1	1	0	0	1	0
8	1	0	0	0	0	0	0	1
9	1	0	0	1	0	0	0	0

$H = \overline{D}$

$G = C$

$F = \overline{B}C + B\overline{C}$

$E = \overline{A}\overline{B}\overline{C}$

Gates: 8

Literals: 9

EXCESS-3

	A	B	C	D	E	F	G	H
0	0	0	1	1	1	1	0	0
1	0	1	0	0	1	0	1	1
2	0	1	0	1	1	0	1	0
3	0	1	1	0	1	0	0	1
4	0	1	1	1	1	0	0	0
5	1	0	0	0	0	1	1	1
6	1	0	0	1	0	1	1	0
7	1	0	1	0	0	1	0	1
8	1	0	1	1	0	1	0	0
9	1	1	0	0	0	0	1	1

$H = \overline{D}$

$G = \overline{C}$

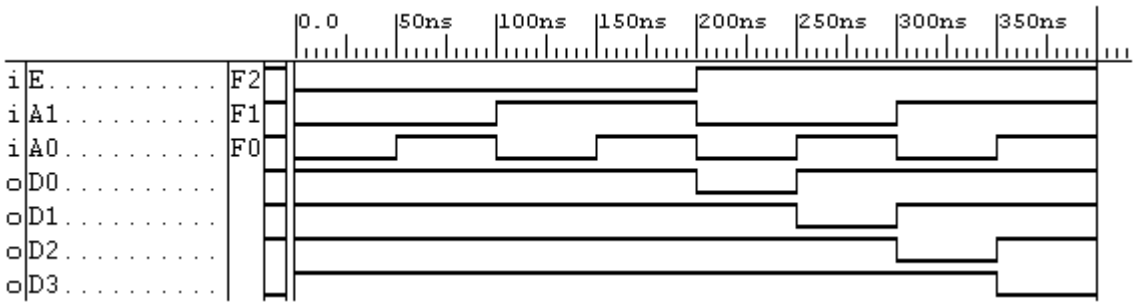
$F = \overline{B}$

$E = \overline{A}$

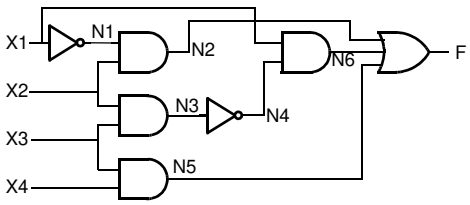
Gates: 4

Literals: 4

3-55.



3-58.



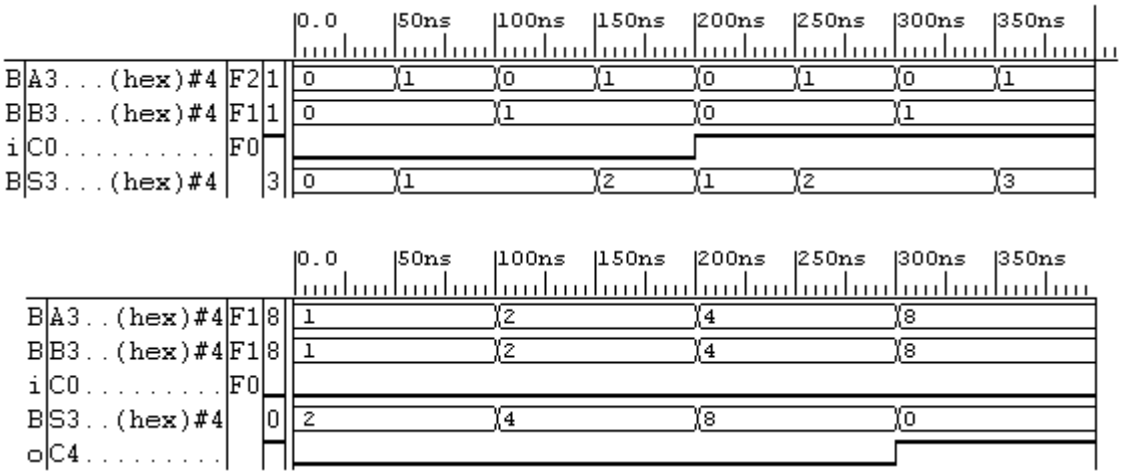
3-62.

From 3-2: F = X Z + Z $\overline{Y}$

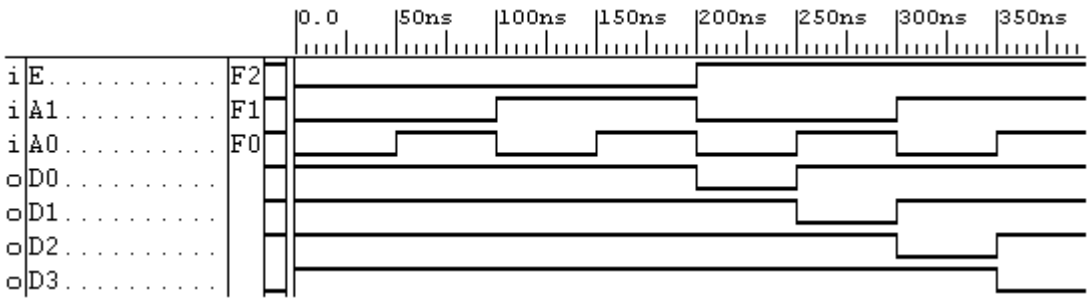
Using Nand Gates:

```
...
signal T: std_logic_vector(0 to 2);
begin
  g0: NOT1 port map (Y, T(0));
  g1: NAND2 port map (X, Z, T(1));
  g2: NAND2 port map (Z, T(0), T(2));
  g3: NAND2 port map (T(1), T(2), F);
 end
```

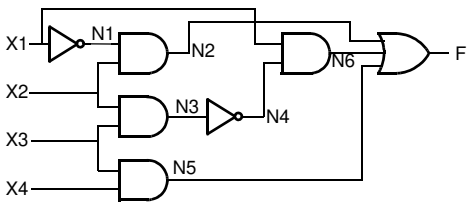
3-66.



3-69.

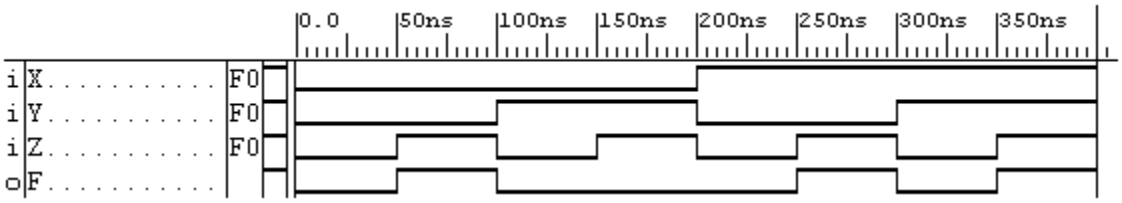


3-72.

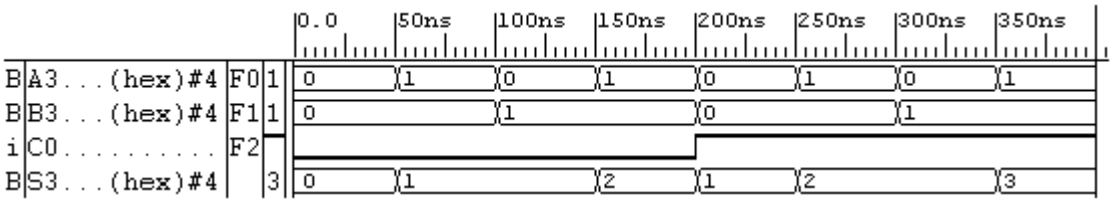


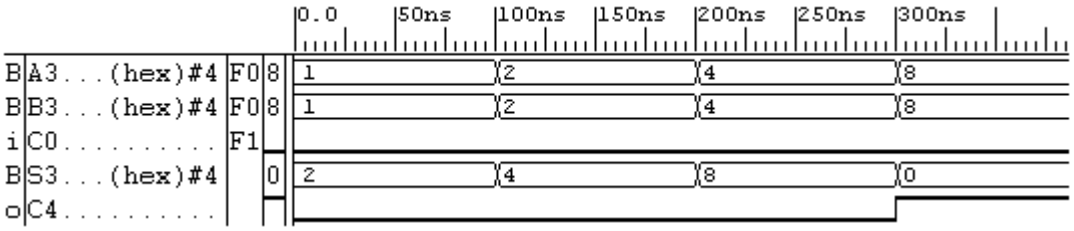
3-76.

```
//Fucntion F from problem 3-2 =  X Z + Z Y

module cicuit_3_76(X, Y, Z, F);
  input X, Y, Z;
  output F;

  assign F = (X & Z) | (Z & ~Y);
endmodule
```



3-80.



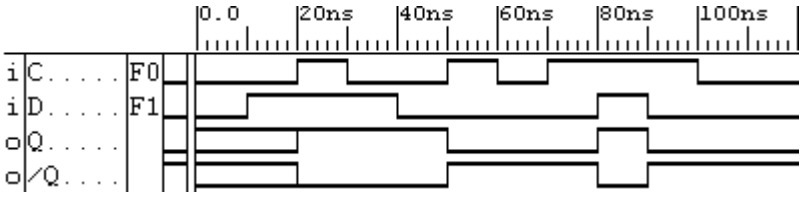


Problem Solutions to Problems Marked With a * in

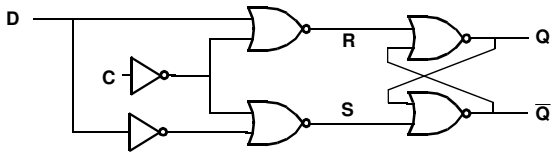
Logic Computer Design Fundamentals, Ed. 2

CHAPTER 4

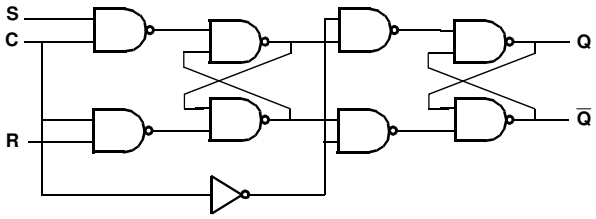
4-3. (All simulations performed using Xilinx Foundation Series software.)



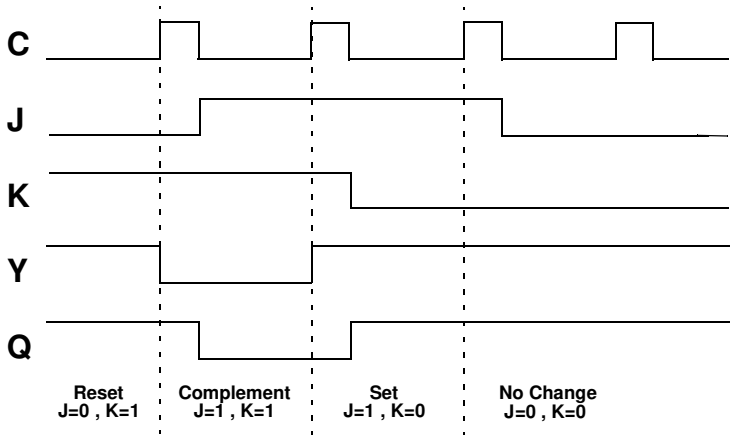
4-4.



4-5.



4-6.



4-10.

J	K	Q(t)	Q(t+1)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

S	R	Q(t)	Q(t+1)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	X
1	1	1	X

D	Q(t)	Q(t+1)
0	0	0
0	1	0
1	0	1
1	1	1

T	Q(t)	Q(t+1)
0	0	0
0	1	1
1	0	1
1	1	0

$Q(t+1) = D$ $\qquad$ $Q(t+1) = T \oplus Q$

$Q(t+1) = J\overline{Q} + \overline{K}Q$ $\qquad$ $Q(t+1) = S + \overline{R}Q$

$J_A = B \qquad K_A = B\overline{X}$

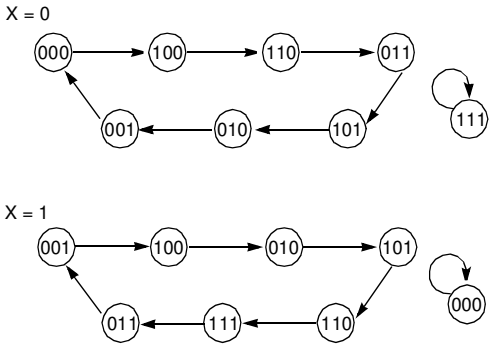
$J_B = \overline{X} \qquad K_B = A\overline{X} + \overline{A}X$

$A(t+1) = J_A\overline{A} + \overline{K}_A A = B\overline{A} + \overline{B}A + XA$

$B(t+1) = J_B\overline{B} + \overline{K}_B B = \overline{X}\,\overline{B} + ABX + \overline{A}B\overline{X}$

4-12.

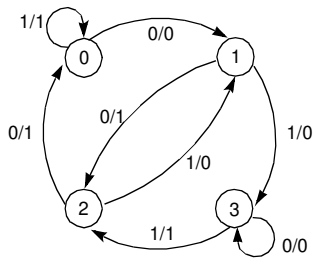
Present state			Input	Next state		
A	*B*	*C*	*X*	*A*	*B*	*B*
0	0	0	0	1	0	0
0	0	0	1	0	0	0
0	0	1	0	0	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	1	0	1
0	1	1	0	1	0	1
0	1	1	1	0	0	1
1	0	0	0	1	1	0
1	0	0	1	0	1	0
1	0	1	0	0	1	0
1	0	1	1	1	1	0
1	1	0	0	0	1	1
1	1	0	1	1	1	1
1	1	1	0	1	1	1
1	1	1	1	0	1	1



State diagram is the combination of the above two diagrams.

4-17.

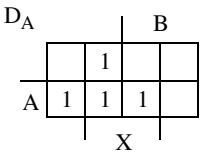
Present state		Input	Next state		Output
A	*B*	*X*	*A*	*B*	*Y*
0	0	0	0	1	0
0	0	1	0	0	1
0	1	0	1	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	0	1	0
1	1	0	1	1	0
1	1	1	1	0	1



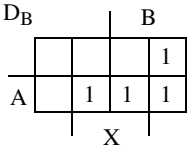
Format: X/Y

4-19.

Present state		Input	Next state	
A	*B*	*X*	*A*	*B*
0	0	0	0	0
0	0	1	1	0
0	1	0	0	1
0	1	1	0	0
1	0	0	1	0
1	0	1	1	1
1	1	0	1	1
1	1	1	0	1



$D_A = A\overline{B} + AX + \overline{B}X$

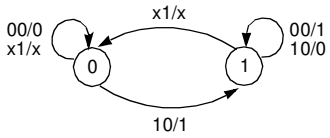


$D_B = AX + B\overline{X}$

4-20.

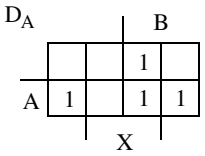
Present state	Inputs		Next state	Output
Q(t)	*X*	*Y*	*Q(t+1)*	*Z*
0	0	0	0	0
0	0	1	0	X
0	1	0	1	1
0	1	1	0	X
1	0	0	1	1
1	0	1	0	X
1	1	0	1	0
1	1	1	0	X

Format: XY/Z (x = unspecified)

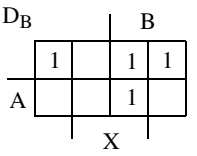


4-24.

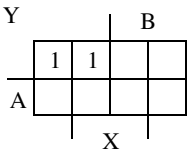
Present state		Input	Next state		Output
A	*B*	*X*	*A*	*B*	*Y*
0	0	0	0	1	1
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	1	1	0
1	0	0	1	0	0
1	0	1	0	0	0
1	1	0	1	0	0
1	1	1	1	1	0



$D_A = A\overline{X} + BX$



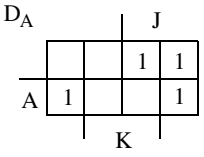
$D_B = BX + \overline{A}\,\overline{X}$



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

4-25.

Present state	Input		Next state
A	*J*	*K*	*A*
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0



$D_A = \overline{A}J + A\overline{K}$

4-30.

Present state		Input	Next state		FF Inputs			
A	*B*	*X*	*A*	*B*	J_A	K_A	J_B	K_B
0	0	0	0	1	0	X	1	X
0	0	1	0	0	0	X	0	X
0	1	0	0	1	0	X	X	0
0	1	1	1	1	1	X	X	0
1	0	0	1	0	X	1	0	X
1	0	1	0	0	X	1	0	X
1	1	0	1	0	X	0	X	1
1	1	1	1	1	X	0	X	0

$J_A = BX$

$K_A = B\,X + \overline{B}\,\overline{X}$

$J_B = \overline{A}\,\overline{X}$

$K_B = A\,\overline{X}$

4-33.

Present state		Inputs		Next state		FF Inputs			
A	*B*	*E*	*X*	*A*	*B*	J_A	K_A	J_B	K_B
0	0	0	0	0	0	0	X	0	X
0	0	0	1	0	0	0	X	0	X
0	0	1	0	1	1	1	X	1	X
0	0	1	1	0	1	0	X	1	X
0	1	0	0	0	1	0	X	X	0
0	1	0	1	0	1	0	X	X	0
0	1	1	0	0	0	0	X	X	1
0	1	1	1	1	0	1	X	X	1
1	0	0	0	1	0	X	0	0	X
1	0	0	1	1	0	X	0	0	X
1	0	1	0	0	1	X	1	1	X
1	0	1	1	1	1	X	0	1	X
1	1	0	0	1	1	X	0	X	0
1	1	0	1	1	1	X	0	X	0
1	1	1	0	1	0	X	0	X	1
1	1	1	1	0	0	X	1	X	1

$J_A = E(BX + \overline{B}\,\overline{X})$

$K_A = E(BX + \overline{B}\,\overline{X})$

$J_B = E$

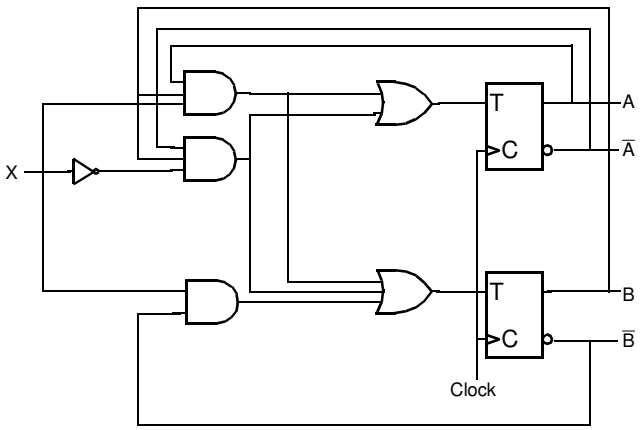
$K_B = E$

4-36.

Present state		Input	Next state		FF Inputs	
A	*B*	*X*	*A*	*B*	T_A	T_B
0	0	0	0	0	0	0
0	0	1	0	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	0	0
1	0	0	1	0	0	0
1	0	1	1	1	0	1
1	1	0	1	1	0	0
1	1	1	0	0	1	1

$T_A = ABX + \overline{A}B\overline{X}$

$T_B = ABX + \overline{A}B\overline{X} + \overline{B}X$



4-37

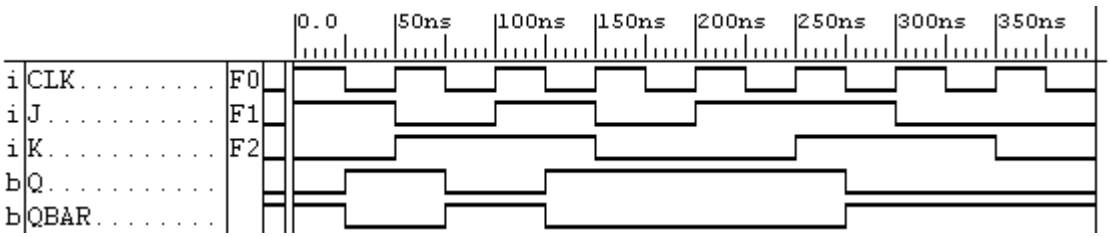
```
library IEEE;
use IEEE.std_logic_1164.all;

entity mux_4to1 is
     port (
          S: in STD_LOGIC_VECTOR (1 downto 0);
          D:  in  STD_LOGIC_VECTOR  (3  downto
0);
          Y: out STD_LOGIC
     );
end mux_4to1;

-- (continued in the next column)
```

```
architecture mux_4to1_arch of mux_4to1 is
begin

process (S, D)
     begin
     case S is
          when "00" => Y <= D(0);
          when "01" => Y <= D(1);
          when "10" => Y <= D(2);
          when "11" => Y <= D(3);
          when others => null;
     end case;

end process;
end mux_4to1_arch;
```

4-40.

```
library IEEE;
use IEEE.std_logic_1164.all;
entity jkff is
   port (
      J,K,CLK: in STD_LOGIC;
      Q: out STD_LOGIC
   );
end jkff;

architecture jkff_arch of jkff is
signal q_out: std_logic;
begin

state_register: process (CLK)
begin
  if CLK'event and CLK='0' then  --CLK falling edge

-- (continued in the next column)
```

```
    case J is
        when '0' =>
            if K = '1' then
                q_out <= '0';
            end if;
        when '1' =>
            if K = '0' then
                q_out <= '1';
            else
                q_out <= not q_out;
            end if;
        when others => null;
    end case;
  end if;
end process;

Q <= q_out;
end jkff_arch;
```



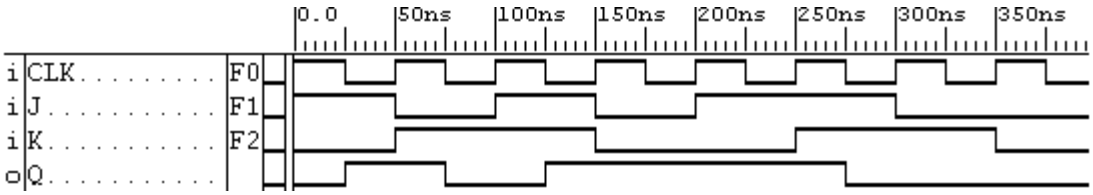
4-45.

```
module problem_4_45 (S, D, Y) ;

input [1:0] S ;
input [3:0] D ;
output Y;
reg Y ;

// (continued in the next column)
```

```
always @(S or D)
    begin

        if (S == 2'b00) Y <= D[0];
        else if (S == 2'b01) Y <= D[1];
        else if (S == 2'b10) Y <= D[2];
        else Y <= D[3];

    end
endmodule
```

4-47.

```
module JK_FF (J, K, CLK, Q) ;

input J, K, CLK ;
output Q;
reg Q;

// (continued in the next column)

always @(negedge CLK)
    case (J)
        0'b0: Q <= K ? 0: Q;
        1'b1: Q <= K ? ~Q: 1;
    endcase
endmodule
```



Problem Solutions to Problems Marked With a * in

Logic Computer Design Fundamentals, Ed. 2

CHAPTER 5

5-3.

1000, 0100, 1010, 1101 0110, 1011, 1101, 1110

5-6.

Shifts:	0	1	2	3	4
A	0110	1011	0101	0010	1001
B	0011	0001	0000	0000	0000
C	0	0	1	1	0

5-8.

Replace each AND gate in Figure 5-6 with an AND gate with one additional input and connect this input to the following:

$$S_1 + \overline{S}_0$$

This will force the outputs of all the AND gates to zero, and, on the next clock edge, the register will be cleared if S1 is 0 and S0 is logic one.

Also, replace each direct shift input with this equation: $S_1\overline{S}_0$ This will stop the shift operation from interfering with the load parallel data operation.

5-10.

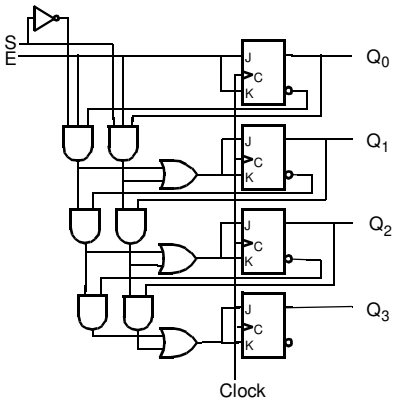
a) 1000, 0100, 0010, 0001, 1000

b) # States = n

5-17.

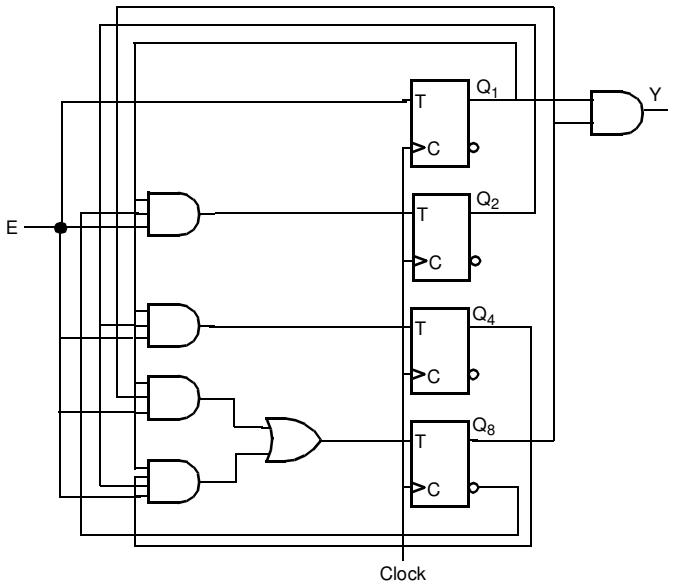
$$Q_0 = \overline{Q}_0E$$
$$Q_1 = (Q_0\overline{Q}_1 + \overline{Q}_0Q_1)E$$
$$Q_2 = (Q_0Q_1\overline{Q}_2 + \overline{Q}_1Q_2 + \overline{Q}_0Q_2)E$$
$$Q_3 = (\overline{Q}_2Q_3 + \overline{Q}_1Q_3 + \overline{Q}_0Q_3 + Q_0Q_1Q_2\overline{Q}_3)E$$

5-21.



5-24.

$$T_{Q8} = (Q_1Q_8 + Q_1Q_2Q_4)E$$
$$T_{Q4} = Q_1Q_2E$$
$$T_{Q2} = Q_1\overline{Q}_8E$$
$$T_{Q1} = E$$
$$Y = Q_1Q_8$$



5-26.

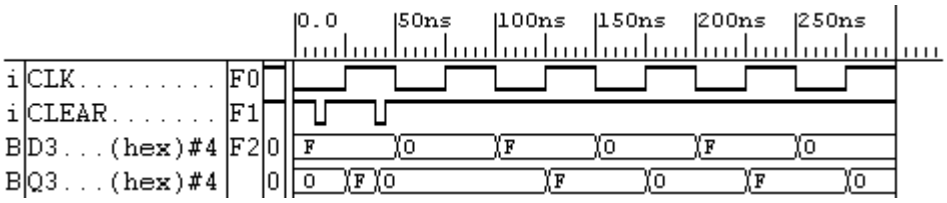
Present state			Next state			FF Inputs					
A	*B*	*C*	*A*	*B*	*C*	J_A	K_A	J_B	K_B	J_C	K_C
	0	0		0	1			0	X	1	X
	0	1		1	0			1	X	X	1
	1	1		0	0			X	1	0	X
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	0	1	1	0	X	X	0	1	X
0	1	1	1	0	0	1	X	X	1	X	1
1	0	0	1	0	1	X	0	0	X	1	X
1	0	1	0	0	0	X	1	0	X	X	1

a) $J_B = C$
$K_B = C$
$J_C = \overline{B}$
$K_C = 1$

b) $J_A = BC$
$K_A = C$
$J_B = \overline{A}C$
$K_B = C$
$J_C = 1$
$K_C = 1$

5-29. (All simulations performed using Xilinx Foundation Series software.)

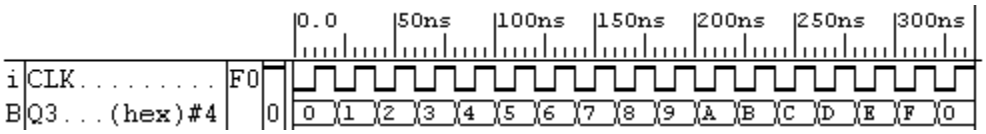
```
library IEEE;
use IEEE.std_logic_1164.all;

entity reg_4_bit is
   port (
      CLEAR, CLK: in STD_LOGIC;
      D: in STD_LOGIC_VECTOR (3 downto 0);
      Q: out STD_LOGIC_VECTOR (3 downto 0)
   );
end reg_4_bit;

architecture reg_4_bit_arch of reg_4_bit is
begin

process (CLK, CLEAR)
begin
  if CLEAR ='0' then                    --asynchronous RESET active Low
    Q <= "0000";
  elsif (CLK'event and CLK='1') then    --CLK rising edge
    Q <= D;
  end if;
end process;

end reg_4_bit_arch;
```



5-33.

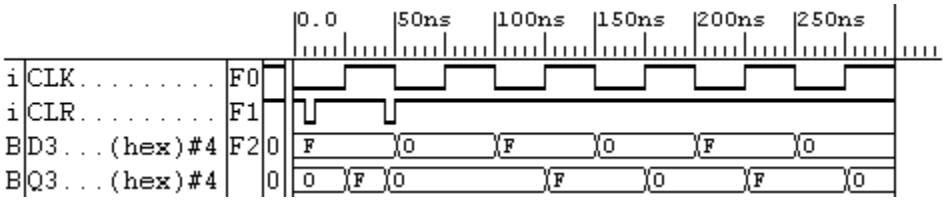
```
library IEEE;
use IEEE.std_logic_1164.all;

entity ripple_1_bit is
   port (
     RESET, CLK, J, K: in STD_LOGIC;
     Q: out STD_LOGIC
   );
end ripple_1_bit;

architecture ripple_arch of ripple_1_bit is
signal Q_out: std_logic;
begin
process (CLK, RESET)
begin
  if RESET ='1' then  -- asynchronous RESET active
Low
    Q_out <= '0';
   elsif (CLK'event and CLK='0') then --CLK falling
edge
          if (J = '1' and K = '1') then
               Q_out <= not Q_out;
          elsif(J = '1' and K = '0') then
               Q_out <= '1';
          elsif (J = '0' and K = '1') then
               Q_out <= '0';
          end if;
     end if;
end process;
Q <= Q_out;

end ripple_arch;

-- (Continued in next column)
```

```
library IEEE;
use IEEE.std_logic_1164.all;

entity ripple_4_bit is
   port (
      RESET, CLK: in STD_LOGIC;
      Q: out STD_LOGIC_VECTOR (3 downto 0)
   );
end ripple_4_bit;

architecture ripple_4_bit_arch of ripple_4_bit is

component ripple_1_bit
   port (
      RESET, CLK, J, K: in STD_LOGIC;
      Q: out STD_LOGIC
   );
end component ;
signal logic_1: std_logic;
signal Q_out: std_logic_vector(3 downto 0);
begin
     bit0: ripple_1_bit port map(RESET, CLK, logic_1, logic_1,
Q_out(0));
     bit1: ripple_1_bit port map(RESET, Q_out(0), logic_1, logic_1,
Q_out(1));
     bit2: ripple_1_bit port map(RESET, Q_out(1), logic_1, logic_1,
Q_out(2));
     bit3: ripple_1_bit port map(RESET, Q_out(2), logic_1, logic_1,
Q_out(3));

logic_1 <= '1';
Q <= Q_out;

end ripple_4_bit_arch;
```



5-35.

```
module register_4_bit (D, CLK, CLR, Q) ;

input [3:0] D ;
input CLK, CLR ;
output [3:0] Q ;
reg [3:0] Q ;

always @(posedge CLK or negedge CLR)
begin
      if (~CLR)                  //asynchronous RESET active low
            Q = 4'b0000;
      else                       //use CLK rising edge
      Q = D;
end
endmodule
```

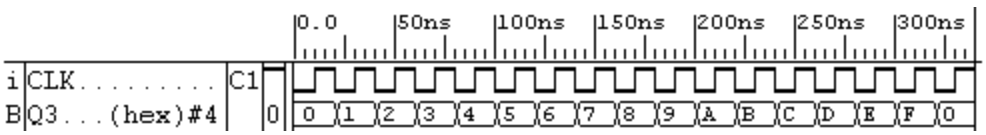


5-39.

```
module jk_1_bit (J, K, CLK, CLR, Q) ;

input J, K, CLK, CLR ;
output Q ;
reg Q;

always @(negedge CLK or posedge CLR)
begin
  if (CLR)
     Q <= 1'b0;
  else if ((J == 1'b1) && (K == 1'b1))
     Q <= ~Q;
  else if (J == 1'b1 && K == 1'b0)
     Q <= 1'b1;
  else if (J == 1'b0 && K == 1'b1)
     Q <= 1'b0;
end
endmodule
```

// (continued in next column)

```
module reg_4_bit (CLK, CLR, Q);

input CLK, CLR ;
output [3:0] Q ;
reg [3:0] Q;

jk_1_bit
      g1(1'b1, 1'b1, CLK, CLR, Q[0]),
      g2(1'b1, 1'b1, Q[0], CLR, Q[1]),
      g3(1'b1, 1'b1, Q[1], CLR, Q[2]),
      g4(1'b1, 1'b1, Q[2], CLR, Q[3]);

endmodule
```



Problem Solutions to Problems Marked With a * in

Logic Computer Design Fundamentals, Ed. 2

C H A P T E R 6

6-1.

a) A = 13, D = 32 b) A = 18, D = 64 c) A = 25, D = 32 d) A = 32, D = 8

6-3.

$(633)_{10} = (10\ 0111\ 1001)_2$, $(2731)_{10} = (0000\ 1010\ 1010\ 1011)_2$

6-9.

a) 32 b) 20,15 c) 5, 5–to–32

6-15.

PTERM		INPUTS			OUTPUTS			
		X	Y	Z	A	$\overline{B}$	C	D
Y Z	1	–	1	1	1	1	–	1
X Y	2	1	1	–	1	–	–	–
$\overline{X}$ Y	3	0	1	–	–	1	1	1
X $\overline{Y}$ $\overline{Z}$	4	1	0	0	–	–	1	1

6-18.

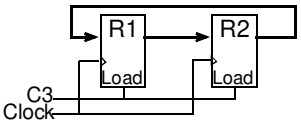
PTERM		INPUTS			
		A	B	C	D
A	1	1	–	-	–
BC	2	–	1	1	–
BD	3	–	1-	-	1
$\overline{B}$C	4	–	0	1	–
$\overline{B}$D	5	–	0	-	1
B$\overline{C}$ $\overline{D}$	6	–	1	0	0
CD	7	–	–	1	1
$\overline{C}$ $\overline{D}$	8	–	–	0	0
-	-	–	–	–	–
$\overline{D}$	9	–	–	–	0
-	-	–	–	–	–
-	-	–	–	–	–

Problem Solutions to Problems Marked With a * in

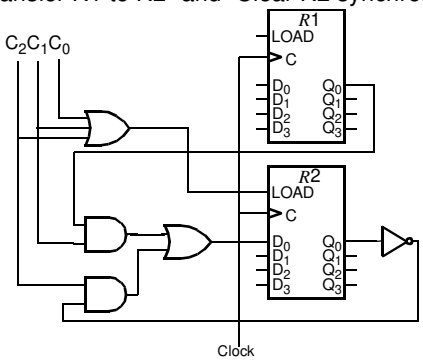
Logic Computer Design Fundamentals, Ed. 2

C H A P T E R 7

7-1.

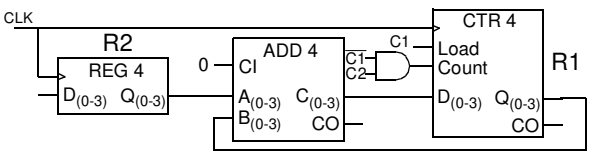


7-3. Errata: Interchange statements “Transfer R1 to R2” and “Clear R2 synchronously with the clock.”

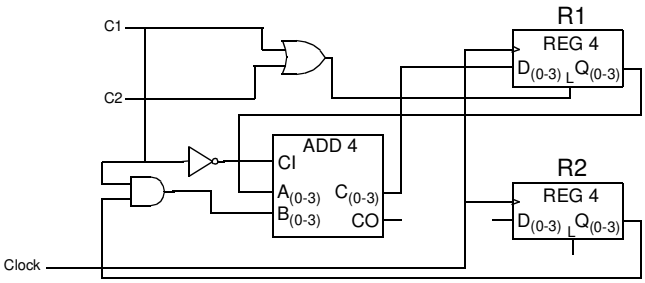


7-6.

a)



b)



7-9.

0101 1110
<u>1100 0101</u>
0100 0100 AND
1101 1111 OR
1001 1011 XOR

7-11.

sl 1001 1010 sr 0010 0110

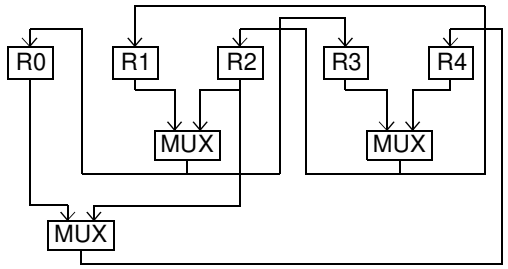
7-14.

a) Destination <- Source Registers
R0 <- R1, R2
R1 <- R4
R2 <- R3, R4
R3 <- R1
R4 <- R0, R2

b) Source Registers -> Destination
R0 -> R4
R1 -> R0, R3
R2 -> R0, R4
R3 -> R2
R4 -> R1, R2

c) The minimum number of buses needed for operation of the transfers is three since transfer Cb requires three different sources.

d)



7-19.

$C = C_8$

$V = C_8 \oplus C_7$

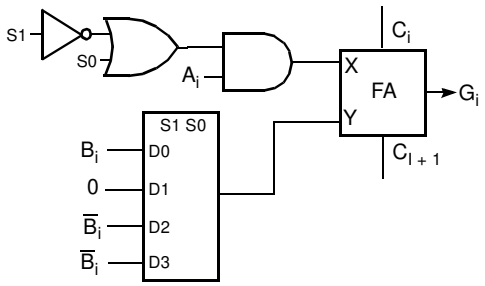
$Z = \overline{F_7 + F_6 + F_5 + F_4 + F_3 + F_2 + F_1 + F_0}$

$N = F_7$

7-22.

$X = A\,\overline{S1} + A\,S0$

$Y = B\,\overline{S1}\,\overline{S0} + \overline{B}\,S_1$



7-24.

a) XOR = 00, NAND = 01, NOR = 10 XNOR = 11

Out = $S_1\,\overline{A}\,\overline{B} + S_0\,\overline{A}\,\overline{B} + \overline{S}_1\,\overline{A}\,B + S_0\,A\,B + \overline{S}_1\,\overline{S}_0\,A\,\overline{B}$

b) The above is a simplest result.

7-26.

(a) 1011 (b) 1010 (c) 0001 (d) 1100

7-28.

(a)	$R5 \leftarrow R4 \wedge \overline{R5}$	R5 = 0000 0100	(d)	$R5 \leftarrow DataIn$	R5 = 0001 0010
(b)	$R6 \leftarrow R2 + \overline{R4} + 1$	R6 = 1111 1110	(e)	$R4 \leftarrow R4 \oplus$ Constant	R4 = 0001 0101
(c)	$R5 \leftarrow srR0$	R5 = 0000 0000	(f)	$R3 \leftarrow R0 \oplus R0$	R3 = 0000 0000

7-31.

Clock	AA	BA	MB	OF/EX:A	OF/EX:B	FS	EX/WB:F	EX/WB:DI	MD	RW	DA	R[DA]
1	2	3	0	—	—	—	—	—	—	—	—	—
2	0	6	0	02	03	5	—	—	—	—	—	—
3	7	0	0	00	06	18	FF	—	0	1	1	—
4	0	0	1	07	00	01	0C	—	0	1	4	FF
5	0	3	0	00	02	02	08	—	0	1	7	0C
6	0	0	0	00	03	00	02	—	0	1	1	08
7	0	0	0	00	00	00	00	—	0	0	0	02
8	—	—	—	00	00	0C	00	Data	1	1	4	—
9	—	—	—	—	—	—	00	—	0	1	5	Data
10	—	—	—	—	—	—	—	—	—	—	—	00

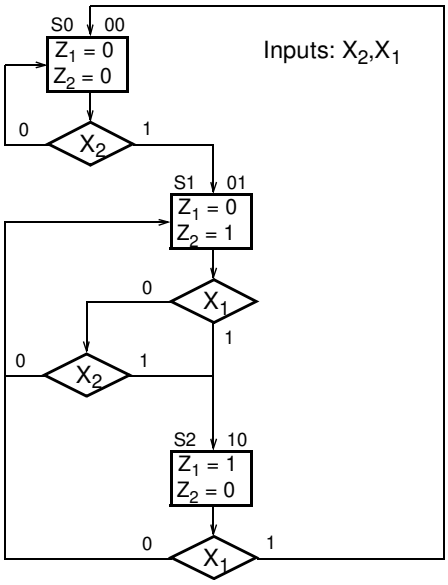
The contents of registers that change for a given clock cycle are shown in the next clock cycle. Values are given in hexadecimal.

Problem Solutions to Problems Marked With a * in

Logic Computer Design Fundamentals, Ed. 2

C H A P T E R 8

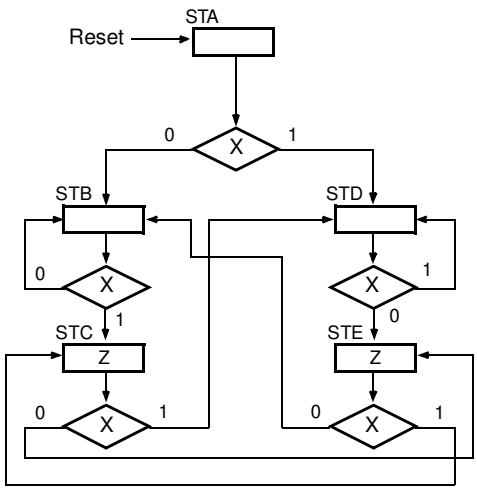
8-1.



8-2.

A:	0	1	1	0	1	1	
B:	1	1	1	1	0	0	
C:	0	1	0	1	0	1	
State:	ST1	ST1	ST2	ST3	ST1	ST2	ST3
Z:	0	0	1	1	0	0	

8-5.

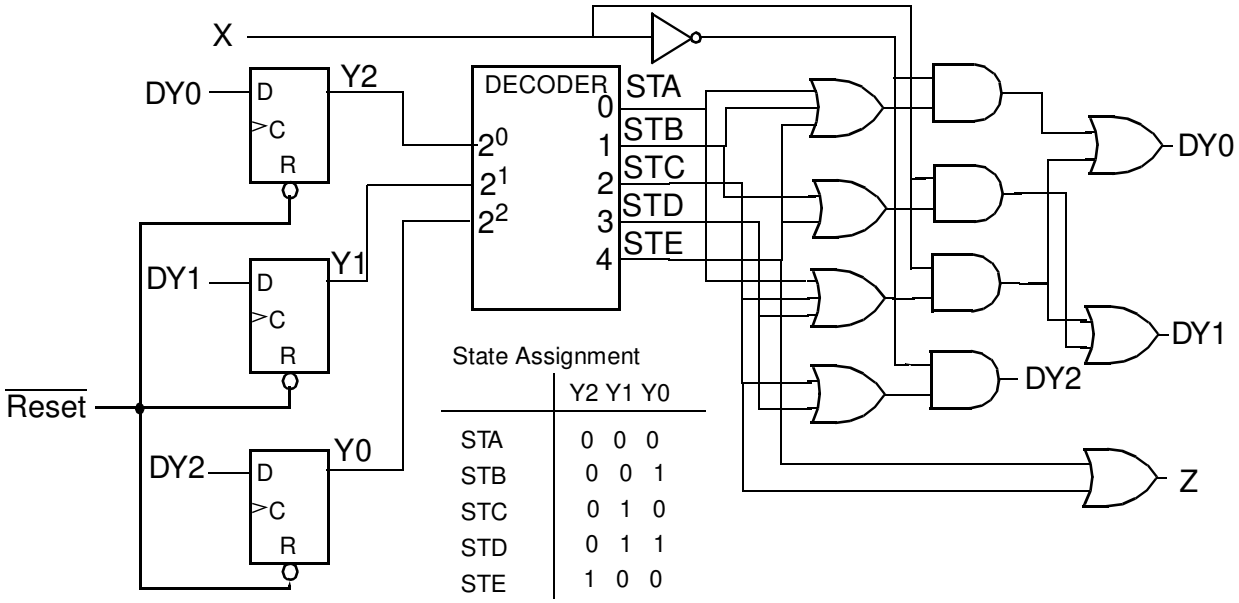


8-8.

ST1(t + 1) = ST1·$\overline{A}$ + ST2·B·C + ST3, ST2(t + 1) = ST1·A, ST3(t + 1)= ST2·($\overline{B}$ + $\overline{C}$), Z = ST2·B + ST3

For the D flip-flops, DSTi = STi(t + 1) and STi = Q(STi).Reset initializes the flip-flops: ST1 = 1, ST2 = ST3 = 0.

8-9.



State Assignment

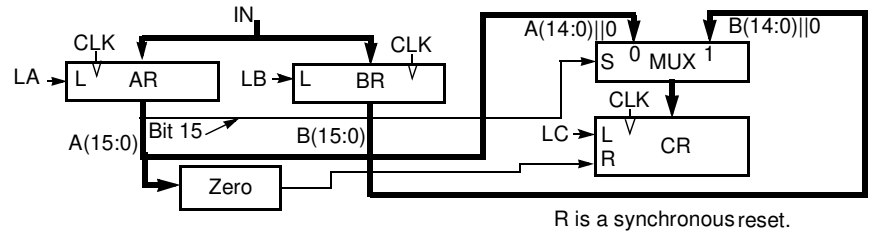
	Y2 Y1 Y0
STA	0 0 0
STB	0 0 1
STC	0 1 0
STD	0 1 1
STE	1 0 0

8-12.

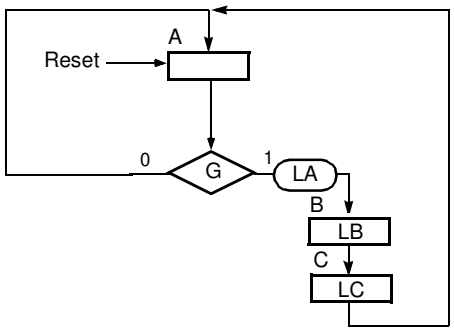
```
   100110    (38)
 × 110101  (× 53)
   100110
  000000
 100110
000000
100110
100110
11111011110   (2014)
```

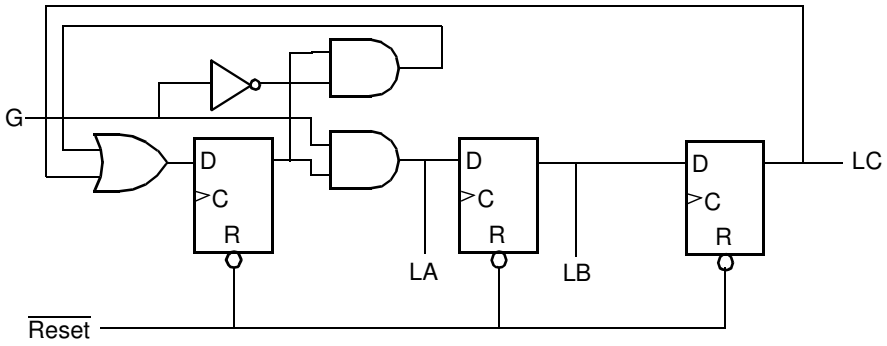
```
100110
110101
000000      Init PP
100110      Add
100110      After Add
0100110     After Shift
00100110    After Shift
100110      Add
10111110    After Add
010111110   After Shift
0010111110  After Shift
100110      Add
1100011110  After Add
01100011110 After Shift
100110      Add
11111011110 After Add
011111011110 After Shift
```

8-17.



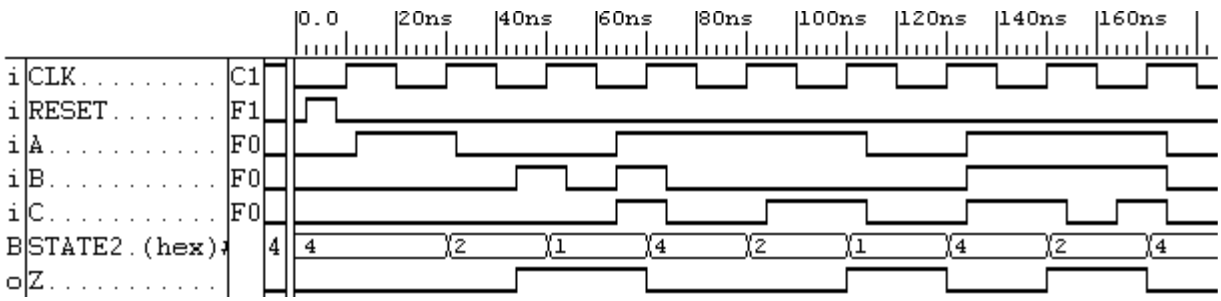
R is a synchronous reset.





8-20.

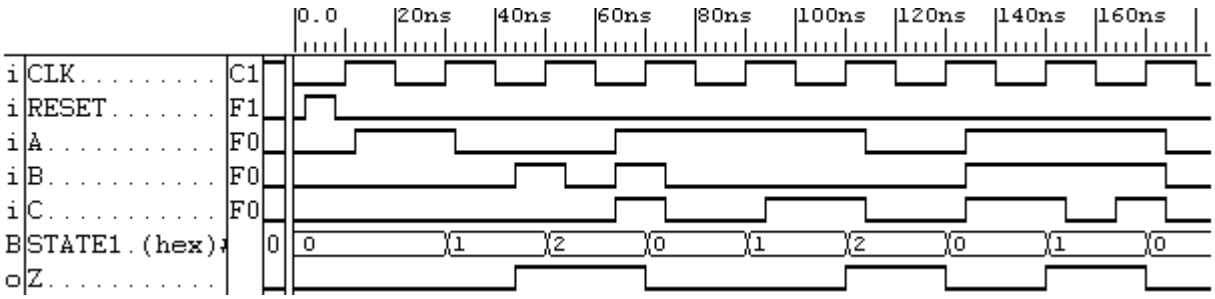
```
library IEEE;
use IEEE.std_logic_1164.all;

entity asm_820 is
	port (
		A, B, C, CLK, RESET: in STD_LOGIC;
		Z: out STD_LOGIC
	);
end asm_820;

architecture asm_820_arch of asm_820 is
	type state_type is (ST1, ST2, ST3);
	signal state, next_state : state_type;
begin

state_register: process (CLK, RESET)
begin
	if RESET='1' then --asynchronous RESET active High
		state <= ST1;
	elsif (CLK'event and CLK='1') then  --CLK rising edge
		state <= next_state;
	end if;
end process;

next_state_func: process (A, B, C, state)
begin
	case (state) is
		when ST1 =>
			if A = '0' then
				next_state <= ST1;
			else
				next_state <= ST2;
			end if;
		when ST2 =>
			if ((B = '1') and (C = '1')) then
				next_state <= ST1;
			else
				next_state <= ST3;
			end if;
		when ST3 =>
			next_state <= ST1;
	end case;
end process;

--Output Z only depends on the state and input 'B'
output_func: process (B, state)
begin
	case (state) is
		when ST1 =>
			Z <= '0';
		when ST2 =>
			if (B = '1') then
				Z <= '1';
			else
				Z <= '0';
			end if;
		when ST3 =>
			Z <= '1';
	end case;
end process;

end asm_820_arch;
```



NOTE: State hex value 4 cooresponds to ST1, 2 to ST2 and 1 to ST3.

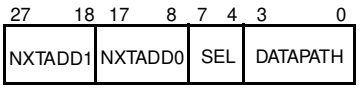
8-21. Errata: A, B, C should be ST1, ST2, ST3.

```
module asm_821 (CLK, RESET, A, B, C, Z) ;

input CLK, RESET, A, B, C ;
output Z ;
reg [1:0] state, next_state;
parameter ST1=2'b00, ST2=2'b01, ST3=2'b10, ST4=2'b11;
reg Z;

always @(posedge CLK or posedge RESET)
begin
  if (RESET)  //asynchronous RESET active High
     state <= ST1;
  else          //use CLK rising edge
     state <= next_state;
end

//Next state function
always @(A or B or C or state)
begin
       case (state)
              ST1: next_state <= A ? ST2: ST1;
              ST2: next_state <= (B && C) ? ST1: ST3;
              ST3: next_state <= ST1;
              ST4: next_state <= ST1; // Next state ST1 is assigned to unused state ST4.
       endcase
end

//Output function
always @(B or state)
begin
       case (state)
              ST1: Z <= 1'b0;
              ST2: Z <= B ? 1'b1: 1'b0;
              ST3: Z <= 1'b1;
              ST4: Z <= 0'b0;
       endcase
end

endmodule
```



NOTE: State hex value 0 cooresponds to ST1, 1 to ST2 and 2 to ST3.

8-28.



28 bits/word

Total = 1024 words x 28 bits/word = 28,672 bits

8-32.

a) Opcode = 8 bits , b) 16 bits c) 65,536 d) –32768 to +32767

8-39. Errata: Problem 8-39(d): C ← 0 should be C = 0.

	ADDR	NXT	MS	MC	IL	PI	PL	TD	TA	TB	MB	FS	MD	RW	MM	MW
a)	—	17	NXT	NXA	NLI	NLP	NLP	DR	SA	SB	Register	F = A – B	FnUt	WR	—	NW
	—	17	001	0	0	0	0	0	0	0	0	00101	0	1	0	0
DR = 3, SA = 1, SB = 2																
b)	—	—	CNT	—	NLI	NLP	NLP	DR	SA	—	Register	F = lsr A	FnUt	WR	—	NW
	—	—	000	0	0	0	0	0	0	0	0	10100	0	1	0	0
DR = 5, SA = 5																
c)	—	21	BNZ	NXA	NLI	NLP	NLP	—	—	—	—	—	—	—	—	—
	—	21	111	0	0	0	0	0	0	0	0	00000	0	0	0	0
d)	—	—	CNT	—	NLI	NLP	NLP	DR	SA	—	Register	F = A	FnUt	WR	—	NW
	—	—	000	0	0	0	0	0	0	0	0	00000	1	1	0	0

DR = 6, SA = 6 Note: For R6 + 0, C = 0.

8-47.

Pipeline Fill	3 cycles
Execution Write-Backs	22 cycles
Final Register Load	1 cycle
TOTAL	26 cycles or $26 \times 5 = 130$ ns

Problem Solutions to Problems Marked With a * in

Logic Computer Design Fundamentals, Ed. 2

C H A P T E R 9

9-2.

a)
```
LD    R1, A
LD    R2, B
LD    R3, C
LD    R4, D
ADD   R3, R1, R3
ADD   R1, R1, R2
MUL   R2, R2, R4
MUL   R1, R3, R1
SUB   R1, R1, R2
ST    Y, R1
```

b)
```
MOV   T1, A
ADD   T1, C
MOV   T2, B
MUL   T2, D
MOV   T3, A
ADD   T3, B
MUL   T3, T1
SUB   T3, T2
MOV   Y, T3
```

c)
```
LD    A
ADD   C
ST    T1
LD    B
MUL   D
ST    T2
LD    A
ADD   B
MUL   T1
SUB   T2
ST    Y
```

9-3.

A) $(A + B) \times (A + C) - (B \times D) = AB + CA + \times BD \times -$

B,C)

PUSH A	PUSH B	ADD	PUSH A	PUSH C	ADD
A	B	A+B	A	C	A+C
	A		A+B	A	A+B
				A+B	

MUL	PUSH B	PUSH D	MUL	SUB
(A+B)x(A+C)	B	D	BxD	(A+B)x(A+C) - BxD
	(A+B)x(A+C)	B	(A+B)x(A+C)	
		(A+B)x(A+C)		

9-6.

a) X = 200 – 208 – 1 = –9 b) X = 1111 1111 1111 0111

9-9.

address field = 0

9-11.

a) 3 Register Fields x 5 bits/Field = 15 bits. 32 bits - 15 bits = 17 bit. 2^{17} = 131,072

b) 256 = 8 bits. 2 Register Fields x 5 bits/Field = 10 bits. 32 bits - 8 bits - 10 bits = 14 Memory Bits

9-13.

Read and Write of the FIFO work in the following manner:

Write: $M[WC] \leftarrow DATA$
$ASC \leftarrow ASC + 1$
$WC \leftarrow WC + 1$

Read: $DST \leftarrow M[RC]$
$ASC \leftarrow ASC - 1$
$RC \leftarrow RC + 1$

	WC	RC	ASC
WR	1	0	1
WR	2	0	2
RD	2	1	1
RD	2	2	0

9-17.

a)
ADD R0, R4
ADC R1, R5
ADC R2, R6
ADC R3, R7

b)
$R0 \leftarrow 8C + 5C$, R0 = E8, C = 0
$R1 \leftarrow 35 + FE + 0$, R1 = 33, C = 1
$R2 \leftarrow D7 + 68 + 1$, R2 = 40, C = 1
$R3 \leftarrow 2B + 11 + 1$, R3 = 3D, C = 0

9-20.

OPP	Result Register	C
SHR	0101 1101	1
SHL	1011 1010	1
SHRA	1101 1101	1
SHLA	1011 1010	1
ROR	0101 1101	1
ROL	1011 1010	1
RORC	1101 1101	0
ROLC	1011 1010	1

9-22.

$$\text{Smallest Number} = 0.5 \times 2^{-255}$$
$$\text{Largest Number} = (1 - 2^{-26}) \times 2^{+255}$$

9-24.

E	e	$(e)_2$
+8	15	1111
+7	14	1110
+6	13	1101
+5	12	1100
+4	11	1011
+3	10	1010
+2	9	1001
+1	8	1000
0	7	0111
–1	6	0110
–2	5	0101
–3	4	0100
–4	3	0011
–5	2	0010
–6	1	0001
–7	0	0000

9-27.

```
TEST  R, (0001)16      (AND Immediate 1 with Register R)
BNZ   ADRS             (Branch to ADRS if Z = 0)
```

9-29.

```
a)      A =    0011 0101      53
        B =    0011 0100    - 52
   A – B  =    0000 0001       1
```

b) C (borrow) = 0, Z = 0

c) BH, BHE, BNE

9-31.

	PC	SP	TOS
a) Initially	2000	2735	3250
b) After Call	2147	2734	2002
c) After Return	2002	2735	3250

9-34.

External Interrupts:
1) Hard Disk
2) Mouse
3) Keyboard
4) Modem
5) Printer

Internal Interrupts:
1) Overflow
2) Divide by zero
3) Invalid opcode
4) Memory stack overflow
5) Protection violation

A software interrupt provides a way to call the interrupt routines normally associated with external or internal interrupts by inserting an instruction into the code. Privileged system calls for example must be executed through interrupts in order to switch from user to system mode. Procedure calls do not allow this change.

Problem Solutions to Problems Marked With a * in

Logic Computer Design Fundamentals, Ed. 2

CHAPTER 10

10-1.

a) $PC \leftarrow M[SP]$, $SP \leftarrow SP + 1$

b) $R6 \leftarrow 0F0F_{16}$

c) $R2 \leftarrow M[255 + R3], M[255 + R3] \leftarrow R2$

d) $M[SP] \leftarrow PC + 2, SP \leftarrow SP–1, PC \leftarrow M[M[PC + 2 + 00F0_{16}]]$

PC and *SP* are the value at the time of the instruction fetch.

10-4.

Register: R3, Register Indirect: 3, Immediate: 2002_{10}, Direct: 1000_{10},
Indexed: 1003_{10}, Indexed Indirect: 1003_{10}, Relative: 3003_{10}, Relative Indirect: 3003_{10}

10-10.

	Sym	RT	MC	MM/ LS	MR/ PS	DSA/ MS	SB	MA	MB	MD	FS/ NA	MO
a)	SHRA0	$R9 \leftarrow SD$	0	0	0	09	0D	0	0	0	10	0
	1	$R9 \leftarrow R9$ (Set MSTS)	0	0	0	09	0	0	0	0	00	F
	2	z: $CAR \leftarrow$ SHRA6	3	0	0	6	00	0	0	0	SHRA6	0
	3	$DD \leftarrow DD(15)\|\|DD(15{:}1)$	0	0	0	0F	0F	0	0	0	15	0
	4	$R9 \leftarrow R9 - 1$	0	0	0	09	00	0	0	0	06	F
	5	z : $CAR \leftarrow SHRA3$	3	0	1	6	00	0	0	0	SHRA3	0
	6	$DD \leftarrow DD, CAR \leftarrow WB0(ROM)$	2	1	4	0F	00	0	0	0	00	D
b)	RLC0	$R9 \leftarrow SD$	0	0	0	09	0D	0	0	0	10	0
	1	$R9 \leftarrow R9$ (Set MSTS)	0	0	0	09	0	0	0	0	00	F
	2	z: $CAR \leftarrow$ RLC6	3	0	0	6	00	0	0	0	RLC6	0
	3	$DD \leftarrow DD(14{:}0)\|\|C, C \leftarrow DD(15)$	0	0	0	0F	0F	0	0	0	1B	D
	4	$R9 \leftarrow R9 - 1$	0	0	0	09	00	0	0	0	06	F
	5	$\bar{z}$: $CAR \leftarrow RLC2$	3	0	1	6	00	0	0	0	RLC2	0
	6	$DD \leftarrow DD, CAR \leftarrow WB0(ROM)$	2	1	4	0F	00	0	0	0	00	D
c)	BV0	V: $CAR \leftarrow BRA$	3	0	0	4	00	0	0	0	BRA	0
	1	$CAR \leftarrow$ INT0(ROM)	2	1	5	00	00	0	0	0	00	0

10-12.

Sym	RT	MC	MM /LS	MR /PS	DSA /MS	SB	MA	MB	MD	FS /NA	MO
MUL0	$R9 \leftarrow 16$	0	0	0	09	10	0	2	0	10	0
1	$R10 \leftarrow DD$	0	0	0	0A	0F	0	0	0	10	0
2	$DD \leftarrow R0$	0	0	0	0F	00	0	0	0	10	0
3	$SD \leftarrow rorc(SD)$	0	0	0	0D	0D	0	0	0	17	D
4	$\bar{C}$: $CAR \leftarrow$ MUL6	3	0	1	3	00	0	0	0	MUL6	0
5	$DD \leftarrow DD + R10$	0	0	0	0F	0A	0	0	0	02	D
6	$DD \leftarrow rorc(DD)$	0	0	0	0F	0F	0	0	0	17	D
7	$R9 \leftarrow R9 - 1$	0	0	0	09	00	0	0	0	06	F
8	z: $CAR \leftarrow MWB0$	3	0	0	6	00	0	0	0	MWB0	0
9	$CAR \leftarrow$ MUL3	3	0	0	0	00	0	0	0	MUL3	0

MWB0 is the write back routine for the Multiply Operation.

10-17.

Cycle 1: PC = 10F

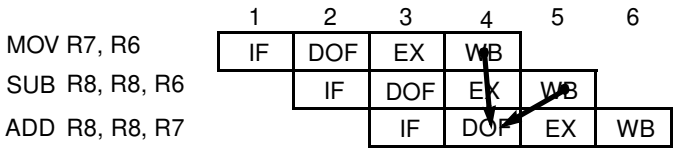
Cycle 2: PC_{-1} = 110, IR = 4418 $2F01_{16}$

Cycle 3: PC_{-2} = 110, RW = 1, DA = 01, MD = 0, BS = 0, PS = X, MW = 0, FS = 02, SH = 01, MA = 0, MB = 1
BUS A = 0000 001F, BUS B = 0000 2F01

Cycle 4: RW = 1, DA = 01, MD = 0, D0 = 0000 2F20, D1 = XXXX XXXX, D2 = 0000 00000

Cycle 5: R1 = 0000 2F20

10-21.



10-23.

a)
```
MOV R7, R6
SUB R8, R8, R6
NOP
ADD R8, R8, R7
```

b)
```
SUB R7, R7, R6
NOP
BNZ R7, 000F
NOP
NOP
AND R8, R7
OR  R5, R7
```

10-28.

a)
```
LD   R1, INDEX
LD   R2, ADDRESS
ADD  R3, R2, R1
LD   R4, R3
SBI  R4, R4, 1
ST   R3, R4
```
Time = 10 RISC Clock Cycles

b) IF = 2 CISC Clock Cycles
1OF = 4 CISC Clock Cycles
EX = 1 CISC Clock Cycles
WB = 2 CISC Clock Cycles
INT = 1 CISC Clock Cycles

Time = 10 CISC Clock Cycles

Time = 30 RISC Clock Cycles