

***Digital Logic and Microprocessor Design with
Interfacing 2nd edition Hwang Solutions Manual***

SKU: 9781305859456-SOLUTIONS

Chapter 2 Solutions

2.1.

- a) 1000010
- b) 110001
- c) 1000000001
- d) 1101100000
- e) 11101101001
- f) 11111011111

2.2.

- a) 30_{10} , 36_8 , $1E_{16}$
- b) 26, 32, 1A
- c) 291, 443, 123
- d) 91, 133, 5B
- e) 878_{10} , 1556_8 , $36E_{16}$
- f) 1514, 2752, 5EA

2.3.

- a) 01100110
- b) 11100011
- c) 0010111111101000
- d) 011111000010
- e) 0101101000101101
- f) 1110000010001011

2.4.

- a) 000011101010
- b) 111100010110
- c) 000010011100
- d) 101111000100
- e) 111000101000

2.5.

	Decimal	Octal	Hexadecimal
a)	-53	713	CB
b)	30	36	1E
c)	-19	55	ED
d)	-167	7531	F59
e)	428	654	1AC

2.6.

- a) 11100101; 229
- b) 10110001; 177
- c) 411010110; 214
- d) 401011101; 93

2.7.

- a) 11100101; -27
- b) 10110001; -79

- c) 411010110; -42
d) 401011101; 93

2.8.

- a) 01101111; 111
b) 11001001; 201
c) 11110000; 240
d) 10110001; 177

2.9.

- a) 01101111; 111
b) 11001001; -55
c) 11110000; -16
d) 10110001; -79

2.10.

Binary calculations	Unsigned decimal calculations	Signed decimal calculations
1001 + 0011 = 1100 No overflow	9 + 3 = 12 No overflow error	-7 + 3 = -4 No overflow error
0110 + 1011 = 40001 Overflow	6 + 11 = 1 Overflow error	6 + (-5) = 1 No overflow error
0101 + 0110 = 1011 No overflow	5 + 6 = 11 No overflow error	5 + 6 = -5 Overflow error
0101 - 0110 = 1111 No overflow	5 - 6 = 15 Overflow error	5 - 6 = -1 No overflow error
1011 - 0101 = 0110 No overflow	11 - 5 = 6 No overflow error	-5 - 5 = 6 Overflow error

2.11.

x	y	z	x'y'z'	x'yz	xy'z'	xyz	F
0	0	0	1	0	0	0	1
0	0	1	0	0	0	0	0
0	1	0	0	0	0	0	0
0	1	1	0	1	0	0	1
1	0	0	0	0	1	0	1
1	0	1	0	0	0	0	0
1	1	0	0	0	0	0	0
1	1	1	0	0	0	1	1

(a)

x	y	z	xy'z	x'yz'	xyz	xyz'	F
0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	1	0	0	1	0	0	1
0	1	1	0	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	0	0	0	1
1	1	0	0	0	0	1	1
1	1	1	0	0	1	0	1

(b)

w	x	y	z	w'xy'z	w'xyz	wxy'z	wxyz	F
0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0
0	0	1	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0
0	1	0	0	0	0	0	0	0
0	1	0	1	1	0	0	0	1
0	1	1	0	0	0	0	0	0
0	1	1	1	0	1	0	0	1
1	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0
1	0	1	0	0	0	0	0	0
1	0	1	1	0	0	0	0	0
1	1	0	0	0	0	0	0	0
1	1	0	1	0	0	1	0	1
1	1	1	0	0	0	0	0	0
1	1	1	1	0	0	0	1	1

(c)

w	x	y	z	wxy'z	w'y'z'	wxz	xyz'	F
0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0
0	0	1	0	0	1	0	0	1
0	0	1	1	0	0	0	0	0
0	1	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0
0	1	1	0	0	1	0	1	1
0	1	1	1	0	0	0	0	0
1	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0
1	0	1	0	0	0	0	0	0
1	0	1	1	0	0	0	0	0
1	1	0	0	0	0	0	0	0
1	1	0	1	1	0	1	0	1
1	1	1	0	0	0	0	1	1
1	1	1	1	0	0	1	0	1

(d)

x	y	z	xy'	x'y'z	xyz'	F
0	0	0	0	0	0	0
0	0	1	0	1	0	1
0	1	0	0	0	0	0
0	1	1	0	0	0	0
1	0	0	1	0	0	1
1	0	1	1	0	0	1
1	1	0	0	0	1	1
1	1	1	0	0	0	0

(e)

w	x	y	z	w'z'	w'xy	wx'z	wxyz	F
0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	0	0	0
0	0	1	0	1	0	0	0	1
0	0	1	1	0	0	0	0	0
0	1	0	0	1	0	0	0	1
0	1	0	1	0	0	0	0	0
0	1	1	0	1	1	0	0	1
0	1	1	1	0	1	0	0	1
1	0	0	0	0	0	0	0	0
1	0	0	1	0	0	1	0	1
1	0	1	0	0	0	0	0	0
1	0	1	1	0	0	1	0	1
1	1	0	0	0	0	0	0	0
1	1	0	1	0	0	0	0	0
1	1	1	0	0	0	0	0	0
1	1	1	1	0	0	0	1	1

(f)

x	y	z	x'	y'	x+y'	yz	(yz)'	[(x+y')(yz)']	xy'	x'y	(xy'+x'y)	F
0	0	0	1	1	1	0	1	1	0	0	0	0
0	0	1	1	1	1	0	1	1	0	0	0	0
0	1	0	1	0	0	0	1	0	0	1	1	0
0	1	1	1	0	0	1	0	0	0	1	1	0
1	0	0	0	1	1	0	1	1	1	0	1	1
1	0	1	0	1	1	0	1	1	1	0	1	1
1	1	0	0	0	1	0	1	1	0	0	0	0
1	1	1	0	0	1	1	0	0	0	0	0	0

(g)

N_3	N_2	N_1	N_0	$N_3'N_2'N_1N_0'$	$N_3'N_2'N_1N_0$	$N_3N_2'N_1N_0'$	$N_3N_2'N_1N_0$	$N_3N_2N_1'N_0'$	$N_3N_2N_1'N_0$	F
0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	1
0	0	1	1	0	1	0	0	0	0	1
0	1	0	0	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0
1	0	1	0	0	0	1	0	0	0	1
1	0	1	1	0	0	0	1	0	0	1
1	1	0	0	0	0	0	0	1	0	1
1	1	0	1	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	1	1

(h)

2.12.

- (a) $F = a'bc' + a'bc + abc'$
 (b) $F = w'x'yz' + w'xy'z' + w'xy'z + w'xyz + wx'y'z + wx'yz' + wxy'z' + wxy'z + wxyz$
 (c) $F_1 = w'x'y'z' + w'x'yz + w'xy'z + w'xyz' + wx'y'z + wx'yz' + wxy'z' + wxyz$
 $F_2 = w'x'y'z' + w'x'yz + w'x'yz' + w'x'yz + w'xy'z + wx'y'z' + wx'y'z + wxy'z' + wxyz' + wxyz$
 (d) $F = N_3'N_2'N_1N_0' + N_3'N_2'N_1N_0 + N_3'N_2N_1N_0' + N_3N_2'N_1N_0' + N_3N_2'N_1N_0 + N_3N_2N_1'N_0' + N_3N_2N_1N_0$

2.14.

(a)

w	x	y	z	$w'z'$	$w'xy$	$wx'z$	$wxyz$	<i>Left Side</i>	$w'z'$	xyz	$wx'y'z$	wyz	<i>Right Side</i>
0	0	0	0	1	0	0	0	1	1	0	0	0	1
0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	1	1	0	0	0	1
0	0	1	1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	1	0	0	0	1	1	0	0	0	1
0	1	0	1	0	0	0	0	0	0	0	0	0	0
0	1	1	0	1	1	0	0	1	1	0	0	0	1
0	1	1	1	0	1	0	0	1	0	1	0	0	1
1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	1	0	1	0	0	1	0	1
1	0	1	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	0	0	1	0	1	0	0	0	1	1
1	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	0	1	1	0	1	0	1	1

(b)

y	z	z	y'	yz'	<i>Left Side</i>	<i>Right Side</i>
0	0	0	1	0	1	1
0	1	1	1	0	1	1
1	0	0	0	1	1	1
1	1	1	0	0	1	1

(c)

x	y	z	xyz'	x'	xyz'	<i>Left Side</i>	x'	z'	<i>Right Side</i>
0	0	0	0	1	0	1	1	1	1
0	0	1	0	1	0	1	1	0	1
0	1	0	0	1	0	1	1	1	1
0	1	1	0	1	0	1	1	0	1
1	0	0	1	0	0	1	0	1	1
1	0	1	0	0	0	0	0	0	0
1	1	0	0	0	1	1	0	1	1
1	1	1	0	0	0	0	0	0	0

(d)

x	y	z	xy	$x'z$	yz	<i>Left Side</i>	xy	$x'z$	<i>Right Side</i>
0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	0	1	1
0	1	0	0	0	0	0	0	0	0
0	1	1	0	1	1	1	0	1	1
1	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0
1	1	0	1	0	0	1	1	0	1
1	1	1	1	0	1	1	1	0	1

(e)

w	x	y	z	$w'x'yz'$	$w'x'yz$	$wx'yz'$	$wx'yz$	$wxyz$	<i>Left Side</i>	x'	wz	$(x' + wz)$	<i>Right Side</i>
0	0	0	0	0	0	0	0	0	0	1	0	1	0
0	0	0	1	0	0	0	0	0	0	1	0	1	0
0	0	1	0	1	0	0	0	0	1	1	0	1	1
0	0	1	1	0	1	0	0	0	1	1	0	1	1
0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	1	0	1	0
1	0	0	1	0	0	0	0	0	0	1	1	1	0
1	0	1	0	0	0	1	0	0	1	1	0	1	1
1	0	1	1	0	0	0	1	0	1	1	1	1	1
1	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	0	0	0	0	0	0	0	1	1	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	1	1	0	1	1	1

(f)

w	x	y	z	w'xy'z	w'xyz	wxy'z	wxyz	Left Side	Right Side
0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0
0	1	0	1	1	0	0	0	1	1
0	1	1	0	0	0	0	0	0	0
0	1	1	1	0	1	0	0	1	1
1	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0
1	1	0	1	0	0	1	0	1	1
1	1	1	0	0	0	0	0	0	0
1	1	1	1	0	0	0	1	1	1

(g)

x_i	y_i	c_i	$x_i y_i$	$x_i + y_i$	$c_i(x_i + y_i)$	Left Side	$x_i y_i c_i$	$x_i y_i c_i'$	$x_i y_i' c_i$	$x_i' y_i c_i$	Right Side
0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0
0	1	0	0	1	0	0	0	0	0	0	0
0	1	1	0	1	1	1	0	0	1	1	1
1	0	0	0	1	0	0	0	0	0	0	0
1	0	1	0	1	1	1	0	0	1	0	1
1	1	0	1	1	0	1	0	1	0	0	1
1	1	1	1	1	1	1	1	0	0	0	1

(h)

x_i	y_i	c_i	$x_i y_i$	$x_i + y_i$	$c_i(x_i + y_i)$	Left Side	$x_i y_i$	$x_i \oplus y_i$	$c_i(x_i \oplus y_i)$	Right Side
0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0
0	1	0	0	1	0	0	0	1	0	0
0	1	1	0	1	1	1	0	1	1	1
1	0	0	0	1	0	0	0	1	0	0
1	0	1	0	1	1	1	0	1	1	1
1	1	0	1	1	0	1	1	0	0	1
1	1	1	1	1	1	1	1	0	0	1

2.19.

(a) $w'z' + w'xy + wx'z + wxyz$
 $= w'x'y'z' + w'x'yz' + w'xy'z' + w'xyz' + w'xyz + wx'y'z + wx'yz + wxyz$
 $= w'x'y'z' + w'x'yz' + w'xy'z' + w'xyz' + w'xyz + wx'y'z + wx'yz + wxyz$
 $= w'z' + w'xyz + wx'y'z + wx'yz + wxyz$
 $= w'z' + (w'+w)xyz + wx'y'z + w(x'+x)yz$
 $= w'z' + xyz + wx'y'z + wyz$

(b) $z + y' + yz'$
 $= z(y'+y) + (z'+z)y' + yz'$

$$\begin{aligned}
&= zy' + zy + z'y' + \cancel{zy} + yz' \\
&= z(y'+y) + z'(y'+y) \\
&= z + z' \\
&= 1
\end{aligned}$$

$$\begin{aligned}
(c) \quad & x y' z' + x' + x y z' \\
&= x z' (y' + y) + x' \\
&= x z' + x' \\
&= x z' + 1 x' \\
&= (x + 1)(x + x')(z' + 1)(z' + x') \\
&= 1 \bullet 1 \bullet 1 (z' + x') \\
&= x' + z'
\end{aligned}$$

$$\begin{aligned}
(d) \quad & xy + x'z + yz \\
&= xy(z'+z) + x'(y'+y)z + (x'+x)yz \\
&= xyz' + xyz + x'y'z + x'yz + \cancel{x'yz} + \cancel{xyz} \\
&= xy(z'+z) + x'(y'+y)z \\
&= xy(1) + x'(1)z \\
&= xy + x'z
\end{aligned}$$

$$\begin{aligned}
(e) \quad & w'x'yz' + w'x'yz + wx'yz' + wx'yz + wxyz \\
&= [w'x'yz' + w'x'yz + wx'yz' + wx'yz] + [wxyz] \\
&= x'y(w'z' + w'z + wz' + wz) + w(x'+x)yz \\
&= x'y + wyz \\
&= y(x' + wz)
\end{aligned}$$

$$\begin{aligned}
(f) \quad & w'xy'z + w'xyz + wxy'z + wxyz \\
&= xy'z(w' + w) + xyz(w' + w) \\
&= xy'z + xyz \\
&= xz(y + y') \\
&= xz
\end{aligned}$$

$$\begin{aligned}
(g) \quad & x_i y_i + c_i (x_i + y_i) \\
&= x_i y_i + x_i c_i + y_i c_i \\
&= x_i y_i (c_i + c_i') + x_i (y_i + y_i') c_i + (x_i + x_i') y_i c_i \\
&= x_i y_i c_i + x_i y_i c_i' + \cancel{x_i y_i c_i} + x_i y_i' c_i + \cancel{x_i y_i c_i} + x_i' y_i c_i \\
&= x_i y_i c_i + x_i y_i c_i' + x_i y_i' c_i + x_i' y_i c_i
\end{aligned}$$

$$\begin{aligned}
(h) \quad & x_i y_i + c_i (x_i + y_i) \\
&= x_i y_i + x_i c_i + y_i c_i \\
&= x_i y_i (c_i + c_i') + x_i (y_i + y_i') c_i + (x_i + x_i') y_i c_i \\
&= x_i y_i c_i + x_i y_i c_i' + \cancel{x_i y_i c_i} + x_i y_i' c_i + \cancel{x_i y_i c_i} + x_i' y_i c_i \\
&= x_i y_i c_i + x_i y_i c_i' + x_i y_i' c_i + x_i' y_i c_i \\
&= x_i y_i (c_i + c_i') + c_i (x_i y_i' + x_i' y_i) \\
&= x_i y_i + c_i (x_i \oplus y_i)
\end{aligned}$$

2.20.

$$\begin{aligned}
(a) \quad & x'y'z' + x'yz + xy'z' + xyz \\
&= (x + x')y'z' + (x + x')yz \\
&= y'z' + yz \\
&= y \odot z
\end{aligned}$$

$$\begin{aligned}
(b) \quad & xy'z + x'yz' + xyz + xyz' \\
&= (x + x')yz' + x(y + y')z \\
&= yz' + xz
\end{aligned}$$

(c) $w'xy'z + w'xyz + wxy'z + wxyz$
 $= xz(w'y' + w'y + wy' + wy)$
 $= xz$

(d) $wxy'z + w'yz' + wxz + xyz'$
 $= wxz + yz'(x+w')$

(e) $xy' + x'y'z + xyz'$
 $= xy'z + xy'z' + x'y'z + xyz'$
 $= xy' + y'z + xz'$
 $= y'(x+z) + xz'$

(f) $w'z' + w'xy + wx'z + wxyz$
 $= w'z' + w'xyz + w'xyz' + wx'z + wxyz$
 $= w'z' + xyz(w' + w) + w'xyz' + wx'z$
 $= w'z' + xyz + wx'z$
 $= w'z' + z(xy + wx')$

(g) $[(x+y')(yz)'](xy' + x'y)$
 $= [(x+y')(y'z')](xy' + x'y)$
 $= [xy' + xz' + y'y' + y'z'] (xy' + x'y)$
 $= [xz' + y'] (xy' + x'y)$
 $= xy'z' + xx'yz' + y'xy' + y'x'y$
 $= xy'z' + xy'$
 $= xy'$

(h) $N_3'N_2'N_1N_0' + N_3'N_2'N_1N_0 + N_3N_2'N_1N_0' + N_3N_2'N_1N_0 + N_3N_2N_1'N_0' + N_3N_2N_1N_0$
 $=$

2.21.

$$\begin{aligned}
 F &= (x' + y' + x'y' + xy)(x' + yz) \\
 &= (x' \bullet 1 + y' \bullet 1 + x'y' + xy)(x' + yz) && \text{by Theorem 6a} \\
 &= (x'(y + y') + y'(x + x') + x'y' + xy)(x' + yz) && \text{by Theorem 9b} \\
 &= (x'y + x'y' + y'x + y'x' + x'y' + xy)(x' + yz) && \text{by Theorem 12a} \\
 &= (x'y + x'y' + y'x + \cancel{y'x'} + \cancel{x'y'} + xy)(x' + yz) && \text{by Theorem 7b} \\
 &= (x'(y + y') + x(y + y'))(x' + yz) && \text{by Theorem 12a} \\
 &= (x' \bullet 1 + x \bullet 1)(x' + yz) && \text{by Theorem 9b} \\
 &= (x' + x)(x' + yz) && \text{by Theorem 6a} \\
 &= 1(x' + yz) && \text{by Theorem 9b} \\
 &= (x' + yz) && \text{by Theorem 6a}
 \end{aligned}$$

2.22.

For three variables (x, y, z), there is a total of eight (2^3) minterms. The function has five minterms, therefore, the inverted function will have three ($8-5=3$) minterms. Hence, implementing the inverted function and then adding a NOT gate at the final output will result in a smaller circuit. The circuit requires 3 AND gates, 1 OR gate, and 1 NOT gate.

2.23.

w	x	y	z	4 AND	4 NAND	4 NOR	4 XOR	4 XNOR
0	0	0	0	0	1	1	0	1
0	0	0	1	0	1	0	1	0

v	w	x	y	z	5 XOR	5 XNOR
0	0	0	0	0	0	0
0	0	0	0	1	1	1

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

(a) (b) (c) (d) (e)

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

(f) (g)

2.24.

w	x	y	z	$(x \odot y)'$	$(xyz)'$	$(x \odot y)' + (xyz)'$	$(w' + x + z)$	F
0	0	0	0	0	1	1	1	1
0	0	0	1	0	1	1	1	1
0	0	1	0	1	1	1	1	1
0	0	1	1	1	1	1	1	1
0	1	0	0	1	1	1	1	1
0	1	0	1	1	1	1	1	1
0	1	1	0	0	1	1	1	1
0	1	1	1	0	0	0	1	0
1	0	0	0	0	1	1	0	0
1	0	0	1	0	1	1	1	1
1	0	1	0	1	1	1	0	0
1	0	1	1	1	1	1	1	1
1	1	0	0	1	1	1	1	1
1	1	0	1	1	1	1	1	1
1	1	1	0	0	1	1	1	1
1	1	1	1	0	0	0	1	0

(a)

x	y	z	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

(b)

w	x	y	z	$w'xy'z$	$(x \oplus y)$	$w'z (y \oplus x)$	$[w'xy'z + w'z (y \oplus x)]$	F
0	0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	0	1
0	0	1	0	0	1	0	0	1
0	0	1	1	0	1	1	1	0
0	1	0	0	0	1	0	0	1
0	1	0	1	1	1	1	1	0
0	1	1	0	0	0	0	0	1
0	1	1	1	0	0	0	0	1
1	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	1
1	0	1	0	0	1	0	0	1
1	0	1	1	0	1	0	0	1
1	1	0	0	0	1	0	0	1
1	1	0	1	0	1	0	0	1
1	1	1	0	0	0	0	0	1
1	1	1	1	0	0	0	0	1

(c)

2.25.

a)

w	x	y	z	$(x \odot y)'$	$(xyz)'$	$(x \odot y)' + (xyz)'$	$(w' + x + z)$	F
0	0	0	0	0	1	1	1	1
0	0	0	1	0	1	1	1	1
0	0	1	0	1	1	1	1	1
0	0	1	1	1	1	1	1	1
0	1	0	0	1	1	1	1	1
0	1	0	1	1	1	1	1	1
0	1	1	0	0	1	1	1	1
0	1	1	1	0	0	0	1	0
1	0	0	0	0	1	1	0	0
1	0	0	1	0	1	1	1	1
1	0	1	0	1	1	1	0	0
1	0	1	1	1	1	1	1	1
1	1	0	0	1	1	1	1	1
1	1	0	1	1	1	1	1	1
1	1	1	0	0	1	1	1	1
1	1	1	1	0	0	0	1	0

(a)

x	y	z	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

(b)

w	x	y	z	$w'xy'z$	$(x \oplus y)$	$w'z (y \oplus x)$	$[w'xy'z + w'z (y \oplus x)]$	F
0	0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	0	1
0	0	1	0	0	1	0	0	1
0	0	1	1	0	1	1	1	0
0	1	0	0	0	1	0	0	1
0	1	0	1	1	1	1	1	0
0	1	1	0	0	0	0	0	1
0	1	1	1	0	0	0	0	1
1	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	1
1	0	1	0	0	1	0	0	1
1	0	1	1	0	1	0	0	1
1	1	0	0	0	1	0	0	1
1	1	0	1	0	1	0	0	1
1	1	1	0	0	0	0	0	1
1	1	1	1	0	0	0	0	1

(c)

b)

$$\begin{aligned}
 F &= [(x \odot y)' + (xyz)'] (w' + x + z) \\
 &= [xy' + x'y + x' + y' + z'] (w' + x + z) \\
 &= (x' + y' + z') (w' + x + z) \\
 &= (ww' + x' + y' + z') (w' + x + yy' + z) \\
 &= (w + x' + y' + z') (w' + x' + y' + z') (w' + x + y + z) (w' + x + y' + z) \\
 &= \Pi(M_7 + M_8 + M_{10} + M_{15})
 \end{aligned}$$

(a)

$$\begin{aligned}
 F &= x \oplus y \oplus z \\
 &= (xy' + x'y)z' + (xy' + x'y)'z \\
 &= xy'z' + x'yz' + xy'z + x'yz \\
 &= (x+y+z)(x+y'+z')(x'+y'+z)(x'+y'+z') \\
 &= \Pi(M_0 + M_3 + M_6 + M_7)
 \end{aligned}$$

(b)

$$\begin{aligned}
 F &= [w'xy'z + w'z (y \oplus x)]' \\
 &= [w'xy'z]' [w'z (y \oplus x)]' \\
 &= [w+x'+y+z'] [w+z'+(y \oplus x)'] \\
 &= [w+x'+y+z'] [w+z'+xy+x'y'] \\
 &= [w+x'+y+z'] [w+x+y'+z'] [w+x'+y+z'] \\
 &= \Pi(M_3 + M_5)
 \end{aligned}$$

(c)

2.26.

$$\begin{aligned}
 F &= [(x \odot y)' + (xyz)'] (w' + x + z) \\
 &= [xy' + x'y + x' + y' + z'] (w' + x + z) \\
 &= (x' + y' + z') (w' + x + z) \\
 &= x'w' + x'z + y'w' + y'x + y'z + z'w' + z'x + z'z \\
 &= w'x' + x'z + w'y' + xy' + y'z + w'z' + xz' \\
 &= (x \oplus z) + xy' + w'x' \\
 &\text{or } (x \oplus z) + xy' + w'z' \\
 &\text{or } (x \oplus z) + y'z + w'x' \\
 &\text{or } (x \oplus z) + y'z + w'z'
 \end{aligned}$$

(a)

$$F = x \oplus y \oplus z$$

(b)

$$\begin{aligned}
 F &= [w'xy'z + w'z(y \oplus x)]' \\
 &= [w'xy'z]' [w'z(y \oplus x)]' \\
 &= [w+x'+y+z'] [w+z'+(y \oplus x)'] \\
 &= [w+x'+y+z'] [w+z'+xy+x'y'] \\
 &= w + w'z' + w'xy + w'x'y' + wx' + x'z' + x'y' + wy + yz' + xy + wz' + z' + xy'z' + x'y'z' \\
 &= w + z' + x'y' + xy \\
 &= w + z' + (x \odot y)
 \end{aligned}$$

(c)

2.27.

x	y	<i>Left Side</i> $x \oplus y$	$x \odot y$	<i>Right Side</i> $(x \odot y)'$
0	0	0	1	0
0	1	1	0	1
1	0	1	0	1
1	1	0	1	0

(a)

x	y	y'	<i>Left Side</i> $x \oplus y'$	<i>Right Side</i> $x \odot y$
0	0	1	1	1
0	1	0	0	0
1	0	1	0	0
1	1	0	1	1

(b)

w	x	y	z	$w \oplus x$	$y \oplus z$	<i>Left Side</i> $(w \oplus x) \odot (y \oplus z)$	$w \odot x$	$y \odot z$	<i>Right Side</i> $(w \odot x) \odot (y \odot z)$	<i>Right Side</i> $((w \odot x) \odot y) \odot z$
0	0	0	0	0	0	1	1	1	1	1
0	0	0	1	0	1	0	1	0	0	0
0	0	1	0	0	1	0	1	0	0	0
0	0	1	1	0	0	1	1	1	1	1
0	1	0	0	1	0	0	0	1	0	0
0	1	0	1	1	1	1	0	0	1	1
0	1	1	0	1	1	1	0	0	1	1
0	1	1	1	1	0	0	0	1	0	0
1	0	0	0	1	0	0	0	1	0	0
1	0	0	1	1	1	1	0	0	1	1
1	0	1	0	1	1	1	0	0	1	1
1	0	1	1	1	0	0	0	1	0	0
1	1	0	0	0	0	1	1	1	1	1
1	1	0	1	0	1	0	1	0	0	0
1	1	1	0	0	1	0	1	0	0	0
1	1	1	1	0	0	1	1	1	1	1

(c)

x	y	z	$(xy)'$	$((xy)'x)'$	$((xy)'y)'$	$[((xy)'x)'((xy)'y)']$	<i>Left Side</i> $[((xy)'x)'((xy)'y)']'$	<i>Right Side</i> $x \oplus y$
0	0	0	1	1	1	1	0	0
0	0	1	1	1	1	1	0	0
0	1	0	1	1	0	0	1	1
0	1	1	1	1	0	0	1	1
1	0	0	1	0	1	0	1	1
1	0	1	1	0	1	0	1	1
1	1	0	0	1	1	1	0	0
1	1	1	0	1	1	1	0	0

(d)

2.28.

$$\begin{aligned}
 (x \oplus y) &= xy' + x'y \\
 &= xx' + xy' + x'y + yy' \\
 &= (x + y)(x' + y') \\
 &= (x'y')'(xy)' \\
 &= [(x'y') + (xy)]' \\
 &= (x \odot y)'
 \end{aligned}$$

(a)

$$\begin{aligned}
 &[(xy)'x]((xy)'y)']' \\
 &= ((xy)'x) + ((xy)'y) \\
 &= (x' + y')x + (x' + y')y \\
 &= xx' + xy' + x'y + yy' \\
 &= xy' + x'y \\
 &= x \oplus y
 \end{aligned}$$

(d)

$$\begin{aligned}
 x \oplus y' &= xy + x'y' \\
 &= x \odot y
 \end{aligned}$$

(b)

2.29.

$$\begin{aligned}
 x \oplus y \oplus z &= (x \oplus y) \oplus z \\
 &= (x'y + xy') \oplus z \\
 &= (x'y + xy')z' + (x'y + xy')z \\
 &= x'yz' + xy'z' + (x'y)'(xy)'z \\
 &= x'yz' + xy'z' + (x+y')(x'+y)z \\
 &= x'yz' + xy'z' + xx'z + xyz + x'y'z + y'y'z \\
 &= x'yz' + xy'z' + xyz
 \end{aligned}$$

2.30.

$$\begin{aligned}
 x \oplus y \oplus z &= (x \oplus y) \oplus z \\
 &= (x'y + xy') \oplus z \\
 &= (x'y + xy')'z + (x'y + xy')z' \\
 &= (x'y)' \cdot (xy)'z + x'yz' + xy'z' \\
 &= (x + y)' \cdot (x' + y)z + x'yz' + xy'z' \\
 &= xx'z + xyz + x'y'z + y'y'z + x'yz' + xy'z' \\
 &= (xy + x'y')z + (x'y + xy')z' \\
 &= (xy + x'y')z + (xy + x'y')'z' \\
 &= (x \odot y)z + (x \odot y)'z' \\
 &= x \odot y \odot z
 \end{aligned}$$

2.31.

- (a) $F(x,y,z) = \Sigma(m_0, m_3, m_4, m_7)$
 (b) $F(x,y,z) = \Sigma(m_2, m_5, m_6, m_7)$
 (c) $F(w,x,y,z) = \Sigma(m_5, m_7, m_{13}, m_{15})$
 (d) $F(w,x,y,z) = \Sigma(m_2, m_6, m_{13}, m_{14}, m_{15})$
 (e) $F(x,y,z) = \Sigma(m_1, m_4, m_5, m_6)$
 (f) $F(w,x,y,z) = \Sigma(m_0, m_2, m_4, m_6, m_7, m_9, m_{11}, m_{15})$
 (g) $F(x,y,z) = \Sigma(m_4, m_5)$
 (h) $F(N_3, N_2, N_1, N_0) = \Sigma(m_2, m_3, m_{10}, m_{11}, m_{12}, m_{15})$

(a)

- (a) $F(x,y,z) = \Pi(M_1, M_2, M_5, M_6)$
 (b) $F(x,y,z) = \Pi(M_0, M_1, M_3, M_4)$
 (c) $F(w,x,y,z) = \Pi(M_0, M_1, M_2, M_3, M_4, M_6, M_8, M_9, M_{10}, M_{11}, M_{12}, M_{14})$
 (d) $F(w,x,y,z) = \Pi(M_0, M_1, M_3, M_4, M_5, M_7, M_8, M_9, M_{10}, M_{11}, M_{12})$
 (e) $F(x,y,z) = \Pi(M_0, M_2, M_3, M_7)$
 (f) $F(w,x,y,z) = \Pi(M_1, M_3, M_5, M_8, M_{10}, M_{12}, M_{13}, M_{14})$
 (g) $F(x,y,z) = \Pi(M_0, M_1, M_2, M_3, M_6, M_7)$
 (h) $F(N_3, N_2, N_1, N_0) = \Pi(M_0, M_1, M_4, M_5, M_6, M_7, M_8, M_9, M_{13}, M_{14})$

(b)

2.32.

- (a) $F(x,y,z) = x'y'z + x'yz + xyz$
 (b) $F(w,x,y,z) = w'x'y'z + w'x'yz + w'xyz$
 (c) $F(x,y,z) = (x+y+z')(x+y'+z')(x'+y'+z)$
 (d) $F(w,x,y,z) = (w+x+y+z')(w+x+y+z)(w+x+y'+z')$
 (e) $F'(x,y,z) = x'y'z' + x'yz' + xy'z' + xy'z + xyz'$
 (f) $F(x,y,z) = (x+y+z)(x+y'+z)(x'+y+z)(x'+y'+z)(x'+y'+z)$

2.33.

F' is expressed as a sum of its 0-minterms. Therefore, F is the sum of its 1-minterms = $\Sigma(0, 2, 4, 5, 6)$. Using three variables, the truth table is as follows:

x	y	z	Minterms	F
0	0	0	$m_0 = x'y'z'$	1
0	0	1	$m_1 = x'y'z$	0
0	1	0	$m_2 = x'y z'$	1
0	1	1	$m_3 = x'y z$	0
1	0	0	$m_4 = x y' z'$	1
1	0	1	$m_5 = x y' z$	1
1	1	0	$m_6 = x y z'$	1
1	1	1	$m_7 = x y z$	0

2.34.

$$\begin{aligned}
 F &= \Sigma(3, 4, 5) = m_3 + m_4 + m_5 \\
 &= x'y'z + xy'z' + xy'z \\
 &= (x' + x + x)(x' + x + y')(x' + x + z) \\
 &\quad (x' + y' + x)(x' + y' + y')(x' + y' + z) \\
 &\quad (x' + z' + x)(x' + z' + y')(x' + z' + z) \\
 &\quad (y + x + x)(y + x + y')(y + x + z) \\
 &\quad (y + y' + x)(y + y' + y')(y + y' + z) \\
 &\quad (y + z' + x)(y + z' + y')(y + z' + z) \\
 &\quad (z + x + x)(z + x + y')(z + x + z) \\
 &\quad (z + y' + x)(z + y' + y')(z + y' + z) \\
 &\quad (z + z' + x)(z + z' + y')(z + z' + z) \\
 &= (x' + y' + z)(x' + y' + z')(x + y + z)(x + y + z')(x + y' + z)
 \end{aligned}$$

2.35.

- a) Product-of-sums (AND-of-OR) format is obtained by using the *duality principle* or De Morgan's Theorem:
 $F' = (x'+y+z) \bullet (x'+y'+z') \bullet (x'+y'+z) \bullet (x'+y'+z')$
- b) Sum-of-products (OR-of-AND) format is obtained by first constructing the truth table for F and then inverting the 0's and 1's to get F' . Then we simply use the AND terms where $F' = 1$.

x	y	z	F	F'
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

$$F' = x'y'z' + x'y'z + x'yz' + x'yz$$

2.36.

a)

$$\begin{aligned}
 F &= w \odot x \odot y \odot z \\
 &= (wx + w'x') \odot y \odot z \\
 &= [(wx + w'x')y + (wx + w'x')'y']z + [(wx + w'x')y + (wx + w'x')'y']'z' \\
 &= wxyz + w'x'yz + (wx)'(w'x')'y'z + [(wx + w'x')y + (wx + w'x')'y']'z' \\
 &= m_{15} + m_3 + (w'+x')(w+x)y'z + [(wx + w'x')y + (wx + w'x')'y']'z' \\
 &= m_{15} + m_3 + w'xy'z + wx'y'z + [(wx + w'x')y + (wx + w'x')'y']'z' \\
 &= m_{15} + m_3 + m_5 + m_9 + [(wx + w'x')y]'[(wx + w'x')'y']'z' \\
 &= m_{15} + m_3 + m_5 + m_9 + [(wx + w'x')' + y'][(wx + w'x') + y]z' \\
 &= m_{15} + m_3 + m_5 + m_9 + [(wx)'(w'x')' + y'] [wxz' + w'x'z' + yz'] \\
 &= m_{15} + m_3 + m_5 + m_9 + [(w'+x')(w+x) + y'] [wxz' + w'x'z' + yz'] \\
 &= m_{15} + m_3 + m_5 + m_9 + [w'x + wx' + y'] [wxz' + w'x'z' + yz'] \\
 &= m_{15} + m_3 + m_5 + m_9 + w'xyz' + wx'y'z' + wxy'z' + w'x'y'z' \\
 &= m_{15} + m_3 + m_5 + m_9 + m_6 + m_{10} + m_{12} + m_0
 \end{aligned}$$

2.37.

a)

```

module P2_24a (
    input w, x, y, z,
    output F
);
    assign F = (~(x^y) | ~(x&y&z)) & (~w|x|z);
endmodule

```

b)

```

module P2_24b (
    input x, y, z,
    output F
);
    assign F = x^y^z;
endmodule

```

c)

```

module P2_24c (
    input w, x, y, z,
    output F
);
    assign F = ~((~w&x&~y&z) | (~w&z&(y^x)));
endmodule

```

2.38.

a)

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.all;

ENTITY P2_24a IS PORT (
    w,x,y,z: IN STD_LOGIC;
    F: OUT STD_LOGIC);
END P2_24a;

ARCHITECTURE Dataflow OF P2_24a IS
BEGIN
    F <= (NOT (x XOR y) OR NOT (x AND y AND z)) AND (NOT w OR x OR z);
END Dataflow;
```

b)

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.all;

ENTITY P2_24b IS PORT (
    x,y,z: IN STD_LOGIC;
    F: OUT STD_LOGIC);
END P2_24b;

ARCHITECTURE Dataflow OF P2_24b IS
BEGIN
    F <= x XOR y XOR z;
END Dataflow;
```

c)

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.all;

ENTITY P2_24c IS PORT (
    w,x,y,z: IN STD_LOGIC;
    F: OUT STD_LOGIC);
END P2_24c;

ARCHITECTURE Dataflow OF P2_24c IS
BEGIN
    F <= NOT((NOT w AND x AND NOT y AND z) OR (NOT w AND z AND (y XOR x)));
END Dataflow;
```

2.39.

```
// this is a Verilog behavioral model of the car security system

module Siren (
    input M, D, V,
    output S
);

    wire term1, term2, term3;

    always @ (M or D or V) begin
        term1 = (M & ~D & V);
        term2 = (M & D & ~V);
        term3 = (M & D & V);
        S = term1 | term2 | term3;
    end

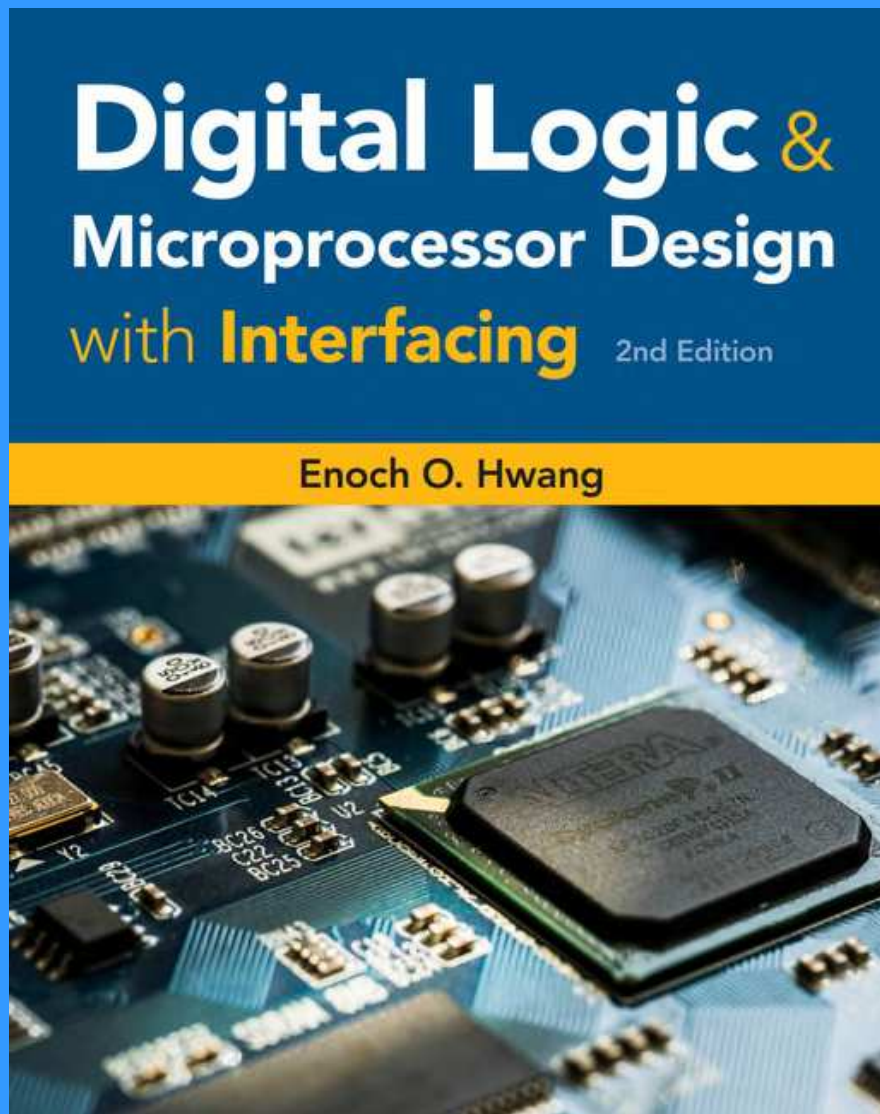
endmodule
```

2.40.

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY Siren IS PORT (
    M, D, V: IN STD_LOGIC;
    S: OUT STD_LOGIC);
END Siren;

ARCHITECTURE Behavioral OF Siren IS
BEGIN
    PROCESS (M, D, V)
    BEGIN
        S <= (M AND NOT D AND V) OR (M AND D AND NOT V) OR (M AND D AND V);
    END PROCESS;
END Behavioral;
```

Chapter 2

Fundamentals of Digital Circuits

© 2018 Cengage Learning®. May not be scanned, copied or duplicated, or posted to a publicly accessible website, in whole or in part.

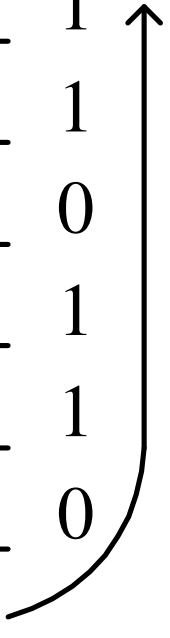
Binary Number

Decimal	Binary	Octal	Hexadecimal
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Value of a Binary Number

- For decimal number:
- $658_{10} = (6 \times 10^2) + (5 \times 10^1) + (8 \times 10^0)$
 $= 600 + 50 + 8 = 658_{10}$
- For binary number:
- $1011011_2 = (1 \times 2^6) + (0 \times 2^5) + (1 \times 2^4) + (1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0)$
 $= 64 + 0 + 16 + 8 + 0 + 2 + 1 = 91_{10}$

Convert Decimal to Digital

2	<u>91</u>	1		Least significant bit	= 1011011
2	<u>45</u>	1			
2	<u>22</u>	0			
2	<u>11</u>	1			
2	<u>5</u>	1			
2	<u>2</u>	0			
1			Most significant bit		

Octal and Hex

■ Octal

001 110 011

1 6 3

5 7 2 4

101 111 010 100

$$5724_8 = (5 \times 8^3) + (7 \times 8^2) \\ + (2 \times 8^1) + (4 \times 8^0)$$

$$= 2560 + 448 + 16 + 4$$

$$= 3028_{10}$$

■ Hex

0110 1101 1011

6 D B

5 C F

0101 1100 1111

$$5CA_{16} = (5 \times 16^2) + (C \times 16^1) + \\ (F \times 16^0)$$

$$= (5 \times 16^2) + (12 \times 16^1) + (15 \times \\ 16^0)$$

$$= 1280 + 192 + 15$$

$$= 1487_{10}$$

Binary Number Arithmetic

$$\begin{array}{r}
 1 \ 0 \ 0 \ 1 \\
 + \ 0 \ 0 \ 1 \ 1 \\
 \hline
 1 \ 1 \ 0 \ 0
 \end{array}$$

$$\begin{array}{r}
 1 \ 1 \ 0 \ 1 \\
 + \ 1 \ 0 \ 1 \ 1 \\
 \hline
 1 \ 1 \ 0 \ 0 \ 0
 \end{array}
 = \begin{array}{r} 13 \\ + \ 11 \\ \hline 24 \end{array}$$

$$\begin{array}{r}
 1 \ 0 \ 0 \ 1 \\
 - \ 0 \ 0 \ 1 \ 1 \\
 \hline
 0 \ 1 \ 1 \ 0
 \end{array}
 = \begin{array}{r} 9 \\ - \ 3 \\ \hline 6 \end{array}$$

$$\begin{array}{r}
 1 \ 1 \ 0 \ 1 \\
 - \ 1 \ 0 \ 1 \ 1 \\
 \hline
 0 \ 0 \ 1 \ 0
 \end{array}
 = \begin{array}{r} 13 \\ - \ 11 \\ \hline 2 \end{array}$$

Negative Number

- Signed or unsigned number representation
- Use two's complement representation for signed numbers
- For signed numbers, the MSB tells whether the number is positive or negative
 - 0 = positive
 - 1 = negative
- If signed number is positive then you can find the value just like for unsigned numbers

Negative Number

- If signed number is negative then you need to do two steps to find its value:
 - (1) Flip all the 1 bits to 0's and all the 0 bits to 1's.
 - (2) Add a 1 to the result obtained from step (1).
- The negated value obtained from step (2) is the value of the original signed number

1001 (original number – MSB is a 1)

(1) 0110 (flip all the bits)

(2) 0111 (add a 1 to the previous number)

0111 = 7, therefore 1001 = -7

Negative Number

4-bit Binary	Two's Complement
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	-8
1001	-7
1010	-6
1011	-5
1100	-4
1101	-3
1110	-2
1111	-1

For 4-bit **unsigned** number:
range is 0 to $2^4 - 1$
= 0 to 15

For 4-bit **signed** number:
range is -2^3 to $2^3 - 1$
= -8 to 7

For n -bit **unsigned** number:
range is 0 to $2^n - 1$

For n -bit **signed** number:
range is -2^{n-1} to $2^{n-1} - 1$

Example

- Find the two's complement number for -58
 - Start with $+58$

2	58	0	Least significant bit Most significant bit
2	29	1	
2	14	0	
2	7	1	
2	3	1	
	1		

= 111010

- Binary for $+58$ is 0111010. Need to add a leading 0, otherwise it is a negative number!

0111010 = 58

1000101 Flip all the bits

1000110 add a 1 to the previous number

Therefore, 1000110 = -58

Sign Extension

- For unsigned numbers, extend with 0
- For signed numbers, extend with the MSB

	Original Number		Sign Extended		Original Number		Sign Extended
	10010		11110010		0101		00000101
Flip bits	01101		00001101				
Add 1	01110		00001110				
Value	– 14		– 14		5		5

Signed Number Arithmetic

$$\begin{array}{rcl}
 0 & 0 & 1 & 1 & = & 3 \\
 + & 1 & 0 & 0 & 1 & = & +(-7) \\
 \hline
 1 & 1 & 0 & 0 & = & -4
 \end{array}$$

$$\begin{array}{rcl}
 3 & = & 0 & 0 & 1 & 1 \\
 + (-3) & = & + & 1 & 1 & 0 & 1 \\
 \hline
 0 & = & \cancel{1} & 0 & 0 & 0 & 0
 \end{array}$$

$$\begin{array}{rcl}
 6 & = & 0 & 1 & 1 & 0 \\
 + & 3 & = & + & 0 & 0 & 1 & 1 \\
 \hline
 9 & \neq & 1 & 0 & 0 & 1
 \end{array}$$

$$\begin{array}{rcl}
 6 & = & 0 & 0 & 1 & 1 & 0 \\
 + & 3 & = & + & 0 & 0 & 0 & 1 & 1 \\
 \hline
 9 & = & 0 & 1 & 0 & 0 & 1
 \end{array}$$

Binary Arithmetic

1. Perform the following 4-bit unsigned number addition. Is there an overflow error?

$$\begin{array}{rcccc} & 0 & 1 & 0 & 1 & = \\ + & 1 & 0 & 1 & 1 & = \\ \hline & & & & & = \end{array}$$

Binary Arithmetic

1. Perform the following 4-bit unsigned number addition. Is there an overflow error?

$$\begin{array}{rcccc} & 0 & 1 & 0 & 1 \\ + & 1 & 0 & 1 & 1 \\ \hline 1 & 0 & 0 & 0 & 0 \end{array} \quad \begin{array}{l} = \\ = \\ = \end{array} \quad \begin{array}{r} 5 \\ +11 \\ \hline \end{array}$$

Binary Arithmetic

1. Perform the following 4-bit unsigned number addition. Is there an overflow error?

$$\begin{array}{rcccc} & 0 & 1 & 0 & 1 \\ + & 1 & 0 & 1 & 1 \\ \hline 1 & 0 & 0 & 0 & 0 \end{array} \quad = \quad \begin{array}{r} 5 \\ +11 \\ \hline 16 \end{array}$$

Yes, there is an overflow error

Binary Arithmetic

2. Perform the following 4-bit unsigned number addition. Is there an overflow error?

$$\begin{array}{rcccc} & 0 & 1 & 0 & 1 & = \\ + & 0 & 1 & 1 & 0 & = \\ \hline & & & & & = \end{array}$$

Binary Arithmetic

2. Perform the following 4-bit unsigned number addition. Is there an overflow error?

$$\begin{array}{rcccc} & 0 & 1 & 0 & 1 & = & & 5 \\ + & 0 & 1 & 1 & 0 & = & + & 6 \\ \hline & 1 & 0 & 1 & 1 & = & & 11 \end{array}$$

No, there is no overflow error

Binary Arithmetic

3. Perform the following 4-bit signed number addition. Is there an overflow error?

$$\begin{array}{rcccc} & 0 & 1 & 0 & 1 & = \\ + & 1 & 0 & 1 & 1 & = \\ \hline & & & & & = \end{array}$$

Binary Arithmetic

3. Perform the following 4-bit signed number addition. Is there an overflow error?

$$\begin{array}{rcccc} & 0 & 1 & 0 & 1 \\ + & 1 & 0 & 1 & 1 \\ \hline 1 & 0 & 0 & 0 & 0 \end{array} \quad \begin{array}{l} = \\ = \\ = \end{array} \quad \begin{array}{l} 5 \\ \underline{+(-5)} \\ 0 \end{array}$$

No, there is no overflow error

Binary Arithmetic

4. Perform the following 4-bit signed number addition. Is there an overflow error?

$$\begin{array}{rcccc} & 0 & 1 & 0 & 1 & = \\ + & 0 & 1 & 1 & 0 & = \\ \hline & & & & & = \end{array}$$

Binary Arithmetic

4. Perform the following 4-bit signed number addition. Is there an overflow error?

$$\begin{array}{rcccc} & 0 & 1 & 0 & 1 & = & & 5 \\ + & 0 & 1 & 1 & 0 & = & + & 6 \\ \hline & 1 & 0 & 1 & 1 & = & & -5 \end{array}$$

Yes, there is an overflow error

Binary Arithmetic

5. Perform the following 4-bit unsigned number subtraction. Is there an overflow error?

$$\begin{array}{r} 0101 \\ - 0110 \\ \hline \end{array} \quad \begin{array}{l} = \\ = \\ = \end{array}$$

Binary Arithmetic

5. Perform the following 4-bit unsigned number subtraction. Is there an overflow error?

$$\begin{array}{r} 0101 \\ - 0110 \\ \hline 1111 \end{array} = \begin{array}{r} 5 \\ - 6 \\ \hline \end{array}$$

Binary Arithmetic

5. Perform the following 4-bit unsigned number subtraction. Is there an overflow error?

$$\begin{array}{r} 0101 \\ - 0110 \\ \hline 1111 \end{array} = \begin{array}{r} 5 \\ - 6 \\ \hline 15 \end{array}$$

Yes, there is an overflow error

Binary Arithmetic

6. Perform the following 4-bit signed number subtraction. Is there an overflow error?

$$\begin{array}{r} 0 \ 1 \ 0 \ 1 \\ - 0 \ 1 \ 1 \ 0 \\ \hline \end{array} \begin{array}{l} = \\ = \\ = \end{array}$$

Binary Arithmetic

6. Perform the following 4-bit signed number subtraction. Is there an overflow error?

$$\begin{array}{r} 0 \ 1 \ 0 \ 1 \\ - 0 \ 1 \ 1 \ 0 \\ \hline 1 \ 1 \ 1 \ 1 \end{array} \quad \begin{array}{l} = \\ = \\ = \end{array} \quad \begin{array}{r} 5 \\ - 6 \\ \hline \end{array}$$

Binary Arithmetic

6. Perform the following 4-bit signed number subtraction. Is there an overflow error?

$$\begin{array}{r} 0 \ 1 \ 0 \ 1 \\ - 0 \ 1 \ 1 \ 0 \\ \hline 1 \ 1 \ 1 \ 1 \end{array} \quad = \quad \begin{array}{r} 5 \\ - 6 \\ \hline -1 \end{array}$$

No, there is no overflow error

Binary Arithmetic

7. Perform the following 4-bit signed number subtraction. Is there an overflow error?

$$\begin{array}{r} 0101 \\ - 1000 \\ \hline \end{array} \begin{array}{l} = \\ = \\ = \end{array}$$

Binary Arithmetic

7. Perform the following 4-bit signed number subtraction. Is there an overflow error?

$$\begin{array}{r} 0101 \\ - 1000 \\ \hline 1101 \end{array} \quad \begin{array}{l} = \\ = \\ = \end{array} \quad \begin{array}{l} 5 \\ -(-8) \\ \end{array}$$

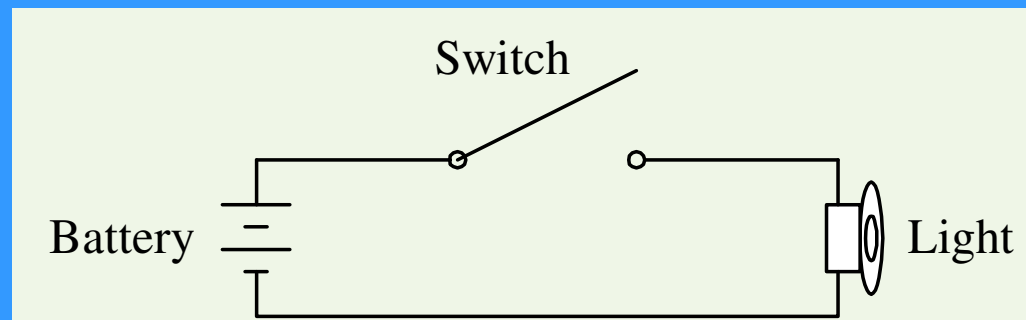
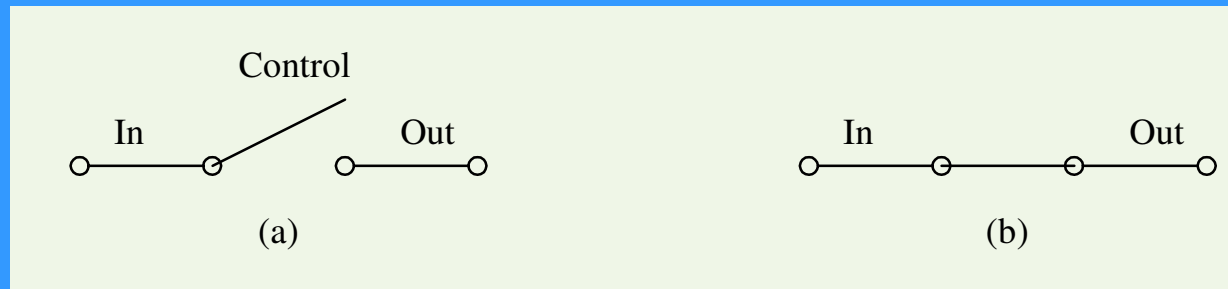
Binary Arithmetic

7. Perform the following 4-bit signed number subtraction. Is there an overflow error?

$$\begin{array}{rcl} 0 & 1 & 0 & 1 & = & 5 \\ - & 1 & 0 & 0 & 0 & = & -(-8) \\ \hline 1 & 1 & 0 & 1 & = & -3 \end{array}$$

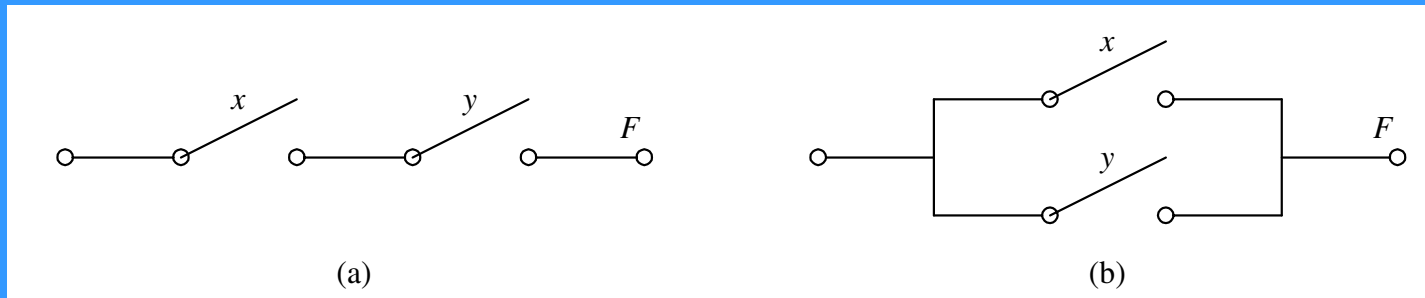
Yes, there is an overflow error

Binary Switch



$$L=X$$

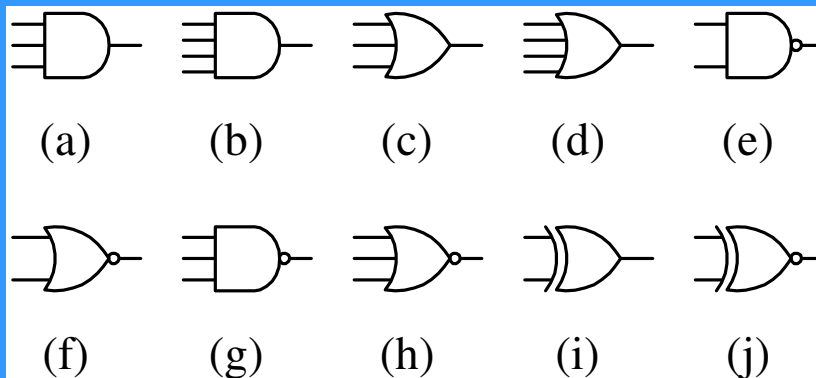
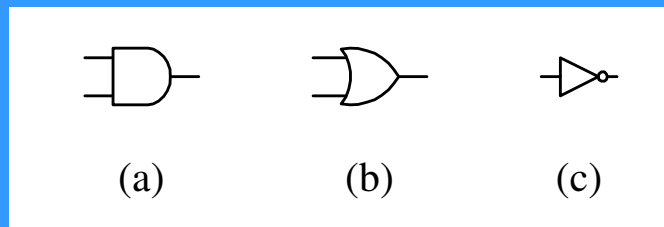
Basic Logic



- AND $F = x \text{ and } y$ $F = x \bullet y$ $F = xy$
- OR $F = x \text{ or } y$ $F = x + y$
- NOT $F = x'$ $F \equiv x$
- Precedence from high to low: NOT/AND/OR
- $F = xy + z'$ $F = x(y + z)'$

Logic Gate

- **Logic gates** are the actual physical devices that implement the logical operators
- Using Logic Symbol to denote logic gates



- NAND
- NOR
- XOR
- XNOR

Logic Gate

- NAND $F = (xy)'$
- NOR $F = (x + y)'$
- XOR $F = x \oplus y = x'y + xy'$
- XNOR $F = x \odot y = x'y' + xy$

For even number of inputs $xor = xnor'$

$$(x \oplus y) = (x \odot y)'$$

For odd number of inputs $xor = xnor$

$$(x \oplus y \oplus z) = (x \odot y \odot z)$$

Truth Table

x	y	F
0	0	0
0	1	0
1	0	0
1	1	1

x	y	F
0	0	0
0	1	1
1	0	1
1	1	1

x	F
0	1
1	0

		2-NAND	2-NOR	2-XOR	2-XNOR
x	y	$(x \bullet y)'$	$(x + y)'$	$x \oplus y$	$x \odot y$
0	0	1	1	0	1
0	1	1	0	1	0
1	0	1	0	1	0
1	1	0	0	0	1

Truth Table

			3-AND	3-OR	3-NAND	3-NOR	3-XOR	3-XNOR
x	y	z	$x \cdot y \cdot z$	$x + y + z$	$(x \cdot y \cdot z)'$	$(x + y + z)'$	$x \oplus y \oplus z$	$x \odot y \odot z$
0	0	0	0	0	1	1	0	0
0	0	1	0	1	1	0	1	1
0	1	0	0	1	1	0	1	1
0	1	1	0	1	1	0	0	0
1	0	0	0	1	1	0	1	1
1	0	1	0	1	1	0	0	0
1	1	0	0	1	1	0	0	0
1	1	1	1	1	0	0	1	1

NAND Gate

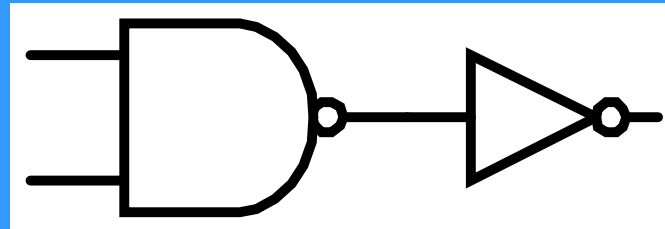
- How can you use a NAND gate to work like an AND gate?

x	y	F
0	0	1
0	1	1
1	0	1
1	1	0

NAND Gate

- How can you use a NAND gate to work like an AND gate?

x	y	F
0	0	1
0	1	1
1	0	1
1	1	0



NAND Gate

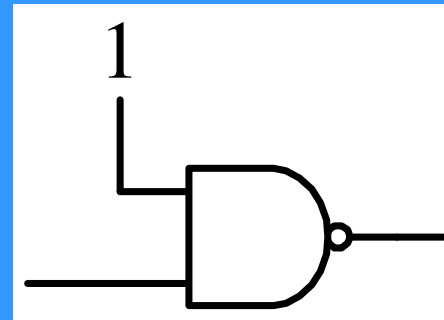
- How can you use a NAND gate to work like a NOT gate?

x	y	F
0	0	1
0	1	1
1	0	1
1	1	0

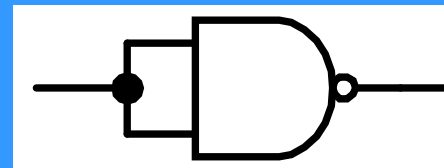
NAND Gate

- How can you use a NAND gate to work like a NOT gate?

x	y	F
0	0	1
0	1	1
1	0	1
1	1	0



or



NAND Gate

- How can you use a NAND gate to work like an OR gate?

x	y	F
0	0	1
0	1	1
1	0	1
1	1	0

NAND Gate

- How can you use a NAND gate to work like an OR gate?

Use DeMorgan's Theorem

x	y	F
0	0	1
0	1	1
1	0	1
1	1	0

NAND Gate

- How can you use a NAND gate to work like an OR gate?

Use DeMorgan's Theorem

x	y	F
0	0	1
0	1	1
1	0	1
1	1	0

$$\begin{aligned}x+y &= (x+y)'' \\ &= \end{aligned}$$

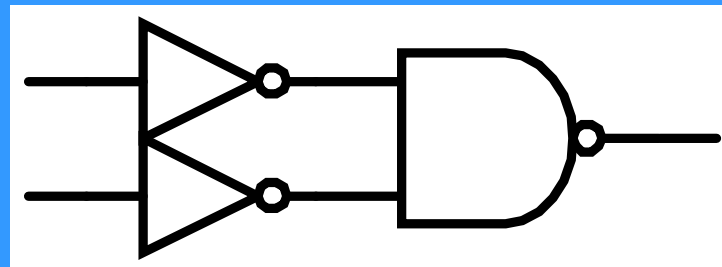
NAND Gate

- How can you use a NAND gate to work like an OR gate?

x	y	F
0	0	1
0	1	1
1	0	1
1	1	0

Use DeMorgan's Theorem

$$\begin{aligned}x + y &= (x + y)'' \\ &= (x'y')'\end{aligned}$$



XOR Gate

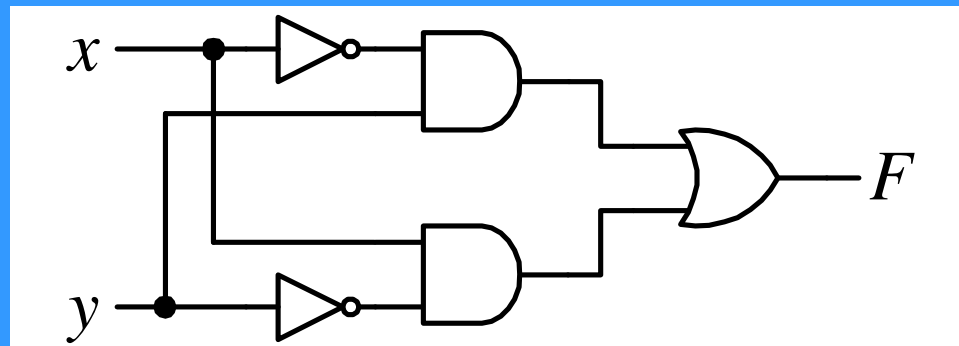
- Use AND, OR, and NOT gates to implement the XOR gate

x	y	F
0	0	0
0	1	1
1	0	1
1	1	0

XOR Gate

- Use AND, OR, and NOT gates to implement the XOR gate

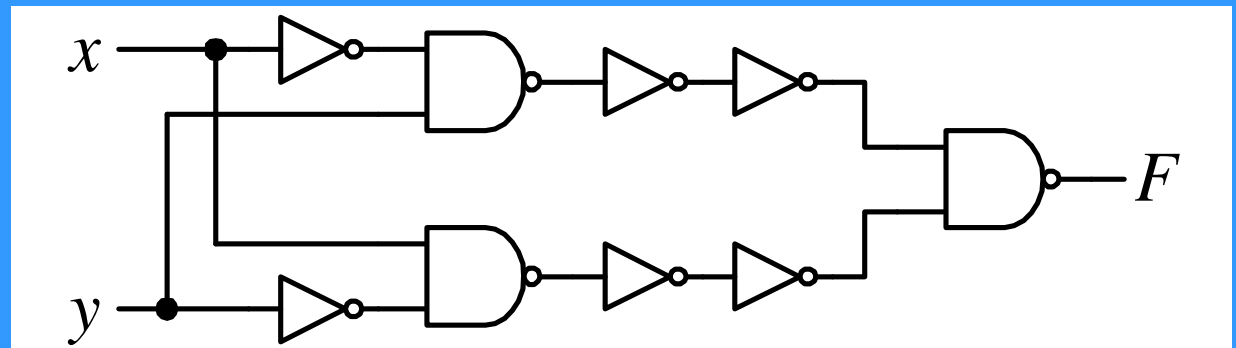
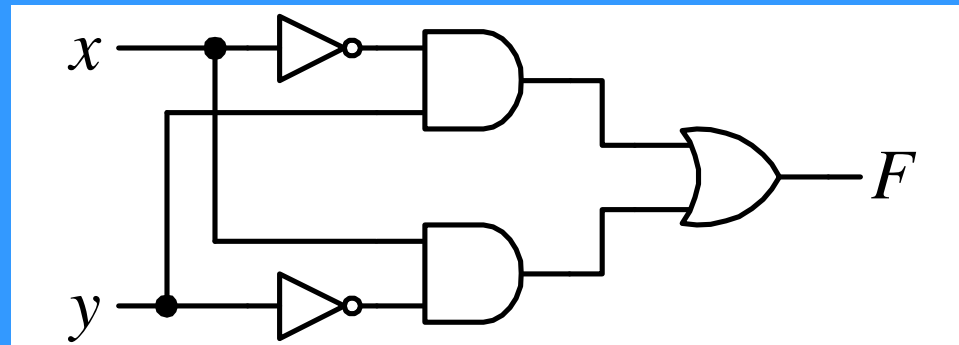
x	y	F
0	0	0
0	1	1
1	0	1
1	1	0



XOR Gate

- Use only NAND gates to implement the XOR gate

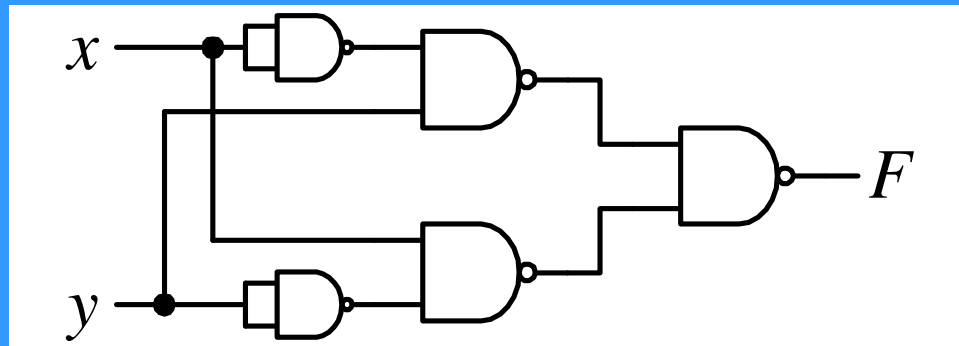
x	y	F
0	0	0
0	1	1
1	0	1
1	1	0



XOR Gate

- Use only NAND gates to implement the XOR gate

x	y	F
0	0	0
0	1	1
1	0	1
1	1	0



Boolean Algebra

- Circuits built with binary switches can be described using Boolean algebra.
- Let $B = \{0,1\}$ be the Boolean algebra. We have axioms, single variable theorems, and two or three variable theorems.
- Can be used to reduce circuit size.

Boolean Algebra

1a.	$0 \bullet 0 = 0$	1b.	$1 + 1 = 1$	
2a.	$1 \bullet 1 = 1$	2b.	$0 + 0 = 0$	
3a.	$0 \bullet 1 = 1 \bullet 0 = 0$	3b.	$1 + 0 = 0 + 1 = 1$	
4a.	$0' = 1$	4b.	$1' = 0$	
(a)				
5a.	$x \bullet 0 = 0$	5b.	$x + 1 = 1$	Null Element
6a.	$x \bullet 1 = 1 \bullet x = x$	6b.	$x + 0 = 0 + x = x$	Identity
7a.	$x \bullet x = x$	7b.	$x + x = x$	Idempotent
8a.	$(x')' = x$			Double Complement
9a.	$x \bullet x' = 0$	9b.	$x + x' = 1$	Inverse
(b)				
10a.	$x \bullet y = y \bullet x$	10b.	$x + y = y + x$	Commutative
11a.	$(x \bullet y) \bullet z = x \bullet (y \bullet z)$	11b.	$(x + y) + z = x + (y + z)$	Associative
12a.	$(x \bullet y) + (x \bullet z) = x \bullet (y + z)$	12b.	$(x + y) \bullet (x + z) = x + (y \bullet z)$	Distributive
13a.	$(x \bullet y)' = x' + y'$	13b.	$(x + y)' = x' \bullet y'$	DeMorgan's
(c)				

Reduce Logic Expression using Boolean Algebra

- Use Boolean algebra to reduce the expression $x + (x \bullet y)$ as much as possible

$$x + (x \bullet y) = (x \bullet 1) + (x \bullet y)$$

$$= x \bullet (1 + y)$$

$$= x \bullet (1)$$

$$= x$$

Boolean Algebra

- Use Boolean algebra to reduce the equation as much as possible

$$F = (x'yz) + (xy'z) + (xyz') + (xyz)$$

Boolean Algebra

- Use Boolean algebra to reduce the equation as much as possible

$$\begin{aligned} F &= (x'yz) + (xy'z) + (xyz') + (xyz) \\ &= (x'yz) + (xy'z) + (xyz') + (xyz) + (xyz) + (xyz) \end{aligned}$$

Boolean Algebra

- Use Boolean algebra to reduce the equation as much as possible

$$\begin{aligned} F &= (x'yz) + (xy'z) + (xyz') + (xyz) \\ &= (x'yz) + (xy'z) + (xyz') + (xyz) + (xyz) + (xyz) \\ &= (x'yz) + (xyz) + (xy'z) + (xyz) + (xyz') + (xyz) \\ &= [(x'yz) + (xyz)] + [(xy'z) + (xyz)] + [(xyz') + (xyz)] \end{aligned}$$

Boolean Algebra

- Use Boolean algebra to reduce the equation as much as possible

$$\begin{aligned} F &= (x'yz) + (xy'z) + (xyz') + (xyz) \\ &= (x'yz) + (xy'z) + (xyz') + (xyz) + (xyz) + (xyz) \\ &= (x'yz) + (xyz) + (xy'z) + (xyz) + (xyz') + (xyz) \\ &= [(x'yz) + (xyz)] + [(xy'z) + (xyz)] + [(xyz') + (xyz)] \\ &= yz(x' + x) + xz(y' + y) + xy(z' + z) \\ &= yz(1) + xz(1) + xy(1) \\ &= yz + xz + xy \\ &= z(y + x) + xy \end{aligned}$$

Duality Principle

- Dual: changing AND with OR and vice versa,
Changing 0 with 1 and vice versa

$$(x \bullet y' \bullet z) + (x \bullet y \bullet z') + (y \bullet z) + 0$$

$$(x + y' + z) \bullet (x + y + z') \bullet (y + z) \bullet 1$$

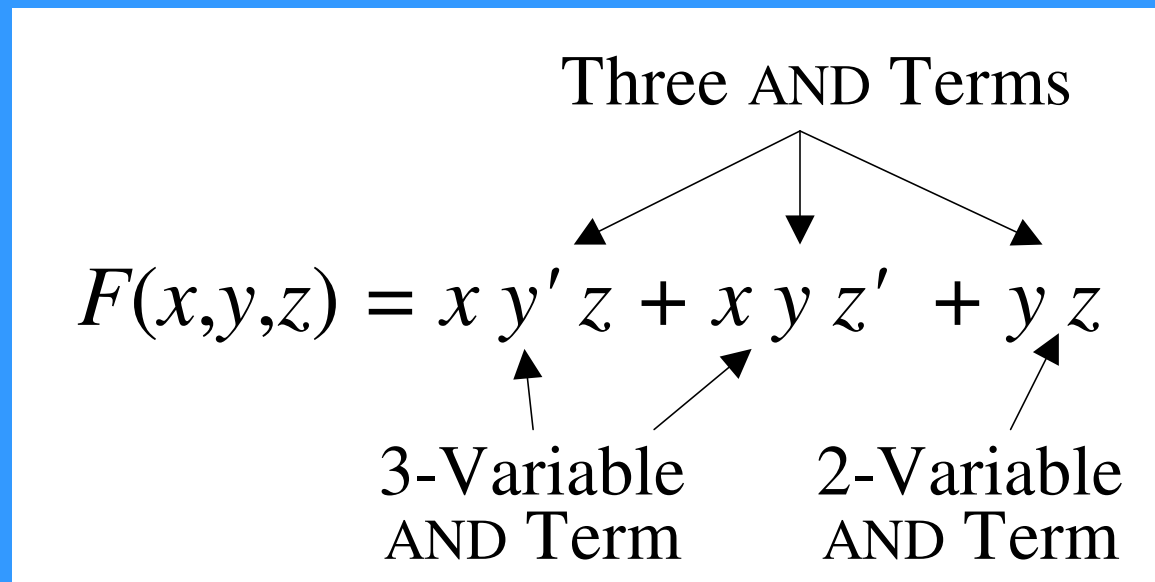
- Duality Principle: if a Boolean expression is true,
then its dual is also true

$$x + 1 = 1 \text{ is true. } x \bullet 0 = 0 \text{ is true}$$

- The inverse of a Boolean expression can be
obtained by taking the dual of that expression and
then complementing each variable.

Boolean Function and Inverse

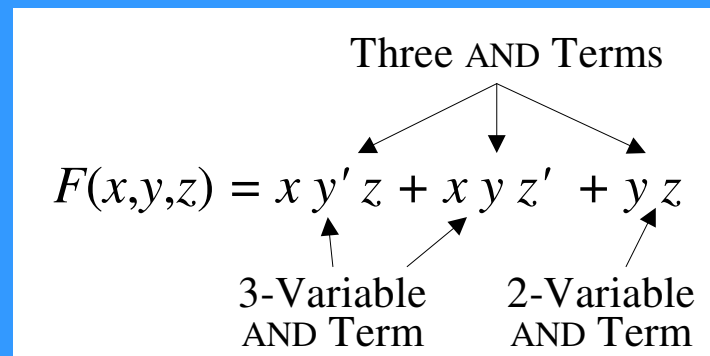
- Boolean function: logic expression to describe digital circuit.



Sum-of-product

Boolean Function and Inverse

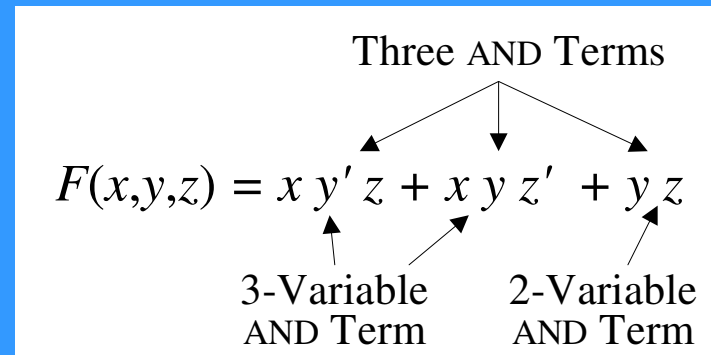
- We are mainly interested in when a function evaluates to a 1



- $F = 1$ when any one of the three AND terms evaluate to a 1
- The first AND term, $xy'z$, equals 1 if

$$x = 1, y = 0, \text{ and } z = 1$$

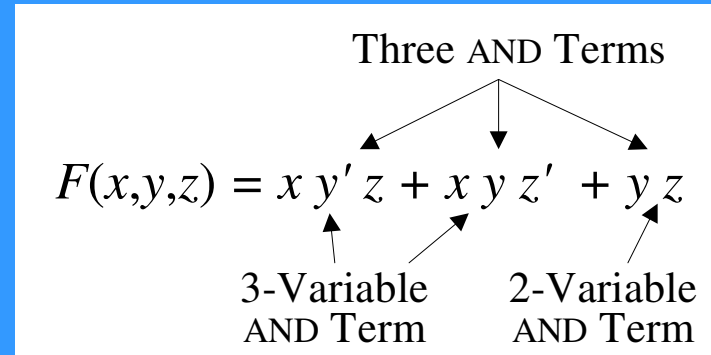
Boolean Function and Inverse



- The last AND term, yz , equals 1 if $y = 1$ and $z = 1$

the missing variable, x , means it doesn't matter what its value is, so it can be either 0 or 1

Boolean Function and Inverse



- Putting everything together, $F = 1$ when

$$x = 1, y = 0, \text{ and } z = 1$$

or $x = 1, y = 1, \text{ and } z = 0$

or $x = 0, y = 1, \text{ and } z = 1$

or $x = 1, y = 1, \text{ and } z = 1$

Boolean Function and Inverse

- It is more convenient to summarize this verbal description of a function with a truth table

x	y	z	F	F'
0	0	0	0	1
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	0

Boolean Function and Inverse

- The inverse of a function, F' , can be obtained easily from the truth table

x	y	z	F	F'
0	0	0	0	1
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	0

- Look at the rows where $F' = 1$
- $F' = (x'y'z') + (x'y'z) + (x'yz') + (xy'z')$

Boolean Function and Inverse

- To get F' using Boolean Algebra requires using DeMorgan's Theorem twice

$$F = xy'z + xyz' + yz$$

$$\begin{aligned} F' &= (xy'z + xyz' + yz)' \\ &= (xy'z)' \cdot (xyz')' \cdot (yz)' \\ &= (x' + y + z') \cdot (x' + y' + z) \cdot (y' + z') \end{aligned}$$

Boolean Function and Inverse

- We have two different equations for F'

$$F' = (x'y'z') + (x'y'z) + (x'yz') + (xy'z')$$

sum-of-products

$$F' = (x'+y+z') \cdot (x'+y'+z) \cdot (y'+z')$$

product-of-sums

Minterms and Maxterms

- Minterm
 - Is a product term that contains all the variables in a function
 - The variable is negated (primed) if the value is a 0
- Maxterm
 - Is a sum term that contains all the variables in a function
 - The variable is negated (primed) if the value is a 1

Minterms and Maxterms

- m_i for minterms
- M_i for maxterms where $0 \leq i < 2^n$ for n variables

x	y	z	Minterm	Notation		Maxterm	Notation
0	0	0	$x' y' z'$	m_0		$x + y + z$	M_0
0	0	1	$x' y' z$	m_1		$x + y + z'$	M_1
0	1	0	$x' y z'$	m_2		$x + y' + z$	M_2
0	1	1	$x' y z$	m_3		$x + y' + z'$	M_3
1	0	0	$x y' z'$	m_4		$x' + y + z$	M_4
1	0	1	$x y' z$	m_5		$x' + y + z'$	M_5
1	1	0	$x y z'$	m_6		$x' + y' + z$	M_6
1	1	1	$x y z$	m_7		$x' + y' + z'$	M_7

Minterm/Maxterm Example

$$\begin{aligned} F &= xy'z + xyz' + yz \\ &= x'yz + xy'z + xyz' + xyz \end{aligned}$$

x	y	z	F	F'	Minterm	Notation
0	0	0	0	1	$x' y' z'$	m_0
0	0	1	0	1	$x' y' z$	m_1
0	1	0	0	1	$x' y z'$	m_2
0	1	1	1	0	$x' y z$	m_3
1	0	0	0	1	$x y' z'$	m_4
1	0	1	1	0	$x y' z$	m_5
1	1	0	1	0	$x y z'$	m_6
1	1	1	1	0	$x y z$	m_7

$$F(x, y, z) = m_3 + m_5 + m_6 + m_7$$

$$F(x, y, z) = \Sigma(3, 5, 6, 7)$$

$$F'(x, y, z) = \Sigma(0, 1, 2, 4)$$

$$\begin{aligned} F &= xy'z + xyz' + yz \\ &= (x + y + z) \cdot (x + y + z') \cdot (x + y' + z) \cdot (x' + y + z) \end{aligned}$$

x	y	z	F	F'	Maxterm	Notation
0	0	0	0	1	$x + y + z$	M_0
0	0	1	0	1	$x + y + z'$	M_1
0	1	0	0	1	$x + y' + z$	M_2
0	1	1	1	0	$x + y' + z'$	M_3
1	0	0	0	1	$x' + y + z$	M_4
1	0	1	1	0	$x' + y + z'$	M_5
1	1	0	1	0	$x' + y' + z$	M_6
1	1	1	1	0	$x' + y' + z'$	M_7

$$F(x, y, z) = M_0 \cdot M_1 \cdot M_2 \cdot M_4$$

$$F(x, y, z) = \Pi(0, 1, 2, 4)$$

$$F'(x, y, z) = \Pi(3, 5, 6, 7)$$

Minterm/Maxterm Example

- Given $F(x, y, z) = \Sigma(1, 2, 3, 5, 6, 7)$
- Write out the full function

Minterm/Maxterm Example

- Given $F(x, y, z) = \Sigma(1, 2, 3, 5, 6, 7)$
- Write out the full function

$$F(x, y, z) = x'y'z + x'yz' + x'yz + x'yz' + xyz' + xyz$$

Minterm/Maxterm Example

- Given $F(x, y, z) = \Sigma(1, 2, 3, 5, 6, 7)$
- What is F using Π ?

Minterm/Maxterm Example

- Given $F(x, y, z) = \Sigma(1, 2, 3, 5, 6, 7)$
- What is F using Π ?
- $F(x, y, z) = \Pi(0, 4)$

Minterm/Maxterm Example

- Given $F(x, y, z) = \Sigma(1, 2, 3, 5, 6, 7)$
- What is F using Π ?
- $F(x, y, z) = \Pi(0, 4)$
- Write out the full function for $\Pi(0, 4)$

Minterm/Maxterm Example

- Given $F(x, y, z) = \Sigma(1, 2, 3, 5, 6, 7)$
- What is F using Π ?
- $F(x, y, z) = \Pi(0, 4)$
- Write out the full function for $\Pi(0, 4)$
- $F(x, y, z) = (x + y + z) \bullet (x' + y + z)$

Minterm/Maxterm Example

- Given $F(x, y, z) = \Sigma(1, 2, 3, 5, 6, 7)$
- What is F' using Σ ?
- $F' = \Sigma(0, 4)$

Minterm/Maxterm Example

- Given $F(x, y, z) = \Sigma(1, 2, 3, 5, 6, 7)$
- What is F' using Π ?
- $F' = \Pi(1, 2, 3, 5, 6, 7)$

Minterm/Maxterm Example

- Given $F(w, x, y, z) = \Sigma(1, 2, 3, 5, 6, 7)$
- Write out the full function

Minterm/Maxterm Example

- Given $F(w, x, y, z) = \Sigma(1, 2, 3, 5, 6, 7)$
- What is F using Π ?

Minterm/Maxterm Example

- Given $F(w, x, y, z) = \Sigma(1, 2, 3, 5, 6, 7)$
- What is F using Π ?
- Write out the full function for Π of 0-Maxterms

Minterm/Maxterm Example

- Given $F(w, x, y, z) = \Sigma(1, 2, 3, 5, 6, 7)$
- What is F' using Σ ?

Minterm/Maxterm Example

- Given $F(w, x, y, z) = \Sigma(1, 2, 3, 5, 6, 7)$
- What is F' using Π ?

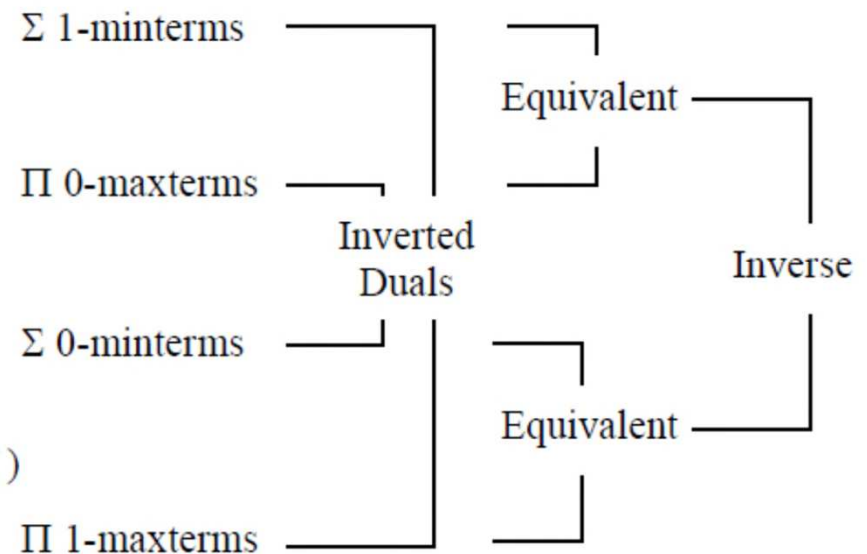
Minterms/Maxterms Relationship

$$\begin{aligned} F(x, y, z) &= x' y z + x y' z + x y z' + x y z \\ &= m_3 + m_5 + m_6 + m_7 \\ &= \Sigma(3, 5, 6, 7) \end{aligned}$$

$$\begin{aligned} &= (x+y+z) \bullet (x+y+z') \bullet (x+y'+z) \bullet (x'+y+z) \\ &= M_0 \bullet M_1 \bullet M_2 \bullet M_4 \\ &= \Pi(0, 1, 2, 4) \end{aligned}$$

$$\begin{aligned} F'(x, y, z) &= x' y' z' + x' y' z + x' y z' + x y' z' \\ &= m_0 + m_1 + m_2 + m_4 \\ &= \Sigma(0, 1, 2, 4) \end{aligned}$$

$$\begin{aligned} &= (x+y'+z') \bullet (x'+y+z') \bullet (x'+y'+z) \bullet (x'+y'+z') \\ &= M_3 \bullet M_5 \bullet M_6 \bullet M_7 \\ &= \Pi(3, 5, 6, 7) \end{aligned}$$



Converting to Minterms/Maxterms

- Given $F(x, y, z) = y + x'z$
- Write it in the Σ of minterms format and Π of maxterms format
- Use Truth Table

Converting to Minterms/Maxterms

- Given $F(x, y, z) = y + x'z$
- Use Truth Table

x	y	z	F	Minterm	Notation
0	0	0		$x' y' z'$	m_0
0	0	1		$x' y' z$	m_1
0	1	0		$x' y z'$	m_2
0	1	1		$x' y z$	m_3
1	0	0		$x y' z'$	m_4
1	0	1		$x y' z$	m_5
1	1	0		$x y z'$	m_6
1	1	1		$x y z$	m_7

Converting to Minterms/Maxterms

- Given $F(x, y, z) = y + x'z$
- Use Truth Table

x	y	z	F	Minterm	Notation
0	0	0	0	$x' y' z'$	m_0
0	0	1	1	$x' y' z$	m_1
0	1	0	1	$x' y z'$	m_2
0	1	1	1	$x' y z$	m_3
1	0	0	0	$x y' z'$	m_4
1	0	1	0	$x y' z$	m_5
1	1	0	1	$x y z'$	m_6
1	1	1	1	$x y z$	m_7

$$F = \Sigma(1, 2, 3, 6, 7) \text{ and } F = \Pi(0, 4, 5)$$

Converting to Minterms/Maxterms

- Given $F(x, y, z) = y + x'z$
- Write it in the Σ of minterms format
- Use Boolean algebra

$$F = y + x'z$$

Converting to Minterms/Maxterms

- Given $F(x, y, z) = y + x'z$
- Write it in the Σ of minterms format
- Use Boolean algebra

$$\begin{aligned} F &= y + x'z \\ &= y(x+x')(z+z') + x'z(y+y') \\ &= xyz + xyz' + x'yz + x'yz' + \cancel{x'yz} + x'y'z \\ &= m_7 + m_6 + m_3 + m_2 + m_1 \\ &= \Sigma(1, 2, 3, 6, 7) \end{aligned}$$

Converting to Minterms/Maxterms

- Given $F(x, y, z) = y + x'z$
- Write it in the Π of maxterms format
- Use Boolean algebra

$$F = y + x'z$$

Converting to Minterms/Maxterms

- Given $F(x, y, z) = y + x'z$
- Write it in the Π of maxterms format
- Use Boolean algebra

$$\begin{aligned} F &= y + x'z \\ &= (y+x')(y+z) \\ &= (y+x'+zz')(y+z+xx') \\ &= (x'+y+z)(x'+y+z')(x+y+z)(\cancel{x'+y+z}) \\ &= M_4 \cdot M_5 \cdot M_0 = \Pi(0, 4, 5) \end{aligned}$$

Converting to Minterms/Maxterms

- Given $F(x, y, z) = y + x'z$
- Write F' in the Σ of minterms format
- Use Boolean algebra

$$F' = (y + x'z)'$$

Converting to Minterms/Maxterms

- Given $F(x, y, z) = y + x'z$

$$\begin{aligned}F' &= (y + x'z)' \\&= y' \cdot (x'z)' \\&= y' \cdot (x+z') \\&= y'x + y'z' \\&= y'x(z+z') + y'z'(x+x') \\&= xy'z + xy'z' + \cancel{xy'z'} + x'y'z' \\&= m_5 + m_4 + m_0 \\&= \Sigma (0, 4, 5)\end{aligned}$$

© 2018 Cengage Learning®. May not be scanned, copied or duplicated, or posted to a publicly accessible website, in whole or in part.

Converting to Minterms/Maxterms

- Given $F(x, y, z) = y + x'z$
- Write F' in the Π of maxterms format
- Use Boolean algebra

$$F' = (y + x'z)'$$

Converting to Minterms/Maxterms

- Given $F(x, y, z) = y + x'z$

$$\begin{aligned}F' &= (y + x'z)' \\&= y' \cdot (x'z)' \\&= y' \cdot (x+z') \\&= (y' + xx' + zz') \cdot (x+z' + yy') \\&= (x+y'+z)(x+y'+z')(x'+y'+z)(x'+y'+z') \\&\quad (x+y+z') \cancel{(x+y'+z')} \\&= M_2 \cdot M_3 \cdot M_6 \cdot M_7 \cdot M_1 \\&= \Pi(1, 2, 3, 6, 7)\end{aligned}$$

© 2018 Cengage Learning®. May not be scanned, copied or duplicated, or posted to a publicly accessible website, in whole or in part.

Converting to Minterms/Maxterms

- Given $F(x, y, z) = y + x'z$

$$F = \Sigma(1, 2, 3, 6, 7) = \Pi(0, 4, 5)$$

$$F' = \Sigma(0, 4, 5) = \Pi(1, 2, 3, 6, 7)$$

Canonical, Standard, and Non-standard Forms

- Canonical: Boolean function expressed in sum-of-minterms or product-of-maxterms
- Standard: sum-of-products or products-of-sum has at least one minterm/maxterm
- Non-standard: not in sum-of-product or product-of-sum format

$$F = x' y z + x y' z + x y z' + x y z$$

$$F' = (x+y'+z') \cdot (x'+y+z') \cdot (x'+y'+z) \cdot (x'+y'+z')$$

$$F_1(x, y, z) = \Sigma(0, 1, 2, 3, 4, 5) \quad F_2(x, y, z) = \Pi(6, 7)$$

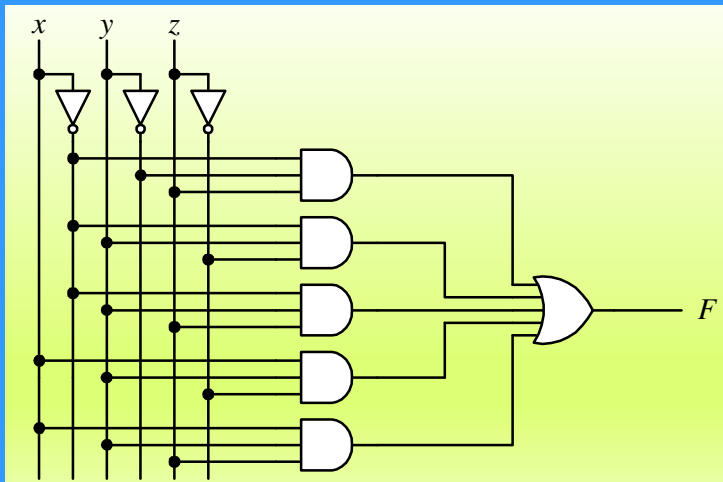
$$F_1(x, y, z) = \Sigma(3, 5, 6) \quad F_2(x, y, z) = \Pi(3, 5, 6)$$

$$F = xy'z + xyz' + yz$$

$$F = x(y'z + yz') + yz$$

Digital Circuit

- Digital circuit is a connection of two or more logic gates
- Digital network can be described using schematic diagrams, Boolean expressions, or truth tables



x	y	z	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

$$F(x, y, z) = x'y'z + x'yz' + x'yz + xyz' + xyz$$

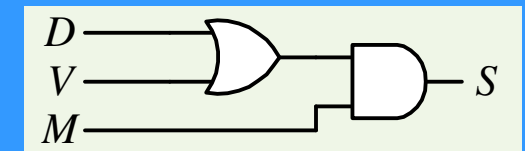
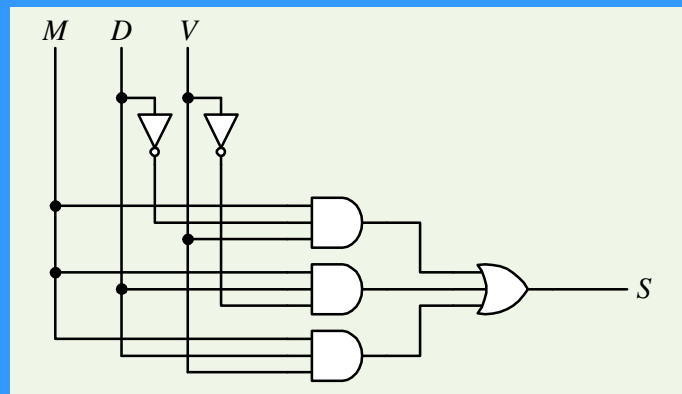
© 2018 Cengage Learning®. May not be scanned, copied or duplicated, or posted to a publicly accessible website, in whole or in part.

Design a Car Security System

- Input: D = Door switch
V = Vibration sensor
M = Motion sensor
Output: S = Siren

$$\begin{aligned} S &= (M D' V) + (M D V') + (M D V) \\ &= M (D' V + D V' + D V) \\ &= M (D' V + D V' + D V + D V) \\ &= M (D(V' + V) + V(D' + D)) \\ &= M (D(1) + V(1)) \\ &= M (D + V) \end{aligned}$$

M	D	V	S
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



Verilog Code for Car System

```
// this is a Verilog dataflow model of the car security system
module Siren (
    input M,
    input D,
    input V,
    output S
);

    wire term1, term2, term3;

    assign term1 = (M & !D & V);
    assign term2 = (M & D & !V);
    assign term3 = (M & D & V);
    assign S = term1 | term2 | term3;

endmodule
```

Verilog Code for Car System

```
// this is a Verilog dataflow model of the car security system
module Siren (
    input M,
    input D,
    input V,
    output S
);

    assign S = (M & !D & V) | (M & D & !V) | (M & D & V);

endmodule
```

Verilog Code for Car System

```
// this is a Verilog dataflow model of the car security system
module Siren (
    input M,
    input D,
    input V,
    output S
);

    assign S = M & (D | V);

endmodule
```

Verilog Code for Car System

```
// this is a Verilog structural model of the car security system
module Siren (
    input M,
    input D,
    input V,
    output S
);

    wire w1;

    or (w1, D, V);
    and (S, M, w1);

endmodule
```

VHDL Code for Car System

```
// this is a VHDL dataflow model of the car security system
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY Siren IS PORT (
    M:      IN STD_LOGIC;
    D:      IN STD_LOGIC;
    V:      IN STD_LOGIC;
    S:      OUT STD_LOGIC);
END Siren;
ARCHITECTURE Dataflow OF Siren IS
    SIGNAL term_1, term_2, term_3: STD_LOGIC;
BEGIN
    term_1 <= M AND (NOT D) AND V;
    term_2 <= M AND D AND (NOT V);
    term_3 <= M AND D AND V;
    S <= term_1 OR term_2 OR term_3;
END Dataflow;
```

VHDL Code for Car System

```
// this is a VHDL dataflow model of the car security system
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY Siren IS PORT (
    M:      IN STD_LOGIC;
    D:      IN STD_LOGIC;
    V:      IN STD_LOGIC;
    S:      OUT STD_LOGIC);
END Siren;
ARCHITECTURE Dataflow OF Siren IS
BEGIN
    S <= M AND (D OR V);
END Dataflow;
```