Solution:

Question 1:

Part a: Express the following decimal numbers as a sum of the values of each digit:

- **568.23**

  The whole number digit 5 has a weight of 100, which is 10^2, the digit 6 has a weight of 10, which is 10^1, the digit 8 has a weight of 1, which is 10^0, the fractional digit 2 has a weight of 0.1, which is 10^{-1}, and the fractional digit 3 has a weight of 0.01, which is 10^{-2}.

$$\begin{aligned} 568.23 &= (5 \times 10^2) + (6 \times 10^1) + (8 \times 10^0) + (2 \times 10^{-1}) + (3 \times 10^{-2}) \\ &= (5 \times 100) + (6 \times 10) + (8 \times 1) + (2 \times 0.1) + (3 \times 0.01) \\ &= \mathbf{500} + \mathbf{60} + \mathbf{8} + \mathbf{0.2} + \mathbf{0.03} \end{aligned}$$

- **70**

  The digit 4 has a weight of 10, which is 10^1, as indicated by its position. The digit 7 has a weight of 1, which is 10^0, as indicated by its position.

$$\begin{aligned} 47 &= (4 \times 10^1) + (7 \times 10^0) \\ &= (4 \times 10) + (7 \times 1) = \mathbf{40} + \mathbf{7} \end{aligned}$$

- **67.924:** Same as above

Part b) Perform the following base conversions (show proper working):

231.34_4 to base 7

$$231.3_4 = 2 \times 16 + 3 \times 4 + 1 + \tfrac{3}{4} = 45.75_{10}$$

$$\begin{array}{lll} 7\,\big/\,45 & & .75 \\ 7\,\big/\,6 & \text{rem. } 3 & \underline{\quad 7} \\ \quad 0 & \text{rem. } 6 & (5).25 \\ & & \underline{\quad 7} \\ & & (1).75 \\ & & \underline{\quad 7} \\ & & (5).25 \\ & & \underline{\quad 7} \\ & & (1).75 \end{array} \qquad 45.75_{10} = 63.5151\ldots_7$$

a) 1001101.0101112 b) 1100101001010111 c) 1111110001011010001 to base 16

$$1001101.010111_2 = \underbrace{0100}_{4}\ \underbrace{1101}_{D}\ .\ \underbrace{0101}_{5}\ \underbrace{1100}_{C} = 4D.5C_{16}$$

$$\underbrace{1100}_{C}\underbrace{1010}_{A}\underbrace{0101}_{5}\underbrace{0111}_{7} = \mathbf{CA57_{16}} \qquad \underbrace{0011}_{3}\underbrace{1111}_{F}\underbrace{0001}_{1}\underbrace{0110}_{6}\underbrace{1001}_{9} = \mathbf{3F169_{16}}$$

Two zeros have been added in part (c) to complete a 4-bit group at the left.

3210_5 to base 3

First find, $3210_5 = 430_{10}$

Then find, $430_{10} = 120221_3$

212_3 to base 5

First, $212_3 = 23_{10}$

Then find, $23_{10} = 43_5$

$123D_{16}$ to base 10

4669_{10}

Part c: Convert to hexadecimal and then to binary:
$(a)\ 757.25_{10}\ (b)\ 123.17_{10}\ (c)\ 356.89_{10}\ (d)\ 1063.5_{10}$

(a) 757.25_{10}

$16 \mid 757$			0.25
$16 \mid 47$	r5		$\times 16$
$16 \mid 2$	$\text{r}15 = \text{F}_{16}$		$(4).00$
0	r2		

$\therefore 757.25_{10} = 2F5.40_{16}$
$= \underset{2}{\underline{0010}}\ \underset{F}{\underline{1111}}\ \underset{5}{\underline{0101}}.\underset{4}{\underline{0100}}\ \underset{0}{\underline{0000}}{}_2$

(b) 123.17_{10}

$16 \mid 123$			0.17
$16 \mid 7$	r11		$\times 16$
0	r7		$(2).72$
			$\times 16$
			$(11).52$
			$\times 16$
			$(8).32$

$\therefore 123.17_{10} = 7B.2B_{16}$
$= \underset{7}{\underline{0111}}\ \underset{B}{\underline{1011}}.\underset{2}{\underline{0010}}\ \underset{B}{\underline{1011}}{}_2$

(c) 356.89_{10}

$16 \mid 356$			0.89
$16 \mid 22$	r4		$\times 16$
$16 \mid 1$	r6		$(14).24$
0	r1		$\times 16$
			$(3).84$
			$\times 16$
			$(13).44$
			$\times 16$
			$(7).04$

$\therefore 356.89_{10} = 164.E3_{16}$
$= \underset{1}{\underline{0001}}\ \underset{6}{\underline{0110}}\ \underset{4}{\underline{0100}}.\underset{E}{\underline{1110}}\ \underset{3}{\underline{0011}}{}_2$

(d) 1063.5_{10}

$16 \mid 1063$			0.5
$16 \mid 66$	r7		$\times 16$
$16 \mid 4$	r2		$(8).00$
0	r4		

$\therefore 1063.5_{10} = 427.8_{16}$
$= \underset{4}{\underline{0100}}\ \underset{2}{\underline{0010}}\ \underset{7}{\underline{0111}}.\underset{8}{\underline{1000}}{}_2$

Part d: Convert to base 6: 3BA.2514 (do all of the arithmetic in decimal)

$$3BA.25_{14} = 3 \times 14^2 + 11 \times 14^1 + 10 \times 14^0 + 2 \times 14^{-1} + 5 \times 14^{-2}$$
$$= 588 + 154 + 10 + 0.1684 = 752.1684_{10}$$

$$\begin{array}{r|l l}
6 & \underline{752} & \\
6 & \underline{125} & r2 \\
6 & \underline{20} & r5 \\
6 & \underline{3} & r2 \\
 & 0 & r3
\end{array}
\qquad
\begin{array}{r}
0.1684 \\
\underline{\quad 6} \\
(1).0104 \\
\underline{\quad 6} \\
(0).0624 \\
\underline{\quad 6} \\
(0).3744 \\
\underline{\quad 6} \\
(2).2464 \\
\underline{\quad 6} \\
(1).4784
\end{array}$$

$\therefore\ 3BA.25_{14} = 752.1684_{10} = 3252.1002_6$

Part e) Add −11 and −20 in 1’s complement (show proper working)

$+11 = 00001011 \qquad +20 = 00010100$

taking the bit-by-bit complement,

-11 is represented by 11110100 and -20 by 11101011

$$\begin{array}{rl}
11110100 & (-11) \\
\underline{11101011} & \underline{+(-20)} \\
(1)\ 11011111 & \\
\longrightarrow 1 & \text{(end-around carry)} \\
\hline
11100000 = -31 &
\end{array}$$

Part f: Add −8 and +19 in 2’s complement

$+8 = 00001000$

complementing all bits to the left of the first 1, −8, is represented by 11111000

$$
\begin{array}{ll}
11111000 & (-8) \\
\underline{00010011} & \underline{+19} \\
\cancel{(1)}00001011 = & +11
\end{array}
$$

(discard last carry)

Note that in both cases, the addition produced a carry out of the furthest left bit position, but there is no overflow because the answer can be correctly represented by eight bits (including sign). A general rule for detecting overflow when adding two n-bit signed binary numbers (1's or 2's complement) to get an n-bit sum is:

> An overflow occurs if adding two positive numbers gives a negative answer or if adding two negative numbers gives a positive answer.

An alternative method for detecting overflow in 2's complement addition is as follows:

> An overflow occurs if and only if the carry out of the sign position is not equal to the carry into the sign position.

Part g: Express the decimal number -39 as an 8-bit number in the sign-magnitude, 1's complement, and 2's complement forms

First, write the 8-bit number for +39.

00100111

In the *sign-magnitude form,* −39 is produced by changing the sign bit to a 1 and leaving the magnitude bits as they are. The number is

10100111

In the *1's complement form,* −39 is produced by taking the 1's complement of +39 (00100111).

11011000

In the *2's complement form,* −39 is produced by taking the 2's complement of +39 (00100111) as follows:

$$
\begin{array}{rl}
11011000 & \text{1's complement} \\
\underline{+ \qquad\quad 1} & \\
\mathbf{11011001} & \text{2's complement}
\end{array}
$$

Part h: Determine the decimal value of this signed binary number expressed in sign-magnitude: 10010101.

The seven magnitude bits and their powers-of-two weights are as follows:

2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	1	0	1	0	1

Summing the weights where there are 1s,

$$16 + 4 + 1 = 21$$

The sign bit is 1; therefore, the decimal number is **−21**.

Part i: Subtract in binary. Place a 1 over each column from which it was necessary to borrow.

The subtraction table for binary numbers is

$0 - 0 = 0$
$0 - 1 = 1$ and borrow 1 from the next column
$1 - 0 = 1$
$1 - 1 = 0$

Borrowing 1 from a column is equivalent to subtracting 1 from that column.

(a)
```
     1←— (indicates
 11101    a borrrow
-10011    from the
  1010    3rd column)
```

(b)
```
 1111←— borrows
 10000
-   11
  1101
```

(c)
```
  111←— borrows
 111001
-  1011
 101110
```

Part j: Add the signed numbers: 01000100, 00011011, 00001110, and 00010010 (show proper working)

The equivalent decimal additions are given for reference.

68	01000100	
+ 27	+ 00011011	Add 1st two numbers
95	01011111	1st sum
+ 14	+ 00001110	Add 3rd number
109	01101101	2nd sum
+ 18	+ 00010010	Add 4th number
127	**01111111**	Final sum

Part k: Construct a table for 4-3-2-1 weighted code and write 9154 using this code. Is it possible to construct a 5-3-1-1 weighted code? A 6-4-1-1 weighted code? Justify your answers.

	4 3 2 1
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 1 0 0
4	1 0 0 0
5	1 0 0 1
6	1 0 1 0
7	1 1 0 0
8	1 1 0 1
9	1 1 1 0

9154 = 1110 0001 1001 1000

5-3-1-1 is possible, but 6-4-1-1 is not, because there is no way to represent 3 or 9.

	5 3 1 1	Alternate Solutions:
0	0 0 0 0	
1	0 0 0 1	(0010)
2	0 0 1 1	
3	0 1 0 0	
4	0 1 0 1	(0110)
5	1 0 0 0	
6	1 0 0 1	(1010)
7	1 0 1 1	
8	1 1 0 0	
9	1 1 0 1	(1110)

Part l: Add the following BCD numbers: (a) 1001 + 0100 (b) 1001 + 1001 (c) 00010110 + 00010101 (d) 01100111 + 01010011 (Show proper workings step by step). When are BCD numbers invalid?

The decimal number additions are shown for comparison.

(a)

	1001		9
	+ 0100		+4
	1101	Invalid BCD number (>9)	13
	+ 0110	Add 6	
0001	**0011**	Valid BCD number	
↓	↓		
1	3		

(b)

	1001		9
	+ 1001		+ 9
1	0010	Invalid because of carry	18
	+ 0110	Add 6	
0001	**1000**	Valid BCD number	
↓	↓		
1	8		

(c)

0001	0110		16
+ 0001	0101		+ 15
0010	1011	Right group is invalid (>9), left group is valid.	31
	+ 0110	Add 6 to invalid code. Add carry, 0001, to next group.	
0011	**0001**	Valid BCD number	
↓	↓		
3	1		

(d)

	0110	0111		67
	+ 0101	0011		+ 53
	1011	1010	Both groups are invalid (>9)	120
	+ 0110	+ 0110	Add 6 to both groups	
0001	**0010**	**0000**	Valid BCD number	
↓	↓	↓		
1	2	0		

Part m: Convert the binary number 11000110 to Gray code. (b) Convert the Gray code 10101111 to binary.

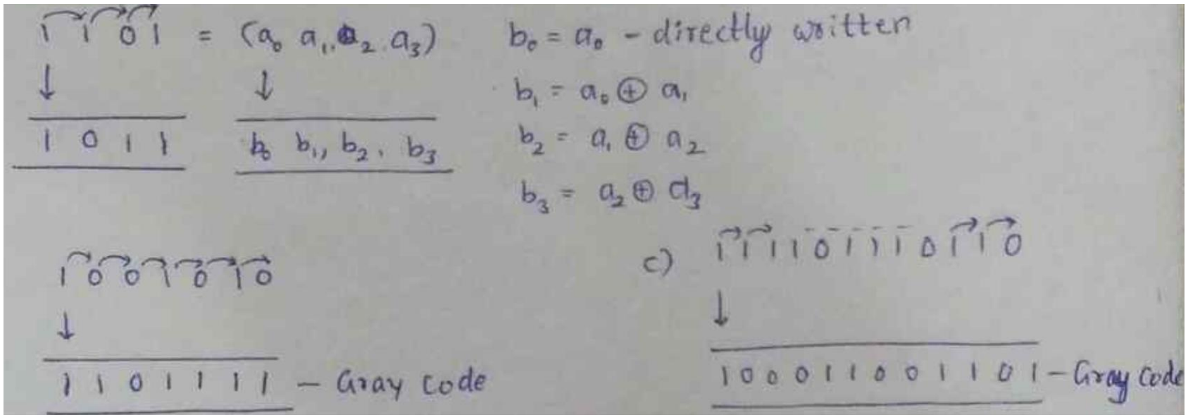
(a) Binary to Gray code:

$$\begin{array}{ccccccccccccccc} 1 & -+\rightarrow & 1 & -+\rightarrow & 0 & -+\rightarrow & 0 & -+\rightarrow & 0 & -+\rightarrow & 1 & -+\rightarrow & 1 & -+\rightarrow & 0 \\ \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow \\ \mathbf{1} & & \mathbf{0} & & \mathbf{1} & & \mathbf{0} & & \mathbf{0} & & \mathbf{1} & & \mathbf{0} & & \mathbf{1} \end{array}$$

(b) Gray code to binary:

$$\begin{array}{ccccccccccccccc} 1 & & 0 & & 1 & & 0 & & 1 & & 1 & & 1 & & 1 \\ \downarrow & + \nearrow & \downarrow & + \nearrow & \downarrow & + \nearrow & \downarrow & + \nearrow & \downarrow & + \nearrow & \downarrow & + \nearrow & \downarrow & + \nearrow & \downarrow \\ \mathbf{1} & & \mathbf{1} & & \mathbf{0} & & \mathbf{0} & & \mathbf{1} & & \mathbf{0} & & \mathbf{1} & & \mathbf{0} \end{array}$$

(c.d.e)


1 1 0 1 = (a_0 a_1 a_2 a_3)

↓

1 0 1 1 b_0 b_1, b_2, b_3

$b_0 = a_0$ - directly written

$b_1 = a_0 \oplus a_1$

$b_2 = a_1 \oplus a_2$

$b_3 = a_2 \oplus a_3$

1 0 0 1 0 1 0

↓

1 1 0 1 1 1 1 – Gray code

c) 1 1 1 1 0 1 1 1 0 1 1 0

↓

1 0 0 0 1 1 0 0 1 1 0 1 – Gray code

Question 2: Boolean Algebra

Part a: Solve the following in order:

- i) **Apply the associative law of addition to the expression A + (B + C + D).**
- ii) **Apply the distributive law to the expression A(B + C + D).**

1. A + (B + C + D) = (A + B + C) + D

2. A (B + C + D) = AB + AC + AD

Part b) Demorgan's law

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

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

The following procedure illustrates the application of DeMorgan's theorems and Boolean algebra to the specific expression

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

Step 1: Identify the terms to which you can apply DeMorgan's theorems, and think of each term as a single variable. Let $\overline{A + B\overline{C}} = X$ and $\overline{D(E + \overline{F})} = Y$.

Step 2: Since $\overline{X + Y} = \overline{X}\,\overline{Y}$,

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

Step 3: Use rule 9 ($\overline{\overline{A}} = A$) to cancel the double bars over the left term (this is not part of DeMorgan's theorem).

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

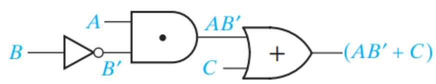
Step 4: Apply DeMorgan's theorem to the second term.

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

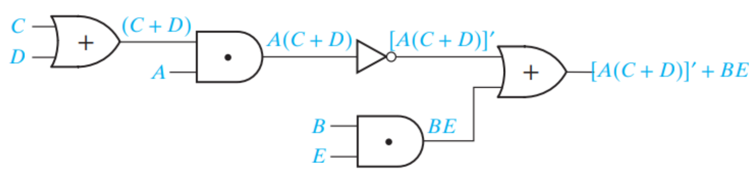
Step 5: Use rule 9 ($\overline{\overline{A}} = A$) to cancel the double bars over the $E + \overline{F}$ part of the term.

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

Part c:



(a)



(b)

a) **Part d: Perform the following:**

i) **Factor C′D + C′E′ + G′H and mention laws that were used in order to obtain the final equation. (So you can practice how factoring works during simplification)**

ii) **Find the inverse of the following function: F = A′B + AB′. Verify that your resultant Boolean equation is indeed the inverse to this equation via a truth table.**

Factor $C'D + C'E' + G'H$.

$C'D + C'E' + G'H = C'(D + E') + G'H$	← first apply the ordinary distributive law, $XY + XZ = X(Y + Z)$
$= (C' + G'H)((D + E') + G'H)$	← then apply the second distributive law
$= (C' + G')(C' + H)(D + E' + G')(D + E' + H)$	← now identify X, Y, and Z in each expression and complete the factoring

i)

The inverse of $F = A'B + AB'$ is

$$F' = (A'B + AB')' = (A'B)'(AB')' = (A + B')(A' + B)$$
$$= AA' + AB + B'A' + BB' = A'B' + AB$$

We will verify that this result is correct by constructing a truth table for F and F':

A B	A'B	AB'	F = A'B + AB'	A'B'	AB	F' = A'B' + AB
0 0	0	0	0	1	0	1
0 1	1	0	1	0	0	0
1 0	0	1	1	0	0	0
1 1	0	0	0	0	1	1

ii)

<u>Part e: Simplify the following to obtain SOP Boolean result (mention laws):</u>

Step 1: Factor BC out of the first and last terms.

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

Step 2: Apply rule 6 ($\overline{A} + A = 1$) to the term in parentheses, and factor $A\overline{B}$ from the second and last terms.

$$BC \cdot 1 + A\overline{B}(\overline{C} + C) + \overline{A}\,\overline{B}\,\overline{C}$$

Step 3: Apply rule 4 (drop the 1) to the first term and rule 6 ($\overline{C} + C = 1$) to the term in parentheses.

$$BC + A\overline{B} \cdot 1 + \overline{A}\,\overline{B}\,\overline{C}$$

Step 4: Apply rule 4 (drop the 1) to the second term.

$$BC + A\overline{B} + \overline{A}\,\overline{B}\,\overline{C}$$

a)

Step 5: Factor $\overline{B}$ from the second and third terms.

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

Step 6: Apply rule 11 ($A + \overline{A}\,\overline{C} = A + \overline{C}$) to the term in parentheses.

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

Step 7: Use the distributive and commutative laws to get the following expression:

$$BC + A\overline{B} + \overline{B}\,\overline{C}$$

Step 1: Apply DeMorgan's theorem to the first term.

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

Step 2: Apply DeMorgan's theorem to each term in parentheses.

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

Step 3: Apply the distributive law to the two terms in parentheses.

$$\overline{A}\,\overline{A} + \overline{A}\,\overline{C} + \overline{A}\,\overline{B} + \overline{B}\,\overline{C} + \overline{A}\,\overline{B}C$$

Step 4: Apply rule 7 ($\overline{A}\,\overline{A} = \overline{A}$) to the first term, and apply rule 10 [$\overline{A}\,\overline{B} + \overline{A}\,\overline{B}C = \overline{A}\,\overline{B}(1 + C) = \overline{A}\,\overline{B}$] to the third and last terms.

$$\overline{A} + \overline{A}\,\overline{C} + \overline{A}\,\overline{B} + \overline{B}\,\overline{C}$$

Step 5: Apply rule 10 [$\overline{A} + \overline{A}\,\overline{C} = \overline{A}(1 + \overline{C}) = \overline{A}$] to the first and second terms.

$$\overline{A} + \overline{A}\,\overline{B} + \overline{B}\,\overline{C}$$

Step 6: Apply rule 10 [$\overline{A} + \overline{A}\,\overline{B} = \overline{A}(1 + \overline{B}) = \overline{A}$] to the first and second terms.

b)

$$\overline{A} + \overline{B}\,\overline{C}$$

Question 3:

Part a: Convert each of the following Boolean expressions to SOP form:

i. **AB + B(CD + EF)**
ii. **(A + B)(B+C+D)**
iii. **((A+B)’+C)’**

(a) $AB + B(CD + EF) = AB + BCD + BEF$

(b) $(A + B)(B + C + D) = AB + AC + AD + BB + BC + BD$

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

Part b: Circuit realization:

$F = (X+Y')Z + X'YZ'$	(from the circuit)
$= (X+Y'+ X'YZ')(Z+X'YZ')$	(distributive law)
$= (X+Y'+X')(X+Y'+Y)(X+Y'+Z')(Z+X')(Z+Y)(Z+Z')$	(distributive law)
$= (1+Y')(X+1)(X+Y'+Z')(Z+X')(Z+Y)(1)$	(complementation laws)
$= (1)(1)(X+Y'+Z')(Z+X')(Z+Y)(1)$	(0 and 1 operations)
$= (X+Y'+Z')(Z+X')(Z+Y)$	(0 and 1 operations)
$G = (X +Y' + Z')(X' + Z)(Y + Z)$	(from the circuit)

Part c: K-Map Simplification

There are four 1s in the output column and the corresponding binary values are 011, 100, 110, and 111. Convert these binary values to product terms as follows:

$$011 \longrightarrow \overline{A}BC$$
$$100 \longrightarrow A\overline{B}\,\overline{C}$$
$$110 \longrightarrow AB\overline{C}$$
$$111 \longrightarrow ABC$$

The resulting standard SOP expression for the output X is

$$X = \overline{A}BC + A\overline{B}\,\overline{C} + AB\overline{C} + ABC$$

For the POS expression, the output is 0 for binary values 000, 001, 010, and 101. Convert these binary values to sum terms as follows:

$$000 \longrightarrow A + B + C$$
$$001 \longrightarrow A + B + \overline{C}$$
$$010 \longrightarrow A + \overline{B} + C$$
$$101 \longrightarrow \overline{A} + B + \overline{C}$$

The resulting standard POS expression for the output X is

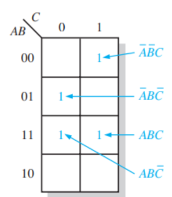
$$X = (A + B + C)(A + B + \overline{C})(A + \overline{B} + C)(\overline{A} + B + \overline{C})$$

i.

ii.

Evaluate the expression as shown below. Place a 1 on the 3-variable Karnaugh map in Figure 4–29 for each standard product term in the expression.

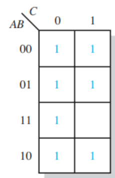
$$\overline{A}\,\overline{B}C + \overline{A}B\overline{C} + AB\overline{C} + ABC$$
$$001 \quad 010 \quad 110 \quad 111$$



The SOP expression is obviously not in standard form because each product term does not have three variables. The first term is missing two variables, the second term is missing one variable, and the third term is standard. First expand the terms numerically as follows:

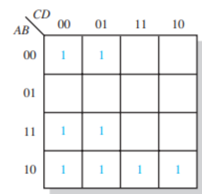
$\overline{A}$	$+ A\overline{B}$	$+ AB\overline{C}$
000	100	110
001	101	
010		
011		

Map each of the resulting binary values by placing a 1 in the appropriate cell of the 3-variable Karnaugh map in Figure 4–31.

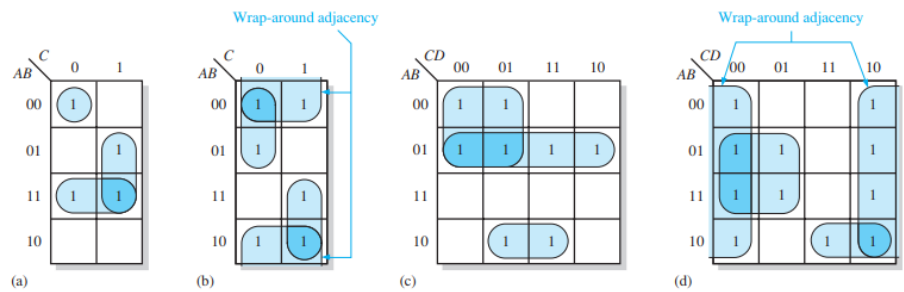


The SOP expression is obviously not in standard form because each product term does not have four variables. The first and second terms are both missing two variables, the third term is missing one variable, and the rest of the terms are standard. First expand the terms by including all combinations of the missing variables numerically as follows:

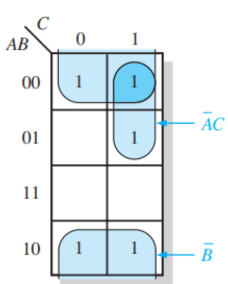
$\overline{B}\,\overline{C}$	$+ A\overline{B}$	$+ AB\overline{C}$	$+ A\overline{B}C\overline{D}$	$+ \overline{A}\,\overline{B}\,\overline{C}D$	$+ A\overline{B}CD$
0000	1000	1100	1010	0001	1011
0001	1001	1101			
1000	1010				
1001	1011				



iii)



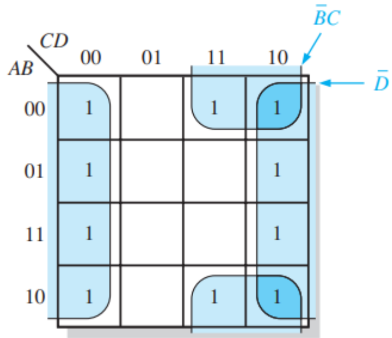
iv)



F= A’C + B’

v)

The first term $\overline{B}\,\overline{C}\,\overline{D}$ must be expanded into $A\overline{B}\,\overline{C}\,\overline{D}$ and $\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$ to get the standard SOP expression, which is then mapped; the cells are grouped as shown in Figure 4–38.



Notice that both groups exhibit "wrap around" adjacency. The group of eight is formed because the cells in the outer columns are adjacent. The group of four is formed to pick up the remaining two 1s because the top and bottom cells are adjacent. The product term for each group is shown. The resulting minimum SOP expression is

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

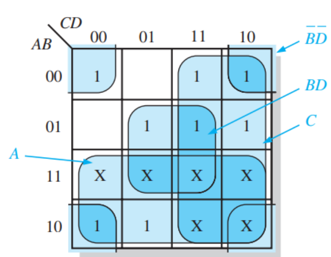
Keep in mind that this minimum expression is equivalent to the original standard expression.

vi)

The expression for segment *a* is

$$a = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}C\overline{D} + \overline{A}\,\overline{B}CD + \overline{A}B\overline{C}D + \overline{A}BC\overline{D} + \overline{A}BCD + A\overline{B}\,\overline{C}\,\overline{D} + A\overline{B}\,\overline{C}D$$

Each term in the expression represents one of the digits in which segment *a* is used. The Karnaugh map minimization is shown in Figure X's (don't cares) are entered for those states that do not occur in the BCD code.



From the Karnaugh map, the minimized expression for segment *a* is

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

Part d: circuits simplifications:

2.13 (a) $F_1 = A'A + B + (B + B) = 0 + B + B = B$

2.13 (b) $F_2 = A'A' + AB' = A' + AB' = A' + B'$

2.13 (c) $F_3 = [(AB + C)'D][(AB + C) + D]$
$= (AB + C)'D\,(AB + C) + (AB + C)'D$
$= (AB + C)'D$ By Th. 5D & Th. 2D

2.13 (d) $Z = [(A + B)C]' + (A + B)CD = [(A + B)C]' + D$
By Th. 11D with $Y = [(A + B)C]'$
$= A'B' + C' + D'$

Question 4:

Part a: An adder is to be designed which adds two 2-bit binary numbers to give a 3-bit binary sum. Find the truth table for the circuit. The circuit has four inputs and three outputs as shown in the diagram.

TRUTH TABLE:

N_1	N_2	N_3
A B	C D	X Y Z
0 0	0 0	0 0 0
0 0	0 1	0 0 1
0 0	1 0	0 1 0
0 0	1 1	0 1 1
0 1	0 0	0 0 1
0 1	0 1	0 1 0
0 1	1 0	0 1 1
0 1	1 1	1 0 0
1 0	0 0	0 1 0
1 0	0 1	0 1 1
1 0	1 0	1 0 0
1 0	1 1	1 0 1
1 1	0 0	0 1 1
1 1	0 1	1 0 0
1 1	1 0	1 0 1
1 1	1 1	1 1 0

From inspection of the table, the output functions are X(A, B, C, D) = Σ m(7, 10, 11, 13, 14, 15) Y(A, B, C, D) = Σ m(2, 3, 5, 6, 8, 9, 12, 15) Z(A, B, C, D) = Σ m(1, 3, 4, 6, 9, 11, 12, 14)

Next we already have studied how to implement these.

Part b:

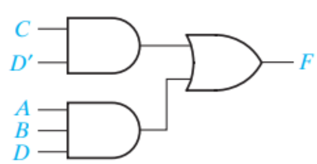
The valid 6-3-1-1 code combinations are listed in Table If any other combination occurs, this is not a valid 6-3-1-1 binary-coded-decimal digit, and the circuit output should be $F = 1$ to indicate that an error has occurred. This leads to the following truth table:

A	B	C	D	F
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	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

The corresponding output function is

$$F = \Sigma\, m(2, 6, 10, 13, 14, 15)$$
$$= A'B'CD' + A'BCD' + AB'CD' + ABCD' + ABC'D + ABCD$$
$$= A'CD' + ACD' + ABD = CD' + ABD$$

The realization using AND and OR gates is



Part c:

x	y	z	V	A	D	E
0	0	0	0	1	0	0
0	0	0	1	1	0	0
0	0	1	0	0	0	1
0	0	1	1	0	1	0
0	1	0	0	1	0	0
0	1	0	1	1	0	0
0	1	1	0	0	0	1
0	1	1	1	0	1	0
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	0	1
1	0	1	1	0	1	0
1	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	1	0

$A(x, y, z, V) = \prod (2, 3, 6, 7, 10, 11, 14, 15)$
$D(x, y, z, V) = \prod (0, 1, 2, 4, 5, 6, 8, 9, 10, 12, 13, 14)$
$E(x, y, z, V) = \prod (0, 1, 3, 4, 5, 7, 8, 9, 11, 12, 13, 15)$

Part d:

Out of sixteen possible combinations of four variables, the combinations in which there are exactly three 1s are listed in Table 5–5 along with the corresponding product term for each.

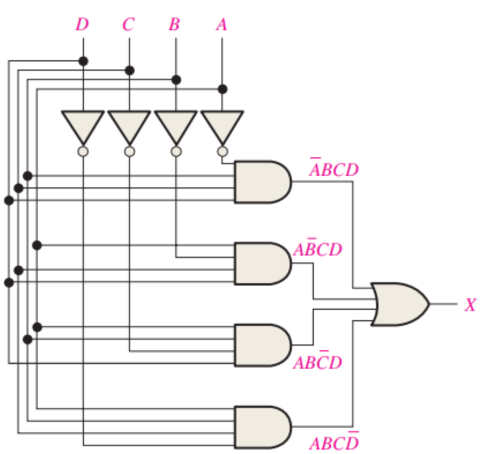
TABLE 5–5

A	B	C	D	Product Term
0	1	1	1	$\bar{A}BCD$
1	0	1	1	$A\bar{B}CD$
1	1	0	1	$AB\bar{C}D$
1	1	1	0	$ABC\bar{D}$

The product terms are ORed to get the following expression:

$$X = \bar{A}BCD + A\bar{B}CD + AB\bar{C}D + ABC\bar{D}$$

This expression is implemented in Figure with AND-OR logic.



Part e:

$ABCDE$		y	z
0 0 0 0 0	less than 10	+	
1 0 0 0 0	at least 10	+	
1 1 0 0 0	at least 20	+	+
1 1 1 0 0	at least 30		+
1 1 1 1 0	at least 40		+
1 1 1 1 1	at least 50		

4.2 (a) $Y = A'B'C'D'E' + AB'C'D'E' + ABC'D'E'$

4.2 (b) $Z = ABC'D'E' + ABCD'E' + ABCDE'$

Part f:

$ABCD$		F
0 0 0 0	$0 \times 0 = 0 \le 2$	1
0 0 0 1	$0 \times 1 = 0 \le 2$	1
0 0 1 0	$0 \times 2 = 0 \le 2$	1
0 0 1 1	$0 \times 3 = 0 \le 2$	1
0 1 0 0	$1 \times 0 = 0 \le 2$	1
0 1 0 1	$1 \times 1 = 1 \le 2$	1
0 1 1 0	$1 \times 2 = 2 \le 2$	1
0 1 1 1	$1 \times 3 = 3 > 2$	0
1 0 0 0	$2 \times 0 = 0 \le 2$	1
1 0 0 1	$2 \times 1 = 2 \le 2$	1
1 0 1 0	$2 \times 2 = 4 > 2$	0
1 0 1 1	$2 \times 3 = 6 > 2$	0
1 1 0 0	$3 \times 0 = 0 \le 2$	1
1 1 0 1	$3 \times 1 = 3 > 2$	0
1 1 1 0	$3 \times 2 = 6 > 2$	0
1 1 1 1	$3 \times 3 = 9 > 2$	0

(a) $F(A, B, C, D) = \sum m(0, 1, 2, 3, 4, 5, 6, 8, 9, 12)$
Refer to FLD p. 695 for full term expansion

(b) $F(A, B, C, D) = \prod M(7, 10, 11, 13, 14, 15)$
Refer to FLD p. 695 for full term expansion

Question 4:

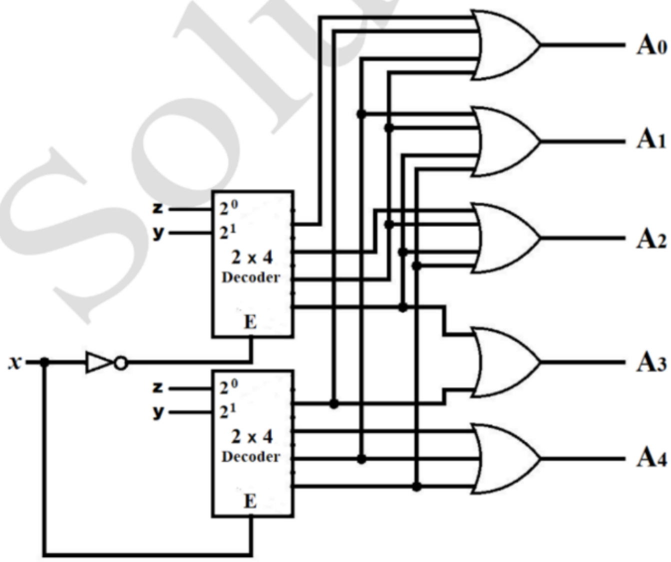
Part a:

The input is a 3-bit number (0 – 7).
The maximum output can be $(7 \times 3) + 1 = 22$, so we need 5 bits at the output.

Input Number	Input Bits			Output Number	Output Bits				
	x	y	z		A_4	A_3	A_2	A_1	A_0
0	0	0	0	$(0 \times 3) + 1 = 1$	0	0	0	0	1
1	0	0	1	$(1 \times 3) + 1 = 4$	0	0	1	0	0
2	0	1	0	$(2 \times 3) + 1 = 7$	0	0	1	1	1
3	0	1	1	$(3 \times 3) + 1 = 10$	0	1	0	1	0
4	1	0	0	$(4 \times 3) + 1 = 13$	0	1	1	0	1
5	1	0	1	$(5 \times 3) + 1 = 16$	1	0	0	0	0
6	1	1	0	$(6 \times 3) + 1 = 19$	1	0	0	1	1
7	1	1	1	$(7 \times 3) + 1 = 22$	1	0	1	1	0

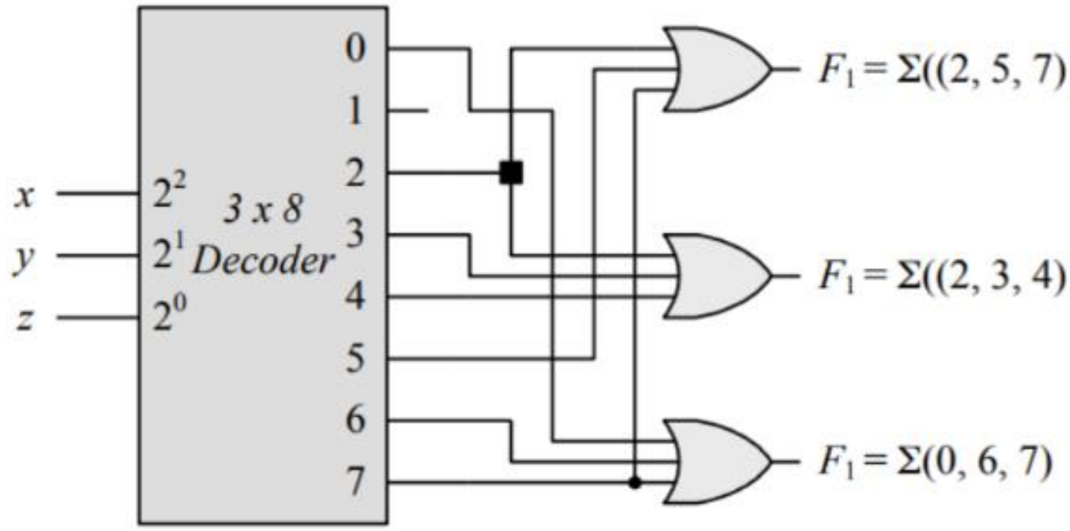
$A_4 = \Sigma(5, 6, 7)$ $A_3 = \Sigma(3, 4)$ $A_2 = \Sigma(1, 2, 4, 7)$
$A_1 = \Sigma(2, 3, 6, 7)$ $A_0 = \Sigma(0, 2, 4, 6)$

$A_4 = \Sigma(5, 6, 7)$ $A_3 = \Sigma(3, 4)$ $A_2 = \Sigma(1, 2, 4, 7)$
$A_1 = \Sigma(2, 3, 6, 7)$ $A_0 = \Sigma(0, 2, 4, 6)$

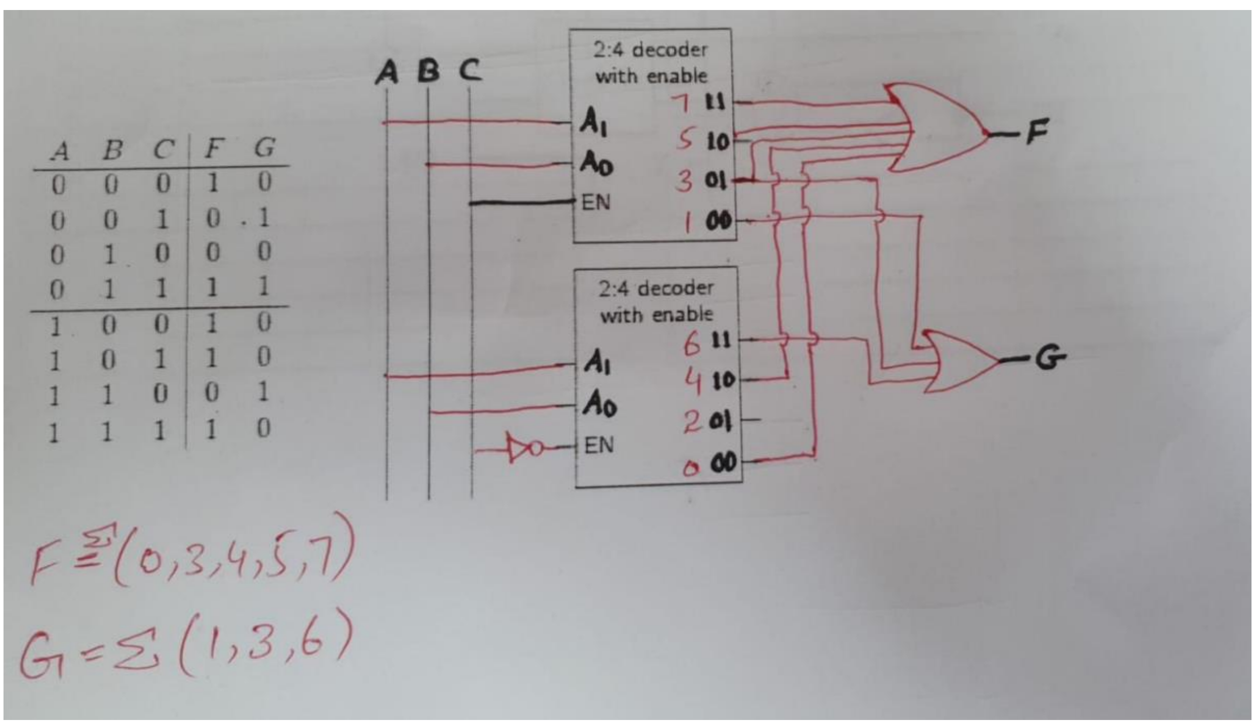


Part b:

$$F_1 = x(y + y')z + x'yz' = xyx + xy'z + x'yz' = \Sigma(2, 5, 7)$$
$$F_2 = xy'z' + x'y = xy'z' + x'yz + x'yz' = \Sigma(2, 3, 4)$$
$$F_3 = x'y'z' + xy(z + z') = x'y'z' + xyz + xyz' = \Sigma(0, 6, 7)$$

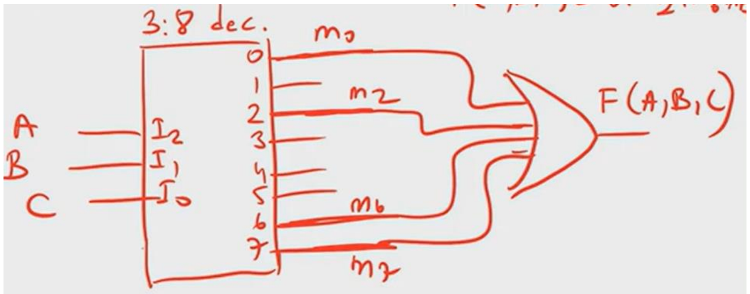


Part c:

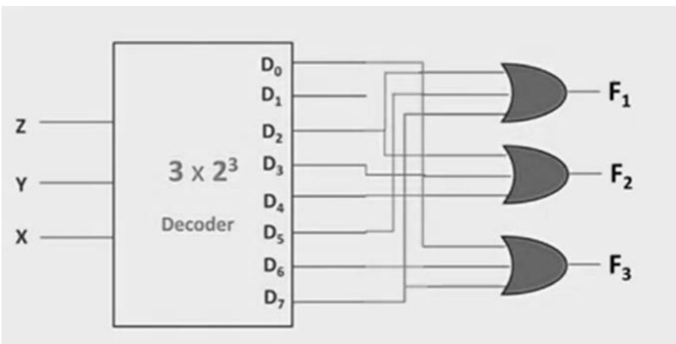


Part d:

i)



ii)



Question 5:

Part a:

The map of the given function:

$$F(A,B,C,D) = A'B'C'D + CD + AC'D$$

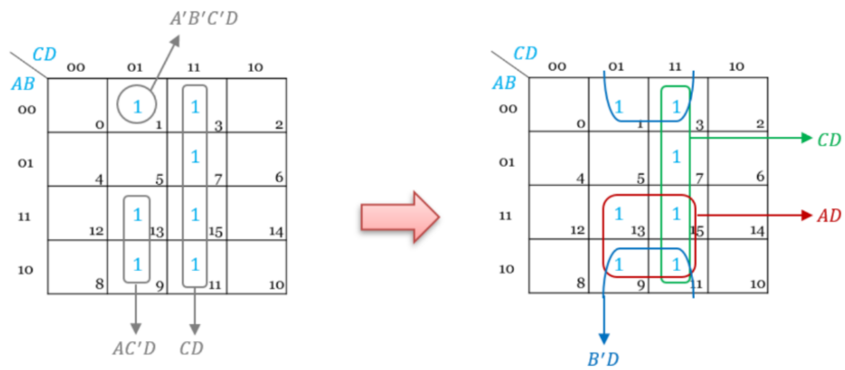
is as shown below in (a). This function can be simplified as shown below in (b). The simplified function is

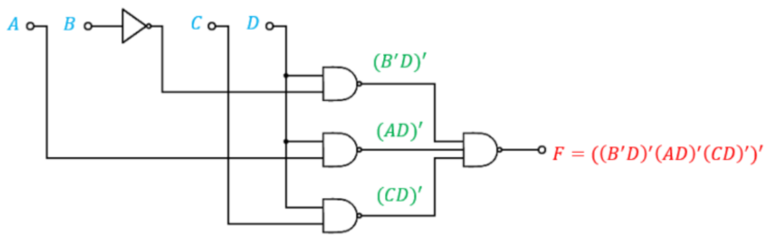
$$F(A,B,C,D) = CD + AD + B'D$$

This function can be expressed as

$$F = (F')' = [(CD + AD + B'D)']' = [(CD)'(AD)'(B'D)']' \Rightarrow DeMorgan's\ theorem$$

Therefore, we can implement the function F using two-level NAND gate circuit as shown below in (c).





(c)

Part B:

First, the 1's of

$$F = \sum(0, 3, 12, 15)$$

K-map plotted as shown below in (a). Then, from the 0's, we get

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

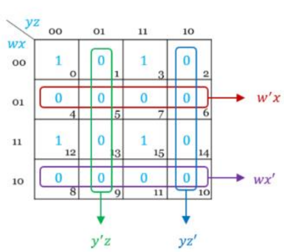
The simplified function in the minimum-product-of-sums form is

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

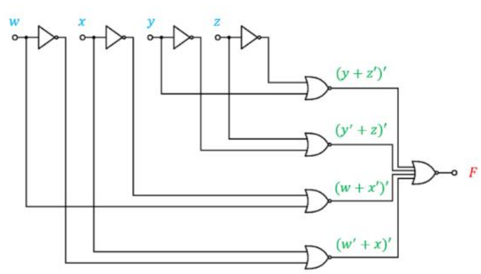
o implement the function F using two-level NOR gate circuit, we reformulate the function as follows:

$$F = (F')' = [\ (y+z')(y'+z)(w+x')(w'+x)\ ]''$$

$$= \{\ (y+z')' + (y'+z)' + (w+x')' + (w'+x)'\ \}'$$

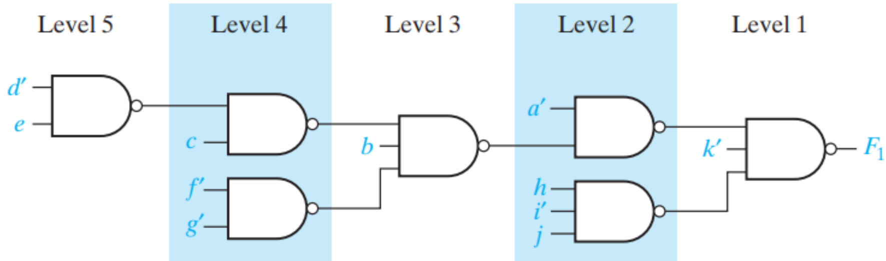


(a)



(b)

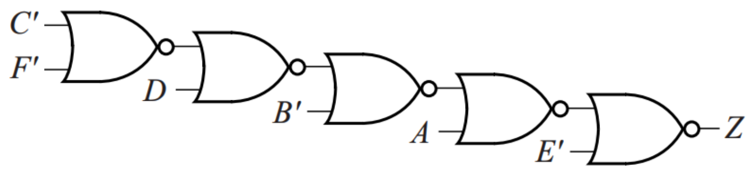
Part c:



(b) NAND network

Part d:

$$
\begin{aligned}
Z &= AE + BDE + BCEF \\
&= E\,(A + BD + BCF) \\
&= E\,[A + B\,(D + CF)]
\end{aligned}
$$

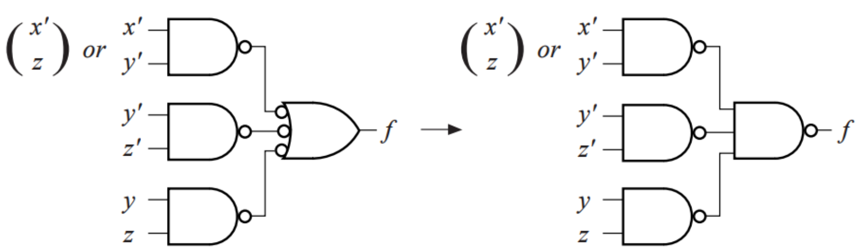


Part e:

yz \ x	0	1
00	1	1
01	1	0
11	1	1
10	0	0

f = yz + y'z' + x'y'

f = yz + y'z' + x'z



Part f:

Using NOR gates only:

cd \ ab	00	01	11	10
00	0	1	0	0
01	0	1	0	0
11	1	1	1	1
10	0	1	0	0

f = (b + c)(b + d)(a' + c)(a' + d)