1.23 ) a) **Sequential**: Since timing is involved and it has to go through a series of steps in a particular order.

b) **Combinational**: No timing involved, just arithmetic operation.

c) **Sequential**: again timing is involved here.

d) **Combinational**: Here we can do a simple comparison operation to sound an alarm, which is combinational logic.

e) **Combinational**: Simple AND gates can be used to do the logic.

f) **Combinational**: This is nothing but XOR gate.

e) **Sequential**: Here again the bits needs to be buffered one after the other which requires sequential logic. (Note: an element of timing is there.)

g) **Combinational**: This is a simple Decoder.

1.30 ) Just modify the State diagram in Pg 21 by drawing an arrow back from “Open” state to “S1” state after a delay for one clock cycle (Delays can be usually given by D-Flip Flops from hardware point of view) NOTE: D-Flip Flops will be dealt with later.

2.2 )

a) $X(Y+Z)$
b) \( XY + XZ \)

\[ 
\begin{array}{c}
X \\
\downarrow \\
Y \\
\downarrow \\
X \\
\downarrow \\
Z \\
\end{array} 
\]

\[ 
\begin{array}{c}
Z \\
\downarrow \\
Y \\
\downarrow \\
X \\
\end{array} 
\]

c) \( X(Y+Z) \)

\[ 
\begin{array}{c}
\neg X \\
\downarrow \\
\neg Y \\
\downarrow \\
\neg Z \\
\end{array} 
\]

d) \( \neg X + \neg YZ \)

\[ 
\begin{array}{c}
X \\
\downarrow \\
\neg X \\
\downarrow \\
Y \\
\downarrow \\
Z \\
\end{array} 
\]

e) \( W(X+YZ) \)

\[ 
\begin{array}{c}
W \\
\downarrow \\
X \\
\downarrow \\
Y \\
\downarrow \\
Z \\
\end{array} 
\]
2.6) 

b) \( X(X+Y) = XX + XY \)
   \( = X+XY \)
   \( = X(1+Y) \)
   \( = X \)

d) \( (X+Y)(X'+Z) = XX'+XZ+YX'+YZ \)
   \( = XZ+X'Y+YZ \)
   \( = XZ+X'Y+YZ(X+X') \)
   \( = XZ(1+Y) + X'Y(1+Z) \)
   \( = XZ+X'Y \) (Consensus Theorem)

2.10) 

b) \( ABC + B(C'+D') \)
    \( = [ABC + B.(C'+D')]' \)
    \( = (ABC)'[B.(C'+D')]' \)
    \( = (A'+B'+C')(B+CD) \)

g) \( X(Y+ZW'+V'S) \)
    \( = [X(Y+ZW'+V'S)]' \)
    \( = X' + (Y+ZW'+V'S)' \)
    \( = X' + [Y'.(ZW')'.(V'S)'] \)
    \( = X' + Y'(Z'+W)(V+S') \)
Appendix Problems

A1) (c) \((0101011)_2 = (1 \times 2^2 + 1 \times 2 + 1 \times 2^0)\) = \((43)_{10}\)

(b) \((123)_8 = (1 \times 8^2 + 2 \times 8 + 3 \times 8^0)\) = \((83)_{10}\)

(i) \(3AE_{16} = (3 \times 16^2 + 10 \times 16 + 14 \times 16^0)\) = \((942)_{10}\)

A2) (c) \((129)_{10} \rightarrow (\_\_\_\_)_2\) = \(129\)

\[
\begin{array}{c|c}
1 & 129 \\
2 & 64 & 1 \\
2 & 32 & -6 \\
2 & 16 & -0 \\
2 & 8 & -0 \\
2 & 4 & -0 \\
2 & 2 & -0 \\
2 & 1 & -0 \\
\end{array}
\]

\(\therefore (129)_{10} = (10000001)_2\)

(b) \((798)_{10} \rightarrow (\_\_\_\_)_8\) = \(798\)

\[
\begin{array}{c|c}
8 & 798 \\
8 & 99 & -6 \\
8 & 12 & -3 \\
1 & 1 & -4 \\
\end{array}
\]

\(\therefore (798)_{10} = (1436)_8\)
(i) \((24_{10})_{10} \rightarrow (\_\_\_)_{16}\)

\[\begin{array}{c}
16 \bar{2} 4_{10} \\
15 - 0
\end{array}\]

\[(24_{10})_{10} \rightarrow (\_\_\_6 (F)_{16})\]

\[A_3\] \(1001000111000101_2\) to base 8

\[= (1 \times 2^7) + (1 \times 2^6) + (1 \times 2^5) + (1 \times 2^4) + (1 \times 2^3) + (1 \times 2^2) + (1 \times 2^1) + (1 \times 2^0)\]

\[= 32768 + 4096 + 256 + 128 + 64 + 1 + 1\]

\[= 37317_{10}\]

Now

\[8 \left[ 37317 \right] \]

\[= 4669 - 5\]

\[8 \left[ 583 - 0 \right] \]

\[8 \left[ 72 - 7 \right] \]

\[8 \left[ 9 - 0 \right] \]

\[8 \left[ 0 - 0 \right] \]

\[8 \left[ 1 - 1 \right] \]

\[\text{ANS} = (1107305)_{8}\]

\[11100011001110001010_2\] to base 16

\[= (1 \times 2^7) + (1 \times 2^6) + (1 \times 2^5) + (1 \times 2^4) + (1 \times 2^3) + (1 \times 2^2) + (1 \times 2^1) + (1 \times 2^0)\]

\[+ (1 \times 2^9) + (1 \times 2^3)\]

\[= (930584)_{10}\]
A4) $C \Rightarrow (101)_8 \rightarrow (1\times 8^2) + (1\times 8^1) = (17)_{10}$

Now $2 \begin{array}{c}17 \\ 8 - 1 \\ 4 - 0 \\ 1 - 0 \end{array} \Rightarrow (10001)_2$.

(b) $(8FC)_{16} \rightarrow (8 \times 16^2) + (15 \times 16^1) + (12 \times 16^0)$

$$= 2048 + 240 + 12 = (2300)_{10}$$

Now $2 \begin{array}{c}2300 \\ 130 - 0 \\ 67 - 0 \\ 287 - 1 \\ 143 - 1 \\ 71 - 1 \\ 35 - 1 \\ 17 - 1 \\ 8 - 0 \\ 4 - 0 \\ 2 - 0 \\ 1 - 0 \\ 0 - 0 \end{array} \Rightarrow (8FC)_{16} = (100011111100)_2$.
A7) (c)  
1 1 1 1 1 0  
+  
1 0 1 1 1  
-------------  
1 0 0 1 0 1 0 1

(b)  
1 0 1 0 1 0 1 0  
+  
0 1 1 1 1 1 1  
-------------  
1 0 0 1 0 1 0 0 1

De) (d)  
1 1 1 0 0 1  
-  
1 0 0 0 1  
-------------  
1 0 1 0 0 0

(c)  
1 1 0 1 1 0 0 1 1 0  
-  
1 0 0 1 1 0 0 1  
-------------  
1 1 0 0 1 0 0 1 1 0 1

A9) (d)  
\( (A9 \ DE)_{16} \rightarrow (\cdot )_{3} \)

\[ (A9 \ DE)_{16} = (10 \times 16^3) + (9 \times 16^2) + (13 \times 16) + (15 \times 16) \]
\[ = 160960 + 2304 + 208 + 14 \]
\[ = 143186 \]
\[ (A9DE)_{16} = (2012122121)_3 \]

(A11) (a) \(-13\) \(
\begin{align*}
2 & \quad \underline{13} \\
2 & \quad \underline{6-1} + 13 = 001101 \\
2 & \quad \underline{3-0} - 13 = 101101 \\
1 & \quad \text{SIGN BIT}
\end{align*}
\)

(A12) (b) \(-27\) \(
\begin{align*}
2 & \quad \underline{27} \\
2 & \quad \underline{13-1} + 27 = 011011 \\
2 & \quad \underline{6-1} - 27 = 100100 \quad \text{ONE'S COMPLEMENT} \\
2 & \quad \underline{3-0} \ \\
1 & \quad \text{COMPLEMENT}
\end{align*}
\)
(b) $-5 \rightarrow 2^5$ Complement

$+5 \rightarrow 000101$

$-5 = 111011$ in $2^5$ Complement
1.3 ) (1) Direct Sequence Binary Encoding: To represent 52 cards we need 6 bits. Each card can be directly encoded in an order (Some extra combinations may not be used as with 6 bits we get 64 combinations but we need only 52 of them).

(2) Use 2 bits to encode which class of card a particular card belongs to (i.e., spades, hearts, etc.) Then 4 bits to represent the card it refers to in that class. (to represent 13 cards in each class we need at least 4 bits, though some combinations may not be used here also) e.g.: 000010 – the most significant 2 bits “00” can refer to spade and the remaining 4 bits can refer to a particular card in spade.

**NOTE:** Omit 1.30 in solution set as it is not included in HW1.