Chapter 4

Fundamental of Computer Logic n PLC

4.1 Introduction.
4.2 Gates and their input-output functions.
4.3 Basic theorems of Boolean Algebra.
4.4 Duality theorem :
4.5 Demorgan's theorem :
4.6 Implementation of Boolean equations using RLL circuit and electronic solid state logic gates
4.7 Logic network design
4.8 Combination and sequential logic networks

4.1 Introduction on Computer Logic

- James Bool, an Irish mathematician, developed Boolean algebra in the 1800's. It was found to be useful technique to design digital and logic circuits in logic control and computer logic circuits.

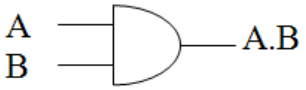
- The engineer or technicians can used this method to model very complicated logic control problem using a single equation. Furthermore, this equation can be simplified using different techniques, e.g. Karnaugh Maps, to optimize the production of the circuit.

- Relay ladder logic can be very easily developed for same Boolean equation. The simplification of the Boolian equation will save the PLC memory during the RLL development.

4.2 Computer Logic Gates

- **The logic gate AND**

For example, A and B often **written** A.B=Boolen equation. The truth table, *IEEE* graphic symbol and Relay Ladder Logic (*RLL*) circuit are given as follows:



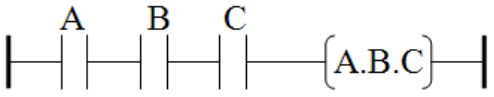
IEEE **symbol of AND gate**

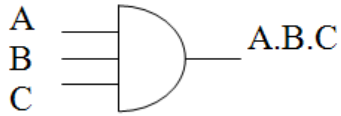
Truth table of AND gate

A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

RLL **for AND gate**

Product of A.B is true when both A and B true, otherwise any combination the results are false. The RLL circuit and IEEE symbol for AND gate with 3 inputs are given as follows:





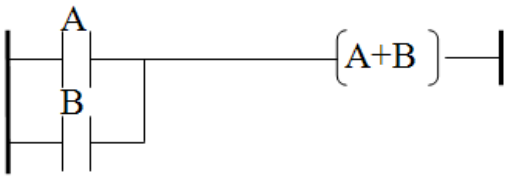
IEEE **symbol of AND gate for 3 input**

RLL **AND gate with 3 inputs.**

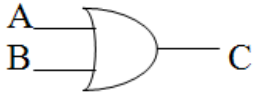
4.2 Computer Logic Gates

- **The logic gate OR**

The logic variable A or with logic variable B is written as A+B. The results of thi equation are true when either A or B or both true, A+B is true. The *IEEE* syr circuit, and truth table for OR gate are given as below:



RLL for OR gate



IEEE symbol OR gate

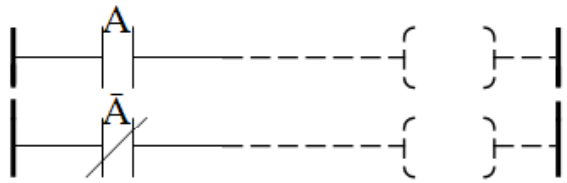
Truth table for OR logic gate

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

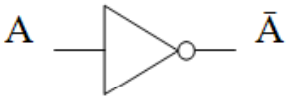
4.2 Computer Logic Gates

- **The logic gate NOT**

It is logic complement of a logic variable, notation Ā. The *IEEE* symbol, *RLL* symbol, and truth table for NOT logic gate are shown below:



RLL symbol for NOT gate (Ā).



IEEE graphic symbol for NOT logic gate

Truth table for NOT logic gate

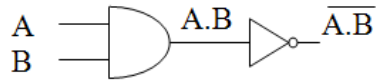
A	NOT (A)
1	0
0	1

4.2 Computer Logic Gates

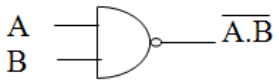
These three logic gates, AND, OR and NOT, can be combined to form other logic devices, e.g. NAND which is a combination of NOT and AND gates. Also, NOR, EOR (exclusive OR gate).

- **The logic gate NAND**

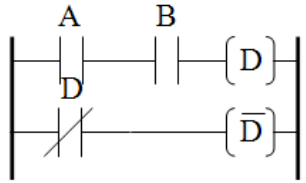
The *IEEE* graphic symbol and truth table are given as follows:



AND and NOT gates used to form NAND



***IEEE* symbol of NAND gate**



***RLL* circuit for NAND logic gate.**

Truth table of NAND gate

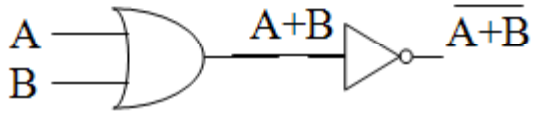
A	B	$\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

4.2 Computer Logic Gates

These three logic gates, AND, OR and NOT, can be combined to form other logic devices, e.g. NAND which is a combination of NOT and AND gates. Also, NOR, EOR (exclusive OR gate).

- **The logic gate NOR**

The IEEE graphic symbol and truth table are given as follows:



OR and NOT gates used to form NAND

***IEEE* symbol of NOR gate**

Truth table of NOR logic gate

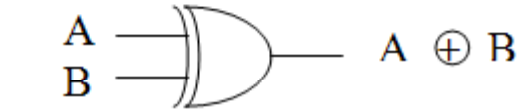
A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

- **The logic gate exclusive OR (EOR)**

The IEEE graphic symbol and truth table are given as follows:

Truth table for EOR logic gate

A	B	A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0



IEEE graphic symbol of EOR logic gate.

4.3 Basic theorems of Boolean Algebra

Table 4.1 Boolean Algebra Theorems

Boolean Law	OR gate	AND gate
1.Commutative Law	A+B=B+A	A.B=B.A
2.Associative Law	A+B+C=A+(B+C)	A.B.C=(A.B).C
	=(A+B)+C	=A.(B.C)
3.Distribution	A+(B.C)=(A+B).(A+C)	A.(B+C)=A.B+A.C
4.Absorptive	A+(A.B)=A	A.(A+B)=A
5.Identities	A+0=A	A.0=0
	A+1=1	A.1=A
	A+A=A	A.A=A
	A+$\bar{A}$=1	A.$\bar{A}$=0
		($\bar{A}$)=A

4.4 Duality theorem :

Dual identity is produced when logic 0 is replaced by logic 1 and logic 1 is replaced by logic 0. Furthermore, the identity "OR" changed to "AND", while the identity "AND" changed to "OR".

Example 1: Produce the dual function for the following identities:

A.1=A	dual function will be =====>	A+0=A
A.A=A	dual function will be =====>	A+A=A
A.0=0	dual function will be =====>	A+1=1
A.$\bar{A}$=0	dual function will be =====>	A+$\bar{A}$=1

Example 2: Prof the absorptive identity?

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

4.5 Demorgan's theorem :

$$\overline{A+B+C+.....+N} = \bar{A}.\bar{B}.\bar{C}.....\bar{N}$$
$$\overline{A.B.C.....N} = \bar{A}+\bar{B}+\bar{C}+...+\bar{N}$$

Example 3: Draw the RLL circuit, logic network and produce the truth table for the following Boolean equation:

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

Solution : *as shown in Fig.* 4.1 and Table 4.2.

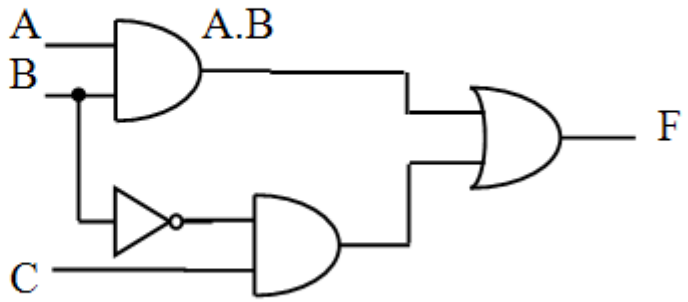


Fig. 4.1 *RLL* AND *IEEE* gate network for example 3.

4.6 Implementation of Boolean equations using RLL and Electronic solid state logic gates

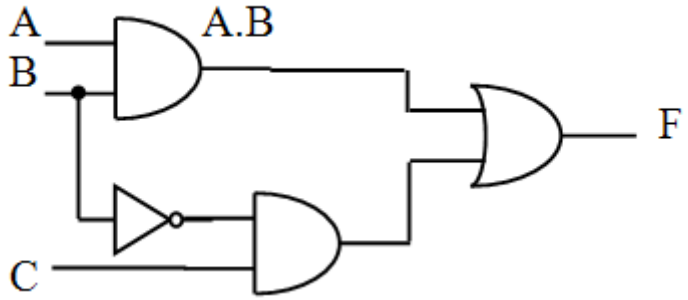


Table 4.2 Truth table for example 3.

A	B	C	A.B	$\overline{B}$	$\overline{B}.C$	$A.B + \overline{B}.C$
0	0	0	0	1	0	0
0	0	1	0	1	1	1
0	1	0	0	0	0	0
0	1	1	0	0	0	0
1	0	0	0	1	0	0
1	0	1	0	1	1	1
1	1	0	1	0	0	1
1	1	1	1	0	0	1

4.6 Implementation of Boolean equations using RLL and Electronic solid state logic gates

Example: Determine X ?

given

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

assuming A=1, B=0, C=1

$$X = \overline{(1 + 0 \cdot 1)} + 1 \cdot (0 + \overline{1})$$

$$X = \overline{(1 + 0)} + 1 \cdot (0 + 0)$$

$$X = \overline{(1)} + 1 \cdot (0)$$

$$X = 0 + 0$$

$$X = 0$$

Example: Simplify;

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

$$X = \overline{(A)} + \overline{(B \cdot C)} + A \cdot (B + \overline{C})$$ ← The higher priority operators are put in parenthases

$$X = \overline{(A)} \cdot \overline{(B \cdot C)} + A \cdot (B + \overline{C})$$ ← DeMorgan's theorem is applied

$$X = \overline{A} \cdot (\overline{B} + \overline{C}) + A \cdot (B + \overline{C})$$ ← DeMorgan's theorem is applied again

$$X = \overline{A} \cdot \overline{B} + \overline{A} \cdot \overline{C} + A \cdot B + A \cdot \overline{C}$$ ← The equation is expanded

$$X = \overline{A} \cdot \overline{B} + (\overline{A} \cdot \overline{C} + A \cdot \overline{C}) + A \cdot B$$ ← Terms with common terms are collected, here it is only NOT C

$$X = \overline{A} \cdot \overline{B} + \overline{C} \cdot (\overline{A} + A) + A \cdot B$$ ← The redundant term is eliminated

$$X = \overline{A} \cdot \overline{B} + \overline{C} + A \cdot B$$ ← A Boolean axiom is applied to simplify the equation further

Example 4: Simplify the following Boolean functions to minimum number of literals? (note; A.B ; A or B called literals)

Solution:

a) $X + \overline{X}.Y = (X + \overline{X}).(X + Y) = 1.(X + Y) = X + Y$

b) $X.(\overline{X} + Y) = X.\overline{X} + X.Y = 0 + X.Y = X.Y$

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

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

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

4.6 Implementation of Boolean equations using RLL and Electronic solid state logic gates

Example 5: Consider a heater oven with two bays can heat one ingot in each bay. When the heater is on it provides enough heat for two ingots. But, if only one ingot is present the oven may become too hot, so a fan is used to cool the oven when it passes a set temperature.

The control problem described as when the temperature of the oven is too high (greater than or equal to as set value) and there is an ingot in only one bay then turn on the fan.

Solution : Assume

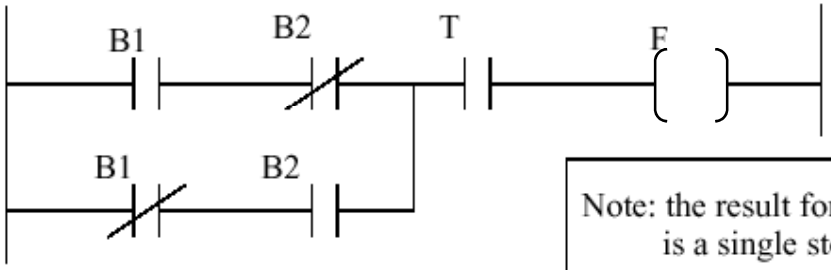
B1= logic sensor indicating ingot exists in bay1,
B2= logic sensor indicating ingot exists in bay 2,
F= actuator of the fan,
T= temperature overheat sensor.

4.6 Implementation of Boolean equations using RLL and Electronic solid state logic gates

$$F = T \cdot (B_1 \oplus B_2)$$

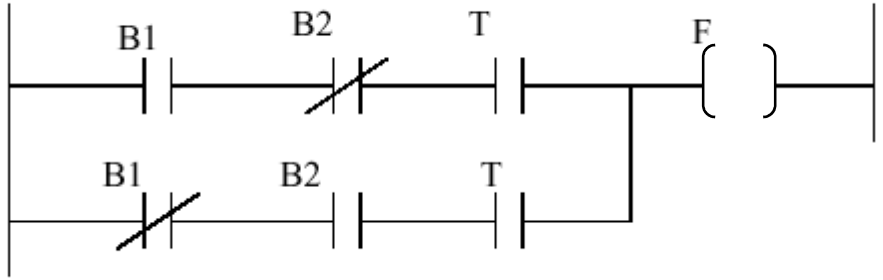
$$F = T \cdot (B_1 \cdot \overline{B_2} + \overline{B_1} \cdot B_2) \qquad (2)$$

$$F = B_1 \cdot \overline{B_2} \cdot T + \overline{B_1} \cdot B_2 \cdot T \qquad (3)$$



Note: the result for conditional logic is a single step in the ladder

Ladder Logic for Equation (3):

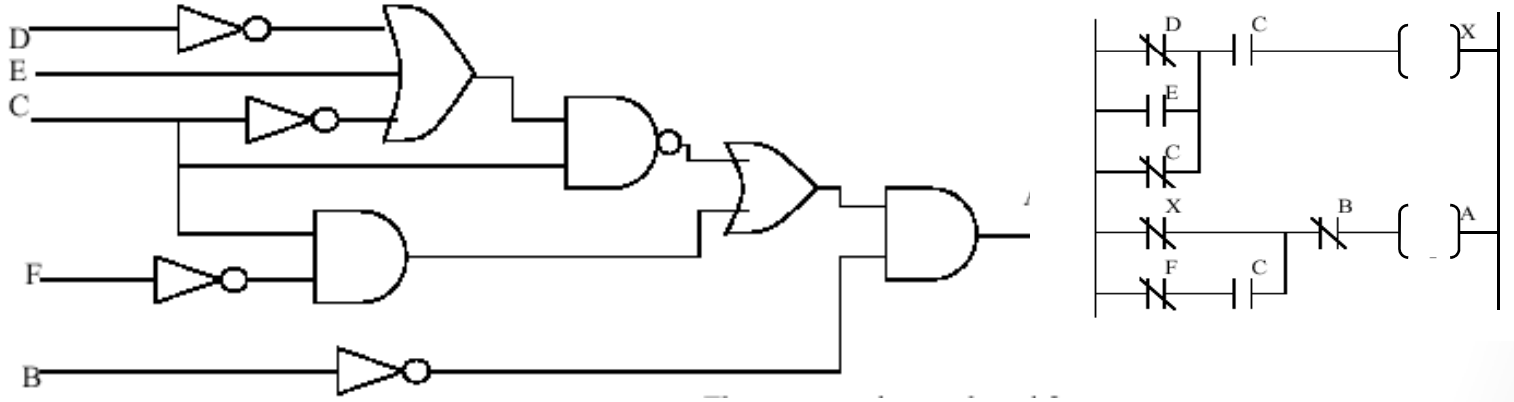


4.6 Implementation of Boolean equations using RLL and Electronic solid state logic gates

Example 6: given the following Boolean equation , sketch logic network and RLL

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

The circuit is given below, and equivalent ladder logic is shown.



4.6 Logic network design

For a given logic control sequence or truth table, It is possible to produce the
Boolean Algebra expression using the following steps:

1. Construct a truth table for the design input/output relationship if not given.
2. For each true statement in an output, form the logic product (AND) of the inputs.
3. For each output have true statements, form the logic Sum (OR) of the logic product produced in step 2.

Example 7: For exclusive OR logic operation (EOR), the output of the network is false only when the inputs are of the same logic level. Otherwise the outputs are true. Develop the combination network of an exclusive OR circuit? Draw network?

Step 1: Truth table development:

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

Step 2: There are two true outputs. Forming logic product using

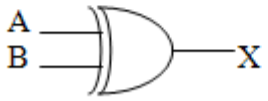
$A = 1 \quad ==> \quad \overline{A} = 0$

$B = 1 \quad ==> \quad \overline{B} = 0$

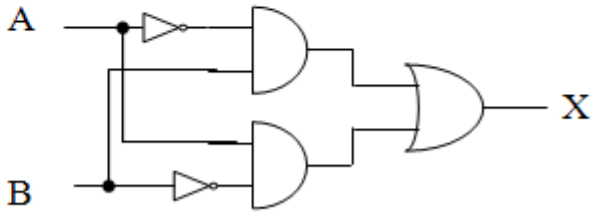
$\overline{A}.B \quad , \quad A\overline{B} \quad ==> \quad X = 1$

Step 3: The logic Sum (OR) of the logical product given in step 2 is:

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



IEEE Symbol for EOR

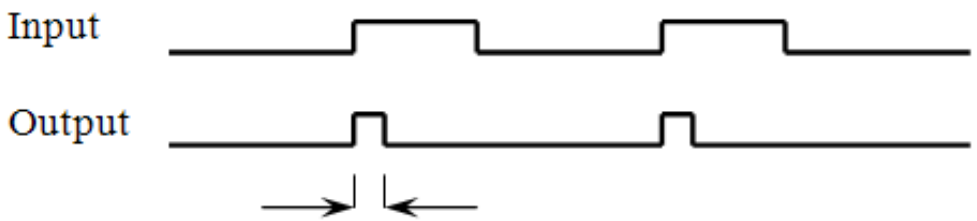


4.6 Combination and Sequential Networks

Combination logic network is a combination of logic operation without consideration of the time required. When time or history is required by the logic function, sequential logic network is used and passes a memory. Three types of logic networks are available :

a) Bi-stable logic network: This has two stable states and known as set/reset or R/S or flip-flop element. Such examples for this network are R/S and J-K flip-flop.

b) Mon-stable logic network: It is one stable state and it can be momentarily take-up the other states for a defined period of time, refer the input/output mono-stable signals. This type of signals most common in PLC and computer networks.



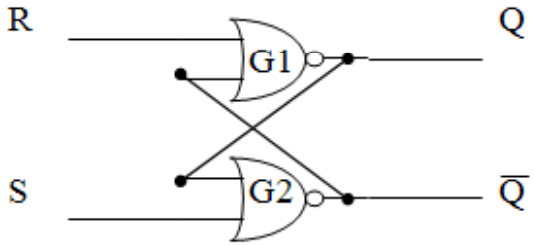
Time can be changed(define)

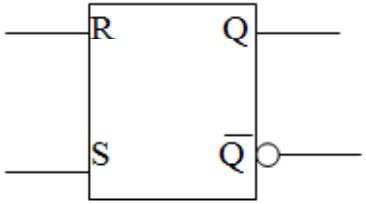
c) Unstable logic network: It has no stable state and used as oscillation.

1) Bi-stable logic network (memory elements)

- *R-S flip-flop*

The R-S flip-flop network is a sequential logic network consist of two NOR logic gates as illustrated in Fig. 4.2:





R=1, S=0

G1: $\overline{1 + Q_n}$ =>0 => Q_{n+1}

G2: $\overline{0 + 0}$ =>1 => $\overline{Q}_{n+1}$

R=0, S=1

G2: $\overline{1 + Q_n}$=0 =>$\overline{Q}_{n+1}$

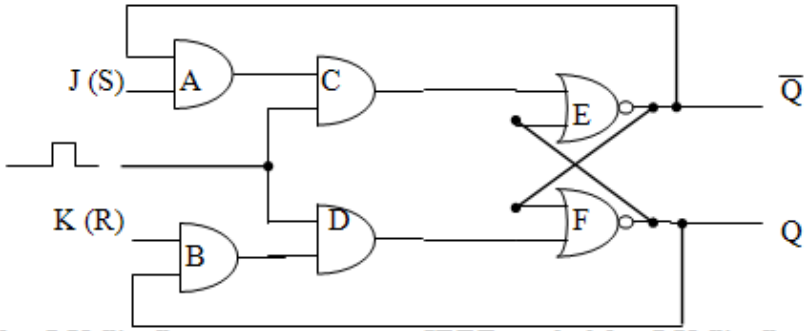
G1: $\overline{0 + 0}$ =1 =>Q_{n+1}

Fig. 4.2 *R-S* Flip-flop *IEEE* network

Input		Output	
S	R	Q_{n+1}	$\overline{Q_{n+1}}$
0	0	Q_n	Q_n
0	1	0	1
1	0	1	0
1	1	Not define	Not define

- *J-K Flip-flop*

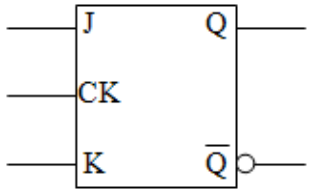
The J-K flip-flop is a small modification on R-S flip-flop, where more gates added to R-S flip-flop. The sequential logic network for J-K flip-flop is given as follows:



The Truth table for J-K flip-flop

Input		Output	
J	K	Q_{n+1}	$\overline{Q}_{n+1}$
0	0	Q_n	$\overline{Q}_n$
0	1	0	1
1	0	1	0
1	1	$\overline{Q}_n$	Q_n

***IEEE* symbol for J-K flip-flop**



For J= 0 and K=1 and initial state $\overline{Q}=0, \quad Q=1$

At gate A: $0.\overline{Q} \Rightarrow C$

At gate B: $1.Q = Q \Rightarrow D$

At gate C: $0.1 = 0 \Rightarrow E$

At gate D: $Q.1 = Q \Rightarrow F$

At gate E: $\overline{0+Q} = \overline{Q} \Rightarrow F$

At gate F: $\overline{Q+\overline{Q}} = 0 \Rightarrow E$

At gate E: $\overline{0+0} = 1 \Rightarrow \overline{Q}$

At gate F: $\overline{1+Q} = 0 \Rightarrow Q$

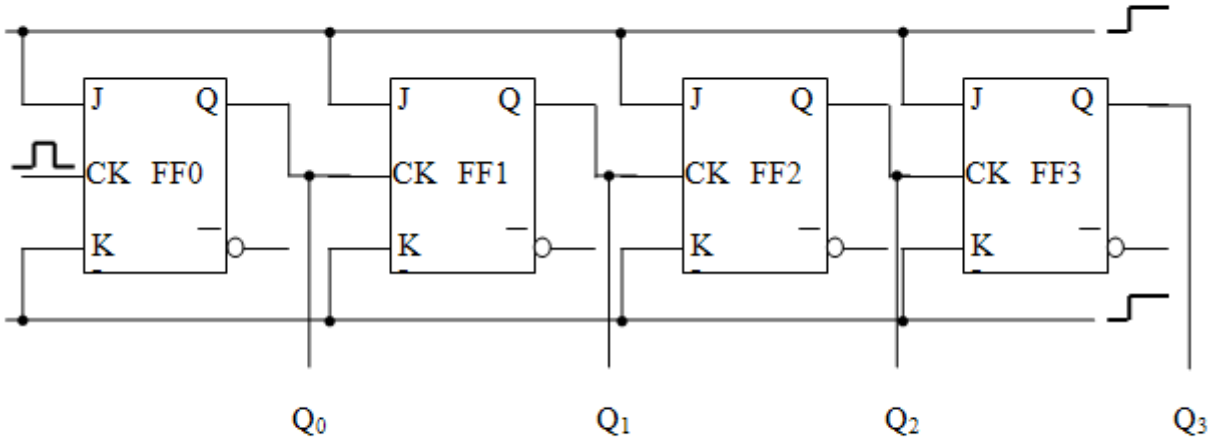
This illustrates prove of the truth table for J=0 and K=1.

*As Home work repeat these procedure for J=1 and K=0 for given initial states.

- *Ripple counter*

Ripple counter, asynchronous binary counter, is another example for sequential logic network. The network is for ripple counter can be developed using multiple J-K flip-flops as shown below:

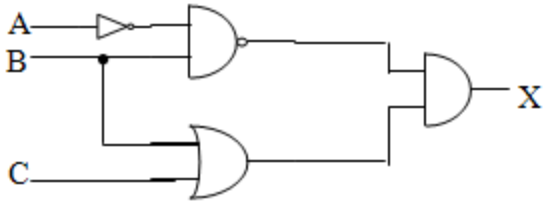
Ripple counter using 4 J-K flip-flops shown in Fig. 4.3.



By setting K = J = Set = Clear = logic 1.

BROBLEMS

4-1) What is the truth table for the network given as follows:



4-2) Develop Boolean expression from the following truth table, simplify it and give for minimum literals?

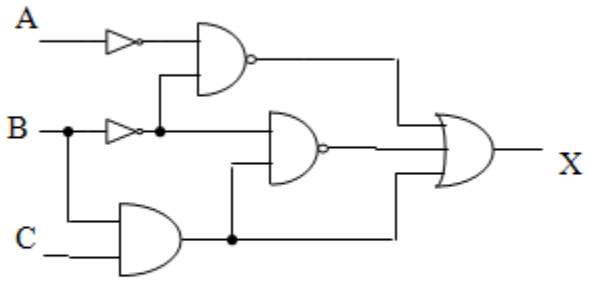
Input 1 A	Input 2 B	Output X
0	0	1
0	1	0
1	0	1
1	1	1

4-3) Develop Boolean expression for the following truth tables? Draw single logic network for the developed Boolean expressions?

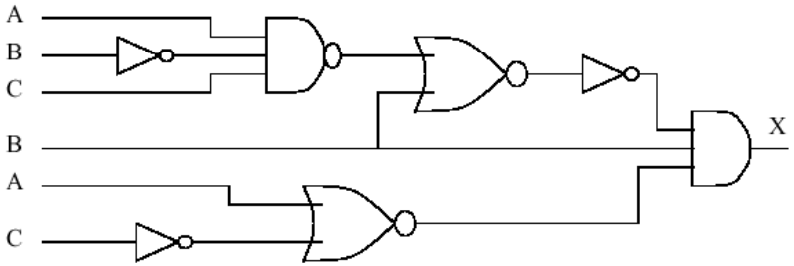
Output X	Input 1 Y	Input 2 Z
0	0	0
0	0	1
1	1	0
1	1	1

4-4) Using the properties of Boolean Algebra, simplify the answer given in question 3.

4-5) Develop and simplify the Boolean expression for the combination network below:



4-6) Develop the Boolean equation for the following logic network, next simplify it and develop *RLL* circuit for the same Boolean equation:



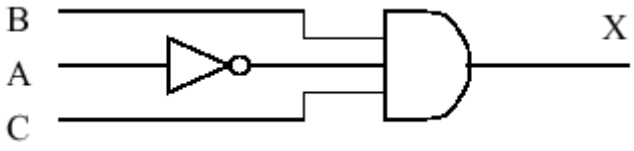
$$X = \overline{\left(\overline{\overline{(A \cdot \overline{B} \cdot C) + B}}\right) \cdot B \cdot \overline{(A + \overline{C})}}$$

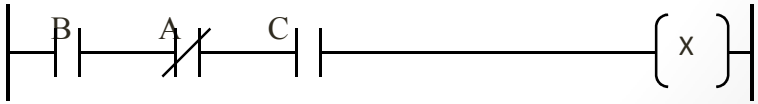
$$X = (\overline{A} + B + \overline{C} + B) \cdot B \cdot (\overline{A} \cdot C)$$

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

$$X = B \cdot \overline{A} \cdot C + B \cdot \overline{A} \cdot C + 0 + B \cdot \overline{A} \cdot C$$

$$X = B \cdot \overline{A} \cdot C$$





4-7) Simplify the following Boolean equations:

a) $A(B+AB)$

b) $\overline{A(\overline{B+AB})}$

c) $\overline{A}(B+AB)$

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

e) $\overline{(A+B)}.\overline{(A+\overline{B})}$

f) $ABCD+\overline{A}BCD+ABC\overline{D}+AB\overline{C}\,\overline{D}$

g) $((A.\overline{B})+\overline{(\overline{B}+A)}).C+(\overline{B}.C+B.C)$

h) $A\overline{B}.C+\overline{(C+B)}$

i) $A.\overline{(\overline{A}+\overline{A}.B)}$

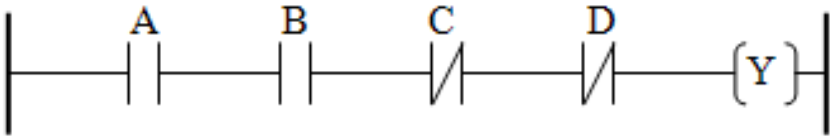
(ans: a) AB b) $\overline{A}+B$ c) $\overline{A}B$ d) $A+B$ e) 0 f) $B(CD+A\overline{D})$ g) C h) $\overline{B}(AC+\overline{C})$ I) A)

4-8) For the given Boolean equations in problem (4-7)a,b and c; draw the unsimplified and simplified logic networks?

4-9) Simplify the following Boolean equation and write the corresponding RLL:

$$Y = \overline{\overline{(AB\overline{C}D + AB\overline{C}\,\overline{D} + \overline{A}BCD + \overline{A}\,\overline{B}CD)} + D}$$

ans: $Y = AB\overline{C}\,\overline{D}$



4-10) For the following Boolean equation:

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

a) Write out the *RLL* of unsimplified equation.
b) Simplify the equation.
c) Write out the RLL for the simplified equation.
(ans: b: $A+DCB$)