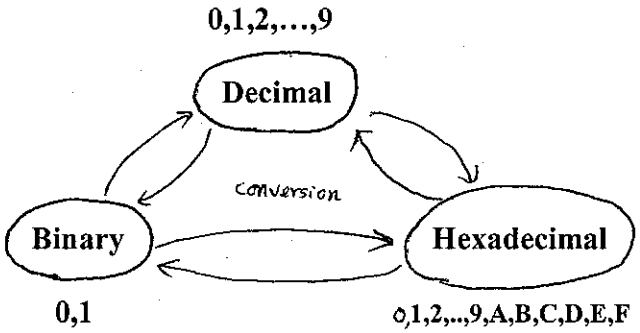
A. Number Systems



Example. 21

(1) Decimal → Binary

2) 21
2) 10 ---- 1
2) 5 ---- 0
2) 2 ---- 1
1 ---- 0

10101

(2) Binary → Decimal

$4\ 3\ 2\ 1\ 0$
10101_2

$2^4+2^2+2^0 = 16+4+1$
$= 21$

(3) Binary ↔ Hexadecimal

0001 0101 ⇔ 15_h

Note.

Decimal fraction → Binary fraction

Ex. 0.81

2×	0.81	
2×	1.62	
2×	1.24	
2×	0.48	
2×	0.96	
2×	1.92	
2×	1.84	
	:	0.110011…

B. Digital Logic

Boolean Algebra (George Boole, 1854):

Triple [K,•, +] consisting of a finite set of elements K, subject to an equivalence relation "=" and two binary operators, "+" and "•", such that for every element x and y ∈ K, the operations x+y and x•y are uniquely defined and the following postulates are satisfied.

1.	The operations are closed for all x and y ∈ K.	x+y ∈ K and x•y ∈ K	
2.	Commutative laws	$A \cdot B = B \cdot A$	$A + B = B + A$
3.	Distributive laws	$A \cdot (B+C) = (A \cdot B) + (A \cdot C)$	$A + (B \cdot C) = (A+B) \cdot (A+C)$
4.	Associative laws	$A \cdot (B \cdot C) = (A \cdot B) \cdot C$	$A + (B+C) = (A+B) + C$
5.	∃ Identity elements	$1 \cdot A = A$	$0 + A = A$
6.	Inverse elements	$A \cdot \bar{A} = 0$	$A + \bar{A} = 1$

"Duality"

Note: When K = {0,1}, it is called 'switching algebra'.

- Logic variables - true (1) or false (0)
- operations

AND	OR	NOT
(∩	∪	~)
(•	+	-)

Boolean expression (Boolean function)

1. 0,1,x,$\bar{x}$ are Boolean expression.
2. If A and B are Boolean expression, then $\bar{A}$,($\bar{A}$), A+B, A•B are also Boolean expression

Boolean Operators

P	Q	$\bar{P}$	P•Q	P+Q	P xor Q	P nand Q	P nor Q
0	0	1	0	0	0	1	1
0	1	1	0	1	1	1	0
1	0	0	0	1	1	1	0
1	1	0	1	1	0	0	0

TABLE 5 Boolean Identities.

Identity	*Name*
$\overline{\overline{x}} = x$	Law of the double complement
$x + x = x$ $x \cdot x = x$	Idempotent laws
$x + 0 = x$ $x \cdot 1 = x$	Identity laws
$x + 1 = 1$ $x \cdot 0 = 0$	Domination laws
$x + y = y + x$ $xy = yx$	Commutative laws
$x + (y + z) = (x + y) + z$ $x(yz) = (xy)x$	Associative laws
$x + yz = (x + y)(x + z)$ $x(y + z) = xy + xz$	Distributive laws
$\overline{(xy)} = \overline{x} + \overline{y}$ $\overline{(x + y)} = \overline{x}\,\overline{y}$	De Morgan's laws
$x + xy = x$ $x(x + y) = x$	Absorption laws
$x + \overline{x} = 1$	Unit property
$x\overline{x} = 0$	Zero property

Solution: The verification of this identity is shown in Table 6. The identity holds because the last two columns of the table agree. ◀

Identities in Boolean algebra can be used to prove further identities. We demonstrate this in Example 7.

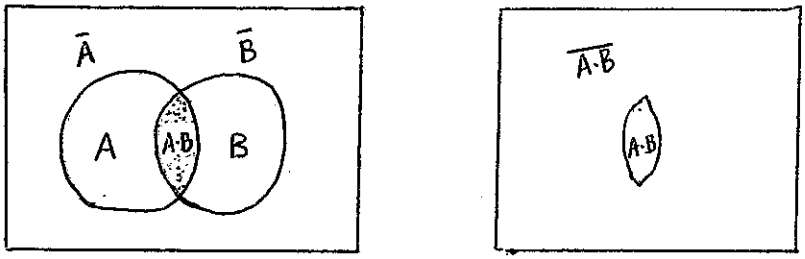
TABLE 6

x	y	z	$y + z$	xy	xz	$x(y + z)$	$xy + xz$
1	1	1	1	1	1	1	1
1	1	0	1	1	0	1	1
1	0	1	1	0	1	1	1
1	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	0	1	1	0	0	0	0
0	0	0	0	0	0	0	0

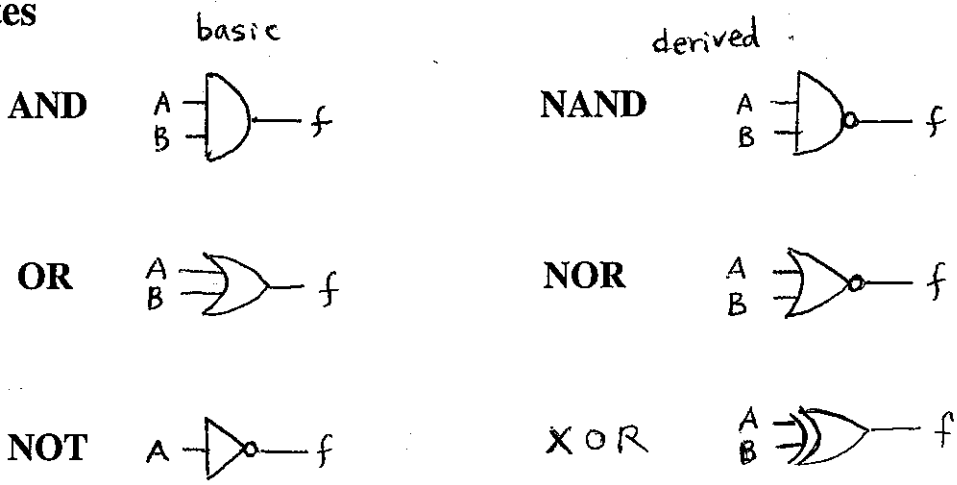
DeMorgan's Theorem

$$\overline{A \cdot B} = \overline{A} + \overline{B} \qquad \overline{A + B} = \overline{A} \cdot \overline{B}$$

Venn Diagram

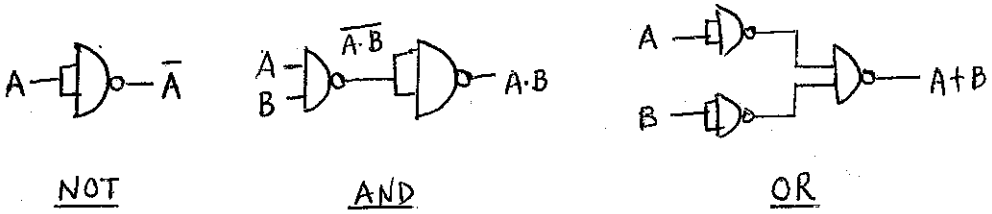


Gates



Functionally complete set of gates

1. **AND, OR, NOT**
2. **AND, NOT** $\quad A + B = \overline{\overline{A + B}} = \overline{\bar{A} \cdot \bar{B}}$
3. **OR, NOT**
4. **NAND**
5. **NOR**



Combinational Circuit — output depends on input only

Sequential Circuit — output depends on input and past history of input (memory)

Implementation of Boolean Functions

Ex.

	A	B	C	F
0	0	0	0	0
1	0	0	1	0
2	0	1	0	1 ✓
3	0	1	1	1 ✓
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1 ✓
7	1	1	1	0

$\bigvee_{i=0}^{7}$ $F(A,B,C) = \Sigma(2,3,6)$ // sigma notation //

↕ dual

$\bigwedge_{i=0}^{7}$ $F(A,B,C) = \Pi(0,1,4,5,7)$ // pi notation //

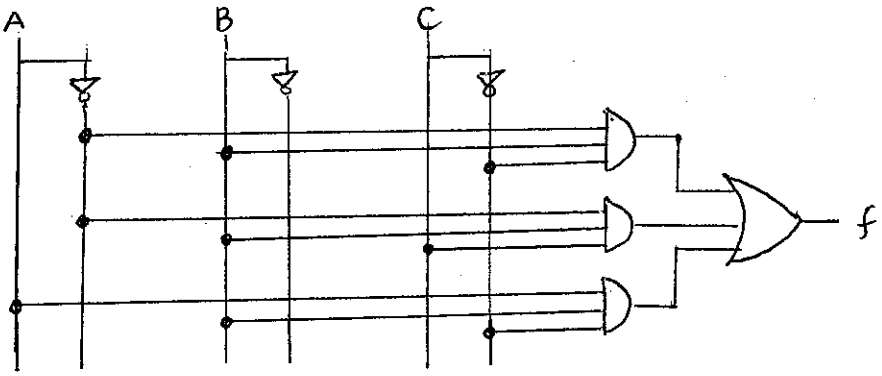
characteristic number $= (01001100)_2$

$= 76$

Canonical Expression

(1) SOP (sum-of-product) form

$$f = \bar{A}\cdot B\cdot\bar{C} + \bar{A}\cdot B\cdot C + A\cdot B\cdot\bar{C}$$



(2) POS (product-of-sum) form

$$f = \overline{(\bar{A}\,\bar{B}\,\bar{C})}\cdot\overline{(\bar{A}\,\bar{B}\cdot C)}\;\overline{(A\,\bar{B}\,\bar{C})}\;\overline{(A\,\bar{B}\,C)}\;\overline{(A\,B\,C)}$$

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

"Equivalent"

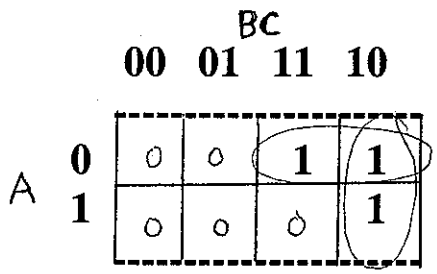
Simplification

Ex. $f = \bar{A}\,B\,\bar{C} + \bar{A}\,B\,C + A\,B\,\bar{C}$

(1) Algebraic Simplification

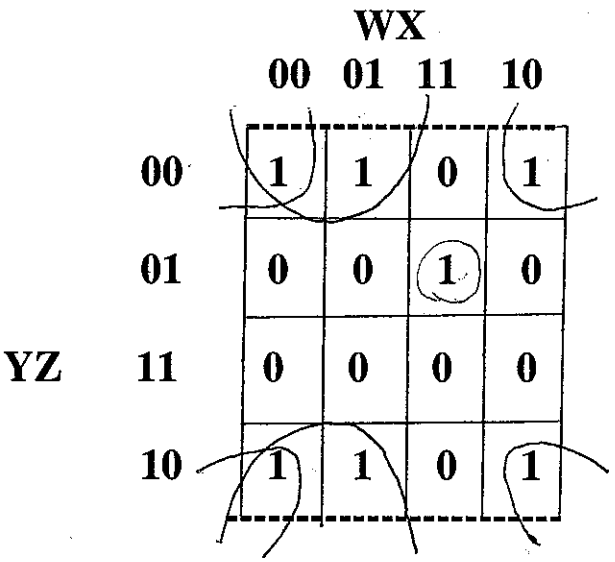
$$f = \bar{A}\,B\,\bar{C} + \bar{A}\,B\,C + A\,B\,\bar{C}$$
$$= \bar{A}\,B\,(\bar{C} + C) + A\,B\,\bar{C}$$
$$= \bar{A}\,B + A\,B\,\bar{C} = \underbrace{\bar{A}\,B + \bar{A}\,B\,\bar{C}}_{\text{absorption}} + A\,B\,\bar{C}$$
$$= \bar{A}\,B + B\,\bar{C}$$
$$(\ = B\,(\bar{A} + \bar{C})\ )$$

(2) Karnaugh Map (K-map)



$$f = \bar{A}\,B + B\,\bar{C}$$

Ex



$$\bar{W}\bar{Z} + \bar{X}\bar{Z} + WX\bar{Y}Z$$

Ex.

Table B.4 Truth Table for the One-Digit Packed Decimal Incrementer

	Input					Output			
Number	A	B	C	D	Number	W	X	Y	Z
0	0	0	0	0	1	0	0	0	1
1	0	0	0	1	2	0	0	1	0
2	0	0	1	0	3	0	0	1	1
3	0	0	1	1	4	0	1	0	0
4	0	1	0	0	5	0	1	0	1
5	0	1	0	1	6	0	1	1	0
6	0	1	1	0	7	0	1	1	1
7	0	1	1	1	8	1	0	0	0
8	1	0	0	0	9	1	0	0	1
9	1	0	0	1	0	0	0	0	0
Don't care condition	1	0	1	0		d	d	d	d
	1	0	1	1		d	d	d	d
	1	1	0	0		d	d	d	d
	1	1	0	1		d	d	d	d
	1	1	1	0		d	d	d	d
	1	1	1	1		d	d	d	d

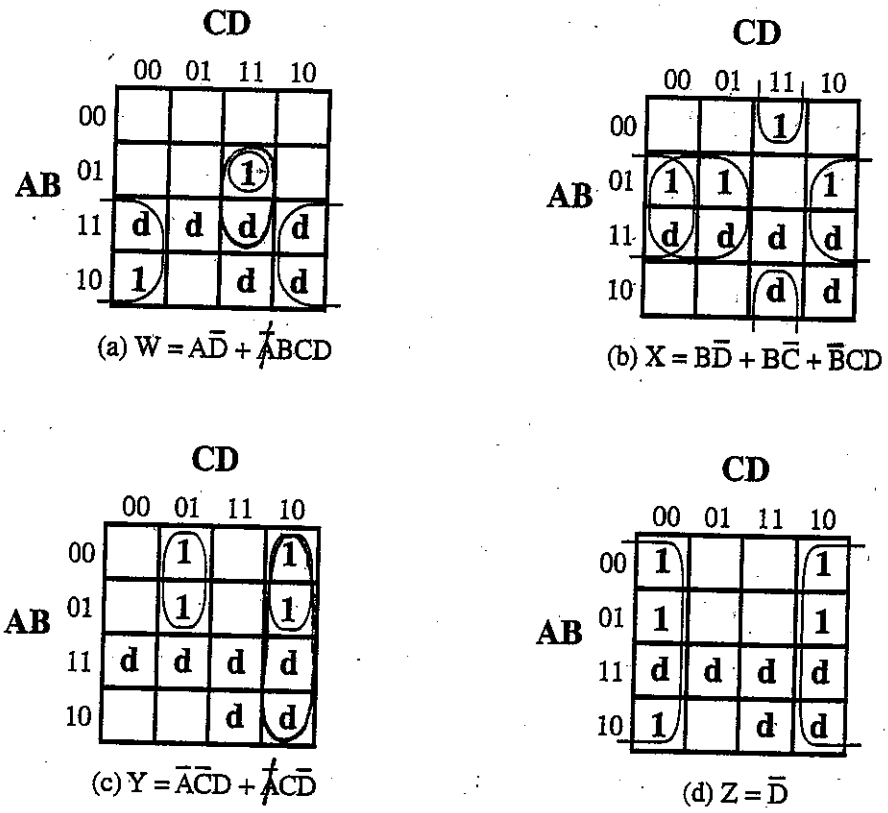


Figure B.10 Karnaugh Maps for the Incrementer

Quine-Mc Cluskey Method

$$F = A\bar{B}\bar{C}\bar{D} + AB\bar{C}\bar{D} + \bar{A}BC\bar{D} + AB\bar{C}D + A\bar{B}\bar{C}D + \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}\bar{C}\bar{D}$$

1000 1100 0110 1101 1001 0001 0000

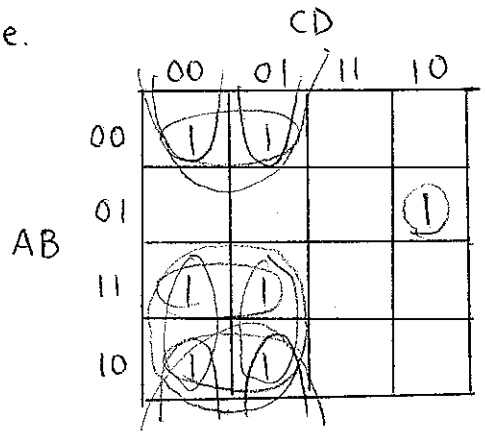
✓ 0000	✓ -000	(-00-)
✓ 1000 ✓	✓ 000-	
✓ 0001 ✓	✓ 1-00	(1-0-)
✓ 1100 ✓	✓ 100- ✓	
(0110)	-001 ✓	
1001 ✓	110- ✓	
1101 ✓	1-01 ✓	

Prime Implicants

	1000	1100	0110	1101	1001	0001	0000
(0110)			(X)				
(-00-)	[X]				[X]	[X]	(X)
(1-0-)	X	[X]		(X)	X		

Essential Prime Implicants: $F = \bar{A}BC\bar{D} + \bar{B}\bar{C} + A\bar{C}$

Note.



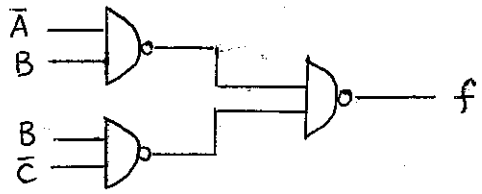
NAND (NOR) Implementation

$$f = B\cdot(\bar{A} + \bar{C})$$

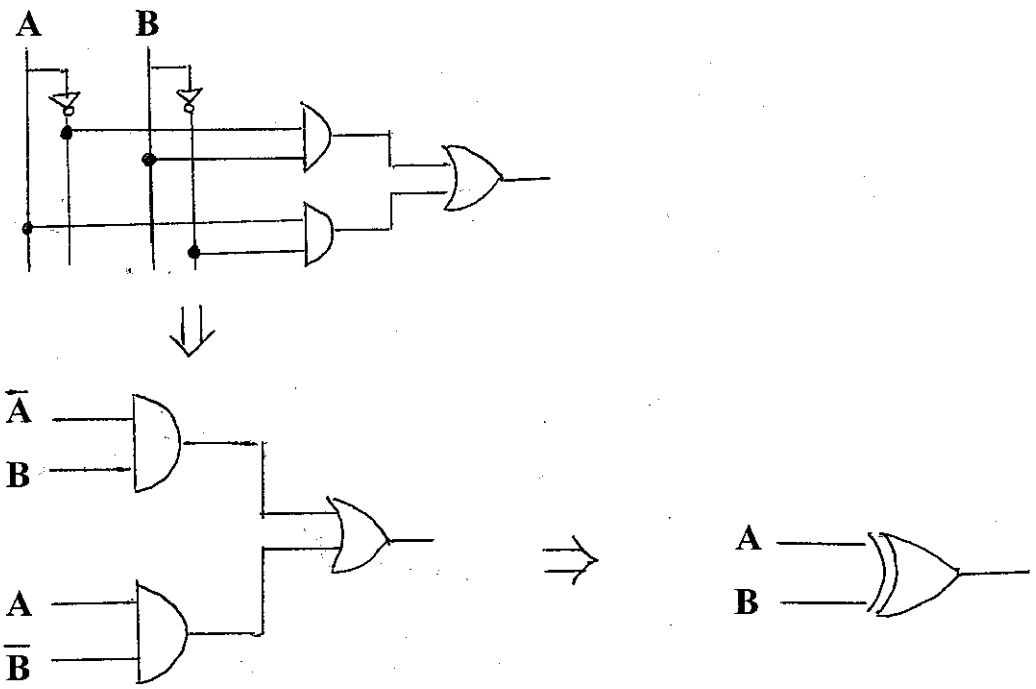
$$= \bar{A}\cdot B + B\cdot\bar{C}$$

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

$$= \overline{\overline{\bar{A}\cdot B}\cdot\overline{B\,\bar{C}}}$$



Exclusive-Or (XOR)



Multiplexers

– multiple input, one output: signal control, data routing

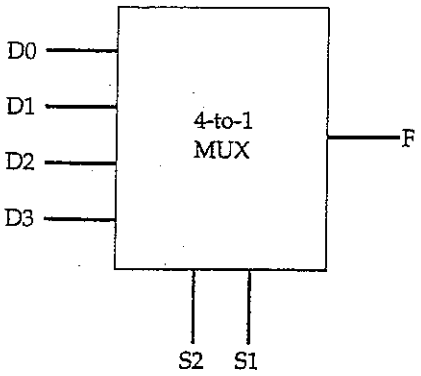


Figure B.12 4-to-1 Multiplexer Representation.

Table B.7 4-to-1 Multiplexer Truth Table

S2	S1	F
0	0	D0
0	1	D1
1	0	D2
1	1	D3

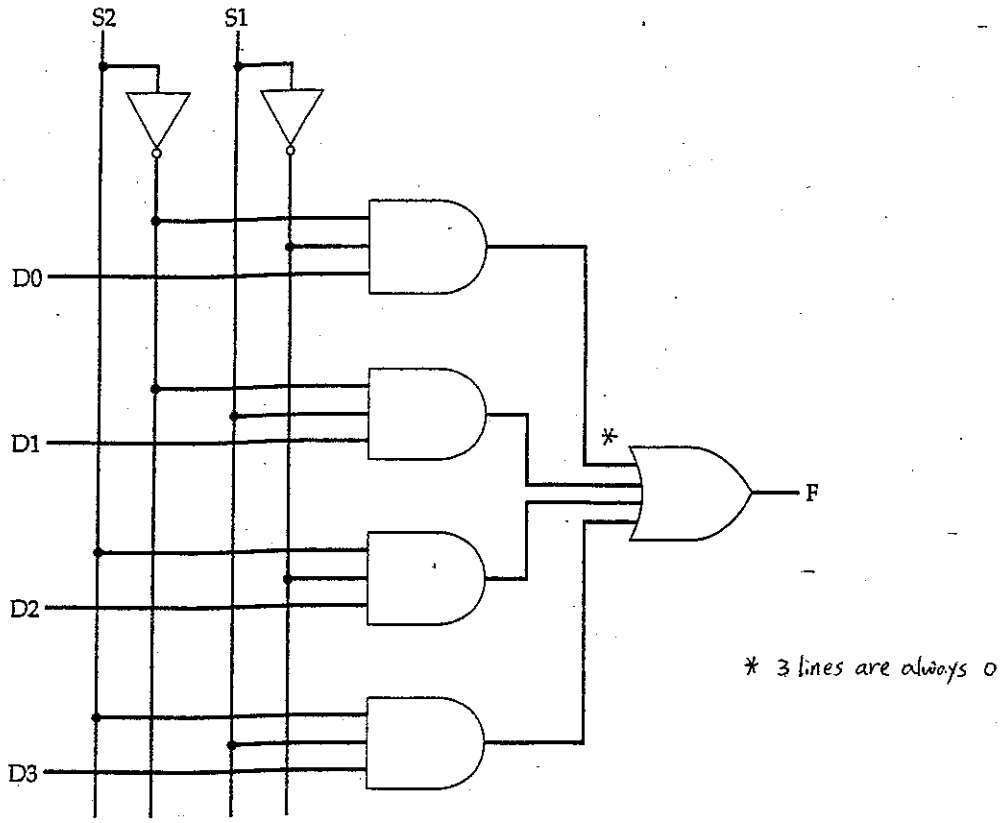


Figure B.13 Multiplexer Implementation.

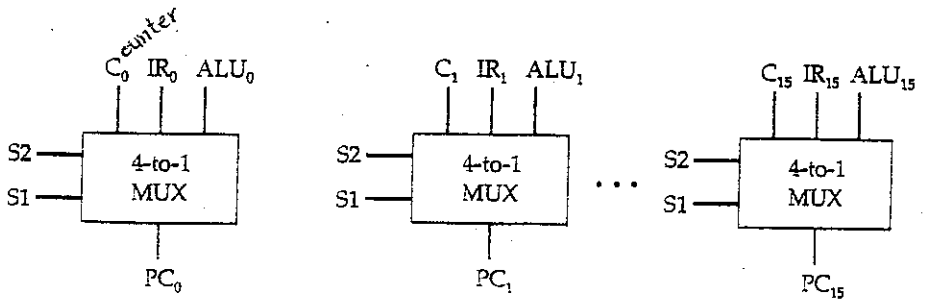


Figure B.14 Multiplexer Input to Program Counter.

Decoders

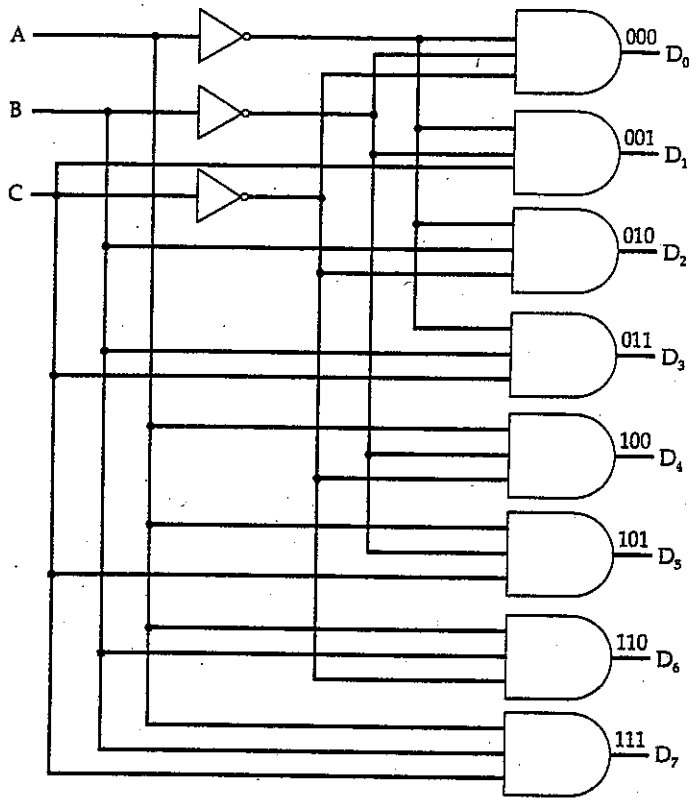


Figure B.15 Decoder with 3 Inputs and $2^3 = 8$ Outputs

Ex. Design 1 K byte memory with 4 256 byte RAM chips.

Address (MAR)	chip
00 00000000 — 00 11111111	0
01 00000000 — 01 11111111	1
10 00000000 — 10 11111111	2
11 00000000 — 11 11111111	3

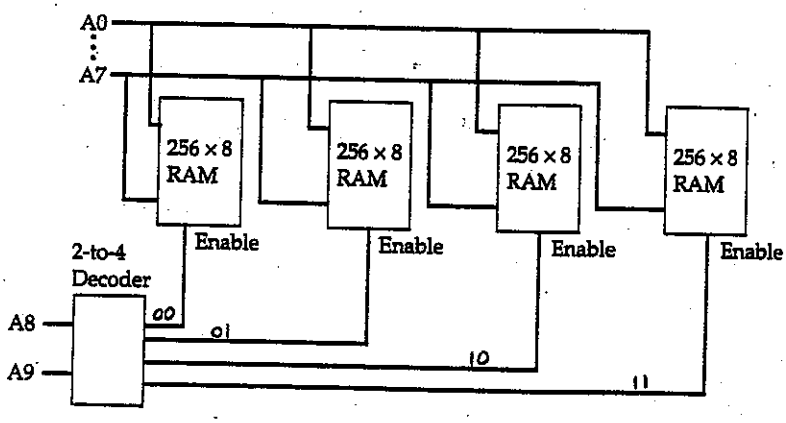


Figure B.16 Address Decoding

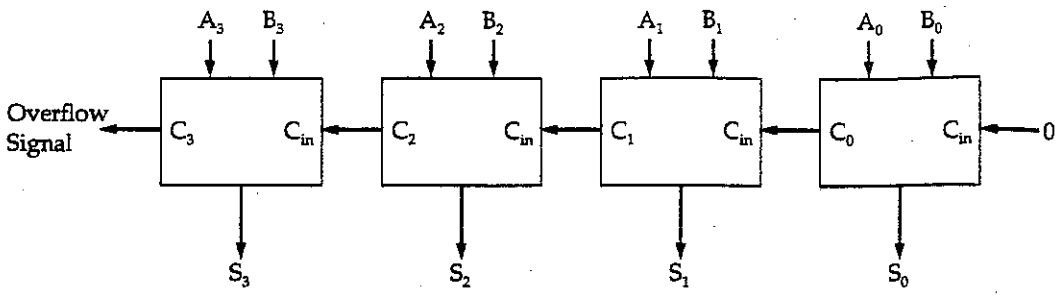
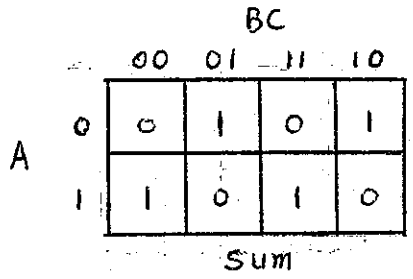
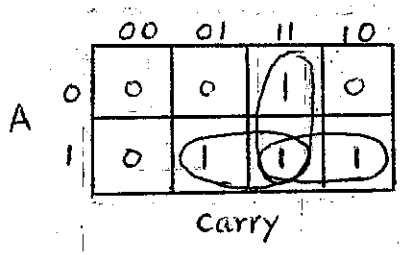


Figure 3.21 4-Bit Adder.

Binary Addition Truth Tables

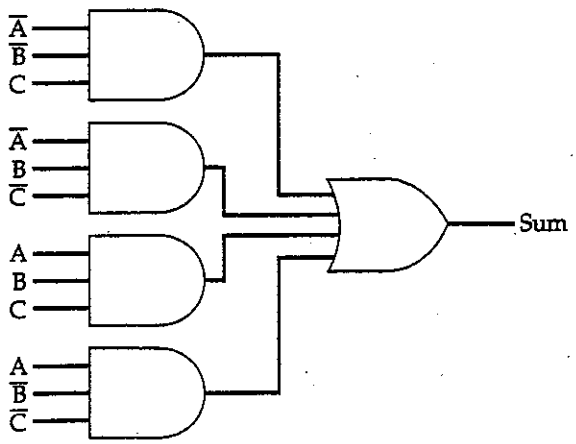
C_{in}	A	B	Sum	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

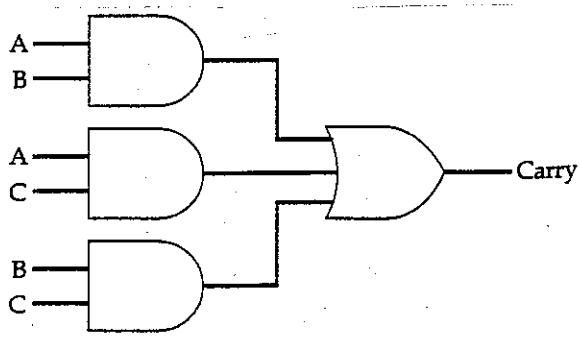




$$\text{Sum} = \overline{A}\overline{B}C + \overline{A}B\overline{C} + ABC + A\overline{B}\overline{C}$$

$$\text{Carry} = AB + AC + BC$$





Carry Lookahead

$$C_0 = A_0 B_0$$

$$C_1 = A_1 B_1 + C_0 (A_1 + B_1)$$

$$= A_1 B_1 + A_0 B_0 (A_1 + B_1)$$

$$= A_1 B_1 + A_0 B_0 A_1 + A_0 B_0 B_1$$

$$C_2 = A_2 B_2 + A_2 A_1 B_1 + A_2 A_1 A_0 B_0 + A_2 B_1 A_0 B_0 + B_2 A_1 B_1 + B_2 A_1 A_0 B_0 + B_2 A_0 B_0 B_1$$

:

$$C_8 = A_8 B_8 + (A_8 + B_8) C_7$$

In general, carry lookahead is done only 4 to 8 bits at a time.

Ex. 32-bit adder using 8-bit adder

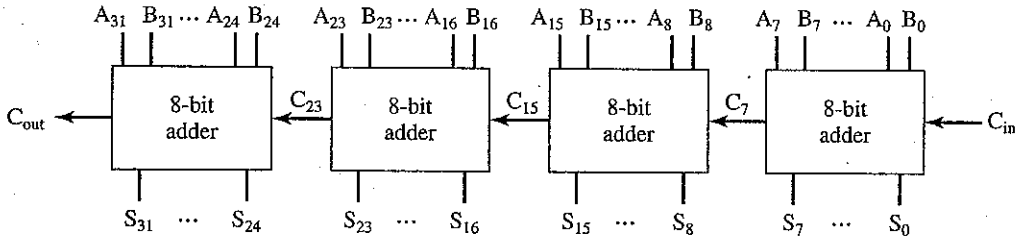
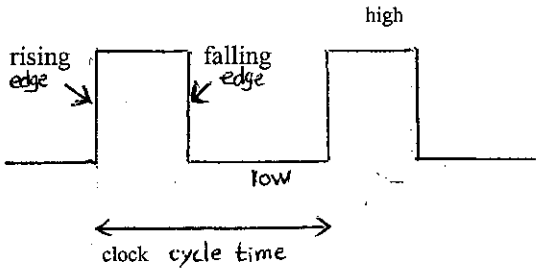


Figure B.23 Construction of a 32-Bit Adder Using 8-Bit Adders

Clocks



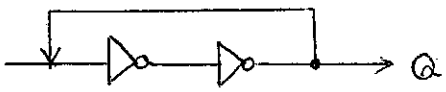
Event (change of state)

- edge-triggered
  - rising edge
  - falling edge ✓
- level-triggerd
  - low
  - high

Feedback

- Output is fed back as an input to the same circuit.

Example.



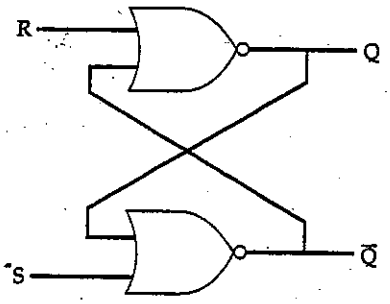
Sequential Circuit

- output depends on current input + past history of inputs

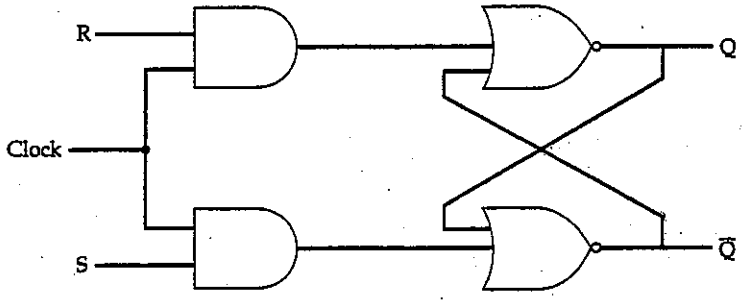
Flip-flop (Latch)

- bistable state (0 or 1) → 1 bit memory
- two outputs, Q and $\overline{Q}$

The S–R Latch



Asynchronous



Synchronous

Table 8.10 The S–R Latch

(a) Characteristic Table

Current Inputs	Current State	Next State
SR	Q_n	Q_{n+1}
00	0	0
00	1	1
01	0	0
01	1	0
10	0	1
10	1	1
11	0	—
11	1	—

⟹

(b) Simplified Characteristic Table

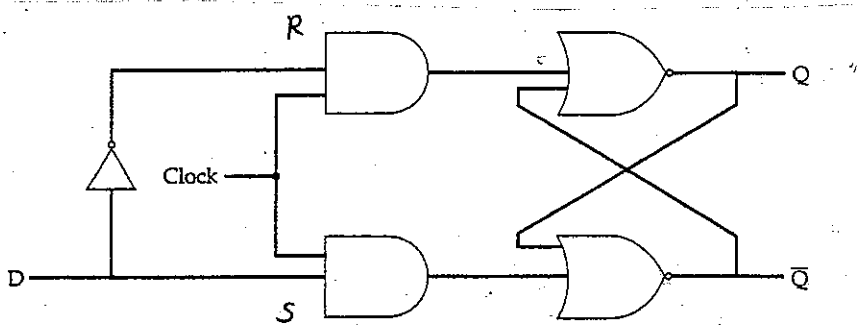
S	R	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	—

(c) Response to Series of Inputs

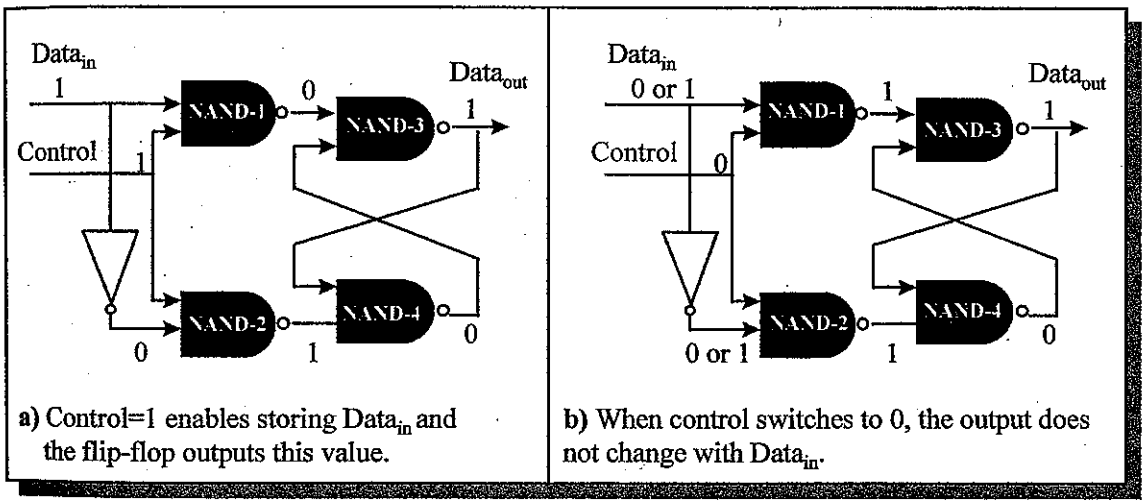
t	0	1	2	3	4	5	6	7	8	9
S	1	0	0	0	0	0	0	0	1	0
R	0	0	0	1	0	0	1	0	0	0
Q_{n+1}	1	1	1	0	0	0	0	0	1	1

D Flip-flop

- modification of S-R flip-flop
- data flip-flop, delay flip-flop



D	Q_{n+1}
0	0
1	1

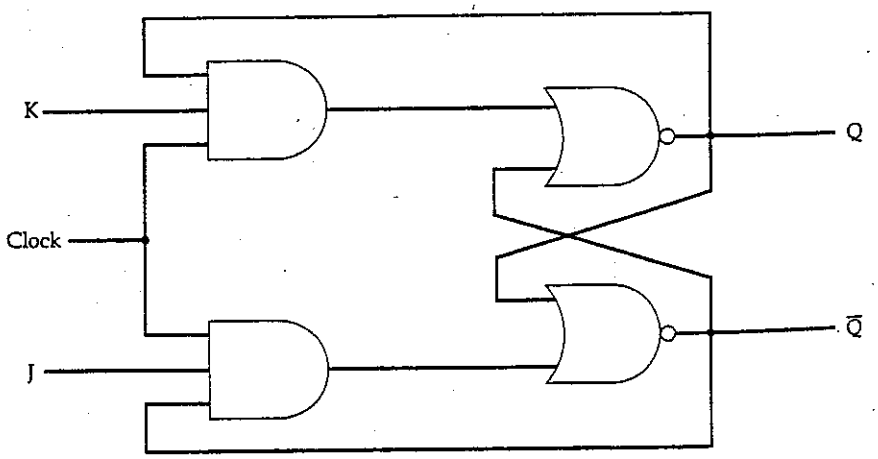


a) Control=1 enables storing $Data_{in}$ and the flip-flop outputs this value.

b) When control switches to 0, the output does not change with $Data_{in}$.

J-K Flip-flop

- universal flip-flop

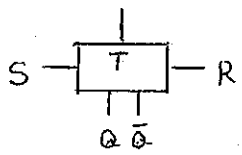


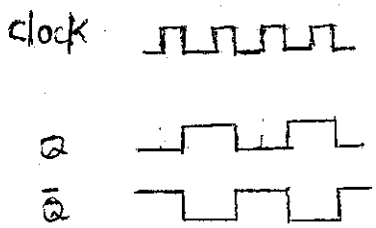
J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q}_{n+1}$

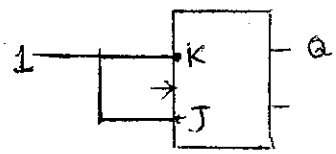
toggle

Note. T flip-flop (toggle)

- counting circut







Implementation

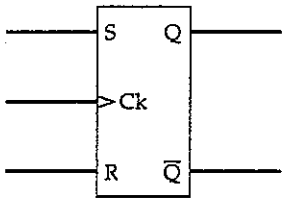
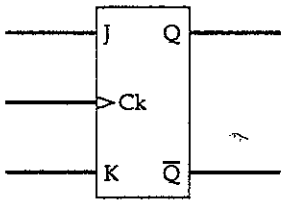
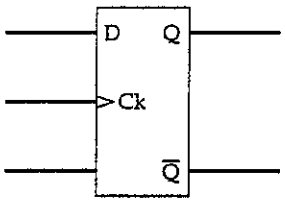
Name	Graphic Symbol	Characteristic Table
S–R		S R Q_{n+1} 0 0 Q_n 0 1 0 1 0 1 1 1 –
J–K		J K Q_{n+1} 0 0 Q_n 0 1 0 1 0 1 1 1 $\overline{Q_n}$
D		D Q_{n+1} 0 0 1 1

Figure B.29 Basic Flip-Flops

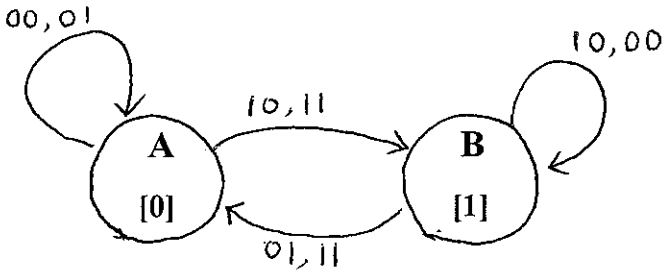
Finite State Machine (FSM)

- **Moore Machine**
  - **Edward Moore (1956)**
  - **Each state is associated with output**

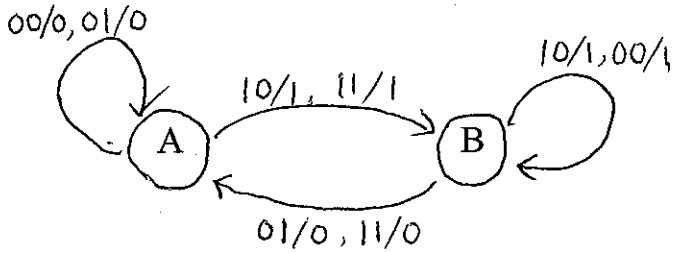
- **Mealy Machine**
  - **George Mealy (1955)**
  - **Each transition is associated with output**

Example. JK Flip-Flop

Moore



Mealy



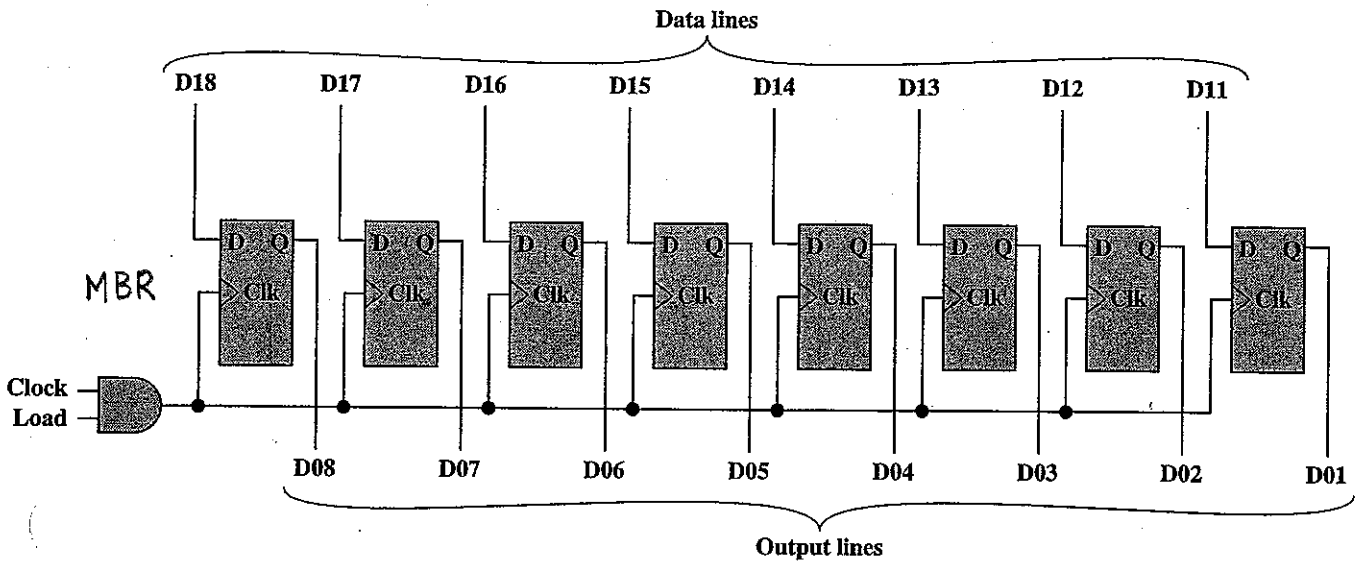


Figure B.30 8-Bit Parallel Register

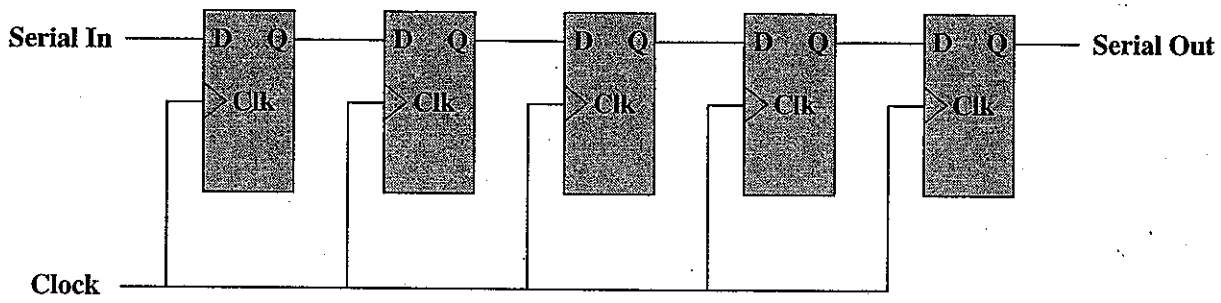
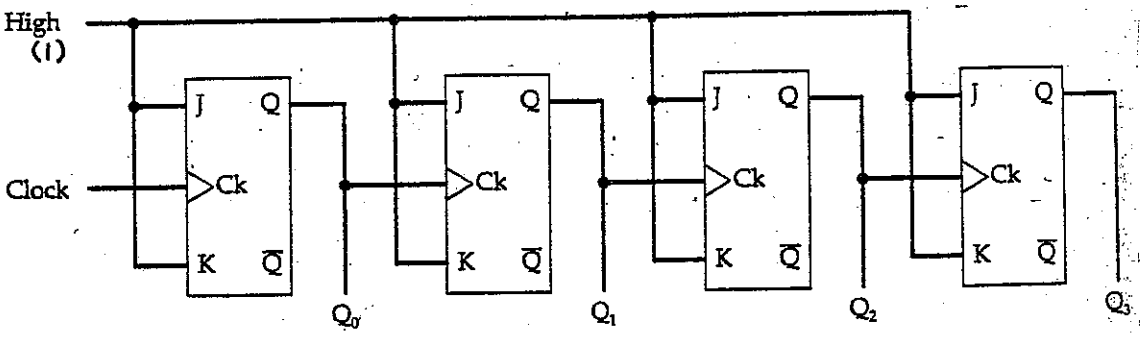
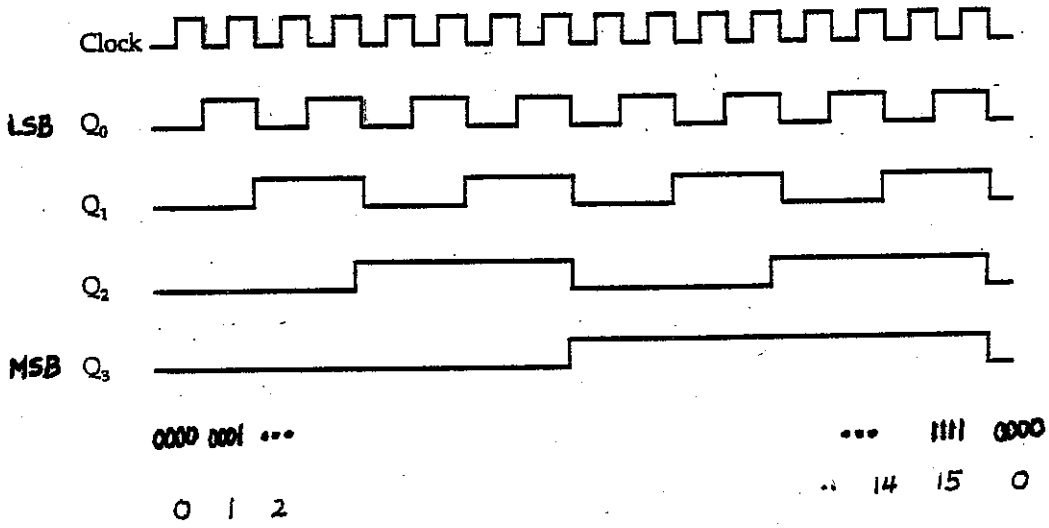


Figure B.31 5-Bit Shift Register

(1) Ripple Counter (Asynchronous Counter)

4-bit counter (Modulo 16 counter)





Note edge-trigger (falling-edge → change)
By using reset, we can create any pulse train, such as $\overbrace{0\text{-}1\text{-}2}\text{-}0\text{-}1\text{-}2\cdots$

Ex 3-bit counter : $0-1-2-3-4-5-6-7$

J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\bar{Q}_n$

Characteristic Table

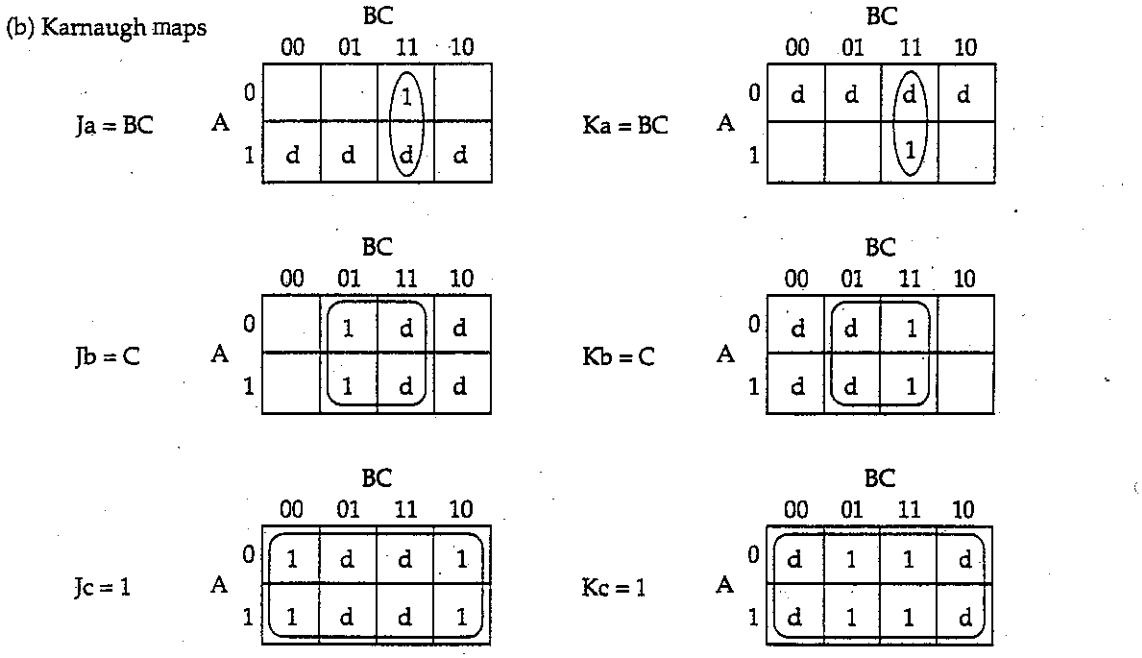
Q_n	J	K	Q_{n+1}
0	0	d	0
0	1	d	1
1	d	1	0
1	d	0	1

Excitation Table

(a) Truth table

A	B	C	Ja	Ka	Jb	Kb	Jc	Kc
0	0	0	0	d	0	d	1	d
0	0	1	0	d	1	d	d	1
0	1	0	0	d	d	0	1	d
0	1	1	1	d	d	1	d	1
1	0	0	d	0	0	d	1	d
1	0	1	d	0	1	d	d	1
1	1	0	d	0	d	0	1	d
1	1	1	d	1	d	1	d	1

(b) Karnaugh maps



(c) Logic diagram

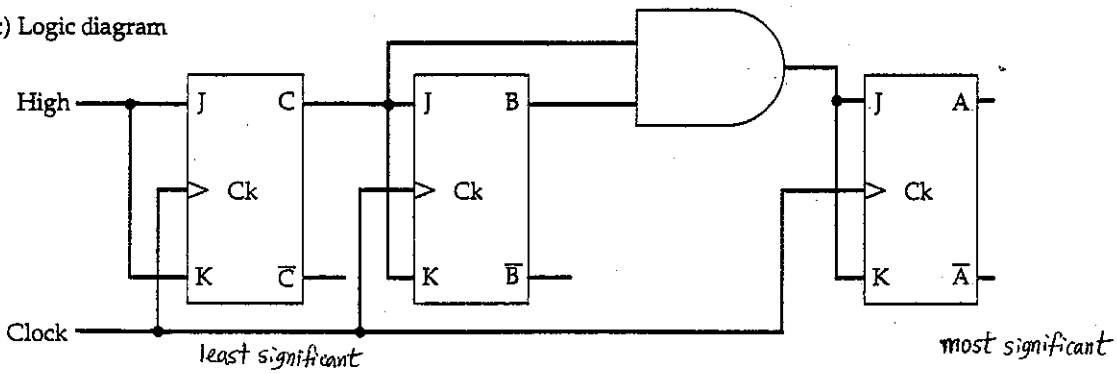


Figure 8.33 Design of a Synchronous Counter