Binary Systems

Cont …

Dr. Bassem A. Abdullah

Computer and Systems Department

Lectures Prepared by Dr.Mona Safar, Edited and Lectured by Dr.Bassem A. Abdullah

Outline

1. Digital Systems
2. Binary Numbers
3. Number Base Conversions
4. Octal and Hexadecimal Numbers
5. Complements
6. Signed Binary Numbers
7. Binary Codes
8. Binary Storage and Resgisters
9. Binary Logic

Table 1.2
Numbers with Different Bases

Decimal (base 10)	Binary (base 2)	Octal (base 8)	Hexadecimal (base 16)
00	0000	00	0
01	0001	01	1
02	0010	02	2
03	0011	03	3
04	0100	04	4
05	0101	05	5
06	0110	06	6
07	0111	07	7
08	1000	10	8
09	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

n bits can represent **2^n unsigned** integers from **0 to 2^n - 1**.
To represent value **x**, then **ceil($\log_2$ x)** bits are needed

Table 1.1
Powers of Two

n	2^n	*n*	2^n	*n*	2^n
0	1	8	256	16	65,536
1	2	9	512	17	131,072
2	4	10	1,024	18	262,144
3	8	11	2,048	19	524,288
4	16	12	4,096	20	1,048,576
5	32	13	8,192	21	2,097,152
6	64	14	16,384	22	4,194,304
7	128	15	32,768	23	8,388,608

Table 1.3
Signed Binary Numbers

Decimal	Signed-2's Complement	Signed-1's Complement	Signed Magnitude
+7	0111	0111	0111
+6	0110	0110	0110
+5	0101	0101	0101
+4	0100	0100	0100
+3	0011	0011	0011
+2	0010	0010	0010
+1	0001	0001	0001
+0	0000	0000	0000
−0	—	1111	1000
−1	1111	1110	1001
−2	1110	1101	1010
−3	1101	1100	1011
−4	1100	1011	1100
−5	1011	1010	1101
−6	1010	1001	1110
−7	1001	1000	1111
−8	1000	—	—

Example

- Using 2's complement, Given X= 1010100, Y= 1000011 subtract:

  a) X-Y,

X =	1010100
2's complement of Y =	+ 0111101
Sum =	10010001
Discard end carry 2^7 =	−10000000
Answer: $X - Y$ =	0010001

Example

- b) Y- X

$$
\begin{aligned}
Y &= \quad 1000011 \\
\text{2's complement of } X &= +\ \underline{0101100} \\
\text{Sum} &= \quad 1101111
\end{aligned}
$$

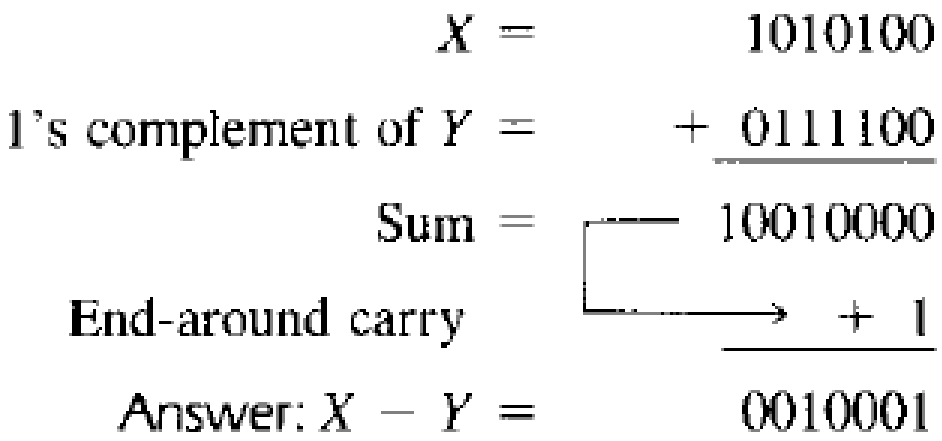
There is no end carry.

Answer: $Y - X = -(\text{2's complement of } 1101111) = -0010001$

Example

- Using 1's complement, Given X= 1010100, Y= 1000011 subtract:

  a) X-Y,



Example

- b) Y- X

$$
\begin{aligned}
Y &= \quad 1000011 \\
\text{1's complement of } X &= +\ \underline{0101011} \\
\text{Sum} &= \quad 1101110
\end{aligned}
$$

There is no end carry.

Answer: $Y - X = -(\text{1's complement of } 1101110) = -0010001$

Binary Codes

- BCD (Binary Coded Decimal)

Decimal	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Why BCD?

- A decimal number n represented in K decimal digits requires 4*k bits
- Example:
  - $185 = (0001\ 1000\ 0101)_{BCD} = (10111001)_2$

BCD Addition

4	0100
+5	+0101
9	1001

4	0100
+8	+1000
12	1100>9
	1100
	+0110 6
1	0010

8	1000
+9	+1001
17	10001>9
	10001
	+0110 6
1	0111

BCD Addition

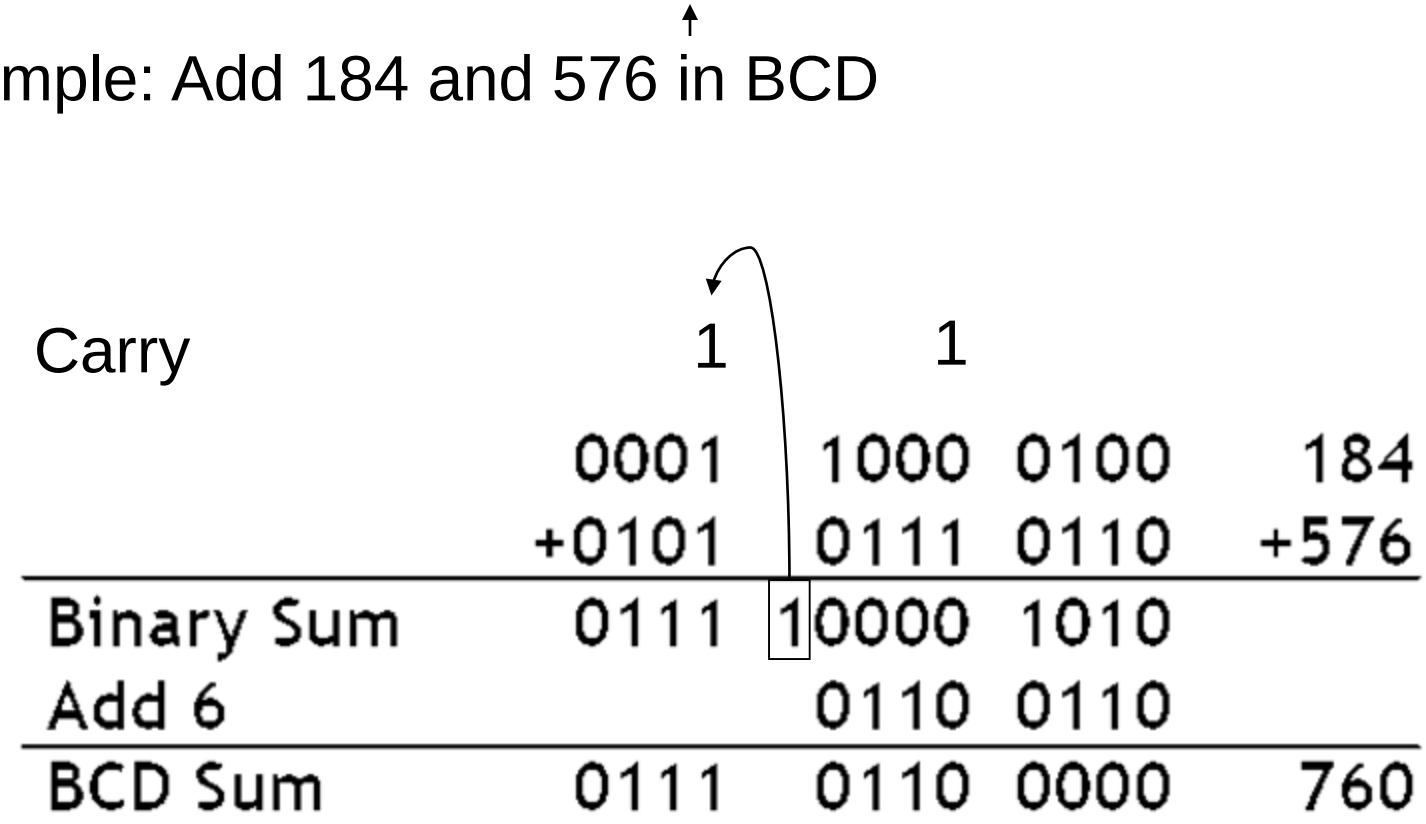
- Rule: we add 6 (0110) if result > 9

Decimal	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	0001 0000
11	0001 0001
12	0001 0010
13	0001 0011
14	0001 0100
15	0001 0101

+6

BCD Addition

Example: Add 184 and 576 in BCD



Carry	1		1	
	0001	1000	0100	184
	+0101	0111	0110	+576
Binary Sum	0111	10000	1010	
Add 6		0110	0110	
BCD Sum	0111	0110	0000	760

Binary Codes for Decimal Digits

Table 1.5
Four Different Binary Codes for the Decimal Digits

Decimal Digit	BCD 8421	2421	Excess-3	8, 4, −2, −1
0	0000	0000	0011	0000
1	0001	0001	0100	0111
2	0010	0010	0101	0110
3	0011	0011	0110	0101
4	0100	0100	0111	0100
5	0101	1011	1000	1011
6	0110	1100	1001	1010
7	0111	1101	1010	1001
8	1000	1110	1011	1000
9	1001	1111	1100	1111
Unused bit combi-nations	1010	0101	0000	0001
	1011	0110	0001	0010
	1100	0111	0010	0011
	1101	1000	1101	1100
	1110	1001	1110	1101
	1111	1010	1111	1110

Self Complementing Code
Can directly get 9's complement

- **Weighted code:**
  - Each binary digit is assigned a weight.
  - For each group of bits corresponding to decimal no., the sum of those bits = the decimal no.
  - BCD(8421), 2421, 84-2-1
  - Excess code is unweighted code
- **Self-complementing:**
  - The sum of the weights = 9
  - Can directly get 9’s complement
  - BCD is not self-complementing

Gray Code

Table 1.6
Gray Code

Gray Code	Decimal Equivalent
0000	0
0001	1
0011	2
0010	3
0110	4
0111	5
0101	6
0100	7
1100	8
1101	9
1111	10
1110	11
1010	12
1011	13
1001	14
1000	15

Only one bit changes from one number to another.

7 → 8

In Binary

0111 → 1000 (4 bits change)

In Gray code

0100 → 1100 (1 bit changes)

Representing Characters
ASCII Code

- ASCII (American Standard Code for Information Interchange) codes are used to represent characters in a computer system.
- Each character we wish to use must be assigned a unique binary code or number to distinguish it from all other characters.
- The characters to be represented include:

  upper-case (A-Z), lower-case(a-z), numerals(0-9), punctuation(, . ; : etc.) and control characters (non-printing, e.g. Esc)
- ASCII is a 7-bit code.

Part of ASCII Code

ASCII	Hex	Symbol
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O

ASCII	Hex	Symbol
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5F	_

ASCII	Hex	Symbol
96	60	`
97	61	a
98	62	b
99	63	c
100	64	d
101	65	e
102	66	f
103	67	g
104	68	h
105	69	i
106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o

ASCII	Hex	Symbol
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	▯

Representing Characters Unicode

- 16-bit standard code for representing characters.
- Unicode allows more than 65,000 characters to be represented and so is sufficient for the alphabet of almost any language (Chinese or Japanese).
- The first 128 Unicode codes correspond to the standard ASCII codes.

Error Detection Code
Even and Odd Parity

Odd parity		Even parity	
Message	P	Message	P
0000	1	0000	0
0001	0	0001	1
0010	0	0010	1
0011	1	0011	0
0100	0	0100	1
0101	1	0101	0
0110	1	0110	0
0111	0	0111	1
1000	0	1000	1
1001	1	1001	0
1010	1	1010	0
1011	0	1011	1
1100	1	1100	0
1101	0	1101	1
1110	0	1110	1
1111	1	1111	0

Example

	Even Parity	Odd Parity
ASCII A=1000001	0 1000001	1 1000001
ASCII T=1010100	1 1010100	0 1010100

What is a Register?

- A register is a group of binary cells
- A register with n cells can store n-bit information
- A register with n cells can be in one of 2^n possible states (0 to 2^n -1)
- A 16-bit example: 1100 0011 1100 1001

Register Transfer

- A register transfer operation is a basic operation in digital systems.
- It transfers binary information from one set of registers to another set of registers.

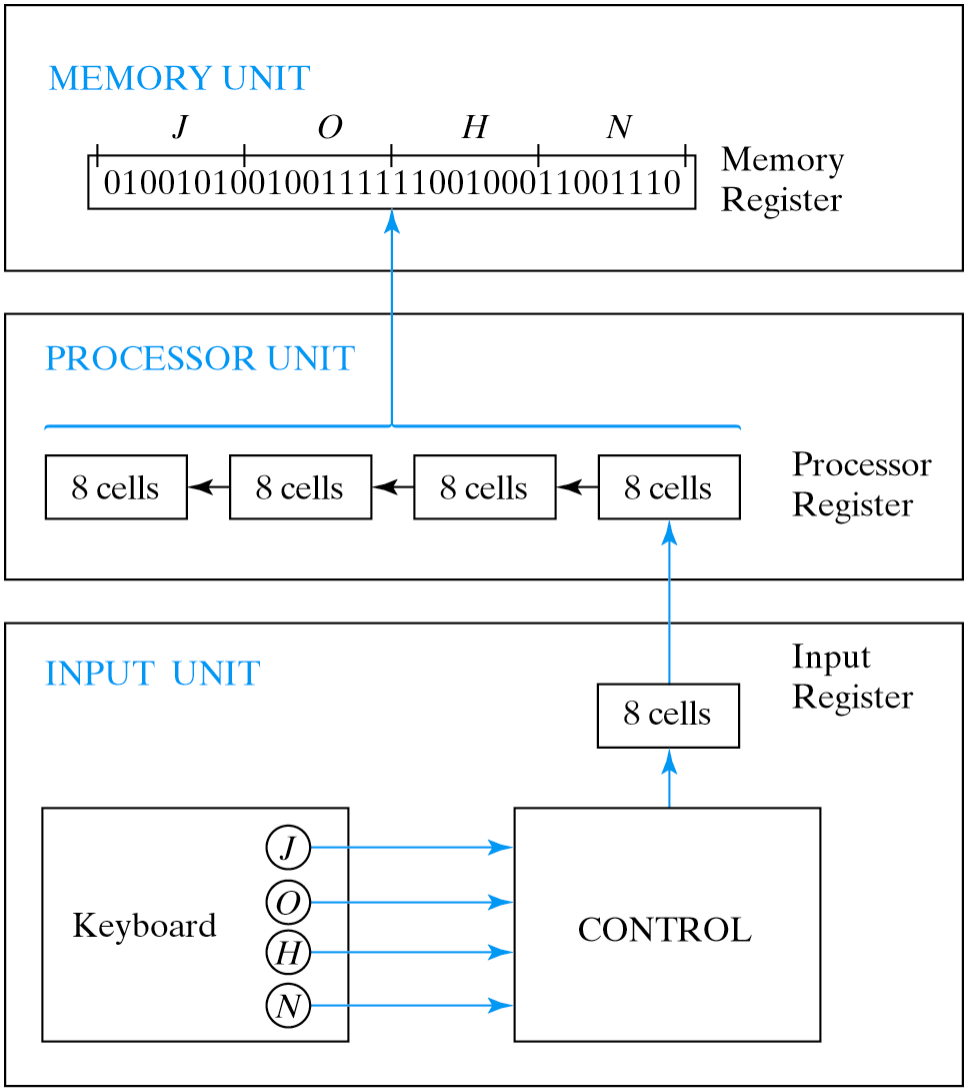


Fig. 1-1 Transfer of information with registers

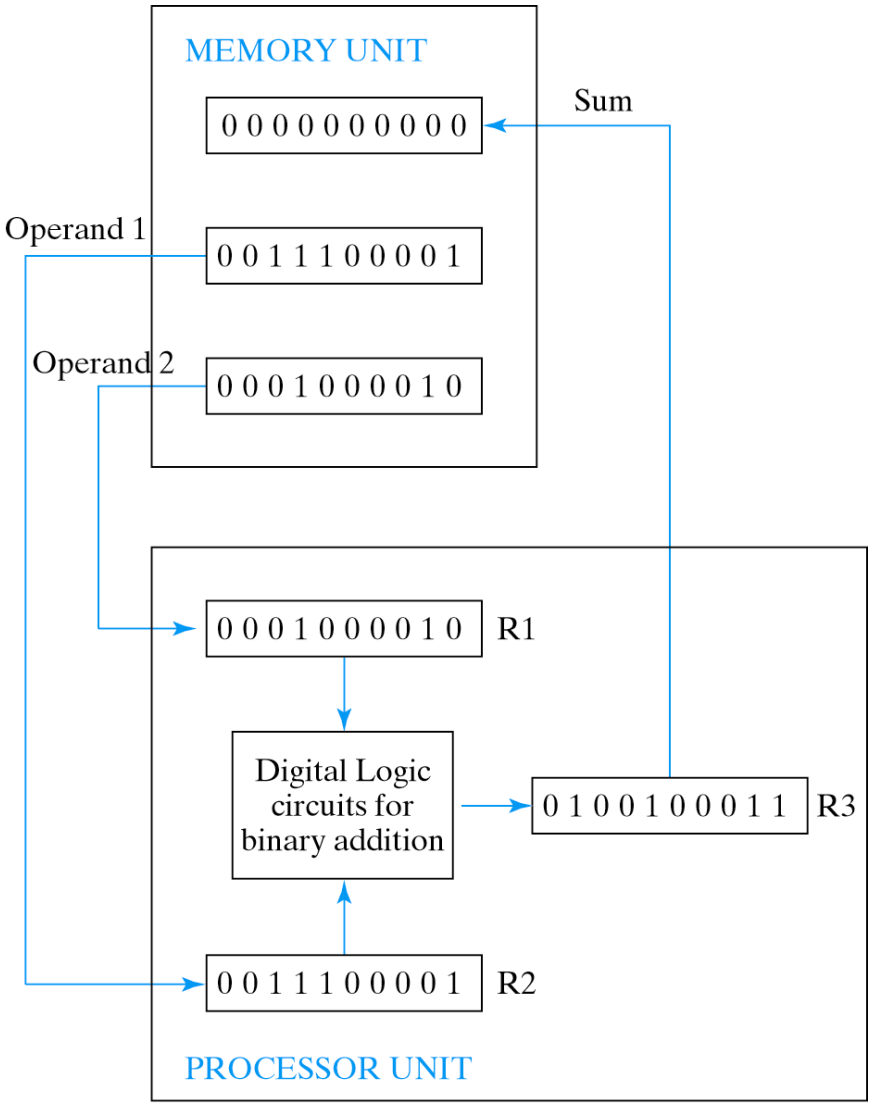


Fig. 1-2 Example of binary information processing

Binary Logic

- 0/1
- Flase/True
- Low/High

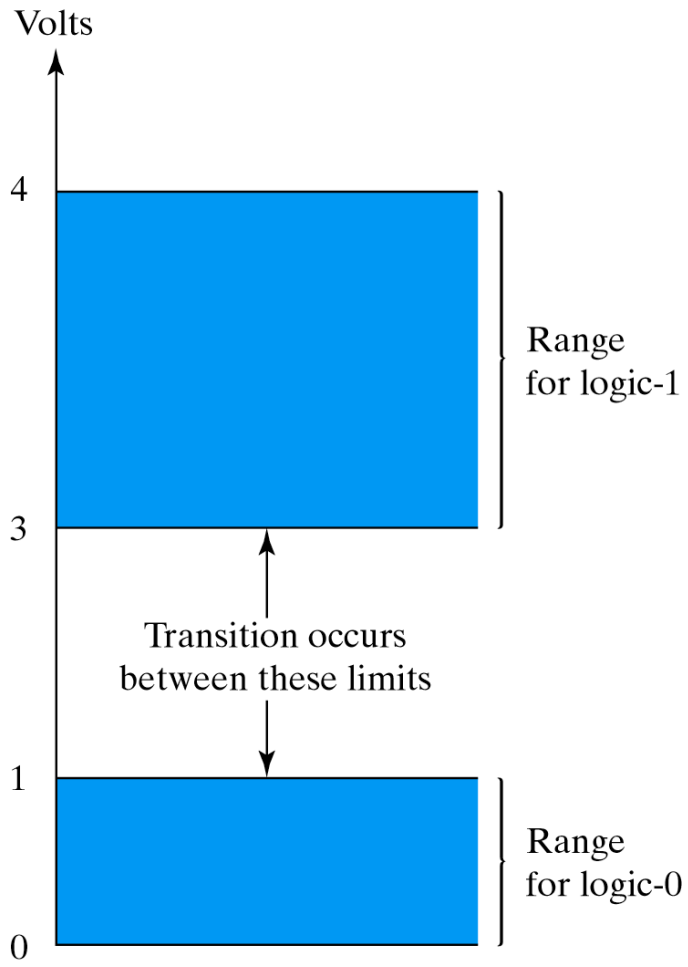


Fig. 1-3 Example of binary signals

Basic Binary Operations

Table 1.8

Truth Tables of Logical Operations

AND

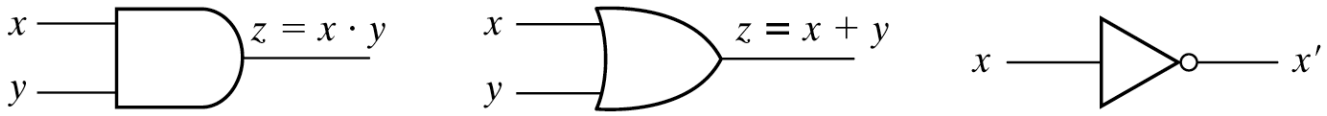
x	y	$x \cdot y$
0	0	0
0	1	0
1	0	0
1	1	1

OR

x	y	$x + y$
0	0	0
0	1	1
1	0	1
1	1	1

NOT

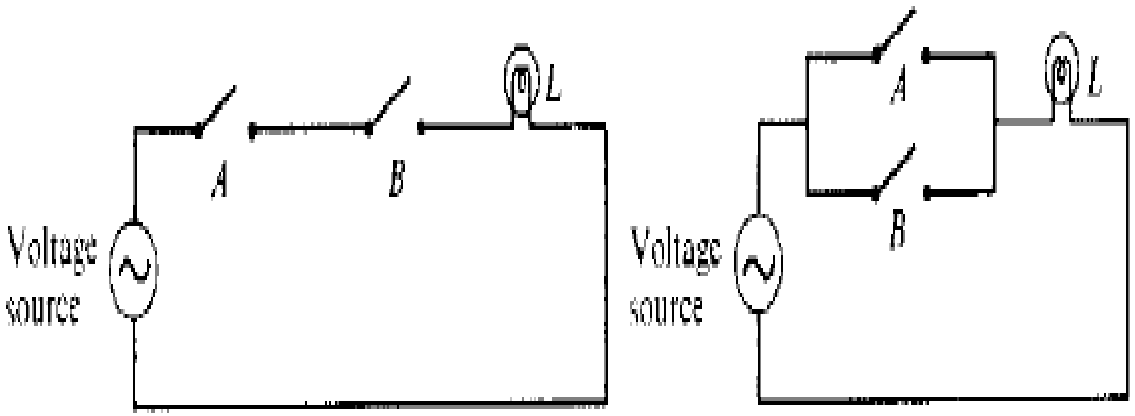
x	x'
0	1
1	0



(a) Two-input AND gate (b) Two-input OR gate (c) NOT gate or inverter

Fig. 1-4 Symbols for digital logic circuits

Switching Circuit Demonstrating Binary Logic



(a) Switches in series – logic AND

(b) Switches in parallel – logic OR

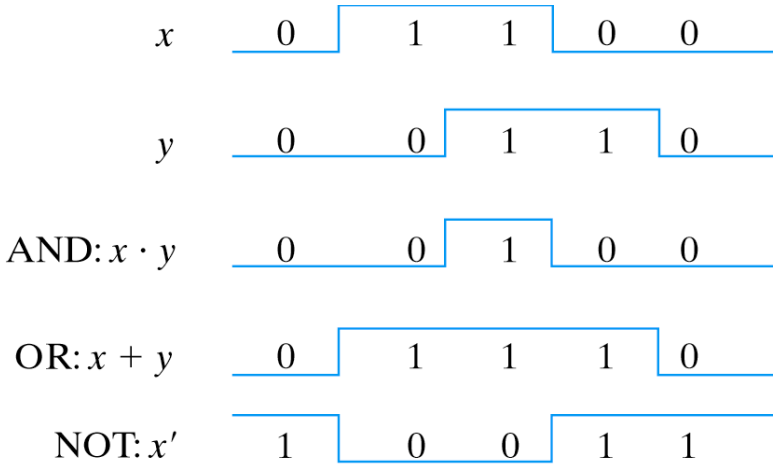
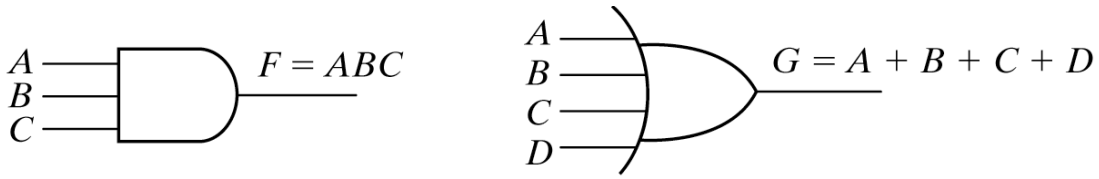


Fig. 1-5 Input-output signals for gates



(a) Three-input AND gate (b) Four-input OR gate

Fig. 1-6 Gates with multiple inputs