EC-121 Digital Logic Design

Lecture 2
Digital Systems and Number System
Dr Hashim Ali

Fall 2017

Department of Computer Science and Engineering
HITEC University Taxila

Overview—What to learn

- Digital Systems
- Number Systems [Binary, Octal and Hexadecimal]
- Complements of Number
- Signed Binary Numbers
- Binary Codes
- Binary Storage and Registers
- Binary Logic

Complements of Binary Numbers
Signed Binary Numbers

Complements of Binary Numbers

- Complements are used for simplifying the subtraction operation and for logical manipulation.
- There are two types of complements for each base-r system: the radix and the diminished radix complements.
- For binary numbers:
  - Radix Complement — 2's complement
  - Diminished Radix Complement — 1's complement

Diminished Radix Complement

[(r-1)'s Complement]

- Given a number N in base-r having n-digits, the (r-1)'s complement of N is defined as $(r^n-1)-N$.
- For example: 1's complement of 1011000
- Here;
  - N = 1011000
  - n = 7 (Total number of bits = 7)
  - r = 2 (Because N is a binary number)
  - $(2^7 - 1) - 1011000 = 1111111 - 1011000 = 0100111$

Radix Complement — [r's Complement]

- The r's complement of an n-digit number in base-r is defined as $r^n - N$, for N=0 and 0 for N=0.
  - Compare with (r – 1)'s complement, the r's complement is (r – 1)'s + 1
  - since $r^n - N = [(r^n - 1) - N] + 1$.

- For example: 2's complement of 1011000

- Here;
  - N = 1011000
  - n = 7 (Total number of bits = 7)
  - r = 2 (Because N is a binary number)
  - $[(2^7 - 1) - 1011000] + 1 = [1111111 - 1011000] + 1 = 0100111 + 1 = 0101000$

Shortcut:
Leaving all least significant 0's and the first 1 unchanged, and others have complemented

Computer Arithmetic

Arithmetic and Logic Unit

- Part of the computer that actually performs arithmetic and logical operations on data.
- All of the other elements of the computer system are there mainly to bring data into the ALU for it to process and then to take the results back out.
- Based on the use of simple digital logic devices that can store binary digits and perform simple Boolean logic operations.

Integer Representation

- In the binary number system arbitrary numbers can be represented with:
  - The digits zero and one.
  - The minus sign (for negative numbers).
  - The period, or radix point (for numbers with a fractional component).
- For purposes of computer storage and processing we do not have the benefit of special symbols for the minus sign and radix point.
- Only binary digits (0,1) may be used to represent numbers.

Sign-Magnitude Representation

- There are several alternative conventions used to represent negative as well as positive integers.
  - All of these alternatives involve treating the most significant (leftmost) bit in the word as a sign bit.
    - If the sign bit is 0 the number is positive.
      - Ex: +5 is 0101
    - If the sign bit is 1 the number is negative.
      - Ex: -5 is 1101

- Sign-magnitude representation is the simplest form that employs a sign bit.

- **Drawbacks:**
  - Addition and subtraction require a consideration of both the signs of the numbers and their relative magnitudes to carry out the required operation.
  - There are two representation of 0, i.e. +0 — 0000 and -0 — 1000.
- Because of these drawbacks, sign-magnitude representation is rarely used in implementing the integer portion of the ALU.

Two's Complement Representation

- Use the most significant bit as a sign bit.
- Differs from the sign-magnitude representation in the way that the other bits are interpreted.

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

Characteristics of 2's Complement Representation

Range	-2^{n-1} through $2^{n-1} - 1$
Number of Rpresentation of Zero	One
Negation	Take the Boolean complement of each bit of the corresponding positive number, then add 1 to the resulting bit pattern viewed as an unsigned integer.
Expansion of Bit Length	Add additional bit positions to the left and fill in with the value of the original sign bit.
Overflow Rule	If two numbers with the same sign (both positive or both negative) are added, then overflow occurs if and only if the result has the opposite sign.
Subtraction Rule	To subtract B from A, take the twos complement of B and add it to A.

Negation

- Two's complement operation
  - Take the Boolean complement of each bit of the integer (including the sign bit).
  - Treating the result as an unsigned binary integer, add 1.

+18 = 00010010 (two's complement)

Bitwise complement = 11101101

+ 1

11101110 = -18

- The negative of the negative of that number is itself:

-18 = 11101110 (two's complement)

Bitwise complement = 00010001

+ 1

00010010 = +18

Range Extension

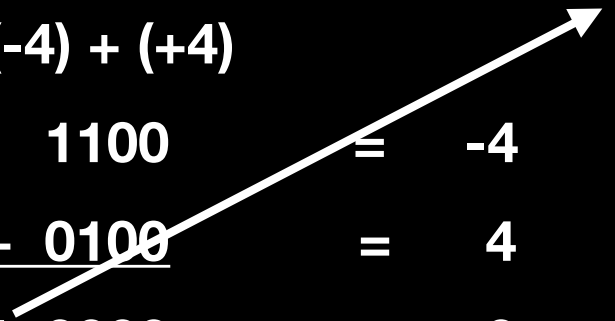
- Range of numbers that can be expressed is extended by increasing the bit length.
- In sign-magnitude notation this is accomplished by moving the sign bit to the new leftmost position and fill in with zeros.
  - Ex: +6 in 4-bit sign-magnitude representation is 0110.
  - +6 in 8-bit representation is 00000110.
- This procedure will not work for two's complement negative integers.
  - Rule is to move the sign bit to the new leftmost position and fill in with copies of the sign bit.
  - For positive numbers, fill in with zeros, and for negative numbers, fill in with ones.
    - Ex: +6 in 4-bit 2's complement representation is 0110.
    - +6 in 8-bit representation is 00000110.
    - -6 in 8-bit representation is 11111010.
  - This is called sign extension.

Addition — 4-bit numbers

a) (-7) + (+5)

1001	=	-7
+ 0101	=	5
1110	=	-2

b) (-4) + (+4)



1100	=	-4
+ 0100	=	4
1 0000	=	0

c) (+3) + (+4)

0011	=	3
+ 0100	=	4
0111	=	7

d) (-4) + (-1)

1100	=	-4
+ 1111	=	-1
1 1011	=	-5

Addition — with Overflow

e) 5 + 4

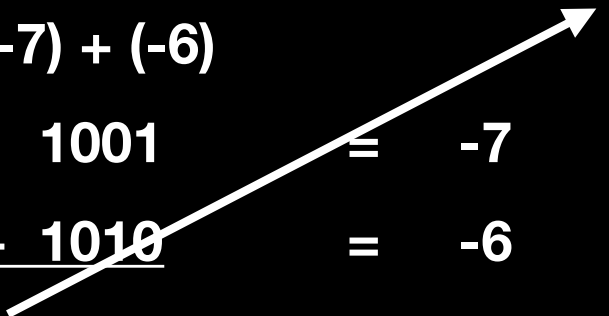
0101	=	5
+ 0100	=	4
1001	=	Overflow

Result has opposite sign

f) (-7) + (-6)

1001	=	-7
+ 1010	=	-6
1 0011	=	Overflow

Discard carry



- **Overflow Rule:**
  - If two numbers are added, and they are both positive or both negative, then overflow occurs if and only if the result has the opposite sign.

Subtraction

- **Subtraction Rule:**
  - To subtract one number (**subtrahend**) from another (**minuend**), take the twos complement (negation) of the subtrahend and add it to the minuend.

Subtraction — 4-bit numbers

a) 2 - 7

Minuend (M) = 2 = 0010

Subtrahend (S) = 7 = 0111

Negation (-S) = 1001

0010	=	2
+ 1001	=	-7
1011	=	-5

b) 0101 - 0010

Minuend (M) = 0101 = 5

Subtrahend (S) = 0010 = 2

Negation (-S) = 1110

Discard carry

0101	=	5
+ 1110	=	-2
1 0011	=	3

Subtraction — 4-bit numbers

a) - 5 - 2

Minuend (M) = -5 = 1011

Subtrahend (S) = 2 = 0010

Negation (-S) = 1110

Discard carry

1011	=	-5
+ 1110	=	-2
1 1001	=	-7

b) 0101 - 1110

Minuend (M) = 0101 = 5

Subtrahend (S) = 1110 = -2

Negation (-S) = 0010

0101	=	5
+ 0010	=	2
0111	=	7

Subtraction — with Overflow

a) 7 - (-7)

Minuend (M) = 7 = 0111

Subtrahend (S) = -7 = 1001

Negation (-S) = 0111

0111	=	7
+ 0111	=	7
1110	=	Overflow

Result has opposite sign

b) - 6 - 4

Minuend (M) = 1010 = -6

Subtrahend (S) = 0100 = 4

Negation (-S) = 1100

Discard carry

1010	=	-6
+ 1100	=	-4
1 0110	=	Overflow

Binary Codes

Binary Codes

- A binary code represents text, computer processor instructions, or other data using any two-symbol system, but often the binary number system's 0 and 1.
  - For example, the lower case *‘a’*, if represented by the bit string *01100001* (as it is in the standard ASCII code), can also be represented as the decimal number *97*.

- In the machine coding, when numbers, letters or words are represented by a specific group of symbols, it is said that the number, letter or word is being encoded. The group of symbols is called as a code.
  - The digital data is represented, stored and transmitted as group of binary bits. This group is also called as binary code. The binary code is represented by the number as well as alphanumeric letter.

- Few examples;
  - Representation of Numbers [We have already seen]
    - 0—0000, 1—0001, 2—0010,…, 9—1001
  - Alphabetic characters
    - h—01101000,…, z—01111010
    - HITEC—01001000 01001001 01010100 01000101 01000011
  - 7 colours of rainbow?
    - Red—0000, Orange—0001, Yellow—0010, Green—0011, Blue—0100, Indigo—0101, Violet—0110

Case sensitive
h ≠ H

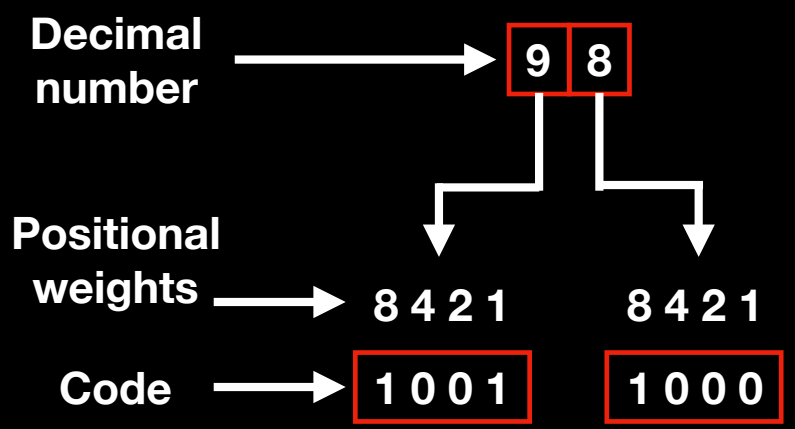
Classification of Binary Codes

- Binary codes are classified in following categories:-
  - Weighted Codes
    - Binary Coded Decimal Code
  - Non-weighted Codes
  - Alphanumeric Codes
  - Error Detecting Codes

Weighted Codes — 8421 Code

- Weighted binary codes obey the positional weight principle.
  - Each position of the number represents a specific weight.

- In weighted codes, each decimal digit is represented by a group of four bits.



Binary Coded Decimal Code (BCD)

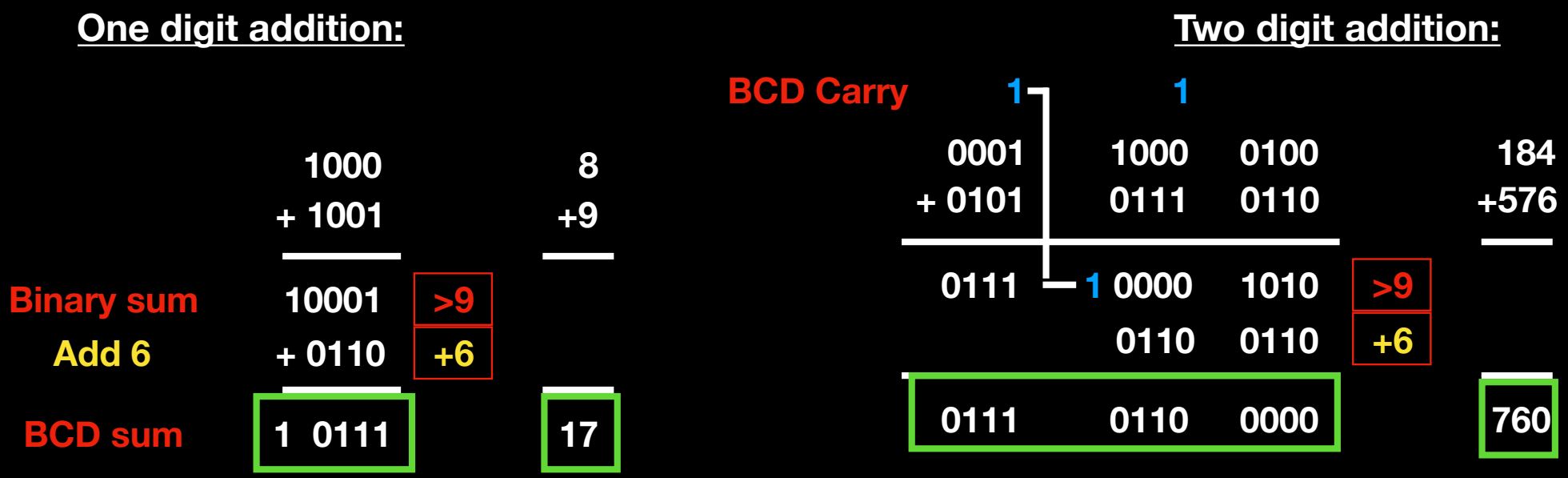
- Decimal digit is represented by 4-bit binary number.
  - BCD is a way to express decimal digit with a binary code.
- In BCD, with 4 bits, we can represent 16 combinations (0000–1111).
  - But only first ten are used [0-0000 to 9-1001].
    - The remaining combinations are invalid.
- Ex: $(185)_{10} = (0001\ 1000\ 0101)_{BCD}$
  $= (101101)_2$

Table 1.4
Binary-Coded Decimal (BCD)

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

BCD Addition

- When the binary sum is greater than 1001, the addition of 6 to the binary sum converts it to the correct digit and also produces a carry as required.



Non-weighted Codes

- In this type of binary codes, the positional weights are not assigned. The examples of non-weighted codes are 2421, Excess-3, 84-2-1 codes and Gray code.

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

Gray Code

- It is the non-weighted code and it is not arithmetic codes. That means there are no specific weights assigned to the bit position.
  - It has a very special feature that, only one bit will change each time the decimal number is incremented as shown in fig.
- As only one bit changes at a time, the gray code is called as a unit distance code.
  - The gray code is a cyclic code.
  - Gray code cannot be used for arithmetic operation.
- Mathematics:
  - Lets say, Num = 8421 BCD of decimal value
  - Grey Code = Num XOR (Num >>1)

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

ASCII Character Code

- ASCII include seven bits, contain 94 graphic characters and 34 control functions as follows table.

- There are three types of control characters:
  - Format effectors: control the layout of printing (BS, HT, CR…).
  - Information separators: used to separate the data into divisions such as paragraphs and pages (RS, FS…).
  - Communication-control characters: it is useful during the transmission of text between remote terminals (STX, ETX..).

ASCII Table

Table 1.7
American Standard Code for Information Interchange (ASCII)

$b_4b_3b_2b_1$	$b_7b_6b_5$ = 000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	“	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	‘	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	–	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	–	o	DEL

Error Detecting Codes

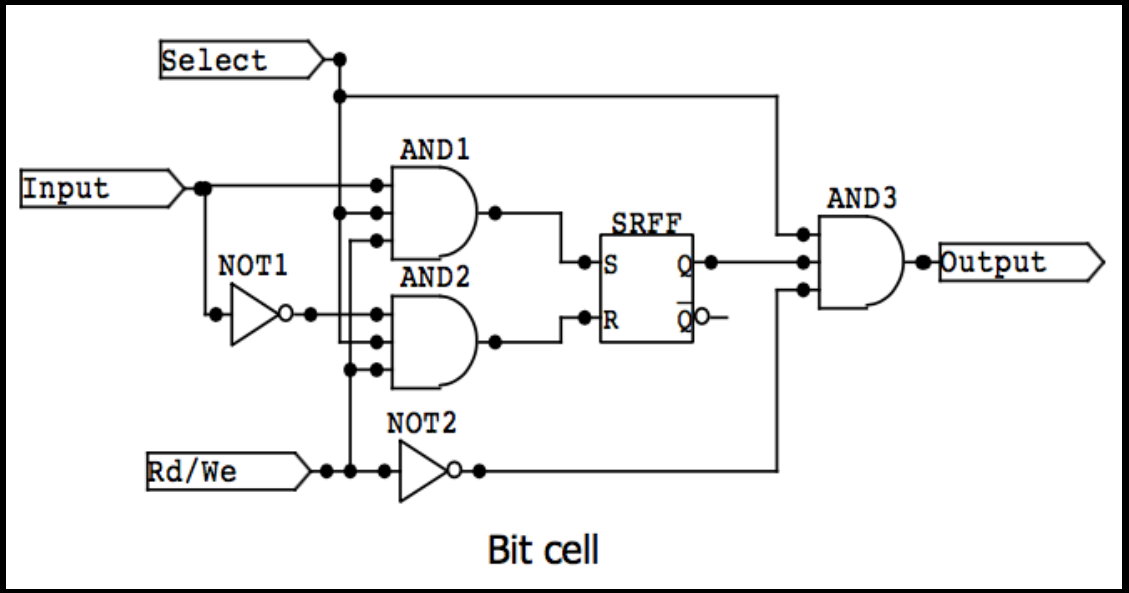
- An eighth bit is added to the ASCII character to indicate its parity. We have following even and odd parity:

	With even parity	With odd parity
ASCII A = 1000001	**01000001**	**11000001**
ASCII T = 1010100	**11010100**	**01010100**

- The even or odd parities can find out only odd combination of errors in each character, an even combination of errors is undetected. May be the hamming code can solve that in some bits range.

Binary Storage and Registers

- A binary cell is a device that processes two stable states and is capable of storing one bit of info, and a register is a group of binary cells.



Bit cell

Binary Logic

Binary Logic

- There are three basic logical operations:-
  - **AND:** This operation is represented as follows
    x . y = z or x y = z,
    z=1 if and only if x=1 and y=1; otherwise z=0
  - **OR:** This operation is represented as follows
    x+y=z
    z=1 if x=1 or if y=1 or x=1 and y=1 ;otherwise z=0
  - **NOT:** This operation is represented as follows
    x' = z or x = z
    e.g. complement operation, changes a 1 to 0, 0 to 1

Similarity and Difference

- Binary logic resembles binary arithmetic, and the operations AND and OR have similarities to multiplication and addition, respectively.
- The symbols used for AND and OR are the same as those used for multiplication and addition.
- Binary logic should not be confused with binary arithmetic.
  - Binary arithmetic: $1 + 1 = 10_2 = 2_{10}$
  - Binary logic: $1 + 1 = 1_2$

Logic Gates

- For each combination of the values of x and y, and output z, it may be listed in a compact form using truth tables.

Table 1.8
Truth Tables of Logical Operations

AND			OR			NOT	
x	y	$x \cdot y$	x	y	$x + y$	x	x'
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1		
1	1	1	1	1	1		

Definition of Logic Signals

- Logic gates are electronic circuits that operate on one or more input signals to produce an output signal.
- Signals such as voltages or currents, we define between some ranges as logic 1 or logic 0.

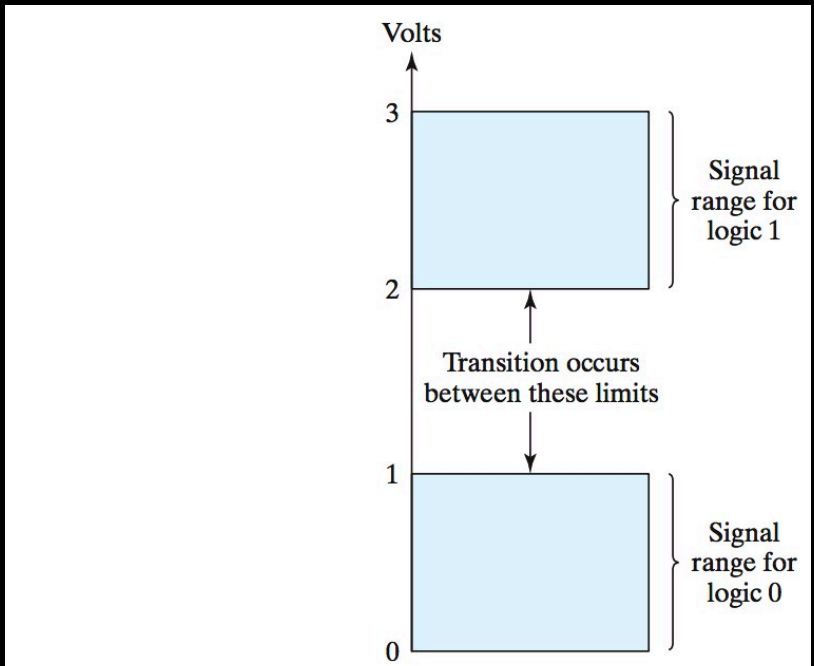
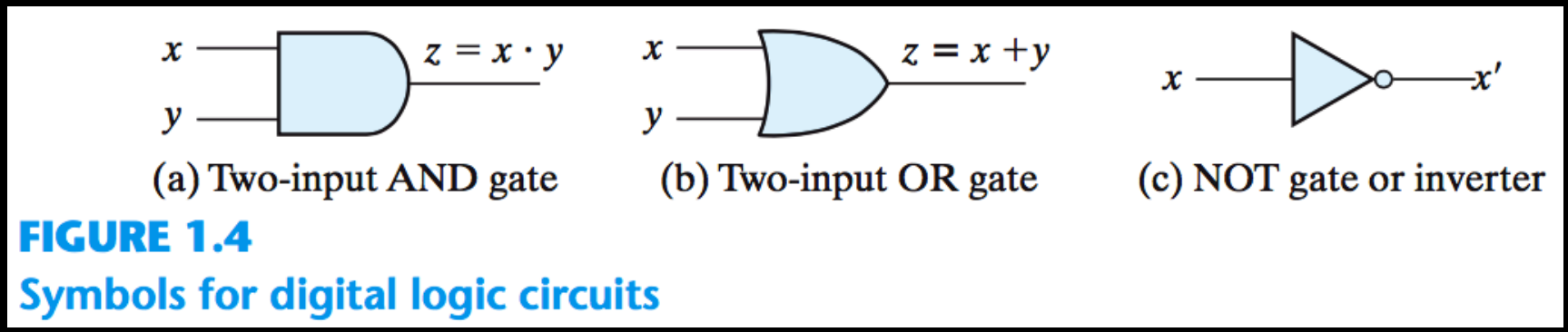


FIGURE 1.3
Signal levels for binary logic values

Logic Gates Symbols

- The graphic symbols used to designate the three types of gates are shown below.

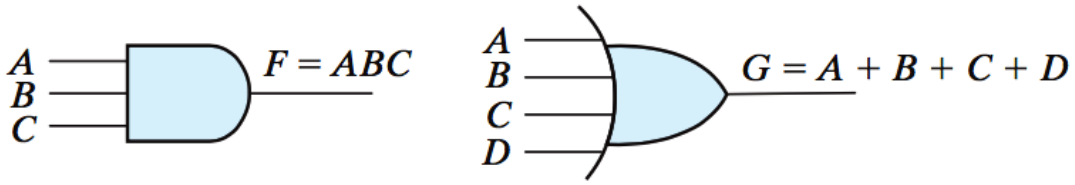


(a) Two-input AND gate (b) Two-input OR gate (c) NOT gate or inverter

FIGURE 1.4
Symbols for digital logic circuits

Gates with Multiple Inputs

- AND and OR gates may have more than two inputs.
- Three–input AND gate responds with logic 1 output if all three inputs are logic 1.
  - When any input is logic 0, output produces logic 0.
- OR gate characteristic have described before in this chapter.



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

FIGURE 1.6
Gates with multiple inputs