CSE 205: Digital Logic Design

Textbook

- Digital Design (5th Edition)
  - M. Morris Mano
  - Michael D. Ciletti

Digital Systems

- Why study digital logic design
  - To design and build a digital system

- Digital Systems
  - Example: digital cameras, digital telephones, digital television, digital computers…
  - Used in communication, business transactions, traffic control, space guidance, medical treatment, weather monitoring, the Internet …

Binary Logic

- In a digital system, all signals take on discrete values.
  - Also referred as states

- Most modern digital systems operate on 2 discrete states
  - Binary logic system
    - Deals with binary variables and a set of logical operations

Binary Logic

- We represent the two states of binary variable as
  - True and false
  - 1 and 0
  - High and Low

- Three basic logic operations
  - AND: $x.y = z \; or \; xy = z$
  - OR: $x + y = z$
  - NOT: $x' = z \; or \; \overline{x} = z$

TRUTH TABLE

- Specifies results for all possible input combinations

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

Logic Gates

- Logic gates are electronic circuits that operate on one or more input signals to produce an output signal.

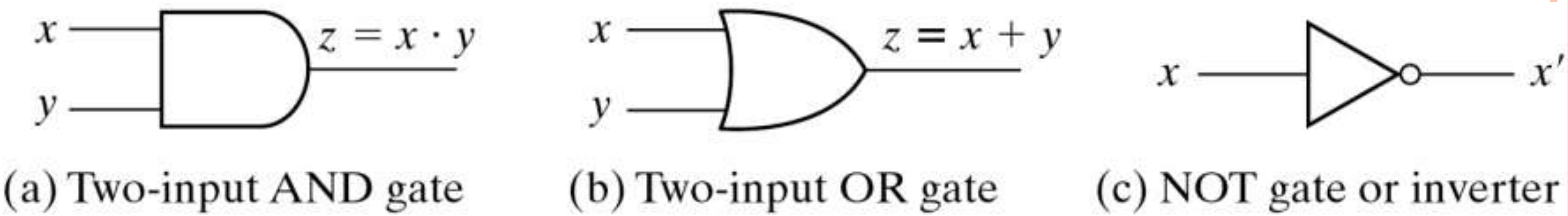


Fig. 1-4 Symbols for digital logic circuits

Logic Gates

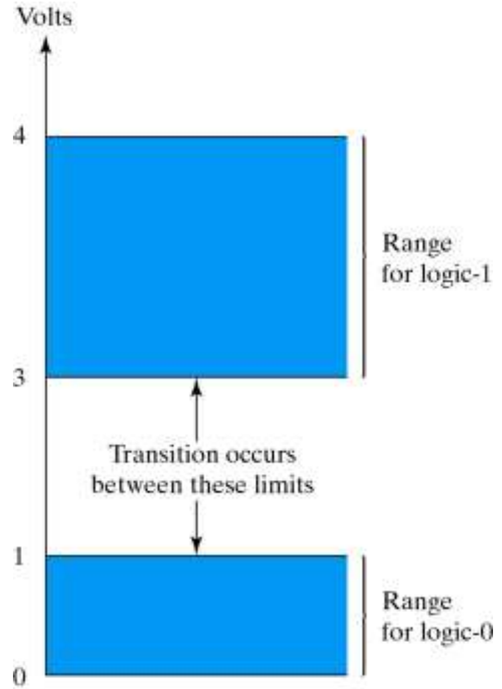
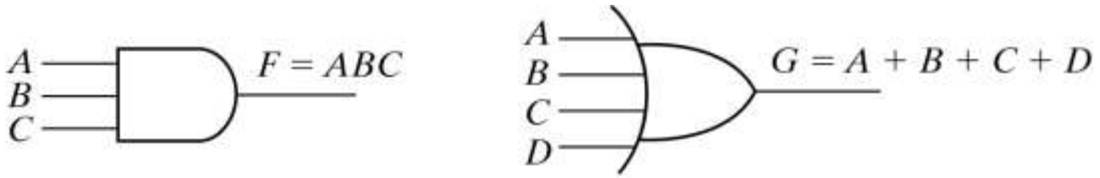


Fig. 1-3 Example of binary signals

Logic Gates



(a) Three-input AND gate

(b) Four-input OR gate

Fig. 1-6 Gates with multiple inputs

Boolean Function

- An algebraic expression
  - Binary variables
  - Constants 0 and 1
  - Logic operation symbols

$$f(x,y,z) = (x + y')z + x'$$

- Some terminology, notation and precedence:
  - f is the name of the function.
  - (x,y,z) are the input variables, each representing 1 or 0.
  - A literal is a single variable within a term, in complemented or uncomplemented form. The function above has four literals: x, y', z, and x'.
  - Parenthesis has the highest precedence, followed by NOT, AND, and then OR.

Gate Implementation of a Function

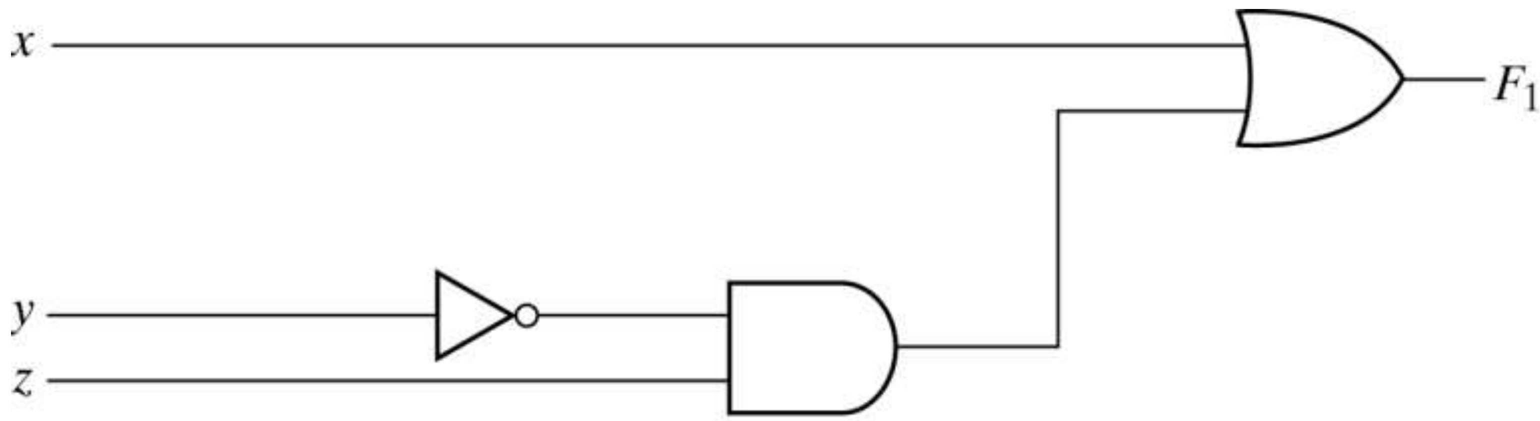
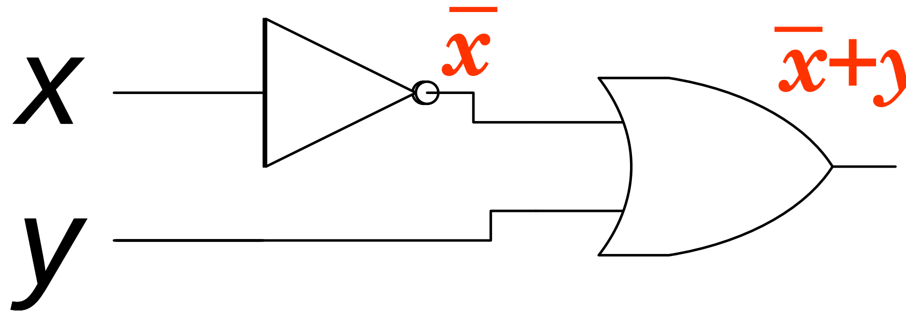


Fig. 2-1 Gate implementation of $F_1 = x + y'z$

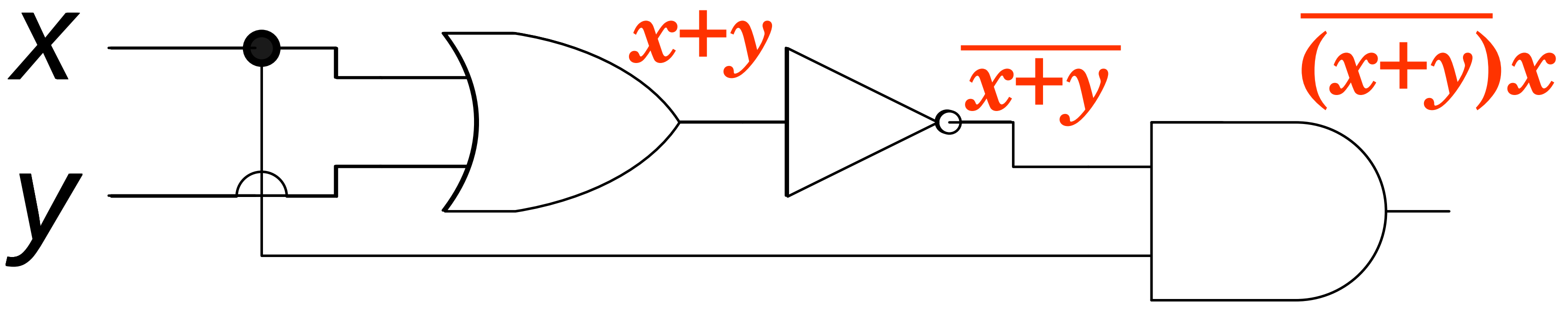
Gate Implementation of a Function

$$f(x,y,z) = x' + y$$



Gate Implementation of a Function

$$f(x,y,z) = \overline{(x + y)}\mathrm{x}$$



Boolean Function

- A Boolean function can be represented in a truth table.

$$f(x,y,z) = (x + y')z + x'$$

$$\begin{aligned}
f(0,0,0) &= (0 + 1)0 + 1 &= 1\\
f(0,0,1) &= (0 + 1)1 + 1 &= 1\\
f(0,1,0) &= (0 + 0)0 + 1 &= 1\\
f(0,1,1) &= (0 + 0)1 + 1 &= 1\\
f(1,0,0) &= (1 + 1)0 + 0 &= 0\\
f(1,0,1) &= (1 + 1)1 + 0 &= 1\\
f(1,1,0) &= (1 + 0)0 + 0 &= 0\\
f(1,1,1) &= (1 + 0)1 + 0 &= 1
\end{aligned}$$

x	y	z	f(x,y,z)
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

BOOLEAN FUNCTION

- A Boolean function can be represented in a truth table.

$$F(x,y,z) = x\bar{z}+y$$

$$F(x,y,z) = x\bar{z}+y$$

x	y	z	$\bar{z}$	$x\bar{z}$	$x\bar{z}+y$
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	0	1
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	0	0	1

Number Systems

- Consists of **TWO Things:**
  - A **BASE or RADIX Value**
  - A **SET of DIGITS**
    - *Digits are symbols representing all values* ***less than the radix value.***
- Example is the Common Decimal System:
  - RADIX (BASE) = 10
  - Digit Set = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}

Decimal Number Systems

- Consider: 5032.21

$$(5032.21)_{10} = 5\times(10)^3 + 0\times(10)^2 + 3\times(10)^1 + 2\times(10)^0 + 2\times(10)^{-1} + 1\times(10)^{-2}$$
$$= 5000 + 0 + 30 + 2 + 0.2 + 0.01$$

- Other Notation: $(5032.21)_{10}$

- In general, a number expressed in a base-*r* system has coefficients multiplied by powers of *r*:

$$a_n \cdot r^n + a_{n-1} \cdot r^{n-1} + \ldots + a_1 \cdot r^1 + a_0 + a_{-1} \cdot r^{-1} + a_{-2} \cdot r^{-2} + \ldots + a_{-m} \cdot r^{-m}$$

Other Number Systems

- Binary
  - Radix = $(2)_{10}$
  - Digit Set = $\{0,1\}$
- Octal
  - Radix = $(8)_{10}$
  - Digit Set = $\{0,1,2,3,4,5,6,7\}$
- Hexadecimal
  - Radix = $(16)_{10}$
  - Digit Set = $\{0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F\}$

Binary Number Systems

- Binary
  - Radix = $(2)_{10}$
  - Digit Set = $\{0,1\}$

- Binary to Decimal:

$$(1101.01)_2 = 1\times 2^3 + 1\times 2^2 + 0\times 2^1 + 1\times 2^0 + 0\times 2^{-1} + 1\times 2^{-2}$$
$$= (13.25)_{10}$$

Binary to Decimal: practice

a) $0110_2 = ?$ 6_{10}

b) $11010_2 = ?$ 26_{10}

c) $0110101_2 = ?$ 53_{10}

d) $11010011_2 = ?$ 211_{10}

Octal Number Systems

- Octal
  - Radix = $(8)_{10}$
  - Digit Set = {0,1,2,3,4,5,6,7}

- Octal to Decimal:

$$(15.2)_8 = 1\times 8^1 + 5\times 8^0 + 2\times 8^{-1}$$
$$= (13.25)_{10}$$

Hexadecimal Number Systems

- Hexadecimal
  - Radix = $(16)_{10}$
  - Digit Set = $\{0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F\}$
- Hexadecimal to Decimal:

$$(D.4)_{16} = D \times (16)^0 + 4 \times (16)^{-1}$$
$$= (13.25)_{10}$$

Decimal (*Integer*) to Binary

- Divide the number by the 'Base' (=2)
- Take the remainder (either 0 or 1) as a coefficient
- Take the quotient and repeat the division

Example: $(13)_{10}$

	Quotient	Remainder	Coefficient
13/ 2 =	6	1	$a_0 = 1$
6 / 2 =	3	0	$a_1 = 0$
3 / 2 =	1	1	$a_2 = 1$
1 / 2 =	0	1	$a_3 = 1$

Answer: $(13)_{10} = (a_3\ a_2\ a_1\ a_0)_2 = (1101)_2$

MSB LSB

Decimal to binary: practice

a) $13_{10} = ?$ $1\,1\,0\,1_2$

b) $22_{10} = ?$ $1\,0\,1\,1\,0_2$

c) $43_{10} = ?$ $1\,0\,1\,0\,1\,1_2$

d) $158_{10} = ?$ $1\,0\,0\,1\,1\,1\,1\,0_2$

DECIMAL (*FRACTION*) TO BINARY

- Multiply the number by the 'Base' (=2)
- Take the integer (either 0 or 1) as a coefficient
- Take the resultant fraction and repeat the division

Example: $(0.625)_{10}$

		Integer		Fraction	Coefficient
0.625	* 2 =	1	.	25	$a_{-1} = 1$
0.25	* 2 =	0	.	5	$a_{-2} = 0$
0.5	* 2 =	1	.	0	$a_{-3} = 1$

Answer: $(0.625)_{10} = (0.a_{-1}\ a_{-2}\ a_{-3})_2 = (0.101)_2$

MSB LSB

Decimal to Octal Conversion

Example: $(175)_{10}$

	Quotient	Remainder	Coefficient
$175 / 8 =$	21	7	$a_0 = 7$
$21 / 8 =$	2	5	$a_1 = 5$
$2 / 8 =$	0	2	$a_2 = 2$

Answer: $(175)_{10} = (a_2\, a_1\, a_0)_8 = (257)_8$

Example: $(0.3125)_{10}$

	Integer		Fraction	Coefficient
$0.3125 * 8 =$	2	.	5	$a_{-1} = 2$
$0.5 * 8 =$	4	.	0	$a_{-2} = 4$

Answer: $(0.3125)_{10} = (0.a_{-1}\, a_{-2}\, a_{-3})_8 = (0.24)_8$

Binary – Octal Conversion

- $8 = 2^3$
- Each group of 3 bits represents an octal digit

Example:

Assume Zeros

$$(\underbrace{10}_{2}\underbrace{110}_{6}.\underbrace{01}_{2})_2$$

$$(2\ 6\ .\ 2)_8$$

Works **both** **ways (Binary to Octal & Octal to Binary)**

Octal	Binary
0	0 0 0
1	0 0 1
2	0 1 0
3	0 1 1
4	1 0 0
5	1 0 1
6	1 1 0
7	1 1 1

BINARY – HEXADECIMAL CONVERSION

- $16 = 2^4$
- Each group of 4 bits represents a hexadecimal digit

Example:

Assume Zeros

$(1\,0110.01)_2$

$(1\quad 6\,.\,4)_{16}$

Works both ways (Binary to Hex & Hex to Binary)

Hex	Binary
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1
A	1 0 1 0
B	1 0 1 1
C	1 1 0 0
D	1 1 0 1
E	1 1 1 0
F	1 1 1 1

Octal – Hexadecimal Conversion

- Convert to Binary as an intermediate step

Example:

$(\;2\quad 6\;.\;2\;)_8$

Assume Zeros

$(\,0\,1\,0\,1\,1\,0\,.\,0\,1\,0\,)_2$

Assume Zeros

$(\;1\quad 6\;.\;4\;)_{16}$

Works both ways (Octal to Hex & Hex to Octal)

Binary Addition

$$
\begin{array}{rccccccccl}
 & 1 & 1 & 1 & 1 & 1 & 1 & & & \\
 & & 1 & 1 & 1 & 1 & 0 & 1 & & = 61 \\
+ & & & 1 & 0 & 1 & 1 & 1 & & = 23 \\
\hline
 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & & = 84
\end{array}
$$

$$\geq (2)_{10}$$

Binary Subtraction

$$
\begin{array}{cccccccccc}
 & & 1 & & & 2 & & & & = (10)_2 \\
 & 0 & \cancel{2} & 2 & 0 & 0 & 2 & & & \\
 & \cancel{1} & 0 & 0 & \cancel{1} & \cancel{1} & 0 & 1 & & = 77 \\
- & & & 1 & 0 & 1 & 1 & 1 & & = 23 \\
\hline
 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & & = 54
\end{array}
$$

COMPLEMENTS

- They are used to simplify the subtraction operation
- Two types (for each *base-r* system)
  - Diminishing radix complement (r-1's complement)
  - Radix complement (r's complement)

For n-digit number N

$$(r^n - 1) - N \longrightarrow r\text{-1's complement}$$

$$r^n - N \longrightarrow r\text{'s complement}$$

Diminished Radix Complement

- **Example for 6-digit <u>decimal</u> numbers**:
  - 9's complement is $(r^n - 1) - N = (10^6 - 1) - N = 999999 - N$
  - 9's complement of 546700:

    $$999999 - 546700 = 453299$$

- **Example for 7-digit <u>binary</u> numbers:**
  - 1's complement is $(r^n - 1) - N = (2^7 - 1) - N = 1111111 - N$
  - 1's complement of 1011000:

    $$1111111 - 1011000 = 0100111$$

Radix Complement

- The r's complement of an n-digit number N in base r is defined as $r^n - N$ for $N \neq 0$ and as 0 for $N = 0$.

- The r's complement can also be obtained by adding 1 to the $(r - 1)$'s complement, since

$$r^n - N = [(r^n - 1) - N] + 1.$$

- Decimal Number: 10^n - N
  - Example: 10's complement of 246700 is 753300

Radix Complement (Binary Number)

- Take 1's complement then add 1

OR

- Toggle all bits to the left of the first '1' from the right
- **Example:**

	1 0 1 1 0 0 0 0	**1 0 1 1 0 0 0 0**
	0 1 0 0 1 1 1 1	
+	**1**	
	0 1 0 1 0 0 0 0	**0 1 0 1 0 0 0 0**

Subtraction with Complements

- $M - N$
  - Add M to r's complement of N
    - $Sum = M + (r^n - N) = M - N + r^n$
  - If $M > N$, Sum will have an end carry r^n , discard it
  - If $M < N$, Sum will not have an end carry and
    - $Sum = r^n - (N - M)$ (r's complement of $N - M$)
    - So $M - N = -$ (r's complement of Sum)

Subtraction with Complements

- 65438 - 5623

	65438
10's complement of 05623	+94377
	159815
Discard end carry 10^5	-100000
Answer	59815

Subtraction with Complements

- 5623 - 65438

	05623
10’s complement of 65438	+34562
→	40185

There is no end carry.

Therefore, the answer is:
– (10's complement of 40185) = – 59815.

Subtraction with Complements

- 10110010 - 10011111

	10110010
2's complement of 10011111	+01100001
	100010011
Discard end carry 2^8	- 100000000
Answer	00010011

Subtraction with Complements

- 10011111 -10110010

	10011111
2’s complement of 10110010	+01001110
	11101101

There is no end carry.

Therefore, the answer is
Y – X = – (2's complement of 11101101) = – 00010011.

Subtraction with Complements

- Subtraction of unsigned numbers can also be done by means of the $(r - 1)$'s complement.
- Remember that the $(r - 1)$'s complement is one less then the r's complement.
- 10110010 - 10011111

	10110010
1's complement of 10011111	+01100000
	100010010
End-around carry	+ 1
Answer	00010011

Subtraction with Complements

- 10011111 -10110010

	10011111
1’s complement of 10110010	+01001101
	11101100

There is no end carry.

Therefore, the answer is
Y – X = – (1's complement of 11101100) = – 00010011.

SIGNED BINARY NUMBERS

- To represent negative integers, we need a notation for negative values.
- It is customary to represent the sign with a bit placed in the leftmost position of the number since binary digits.
- The convention is to make the sign bit 0 for positive and 1 for negative.
- Example:

Signed-magnitude representation:	10001001
Signed-1's-complement representation:	11110110
Signed-2's-complement representation:	11110111

Signed Binary Numbers Arithmetic Addition

+ 5	00000101
+11	00001011
+16	00010000

Discard

- 5	11111011
+11	00001011
+6	100000110

+ 5	00000101
-11	11110101
-6	11111010

- 5	11111011
-11	11110101
-16	111110000

Discard

Signed Binary Numbers Arithmetic Subtraction

- In 2's-complement form:

1. Take the 2's complement of the subtrahend (including the sign bit) and add it to the minuend (including sign bit).
2. A carry out of sign-bit position is discarded.

$$(\pm A) - (+B) = (\pm A) + (-B)$$
$$(\pm A) - (-B) = (\pm A) + (+B)$$

- Example:

(– 6) – (– 13) → (11111010 – 11110011)

→ (11111010 + 00001101)

→ 00000111 (+ 7)

BINARY CODES

- Generally Two Types:

  1. Alphanumeric Codes
  2. Numeric Codes

- Alphanumeric Codes: ASCII Code (7-bit)
  EBCDIC Code (8-bit)
- Numeric Codes: Weighted and Un-Weighted
- Weighted Codes: 8-4-2-1, 2-4-2-1, 3-3-2-1, etc
- Un-Weighted Codes: Excess-3 and Gray Codes (**reflected binary code (RBC)**)

Binary Code: BCD Code

- A number with k decimal digits will require 4k bits in BCD.

- A decimal number in BCD is the same as its equivalent binary number only when the number is between 0 and 9.

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

Binary Code: BCD Code

- BCD Addition

$$
\begin{array}{r}
4 \\
+5 \\
\hline
9
\end{array}
\qquad
\begin{array}{r}
0100 \\
+0101 \\
\hline
1001
\end{array}
\qquad
\begin{array}{r}
4 \\
+8 \\
\hline
12
\end{array}
\qquad
\begin{array}{r}
0100 \\
+1000 \\
\hline
1100 \\
+0110 \\
\hline
10010
\end{array}
\qquad
\begin{array}{r}
8 \\
+9 \\
\hline
17
\end{array}
\qquad
\begin{array}{r}
1000 \\
+1001 \\
\hline
10001 \\
+0110 \\
\hline
10111
\end{array}
$$

Binary Code: BCD Code

- Example:
  - Consider the addition of 184 + 576 = 760 in BCD:

BCD	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 Code: BCD Code

- Decimal Arithmetic: (+375) + (-240) = +135

$$\begin{array}{rr} 0 & 375 \\ \underline{+9} & \underline{760} \\ 0 & 135 \end{array}$$

Binary Codes
Other decimal codes

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
	1010	0101	0000	0001
Unused	1011	0110	0001	0010
bit	1100	0111	0010	0011
combi-	1101	1000	1101	1100
nations	1110	1001	1110	1101
	1111	1010	1111	1110

Binary Code: 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

BINARY CODES: ASCII CHARACTER CODE

- American Standard Code for Information Interchange (Refer to Table 1.7)
- A popular code used to represent information sent as character-based data.
- It uses 7-bits to represent:
  - 94 Graphic printing characters.
  - 34 Non-printing characters.
- Some non-printing characters are used for text format (e.g. BS = Backspace, CR = carriage return).
- Other non-printing characters are used for record marking and flow control (e.g. STX and ETX start and end text areas).

BINARY CODES: ASCII CHARACTER CODE

Table 1.7
American Standard Code for Information Interchange (ASCII)

	$b_7b_6b_5$							
$b_4b_3b_2b_1$	**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

Binary Codes: Error-Detecting Code

- To detect errors in data communication and processing, an eighth bit is sometimes added to the ASCII character to indicate its parity.
- A parity bit is an extra bit included with a message to make the total number of 1's either even or odd.
- Example:
  - Consider the following two characters and their even and odd parity:

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

Logic Gates

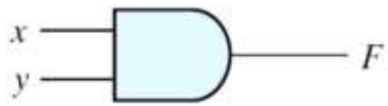
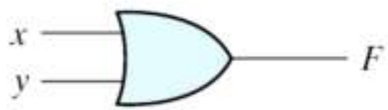
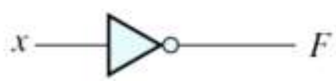
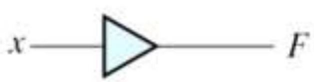
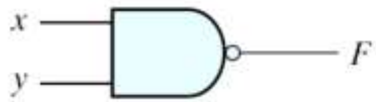
Name	Graphic symbol	Algebraic function	Truth table
AND		$F = xy$	x y | F 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1
OR		$F = x + y$	x y | F 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1
Inverter		$F = x'$	x | F 0 | 1 1 | 0
Buffer		$F = x$	x | F 0 | 0 1 | 1

Figure 2.5 Digital logic gates

Logic Gates

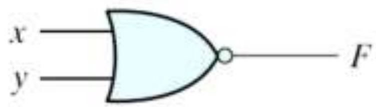
NAND



$F = (xy)'$

x	y	F
0	0	1
0	1	1
1	0	1
1	1	0

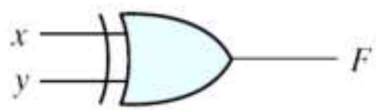
NOR



$F = (x + y)'$

x	y	F
0	0	1
0	1	0
1	0	0
1	1	0

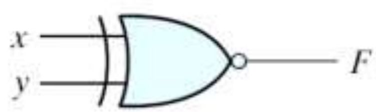
Exclusive-OR (XOR)



$F = xy' + x'y$
$= x \oplus y$

x	y	F
0	0	0
0	1	1
1	0	1
1	1	0

Exclusive-NOR or equivalence



$F = xy + x'y'$
$= (x \oplus y)'$

x	y	F
0	0	1
0	1	0
1	0	0
1	1	1

Figure 2.5 Digital logic gates

Practice: Truth table to Boolean Function

x	y	z	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

$$F = xyz + xyz' + x'yz + x'yz'$$
$$= y$$

Syllabus

- Chapter 1 (Excluding Section 1.8)