Computer Science Department

CS504

Digital Logic & Computer Organization

Lecture 1

Course Administration

Assist. Prof. Dr. Ahmed Hamza

Computer Science Department

Faculty of Graduate Studies for Statistical Research (FGSSR)

Cairo University

- **Office:** ***FGSSR Main Building, 5th Floor, Room No. 521***
- **Office Hours:** ***Saturday (10-12 AM), Thursday (2-4 PM)***

Is It Worth The Effort?

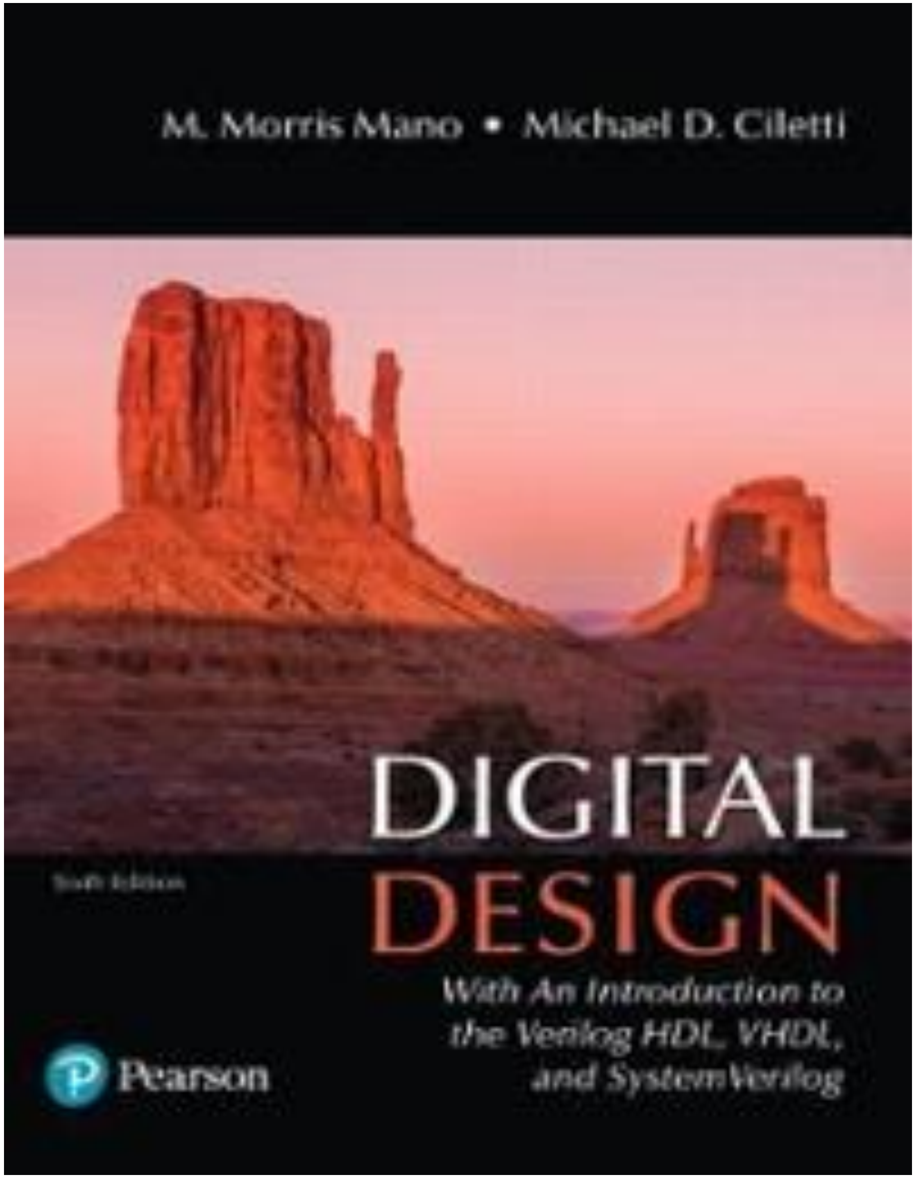
- The present technological period is the digital age.
- Digital systems have such a prevalent role in everyday life:
  - Digital Computers
  - Digital Phones
  - Digital Cameras
  - Digital TVs, etc.
- This course presents the basic concepts and tools for the analysis and design of digital circuits and systems.

List Of References

"Digital Design"

by Mano M. Morris, 6th Edition

Pearson Education, 2018



Course Outline

- ★ Chapter 1: Digital Systems
- ★ Chapter 2: Boolean Algebra And Logic Gates
- ★ Chapter 3: Gate-Level Minimization
- ★ Chapter 4: Combinational Logic
- ★ Chapter 5: Synchronous Sequential Logic

Lecture Outline (Chapter 1)

- **Digital Systems (Section 1.1)**
  - **Analog Versus Digital**
  - **Advantages Of Digital Systems**
  - **Building Blocks Of Digital System**
  - **Digital Computer**
  - **Digitization Of Analog Signal**
  - **Information Representation In Digital System**
- **Binary Codes (Section 1.7)**
  - **Binary Coded Decimal (BCD)**
  - **Excess-3**
  - **Other Binary Codes**

Lecture Outline (Chapter 1) – Cont'd

- ❖ Gray Code
- ❖ Alphanumeric codes
- ❖ Error Detecting Code

★ **Binary Logic (Section 1.9)**

- ❖ Logical Operations
- ❖ Operator Definitions
- ❖ Truth Table
- ❖ Logic Gates
- ❖ Logic Gates Behavior
- ❖ Logic Diagrams And Expressions
- ❖ Logic Gate Delay

Digital Systems

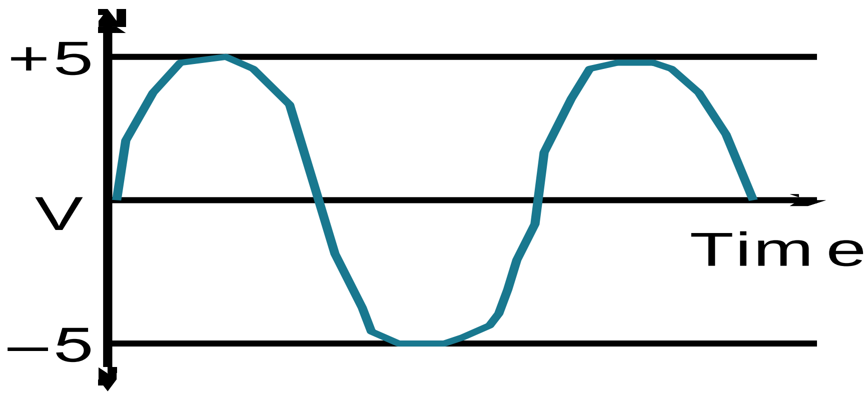
★ Analog Versus Digital

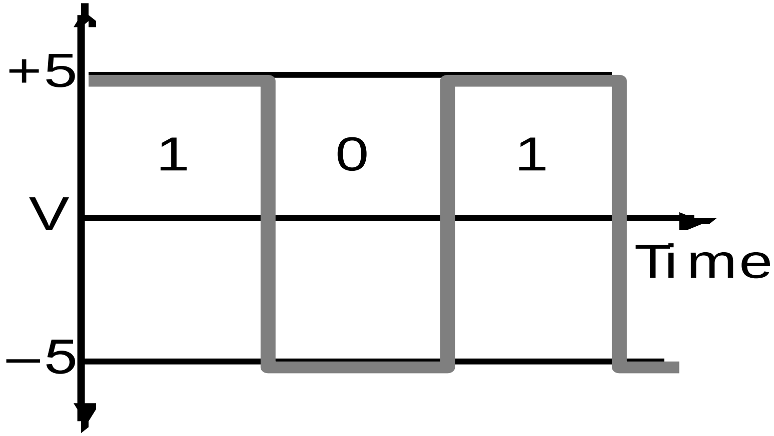
- ❖ **Analog means continuous.**
- ❖ **Analog parameters have continuous range of values.**
  - ✓ **Example: Temperature is an analog parameter, so it increases/decreases continuously**

- ❖ **Digital means discrete.**
- ❖ **Digital parameters have finite set of discrete values.**
  - ✓ **Example: Month number ∈ {1, 2, 3, …, 12} (discrete), so it is a digital parameter (cannot be 1.5!).**

Digital Systems (2)

★ Analog Versus Digital





Analog Signal

Digital Signal

Takes continuous values over a broad range

Takes discrete values only

Digital Systems (3)

★ Analog Versus Digital

Time

Analog

Continuous in
Value & Time

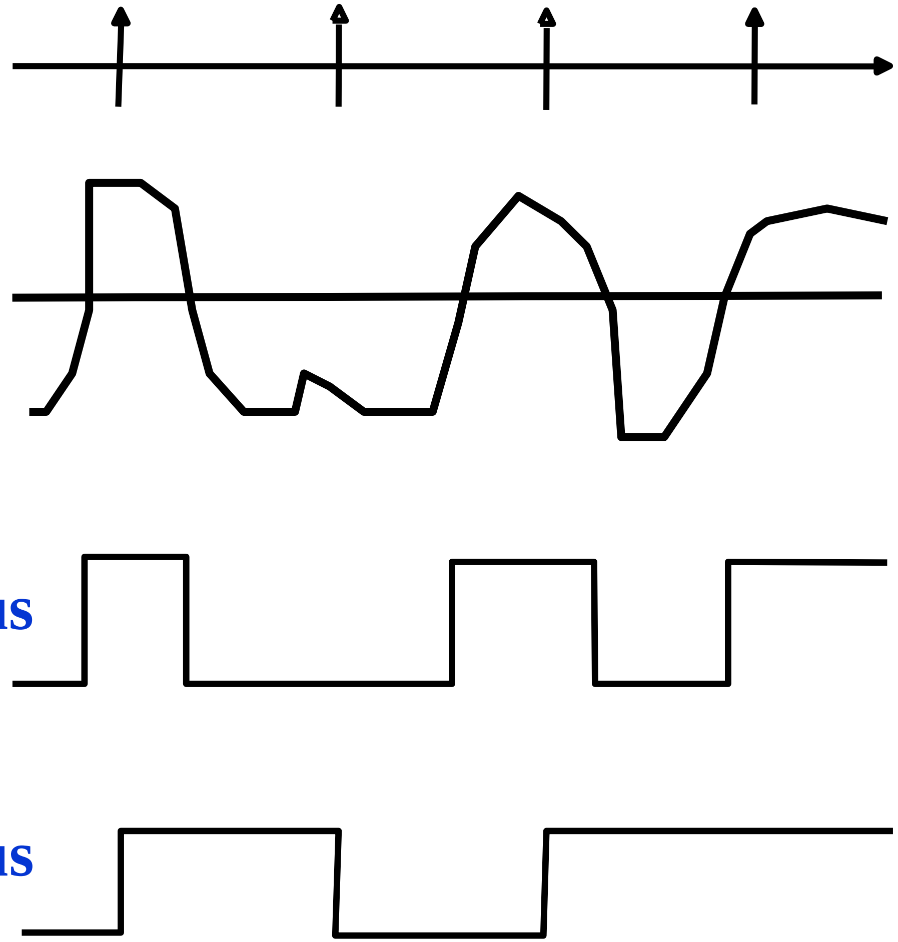
Digital

Asynchronous

Discrete in Value
&
Continuous in Time

Synchronous

Discrete in
Value & Time



Digital Systems (4)

★ Advantages Of Digital Systems

- ❖ **Digital systems are easier to design, because they deal with a limited set of values rather than an infinitely large range of continuous values.**

- ❖ **Most digital devices are programmable (changing the program in a programmable device), so the same underlying hardware can be used for many different applications.**

- ❖ **Digital systems can be made to operate with extreme reliability by using error-correcting codes.**

Digital Systems (5)

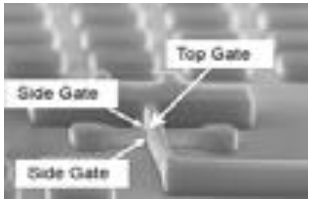
★ **Building Blocks Of Digital System**

Digital System

Circuit Board

Chip

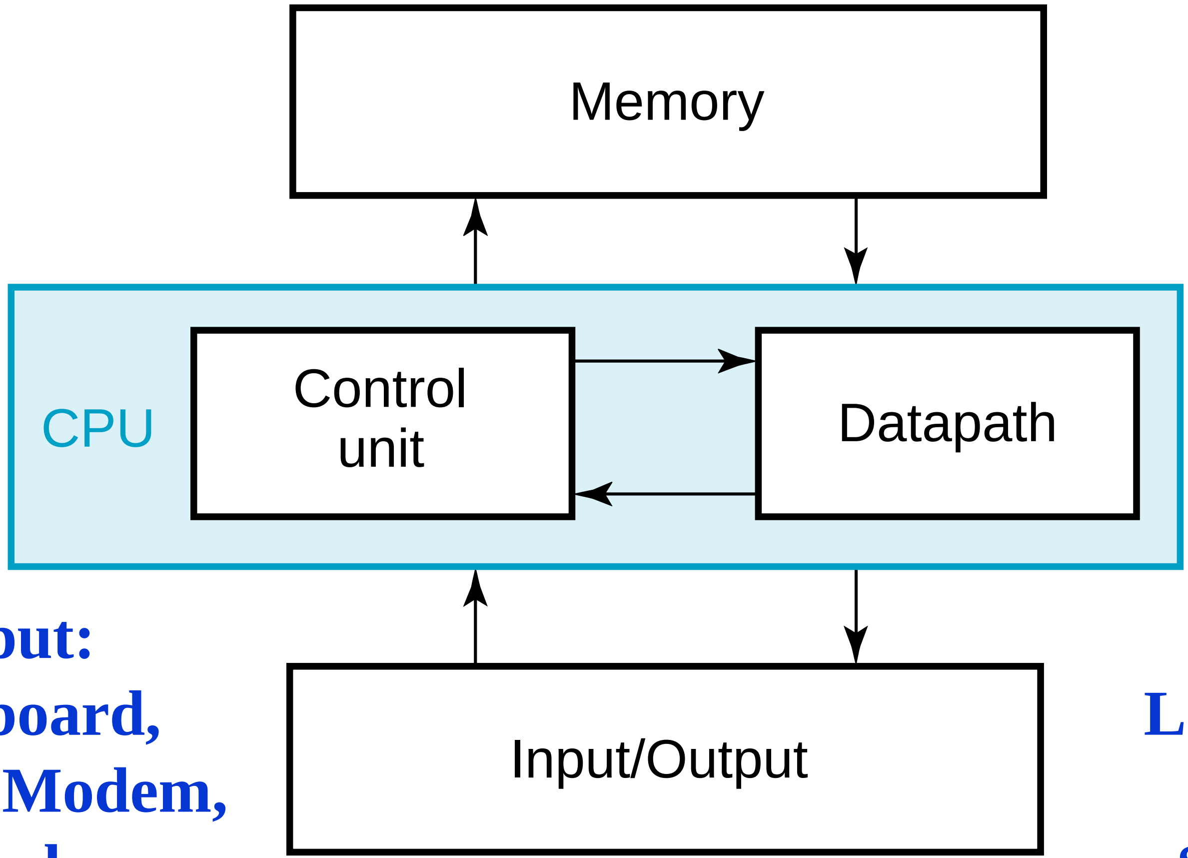
Transistor



Logic Gate

Digital Systems (6)

★ A Digital Computer Example



Input:
Keyboard,
Mouse, Modem,
Microphone

Output:
LCD, LED,
Modem,
Speakers

Digital Systems (7)

★ Digital Computer

- The most striking property of it is its generality.

- It is programmable because it can follow a sequence of instructions, called a program, that operates on given data where the user can specify and change the program or the data according to the specific need.

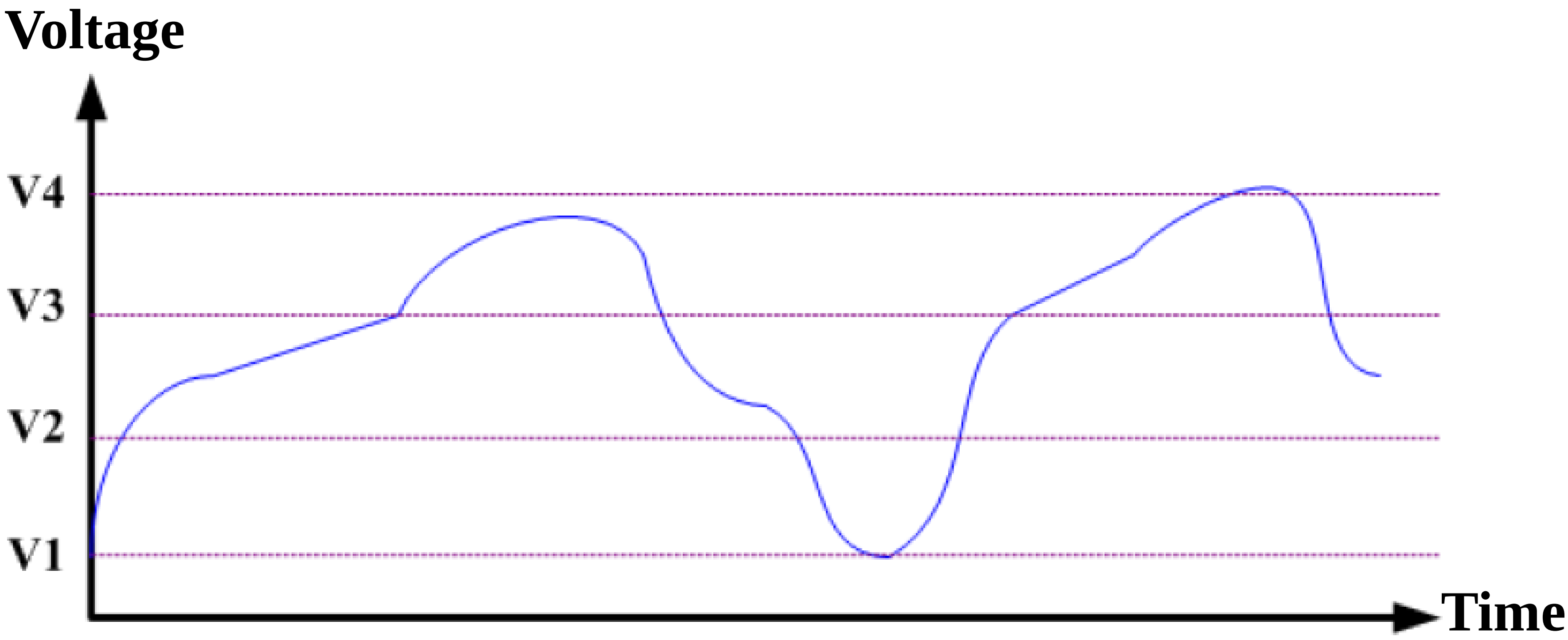
- The supplied data to the digital computer must be in a digital form, but the world around us is analog!!!

- It is common to convert analog parameters into digital form by a process that is called **digitization**.

Digital Systems (8)

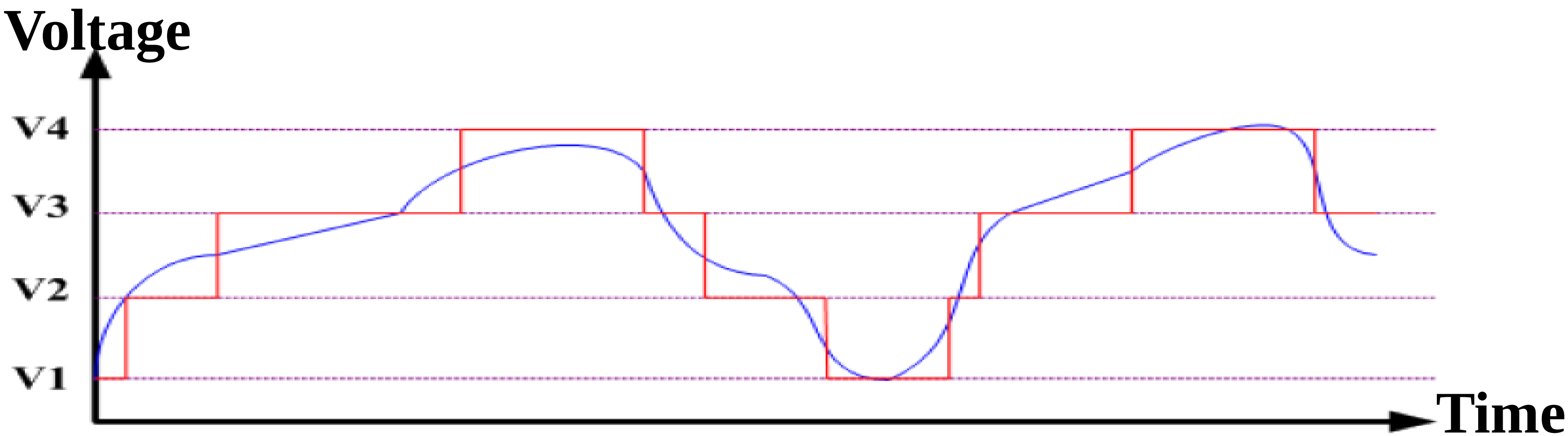
★ Digitization Of Analog Signal

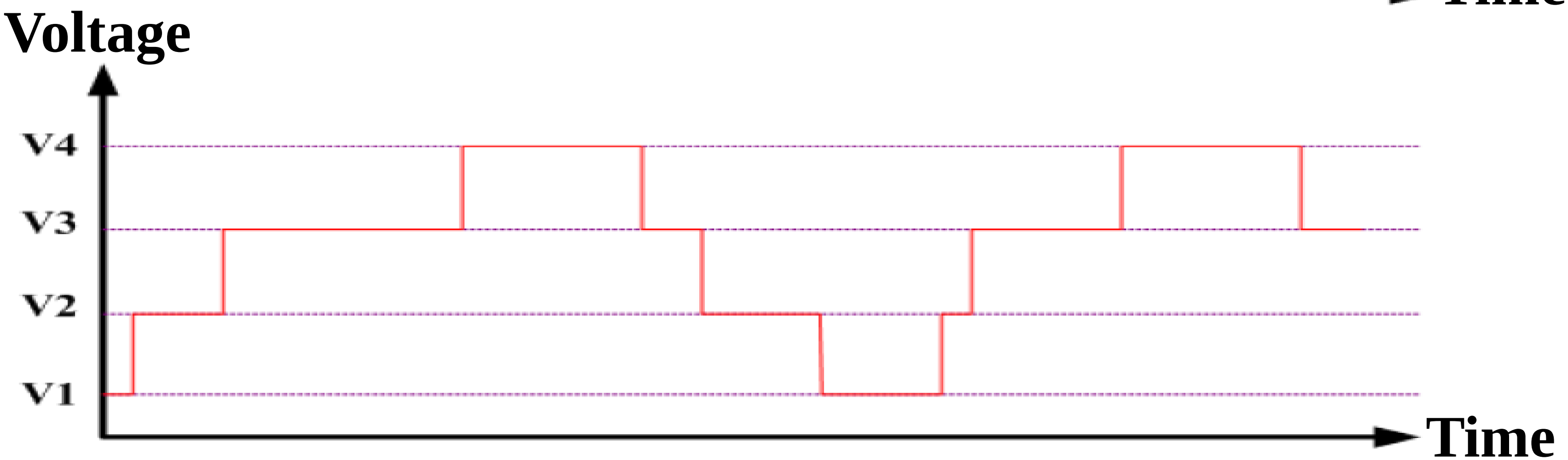
- ❖ **Digitization**: converting an analog signal into digital form.
- ❖ Example: consider digitizing an analog voltage signal.
- ❖ Digitized output is limited to four values = {V1,V2,V3,V4}.



Digital Systems (9)

★ Digitization Of Analog Signal





- ❖ **Some loss of accuracy, why?**
- ❖ **How to improve accuracy?** **Add more voltage values**

Digital Systems (10)

★ Information Representation In Digital System

- ❖ An information variable is represented by a physical electrical quantity (voltage or current) called signal.

- ❖ The electrical signals in most present-day digital systems use just two discrete values and are therefore said to be binary.

- ❖ Binary values are represented abstractly by:
  - ➢ Words (Symbols) Low (L) and High (H)
  - ➢ Words (Symbols) False (F) and True (T)
  - ➢ Words Off and On
  - ➢ Binary Digits 0 and 1

- ❖ Any group of binary digits (bits) can be used to represent any information of any type (number, character, text, image, audio, video).

Digital Systems (11)

- **Information Representation In Digital System**
  - **What are other physical quantities represent 0 and 1?**

CPU	Electrical Voltage
Disk	Magnetic Field Direction
CD	Surface Pits/Light
Dynamic RAM	Electrical Charge

Digital Systems (12)

★ **Information Representation In Digital System**

❖ **Electrical binary signal with two logic values: 0 or 1**

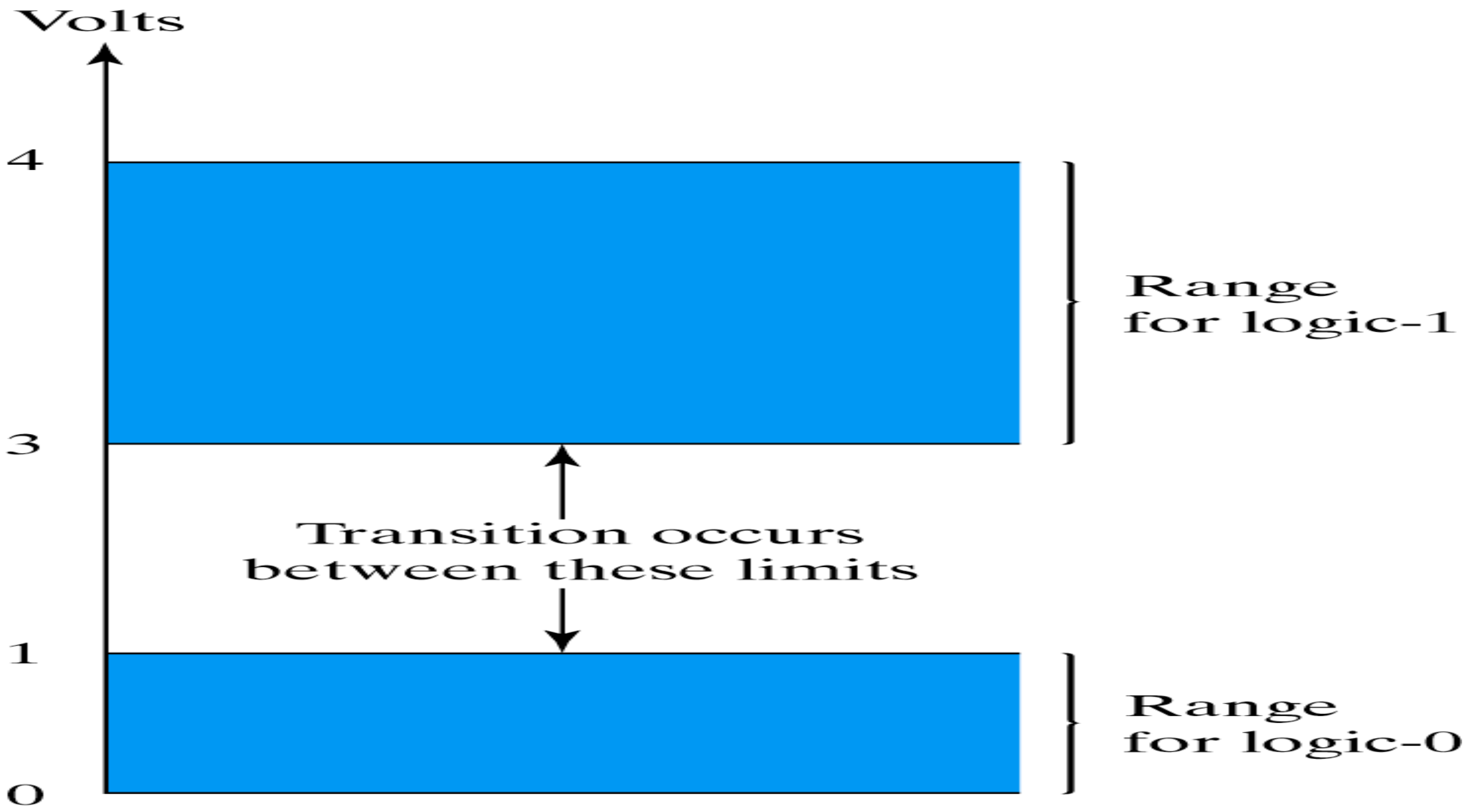
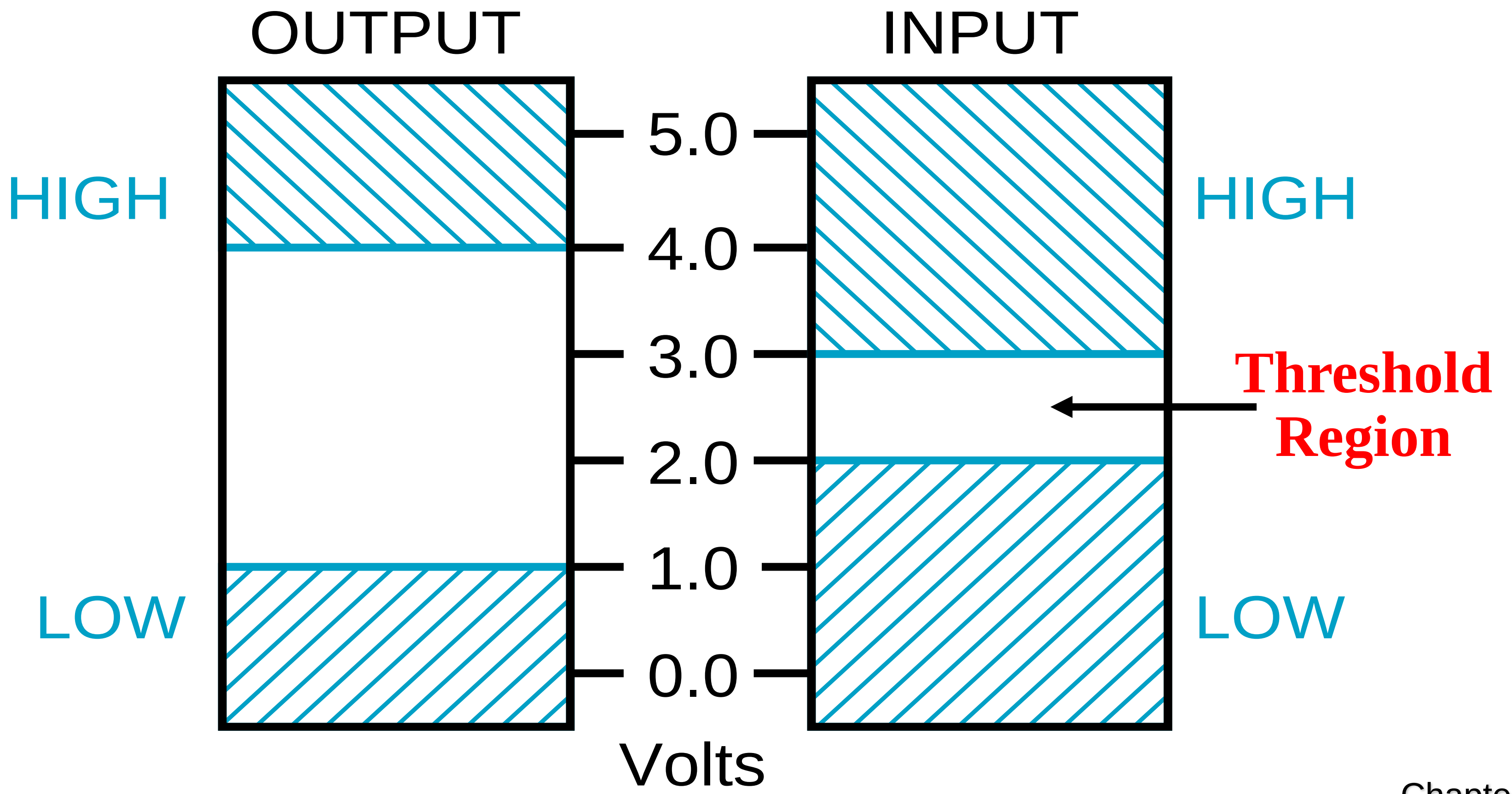


Fig. 1-3 Example of binary signals

Digital Systems (13)

★ **Information Representation In Digital System**

❖ **Electrical Binary Signal Example**



Binary Codes

- Digital systems represent and manipulate not only numbers, but also many other discrete elements of information.

- Any discrete element of information that is distinct among a group of elements can be represented with a unique binary code (i.e. , a pattern of 0 's and 1's).

- The minimum number of bits required to code 2^n distinct elements is n.

  (i.e. , set of four elements can be coded with two bits, with each element assigned one unique of following bits combinations: 00,01,10,11)

Binary Codes (2)

- Non-numeric Binary Codes
  - Example: A binary code for the seven colors of the rainbow
  - Code 100 is not used

Color	Binary Number
Red	000
Orange	001
Yellow	010
Green	011
Blue	101
Indigo	110
Violet	111

Binary Coded Decimal (BCD)

- ★ **Weighted code where each digit is represented in 4-bit.**
- ★ **The weights are 8,4,2,1.**
- ★ **There are six invalid code words:**

1010, 1011, 1100, 1101, 1110, 1111

❖ **For example: To represent $(945)_{10}$ in BCD**

$$\begin{array}{ccc} 9 & 4 & 5 \\ \downarrow & \downarrow & \downarrow \\ 1001 & 0100 & 0101 \end{array}$$

$$\therefore\ (945)_{10} = (100101000101)_{BCD}$$

Binary Coded Decimal (BCD) (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

Warning: Conversion or Coding?

- Do NOT mix up conversion of a decimal number to a binary number with coding a decimal number with a binary code
- $13_{10} = 1101_2$ (This is conversion)
- $13_{BCD} \Leftrightarrow 00010011$ (This is coding)
- In general, coding requires more bits than conversion.
- A number with *n* decimal digits is coded with 4*n* bits in BCD.

Is BCD Useful?

- **Disadvantage**
  - **The representation of a decimal number in BCD needs more bits than its equivalent binary value when the decimal number isn't between 0 and 9.**

- **Advantages**
  - **BCD numbers are decimal numbers coded with binary symbols, so they are more convenient to the computer users whose their inputs and outputs are decimal numbers.**

  - **The need to remember the binary equivalent of decimal numbers from 0 to 9 only.**

Excess-3 (ex-3)

- Excess-three (ex-3) is **unweighted** code to represent a number.
- (ex-3) is like (BCD) in the way of representing a digit.
- Each digit is represented in 4-bit, except that each digit is firstly incremented by 3

❖ For example: to represent $(945)_{10}$ in ex-3

$$\begin{array}{ccc} 9 & 4 & 5 \\ \downarrow & \downarrow & \downarrow \\ 12 & 7 & 8 \\ \downarrow & \downarrow & \downarrow \\ 1100 & 0111 & 1000 \end{array}$$

$$\therefore \ (945)_{10} = (110001111000)_{ex-3}$$

Other Binary Codes

- Other binary codes also assign 4-bit code to 10 decimal digits.
- Each code uses only 10 combinations out of 16 to represent 10 decimal digits from 0 to 9.
- 2421 and 8,4,-2,-1 are also weighted codes as BCD.
- 2421, Excess-3 and 8,4,-2,-1 are self-complementing codes while BCD is NOT.
- Self-complementing property means that the 9's complement of a decimal number is obtained directly by changing 1's to 0's and 0's to 1's.
- $(395)_{10}$ is represented in the excess-3 code as 0110 1100 1000 and the 9 's complement of 395 is $(604)_{10}$ which is represented in excess-3 code as 1001 0011 0111.

Other Binary Codes (2)

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

Gray Code

- ★ As we count up/down using binary codes, the number of bits that change from one binary value to the next varies

000 → 001 (1-bit change)
001 → 010 (2-bit change)
011 → 100 (3-bit change)

- ★ Gray code: only 1 bit changes as we count up or down

- ★ Gray code can be used in low-power logic circuits that count up or down, because only 1 bit changes per count.

Digit	Binary	Gray Code
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101

- ★ Error correction during transmission of gray-coded numbers is easier than using other binary codes.

Alphanumeric Codes

- The alphanumeric characters set is a set of 128 elements that includes 10 decimal digits, 52 letters of the alphabet (uppercase & lowercase), 32 printable symbols (%,$,#,…) and 34 non-printable special characters (Ctrl, Alt, Shift,…).

- Alphanumeric characters set encoding:
  - Standard ASCII: 7-bit character codes (0 – 127).
  - ASCII is an abbreviation of "American Standard Code for Information Interchange".
  - Extended ASCII: 8-bit character codes (0 – 255).
  - Unicode: 16-bit character codes (0 – 65,535)
    - Unicode standard represents a universal character set.
    - Defines codes for characters used in all languages.

Alphanumeric Codes (2)

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	NULL	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 Code

- ★ Binary data are typically transmitted between computers.
- ★ Because of noise, a corrupted bit will change value.
- ★ To detect errors, extra bits are added to each data value.
- ★ Parity bit: is used to make the number of 1's odd or even.
- ★ Even parity: number of 1's in the transmitted data is even.
- ★ Odd parity: number of 1's in the transmitted data is odd.

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

Error Detecting Code (2)

Parity bit

Odd parity Message	P	Even parity 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

Error Detecting Code (3)

Sender → Receiver

7-bit ASCII character + 1 Parity bit

Sent ‘A’=01000001 , Received ‘A’=01000101

- Suppose we are transmitting 7-bit ASCII characters
- A parity bit is added to each character to make it 8 bits
- Parity can detect all single-bit errors.
  - If even parity is used and a single bit changes, it will change the parity to odd, which will be detected at the receiver end.
  - The receiver end can detect the error, but cannot correct it because it does not know which bit is erroneous.
- Can also detect some multiple-bit errors
  - Error in an odd number of bits.
- Cannot detect an even number of erroneous bits, so additional error detection codes may be needed to take care of that possibility.

Binary Logic

- We study binary logic as a foundation for analyzing and designing digital systems.
- Binary logic consists of binary logical variables and a set of binary logical operations.
- Binary logical variables take only one of two discrete values:1 or 0.
- Binary logical variables are designated by letters of the alphabet, such as A, B, C, x, y, z, etc.

Logical Operations

- There are three basic binary logical operations: **AND**, **OR** and **NOT**.
- Each operation produces a binary result of **1** or **0**.
- AND is denoted by a dot (·).
- OR is denoted by a plus (+).
- NOT is denoted by an over bar (¯) the variable.

Notation Examples

- **Examples:**
  - $Y = A \cdot B$ is read **"Y is equal to A AND B."**
  - $Z = X + Y$ is read **"z is equal to x OR y."**
  - $X = \overline{A}$ is read **"X is equal to NOT A."**

- **Note: The statement:**

1 + 1 = 2 (is read "one <u>plus</u> one equals two")

is not the same as

1 + 1 = 1 (is read "1 <u>or</u> 1 equals 1").

Operator Definitions

- Operations are defined on the values "0" and "1" for each operator:

AND

$$0 \cdot 0 = 0$$
$$0 \cdot 1 = 0$$
$$1 \cdot 0 = 0$$
$$1 \cdot 1 = 1$$

OR

$$0 + 0 = 0$$
$$0 + 1 = 1$$
$$1 + 0 = 1$$
$$1 + 1 = 1$$

NOT

$$\overline{0} = 1$$
$$\overline{1} = 0$$

Truth Tables

- Tabular listing of the values of a function for all possible combinations of values on its arguments
- Example: Truth tables for the basic logic operations:

AND		
X	Y	$Z = X \cdot Y$
0	0	0
0	1	0
1	0	0
1	1	1

OR		
X	Y	$Z = X+Y$
0	0	0
0	1	1
1	0	1
1	1	1

NOT	
X	$Z = \overline{X}$
0	1
1	0

Truth Tables – Cont'd

- Used to evaluate any logic function

Consider $F(X, Y, Z) = X\,Y + \overline{Y}\,Z$

X	Y	Z	$X\,Y$	$\overline{Y}$	$\overline{Y}\,Z$	$F = X\,Y + \overline{Y}\,Z$
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

Logic Gates

- The logical operation are represented by logic gates.
- The logic gate is an electronic circuit that operates on one or more input signals to produce an output signal.
- Logic gates have special symbols:

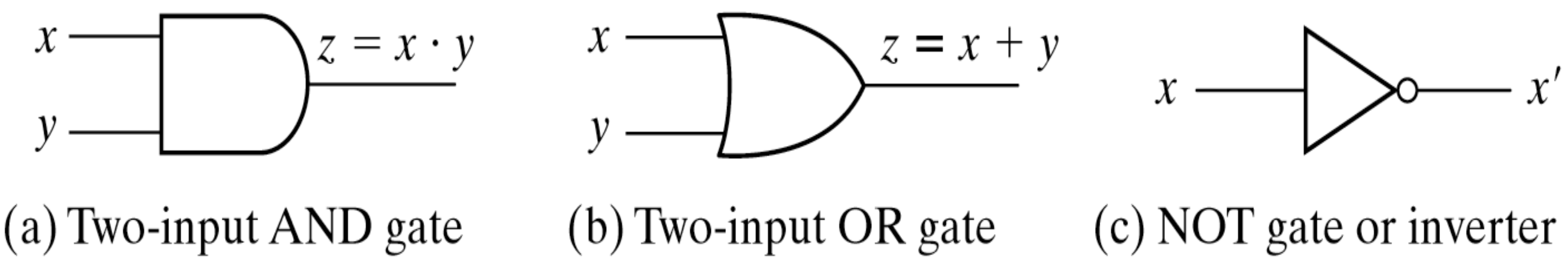
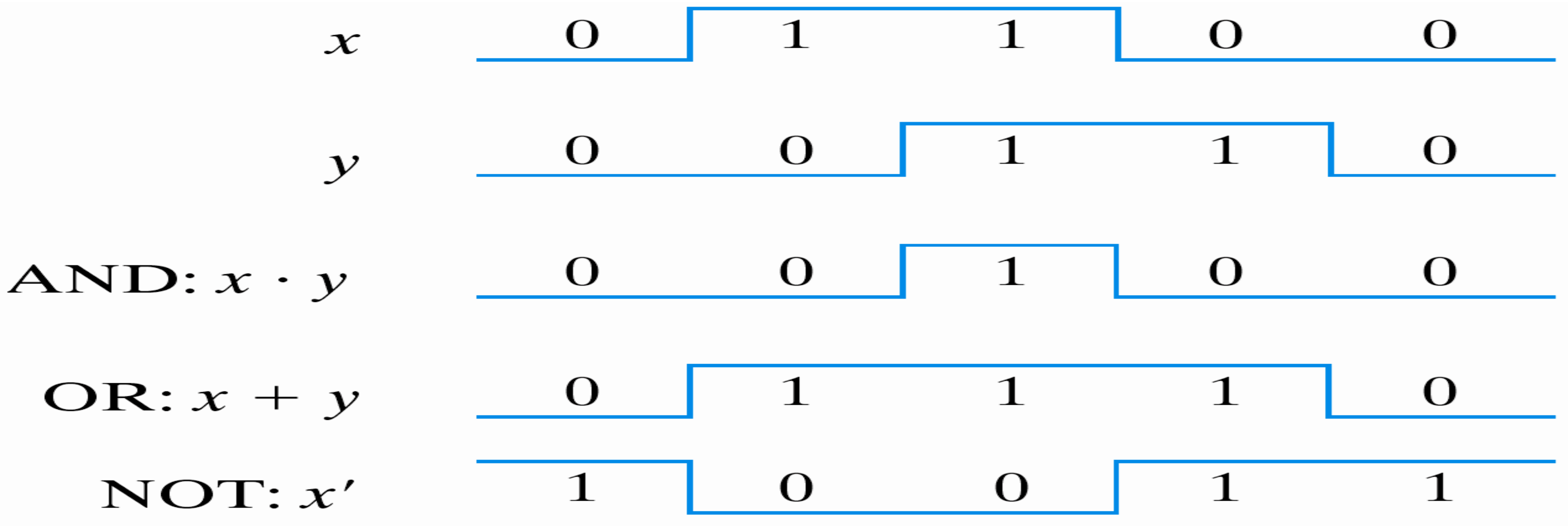
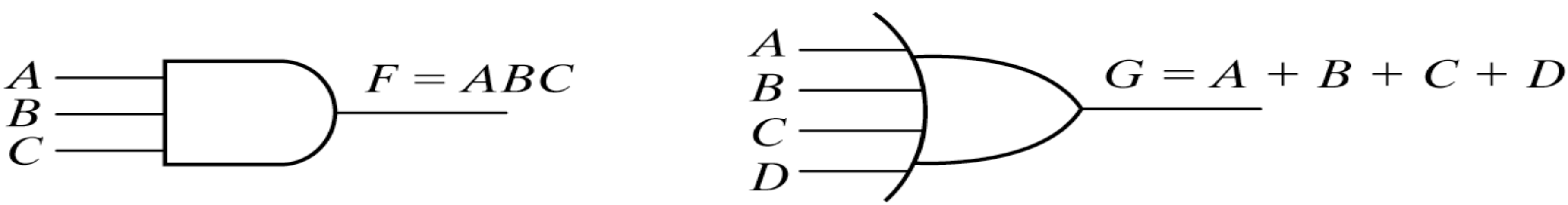


Fig. 1-4 Symbols for digital logic circuits

Logic Gates Behavior





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

Fig. 1-6 Gates with multiple inputs

Logic Diagrams and Expressions

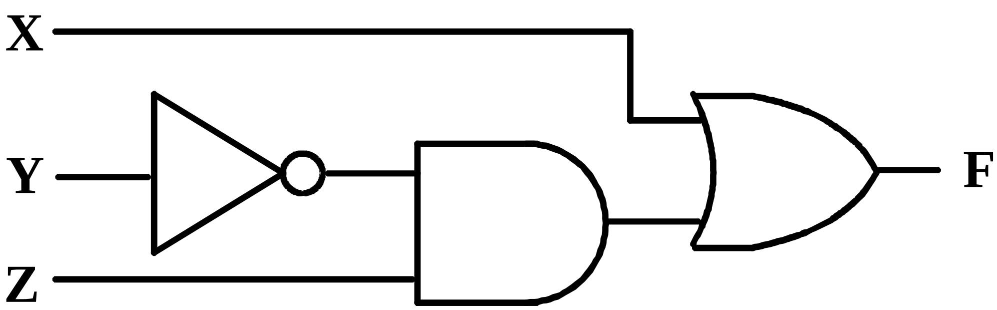
Truth Table

X Y Z	$F = X + \overline{Y} \cdot Z$
0 0 0	0
0 0 1	1
0 1 0	0
0 1 1	0
1 0 0	1
1 0 1	1
1 1 0	1
1 1 1	1

Logic Equation

$$F = X + \overline{Y}\,Z$$

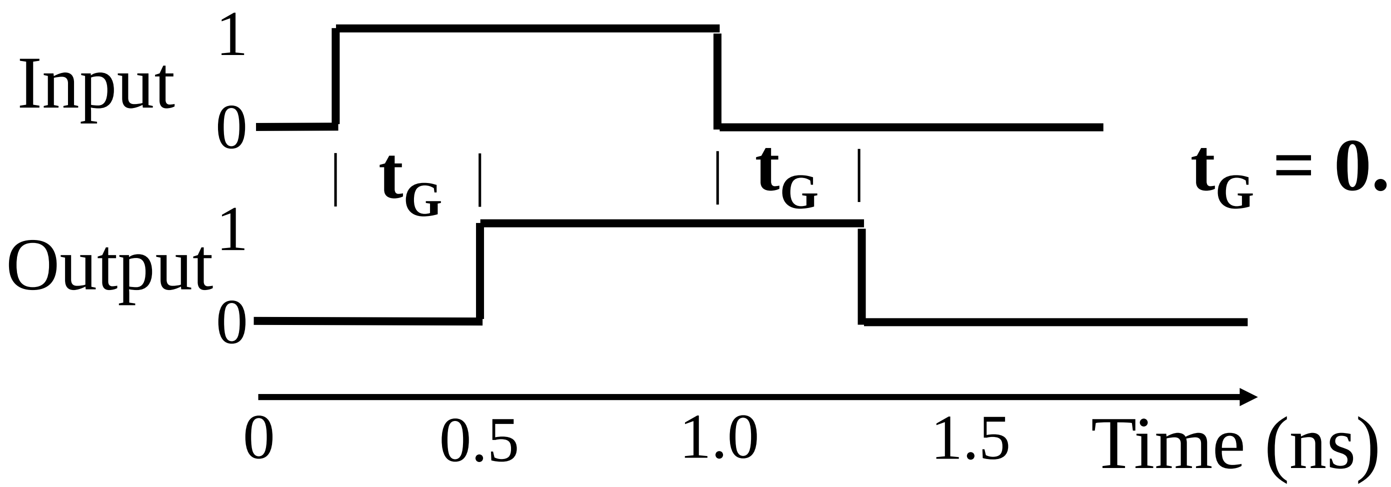
Logic Diagram



- Boolean equations, truth tables and logic diagrams describe the same function.
- Truth tables are <u>unique</u>, but Boolean equations and logic diagrams are not. This gives flexibility in implementing functions.

Logic Gate Delay

- ★ In actual physical gates, if an input changes that causes the output to change, **the output change does not occur instantaneously.**
- ★ The delay between an input change and the output change is the <u>gate delay</u> denoted by tG.



t_G = 0.3 ns

The End

Questions?