Bahria University
Discovering Knowledge

Computer Architecture and Logic Design (CALD)

Lecture 09

Dr. Sorath Hansrajani

Assistant Professor

Department of Software Engineering

Bahria University Karachi Campus

Email: sorathhansrajani.bukc@bahria.edu.pk

Digital Systems and Binary Numbers

These slides were assembled by Mustafa Kemal Uyguroğlu, with grateful acknowledgement of the many others who made their course materials freely available online. Feel free to reuse or adapt these slides for your own academic purposes, provided that you include proper attribution.

Outline of Chapter 1

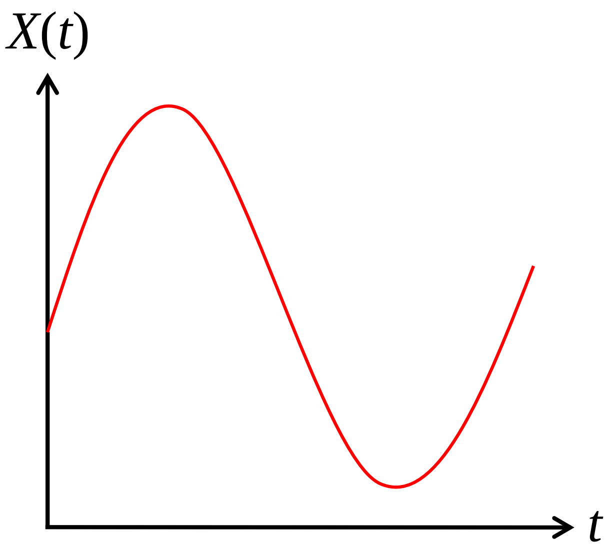
- 1.1 Digital Systems
- 1.2 Binary Numbers
- 1.3 Number-base Conversions
- 1.4 Octal and Hexadecimal Numbers
- 1.5 Complements
- 1.6 Binary Storage and Registers
- 1.7 Binary Logic

Digital Systems and Binary Numbers

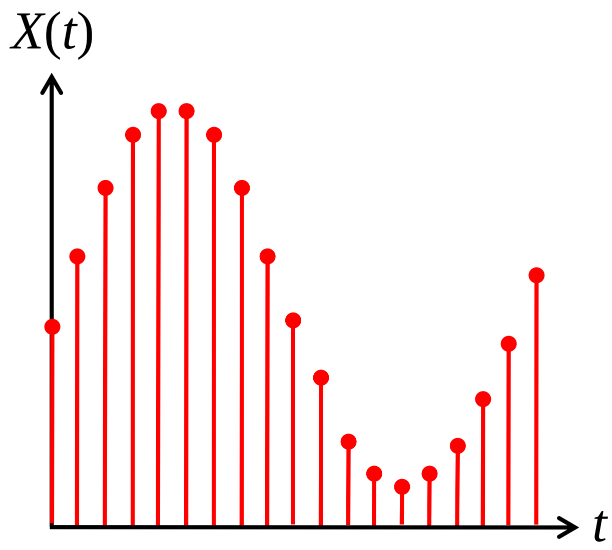
- Digital age and information age
- Digital computers
  - General purposes
  - Many scientific, industrial and commercial applications
- Digital systems
  - Telephone switching exchanges
  - Digital camera
  - Electronic calculators, PDA's
  - Digital TV
- Discrete information-processing systems
  - Manipulate discrete elements of information
  - For example, {1, 2, 3, …} and {A, B, C, …}…

Analog and Digital Signal

- Analog system
  - The physical quantities or signals may vary continuously over a specified range.
- Digital system
  - The physical quantities or signals can assume only discrete values.
  - Greater accuracy



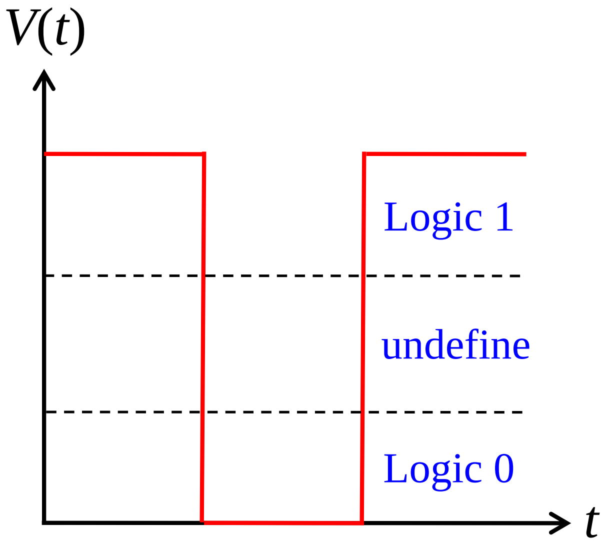
Analog signal



Digital signal

Binary Digital Signal

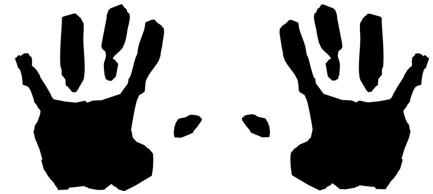
- An information variable represented by physical quantity.
- For digital systems, the variable takes on discrete values.
  - Two level, or binary values are the most prevalent values.
- Binary values are represented abstractly by:
  - Digits 0 and 1
  - Words (symbols) False (F) and True (T)
  - Words (symbols) Low (L) and High (H)
  - And words On and Off
- Binary values are represented by values or ranges of values of physical quantities.



Binary digital signal

Decimal Number System

- Base (also called radix) = 10
  - 10 digits { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 }
- Digit Position
  - Integer & fraction
- Digit Weight
  - Weight = $(Base)^{Position}$
- Magnitude
  - Sum of "*Digit* x *Weight*"
- Formal Notation



2	1	0		-1	-2
5	1	2	.	7	4
100	10	1		0.1	0.01
			.		
500	10	2		0.7	0.04

$$d_2{*}B^2+d_1{*}B^1+d_0{*}B^0+d_{-1}{*}B^{-1}+d_{-2}{*}B^{-2}$$

$$(512.74)_{10}$$

Octal Number System

- Base = 8
  - 8 digits { 0, 1, 2, 3, 4, 5, 6, 7 }
- Weights
  - Weight = (*Base*) $^{\textit{Position}}$
- Magnitude
  - Sum of "*Digit* x *Weight*"
- Formal Notation

64	8	1		1/8	1/64
5	1	2	.	7	4
2	1	0		-1	-2

$$5 *8^{2}+1 *8^{1}+2 *8^{0}+7 *8^{-1}+4 *8^{-2}$$

$$=(330.9375)_{10}$$

$$(512.74)_{8}$$

Binary Number System

- Base = 2
  - 2 digits { 0, 1 }, called *b*inary dig*its* or "*bits*"
- Weights
  - Weight = (*Base*) ^{Position}
- Magnitude
  - Sum of "*Bit* x *Weight*"
- Formal Notation
- Groups of bits   4 bits = *Nibble*

  8 bits = *Byte*

4	2	1		1/2	1/4
1	0	1	•	0	1
2	1	0		-1	-2

$1 * 2^{2} + 0 * 2^{1} + 1 * 2^{0} + 0 * 2^{-1} + 1 * 2^{-2}$

$= (5.25)_{10}$

$(101.01)_{2}$

1 0 1 1

1 1 0 0 0 1 0 1

Hexadecimal Number System

- Base = 16
  - 16 digits { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F }
- Weights
  - Weight = $(Base)^{Position}$
- Magnitude
  - Sum of "*Digit* x *Weight*"
- Formal Notation

256	16	1		1/16	1/256
1	E	5	•	7	A
2	1	0		-1	-2

$$1 * 16^2 + 14 * 16^1 + 5 * 16^0 + 7 * 16^{-1} + 10 * 16^{-2}$$

$$=(485.4765625)_{10}$$

$$(1E5.7A)_{16}$$

The Power of 2

n	2^n
0	$2^0=1$
1	$2^1=2$
2	$2^2=4$
3	$2^3=8$
4	$2^4=16$
5	$2^5=32$
6	$2^6=64$
7	$2^7=128$

n	2^n	
8	$2^8=256$	
9	$2^9=512$	
10	$2^{10}=1024$	Kilo
11	$2^{11}=2048$	
12	$2^{12}=4096$	
20	$2^{20}=1M$	Mega
30	$2^{30}=1G$	Giga
40	$2^{40}=1T$	Tera

Addition

- Decimal Addition

	1	1		← Carry
		5	5	
+		5	5	
	1	1	0	

Binary Addition

- Column Addition

	1	1	1	1	1	1		
		1	1	1	1	0	1	= 61
+			1	0	1	1	1	= 23
	1	0	1	0	1	0	0	= 84

Binary Subtraction

- Borrow a “Base” when needed

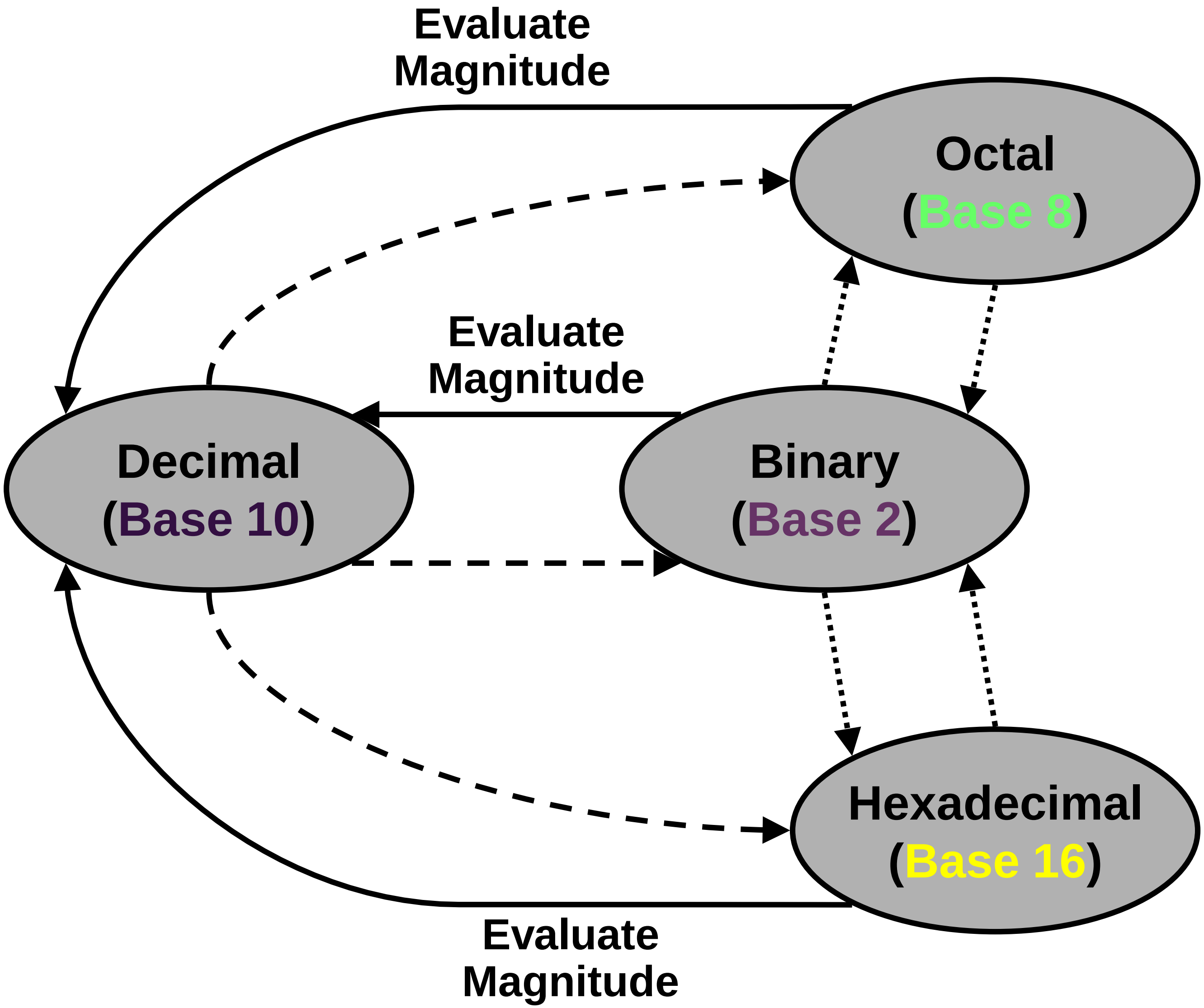
$$
\begin{array}{cccccccccl}
 & & 1 & & & 2 & & & & = (10)_2 \\
 & 0 & \cancel{2} & 2 & 0 & 0 & 2 & & & \\
 & \cancel{1} & 0 & 0 & \cancel{1} & \cancel{1} & 0 & 1 & & = 77 \\
- & & & 1 & 0 & 1 & 1 & 1 & & = 23 \\
\hline
 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & & = 54
\end{array}
$$

Binary Multiplication

- Bit by bit

				1	0	1	1	1
x					1	0	1	0
				0	0	0	0	0
			1	0	1	1	1	
		0	0	0	0	0		
	1	0	1	1	1			
	1	1	1	0	0	1	1	0

Number Base Conversions



Decimal (*Integer*) to Binary Conversion

- 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 (*Fraction*) to Binary Conversion

- 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

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

Example:

Assume Zeros

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

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

Works **<u>both</u>** **ways (*Binary* to *Octal* & *Octal* to *Binary*)**

Binary – Hexadecimal Conversion

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

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

Example:

Assume Zeros

$$(\ \underbrace{1}\ \underbrace{0\,1\,1\,0}\ .\ \underbrace{0\,1}\ )_2$$

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

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

Octal – Hexadecimal Conversion

- Convert to Binary as an intermediate step

Example:

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

Assume Zeros

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

Assume Zeros

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

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

Decimal, Binary, Octal and Hexadecimal

Decimal	Binary	Octal	Hex
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

1.5 Complements

- There are two types of complements for each base-r system: the radix complement and diminished radix 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:

$$(r^n - 1) - N$$

- **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 is 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 is 1111111–1011000 = 0100111

- **Observation:**
  - Subtraction from $(r^n - 1)$ will never require a borrow
  - Diminished radix complement can be computed digit-by-digit
  - For binary: 1 – 0 = 1 and 1 – 1 = 0

Complements

- 1's Complement (*Diminished Radix* Complement)
  - All '0's become '1's
  - All '1's become '0's

  Example $(10110000)_2$

  ⇨ $(01001111)_2$

  If you add a number and its 1's complement …

$$
\begin{array}{r}
1\,0\,1\,1\,0\,0\,0\,0 \\
+\;\;0\,1\,0\,0\,1\,1\,1\,1 \\
\hline
1\,1\,1\,1\,1\,1\,1\,1
\end{array}
$$

Complements

- 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 $N = 0$. Comparing with the $(r - 1)$'s complement, we note that the r's complement is obtained by adding 1 to the $(r - 1)$'s complement, since $r^n - N = [(r^n - 1) - N] + 1$.

- Example: Base-10

The 10's complement of 012398 is 987602
The 10's complement of 246700 is 753300

- Example: Base-2

The 2's complement of 1101100 is 0010100
The 2's complement of 0110111 is 1001001

Complements

- 2's Complement (*Radix* Complement)
  - Take 1's complement then add 1
  - Toggle all bits to the left of the first '1' from the right

OR

Example:

Number:	1 0 1 1 0 0 0 0	1 0 1 1 0 0 0 0
1's Comp.:	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

Complements

- **Subtraction with Complements**
  - The subtraction of two n-digit unsigned numbers $M - N$ in base r can be done as follows:

1. Add the minuend M to the r's complement of the subtrahend N. Mathematically, $M + (r^n - N) = M - N + r^n$.
2. If $M \geqq N$, the sum will produce and end carry r^n, which can be discarded; what is left is the result $M - N$.
3. If $M < N$, the sum does not produce an end carry and is equal to $r^n - (N - M)$, which is the r's complement of $(N - M)$. To obtain the answer in a familiar form, take the r's complement of the sum and place a negative sign in front.

Complements

- Example 1.5
  - Using 10's complement, subtract 72532 – 3250.

$$
\begin{aligned}
M &= \quad 72532 \\
\text{10's complement of } N &= \underline{+96750} \\
\text{Sum} &= \quad 169282 \\
\text{Discard end carry } 10^5 &= \underline{-100000} \\
\text{Answer} &= \quad 69282
\end{aligned}
$$

- Example 1.6
  - Using 10's complement, subtract 3250 – 72532.

$$
\begin{aligned}
\text{M} &= \quad 03250 \\
\text{10's complement of N} &= \underline{+27468} \\
\text{Sum} &= \quad 30718
\end{aligned}
$$

There is no end carry.

Therefore, the answer is – (10's complement of 30718) = – 69282.

Complements

- Example 1.7
  - Given the two binary numbers X = 1010100 and Y = 1000011, perform the subtraction (a) $X - Y$; and (b) $Y - X$, by using 2's complement.

$$
\begin{array}{lrr}
\text{(a)} & X = & 1010100 \\
 & \text{2's complement of } Y = & \underline{+0111101} \\
 & \text{Sum} = & 10010001 \\
 & \text{Discard end carry } 2^7 = & \underline{-10000000} \\
 & \text{Answer. } X - Y = & 0010001
\end{array}
$$

$$
\begin{array}{lrr}
\text{(b)} & Y = & 1000011 \\
 & \text{2's complement of } X = & +\underline{0101100} \\
 & \text{Sum} = & 1101111
\end{array}
$$

There is no end carry. Therefore, the answer is Y – X = – (2's complement of 1101111) = – 0010001.

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.

- Example 1.8
  - Repeat Example 1.7, but this time using 1's complement.

$$
\begin{array}{lrr}
\text{(a)} \quad X - Y = 1010100 - 1000011 & & \\
 & X = & 1010100 \\
 & \text{1's complement of } Y = & \pm\,0111100 \\
 & \text{Sum} = & 10010000 \\
 & \text{End-around carry} = & \underline{+ \qquad 1} \\
 & \text{Answer. } X - Y = & 0010001
\end{array}
$$

$$
\begin{array}{lrr}
\text{(b)}\, Y - X = 1000011 - 1010100 & & \\
 & Y = & 1000011 \\
 & \text{1's complement of } X = & \underline{+\,0101011} \\
 & \text{Sum} = & 1101110
\end{array}
$$

There is no end carry, Therefore, the answer is Y – X = – (1's complement of 1101110) = – 0010001.

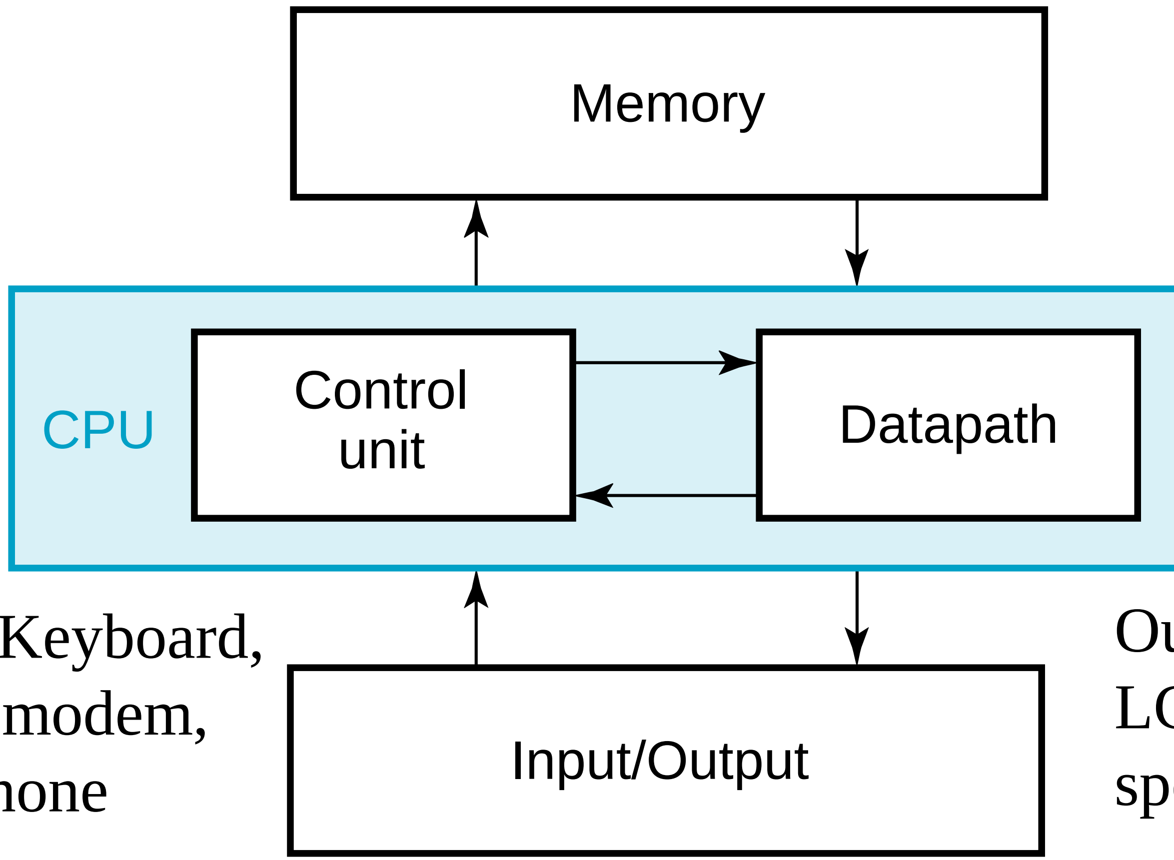
1.8 Binary Storage and Registers

- Registers
  - A binary cell is a device that possesses two stable states and is capable of storing one of the two states.
  - A register is a group of binary cells. A register with *n* cells can store any discrete quantity of information that contains *n* bits.

n cells → 2^n possible states

- A binary cell
  - Two stable state
  - Store one bit of information
  - Examples: flip-flop circuits, capacitor
- A register
  - A group of binary cells
  - AX in x86 CPU
- Register Transfer
  - A transfer of the information stored in one register to another.
  - One of the major operations in digital system.

A Digital Computer Example



Inputs: Keyboard, mouse, modem, microphone

Outputs: CRT, LCD, modem, speakers

Synchronous or Asynchronous?

Transfer of information

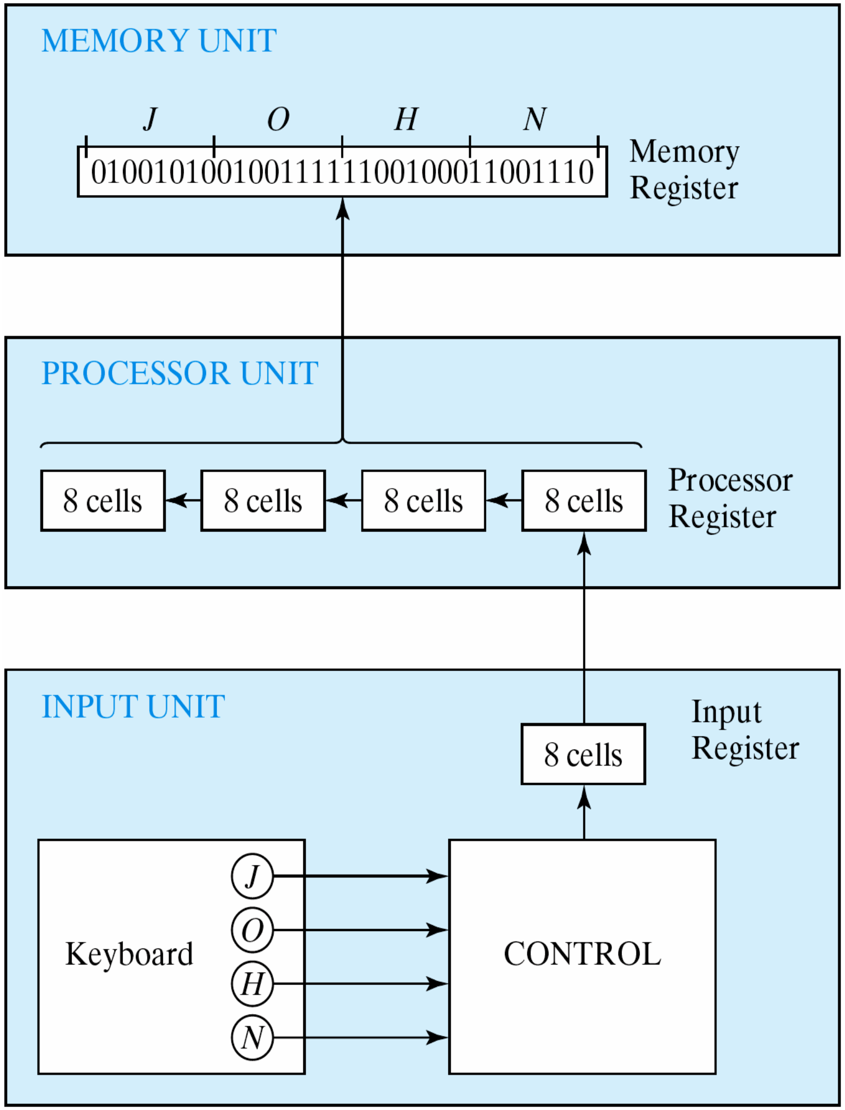


Figure 1.1 Transfer of information among register

Transfer of information

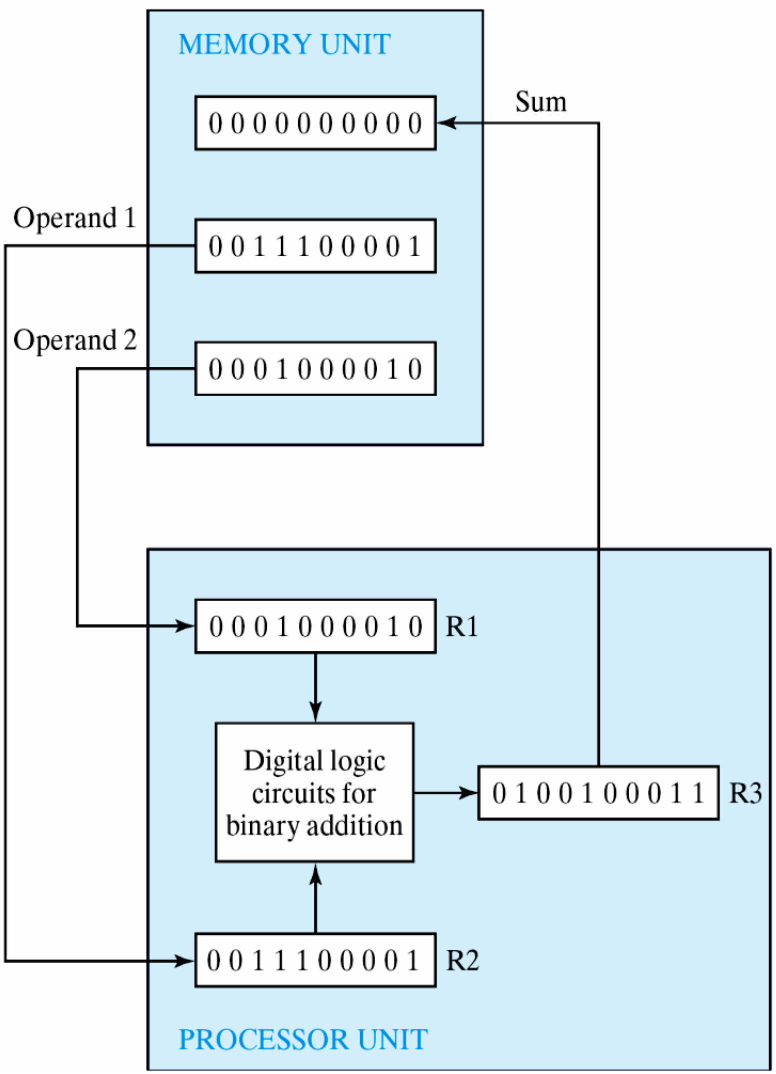


Figure 1.2 Example of binary information processing

- The other major component of a digital system
  - Circuit elements to manipulate individual bits of information
  - Load-store machine

    LD R1;

    LD R2;

    ADD R3, R2, R1;

    SD R3;

1.9 Binary Logic

- Definition of Binary Logic
  - Binary logic consists of binary variables and a set of logical operations.
  - The variables are designated by letters of the alphabet, such as A, B, C, x, y, z, etc, with each variable having two and only two distinct possible values: 1 and 0,
  - Three basic logical operations: AND, OR, and NOT.

1.9 Binary Logic

1. AND: This operation is represented by a dot or by the absence of an operator. For example, $x \cdot y = z$ or $xy = z$ is read "x AND y is equal to z," The logical operation AND is interpreted to mean that $z = 1$ if only $x = 1$ and $y = 1$; otherwise $z = 0$. (Remember that x, y, and z are binary variables and can be equal either to 1 or 0, and nothing else.)
2. OR: This operation is represented by a plus sign. For example, $x + y = z$ is read "x OR y is equal to z," meaning that $z = 1$ if $x = 1$ or $y = 1$ or if both $x = 1$ and $y = 1$. If both $x = 0$ and $y = 0$, then $z = 0$.
3. NOT: This operation is represented by a prime (sometimes by an overbar). For example, $x' = z$ (or $\bar{x} = z$) is read "not x is equal to z," meaning that z is what z is not. In other words, if $x = 1$, then $z = 0$, but if $x = 0$, then $z = 1$, The NOT operation is also referred to as the complement operation, since it changes a 1 to 0 and a 0 to 1.

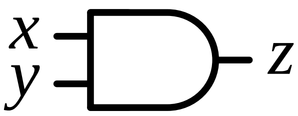
Binary Logic

- Truth Tables, Boolean Expressions, and Logic Gates

AND

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

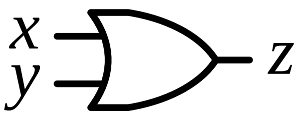
$$z = x \bullet y = x\,y$$



OR

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

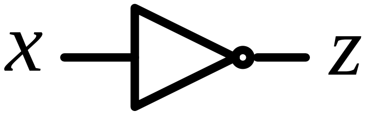
$$z = x + y$$



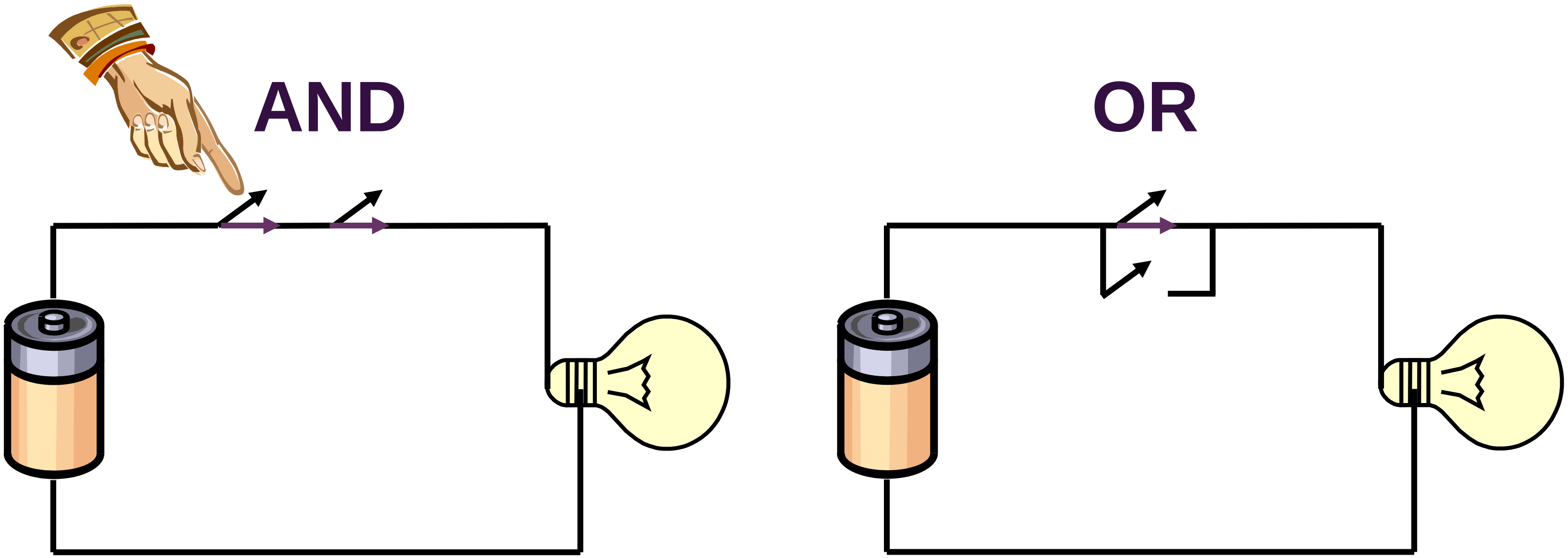
NOT

x	z
0	1
1	0

$$z = \overline{x} = x'$$

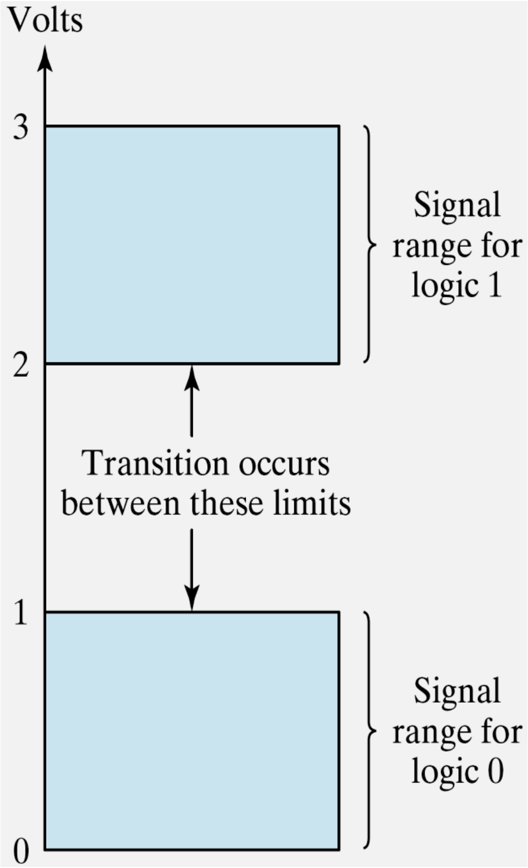


Switching Circuits



Binary Logic

- Logic gates
  - Example of binary signals



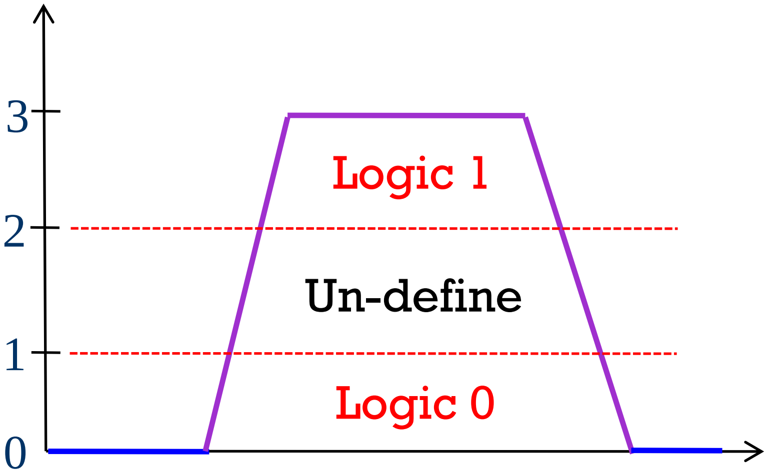
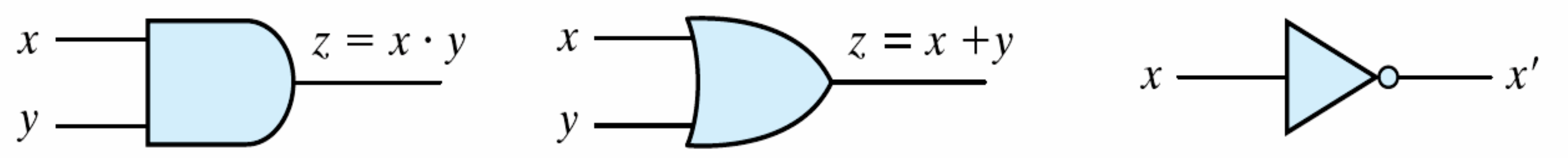


Figure 1.3 Example of binary signals

Binary Logic

- Logic gates
  - Graphic Symbols and Input-Output Signals for Logic gates:



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

Fig. 1.4 Symbols for digital logic circuits

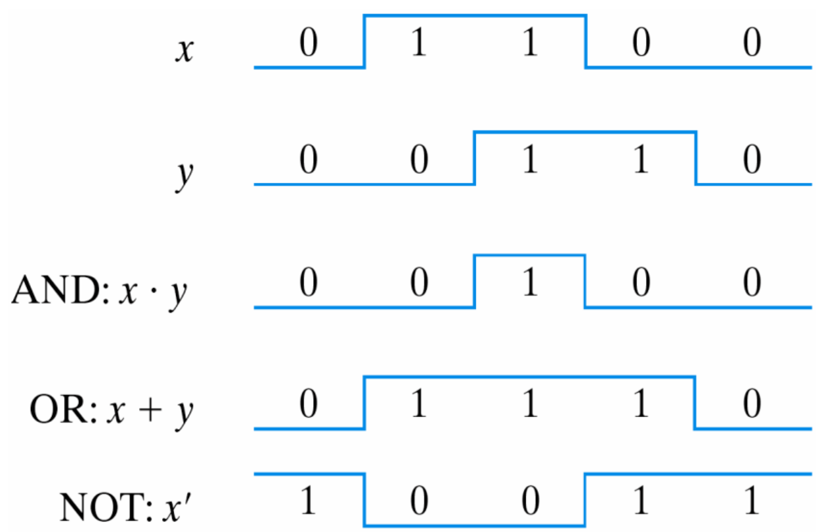
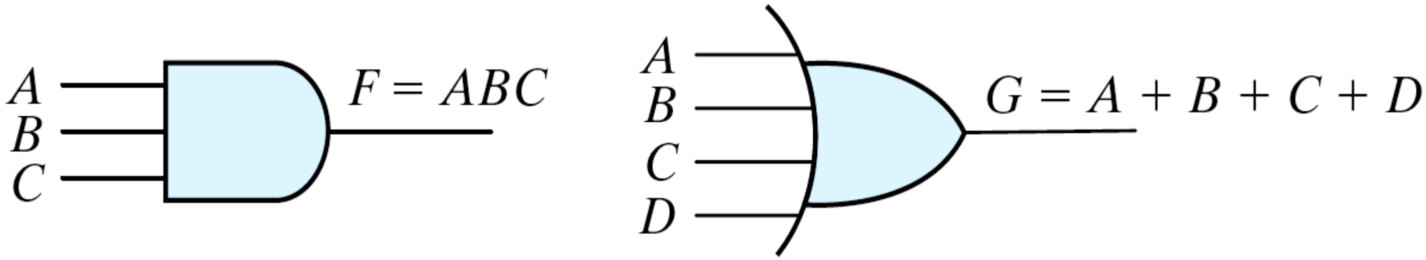


Fig. 1.5 Input-Output signals for gates

Binary Logic

- Logic gates
  - Graphic Symbols and Input-Output Signals for Logic gates:



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

Fig. 1.6 Gates with multiple inputs