Lecture (03)
Binary Codes
Registers and Logic Gates

By:

Dr. Ahmed ElShafee

Binary Codes

- Digital systems use signals that have two distinct values and circuit elements that have two stable states.
- binary number of *n* digits, for example represented by *n* binary circuit elements, each having an output signal equivalent to 0 or 1.
- Digital systems many also presents other discrete elements of information that is distinct among a group of quantities can be represented with a binary code.

- A set of four elements can be coded with two bits, with each element assigned one of the following bit combinations: 00, 01, 10, 11.
- A set of eight elements requires a three-bit code
- A set of 16 elements requires a four-bit code
- The bit combination of an *n*-bit code is determined from the count in binary from 0 to $2^n - 1$

Binary-Coded Decimal Code

- *binary-coded decimal* and is commonly referred to as BCD.

number with *k* decimal digits will require 4*k* bits in BCD

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

Decimal	BCD	Binary
396	0011 1001 0110	110001100
$(185)_{10}$ =	$(0001\ 1000\ 0101)_{BCD}$	$(10111001)_2$

- The BCD value has 12 bits to encode the characters of the decimal value, but the equivalent binary number needs only 8 bits.
- BCD number needs more bits than its equivalent binary value.
- The advantage of BCD, is that the computer input and output data are generated by people who use the decimal system.

BCD Addition

- When the binary sum is equal to or less than 1001_{bcd} 9_{10} (without a carry), the corresponding BCD digit is correct.
- However, when the binary sum is greater than or equal to 1010_{bcd}, 10_{10} the result is an invalid BCD digit
- Add $6_{10} = (0110)_2$ to the binary sum converts it to the correct digit and also produces a carry as required.

-

4	0100	4	0100	8	1000
+5	+0101	+8	+1000	+9	1001
9	1001	12	1100	17	10001
			+0110		+0110
			000 10010		000 10111

- 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

Decimal Arithmetic & signed numbers in BCD

- Uses signed-magnitude system or signed-complement system.
- The sign of a decimal number is usually represented with four bits to conform to the four-bit code of the decimal digits.
- designate a plus with four 0's and a minus with the BCD equivalent of 9, which is 1001.
- The signed-complement system can be either the 9's or the 10's complement, but the 10's complement is the one most often used.
- 10's complement of a BCD number, we first take the 9's complement and then add 1 to the least significant digit
- For subtraction, Take the 10's complement of the subtrahend and add it to the minuend.

- (+375) + (-240) =
- Find 10's-complement of -240 = 9 760
-

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

Other Decimal Codes

- Binary codes for decimal digits require a minimum of four bits per digit.
- Each code uses only 10 out of a possible 16 bit combinations that can be arranged with four bits.
- The other six unused combinations have no meaning and should be avoided.

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

- BCD and the 2421 , and 84-2-1 code are examples of weighted codes. In a weighted code, each bit position is assigned a weighting factor in such a way that each digit can be evaluated by adding the weights of all the 1's in the coded combination.
- BCD code has weights of 8, 4, 2, and 1.
- which correspond to the power-of-two values of each bit.
- 0110 = 8 * 0 + 4 * 1 + 2 * 1 + 1 * 0 = 6.
- 2421 code has weights of 2, 4, 2, and 1.
- 2 * 1 + 4 * 1 + 2 * 0 + 1 * 1 = 7.
- 84-2-1,weights of 8, 4, -2, -1
- 8 * 0 + 4 * 1 + (-2) * 1 + (-1) * 0 = 2.

- The 2421 and the excess-3 codes are examples of self-complementing codes.
- Such codes have the property 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
- $395_{10} = (0110\ 1100\ 1000)_{excess\text{-}3}$
- 2's complement
- $1001\ 0011\ 0111_{excess\text{-}3} = 604_{10}$

Gray Code

- The output data of many physical systems are quantities that are continuous.
- Continuous or analog information is converted into digital form by means of an analog- to-digital converter
- It is sometimes convenient to use the Gray code to represent digital data that have been converted from analog data
- advantage of the Gray code that only one bit in the code group changes in going from one number to the next
- going from 7 to 8, the Gray code changes from 0100 to 1100

-

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

- The standard binary code for the alphanumeric characters is the American Standard Code for Information Interchange (ASCII),
- which uses seven bits to code 128 characters,
- The seven bits of the code are designated by *b*1 through *b*7,
- with *b*7 the most significant bit. The letter *A*, for example, is represented in ASCII a 1000001 (column 100, row 0001).
- ASCII code also contains 94 printed graphic characters and 34 nonprinting characters used for various control functions

- The graphic characters consist of the
  - 26 uppercase letters (A through Z), the
  - 26 lowercase letters (a through z), the
  - 10 numerals (0 through 9), and
  - 32 special printable characters, such as %, *, and $.

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

Control Characters

NUL	Null	DLE	Data-link escape
SOH	Start of heading	DC1	Device control 1
STX	Start of text	DC2	Device control 2
ETX	End of text	DC3	Device control 3
EOT	End of transmission	DC4	Device control 4
ENQ	Enquiry	NAK	Negative acknowledge
ACK	Acknowledge	SYN	Synchronous idle
BEL	Bell	ETB	End-of-transmission block
BS	Backspace	CAN	Cancel
HT	Horizontal tab	EM	End of medium
LF	Line feed	SUB	Substitute
VT	Vertical tab	ESC	Escape
FF	Form feed	FS	File separator
CR	Carriage return	GS	Group separator
SO	Shift out	RS	Record separator
SI	Shift in	US	Unit separator
SP	Space	DEL	Delete

- An additional 128 eight-bit characters with the most significant bit set to 1 are used for other symbols, such as the Greek alphabet or italic type font.

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.

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

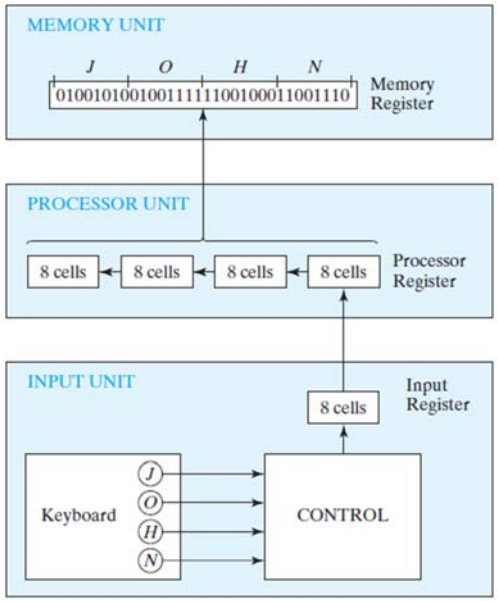
Binary Storage and Registers

- binary information must have a physical existence in some medium for storing
- *binary cell* is capable of storing one bit (0 or 1) of information.
- The information stored in a cell is 1 when the cell is in one stable state and 0 when the cell is in the other stable state.
- **Registers**
- A *register* is a group of binary cells
- A register with *n* cells can store any discrete quantity of information that contains *n* bits
- A register with *n* cells can store any discrete quantity of information that contains *n* bits

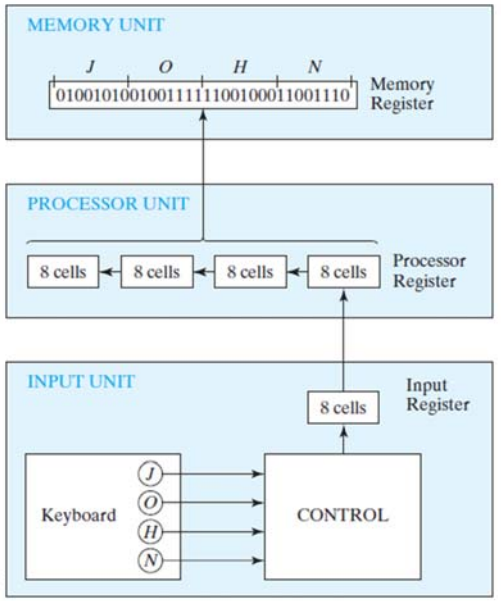
1100001111001001

- A register with 16 cells can be in one of 2^{16} possible states, register can store any binary number from 0 to 2^{16} - 1.
- assumes that the register stores alphanumeric characters of an eight-bit code, then the content of the register is any two meaningful characters.
- **Register Transfer**
- *register transfer* operation is a basic operation that consists of a transfer of binary information from one set of registers into another set of registers.
- The transfer may be direct, from one register to another, or may pass through data-processing circuits to perform an operation

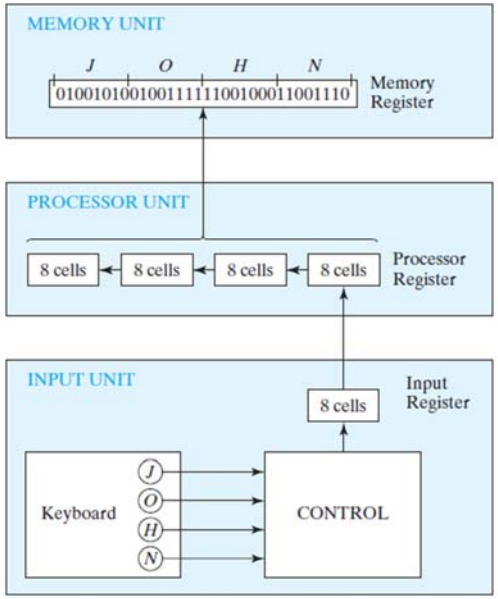
- transfer of binary information from a keyboard into a register in the memory unit.
- The input unit is assumed to have a keyboard, a control circuit, and an input register.
- Each time a key is struck, the control circuit enters an equivalent eight-bit alphanumeric character code into the input register



- The information from the input register is transferred into the eight least significant cells of a processor register.
- After every transfer, the input register is cleared to enable the control to insert a new eight-bit code when the keyboard is struck again.

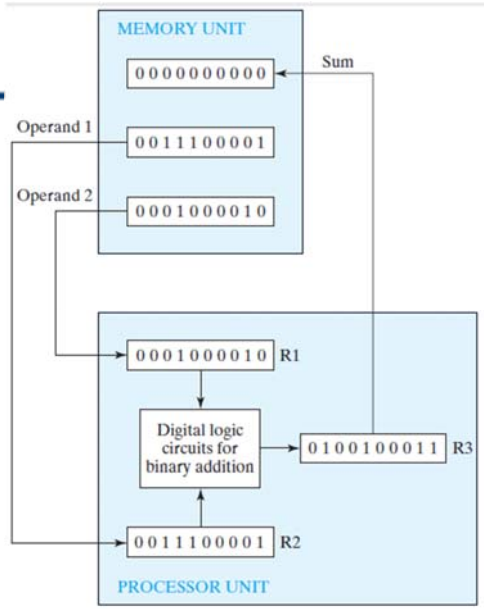


- Each eight-bit character transferred to the processor register is preceded by a shift of the previous character to the next eight cells on its left.
- When a transfer of four characters is completed, the processor register is full, and its contents are transferred into a memory register.

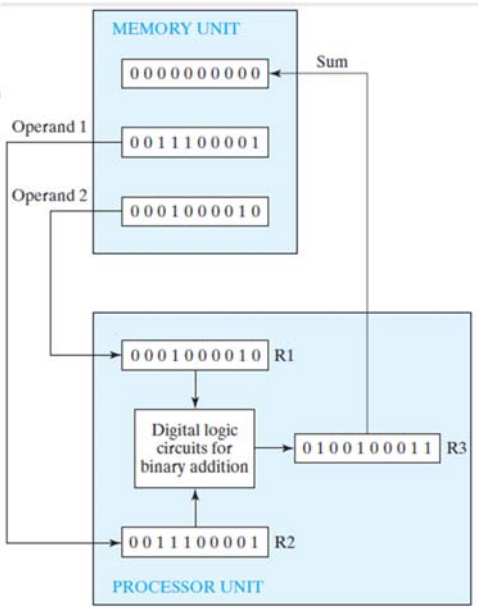


- The devices that hold the data to be processed and with circuit elements called **registers.**
- Binary variables are manipulated by means of digital logic circuits.

- processor unit shown consists of three registers—*R1*, *R2*, and *R3*
- Memory registers store information and are incapable of processing the two operands.
- The information stored in memory can be transferred to processor registers, and the results obtained in processor registers can be transferred back into a memory register for storage until needed again.



- diagram shows the contents of two operands transferred from two memory registers into *R1* and *R2.*
- digital logic circuits produce the sum, which is transferred to register *R3*.
- The contents of *R3* can now be transferred back to one of the memory register



Binary Logic

- Binary logic deals with variables that take on two discrete values (True, False) or bits (values 1 and 0) and with operations that assume logical meaning.
- Binary logic operations equivalent to an algebra called Boolean algebra

- **Definition of Binary Logic**
- There are three basic logical operations: AND, OR, and NOT.

1. AND

- represented by a dot or by the absence of an operator

$$x \cdot y = z \text{ or } xy = z$$

- Definition:
  - $z = 1$ if and only if $x = 1$ and $y = 1$;
  - otherwise $z = 0$.
- equivalent to multiplication

2. OR:

- represented by a plus sign

$$x + y = z$$

- Definition:
  - $z = 1$ if $x = 1$ or if $y = 1$ or if both $x = 1$ and $y = 1$.
  - If both $x = 0$ and $y = 0$, then $z = 0$
- equivalent to addition

3. NOT:

- represented by a ‘ (sometimes by an overbar).

$$x' = z \text{ (or } \overline{x} = z)$$

- Definition:
  - if $x = 1$, then $z = 0$, but
  - if $x = 0$, then $z = 1$

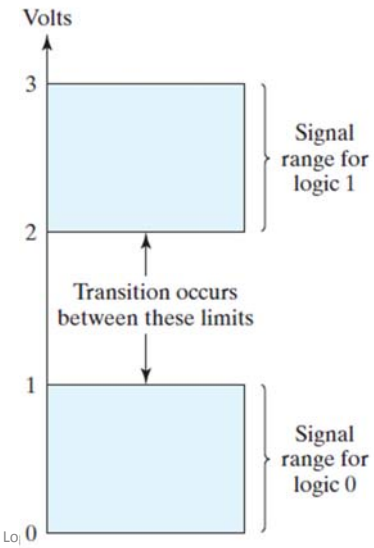
Truth Tables of Logical Operations

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		

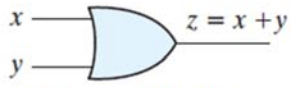
Page 31

- **Logic Gates**
- Logic gates are electronic circuits that operate on one or more input signals to produce an output signal.
- Voltage-operated logic circuits respond to two separate voltage levels that represent a binary variable equal to logic 1 or logic 0.
- For example, a particular digital system may define logic 0 as a signal equal to 0 V and logic 1 as a signal equal to 3 V.

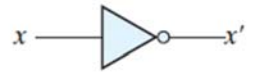
- The input terminals of digital circuits accept binary signals within the allowable range and respond at the output terminals with binary signals that fall within the specified range.
- The intermediate region between the allowed regions is crossed only during a state transition.



(a) Two-input AND gate



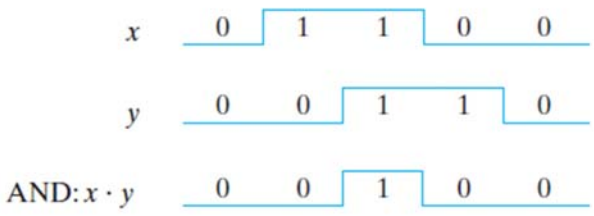
(b) Two-input OR gate



(c) NOT gate or inverter

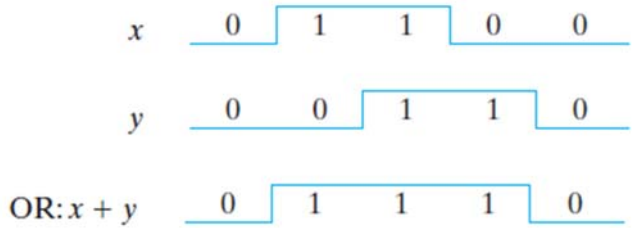
- The input signals *x* and *y* in the AND and OR gates may exist in one of four possible states: 00, 10, 11, or 01.
- The horizontal axis of the timing diagram represents the time, and the vertical axis shows the signal as it changes between the two possible voltage levels.

-



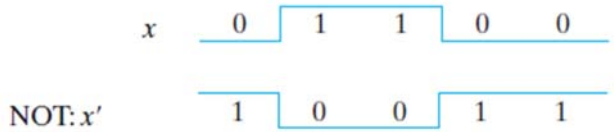
(a) Two-input AND gate

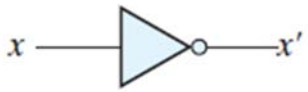
-



(b) Two-input OR gate

•

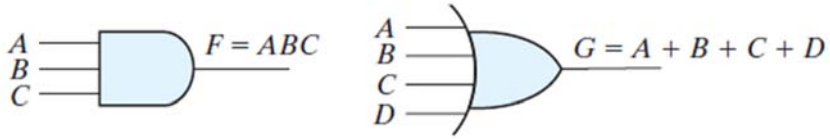




(c) NOT gate or inverter

The NOT gate is commonly referred to as an inverter

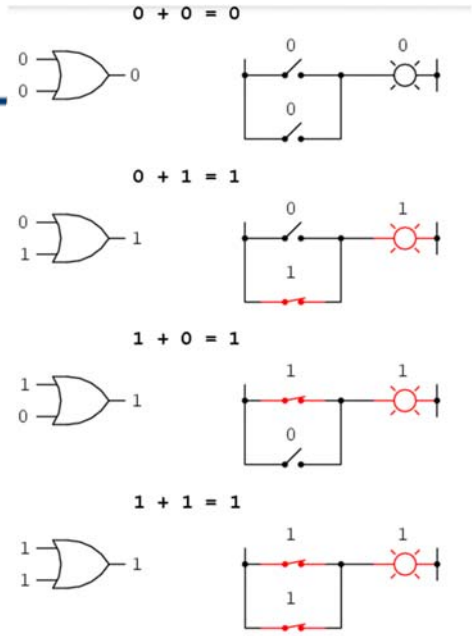
- AND and OR gates may have more than two inputs.

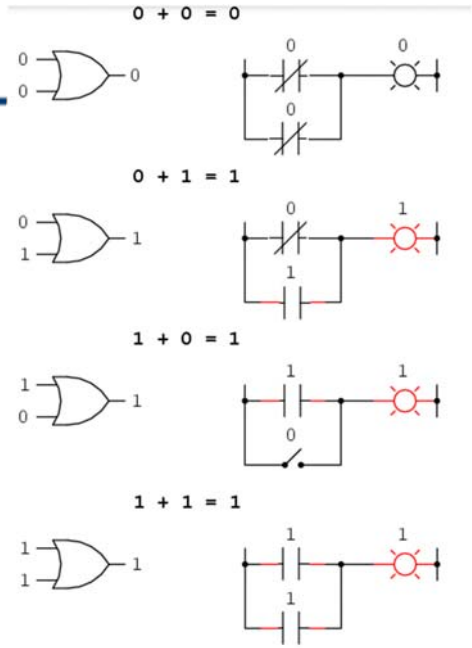


(a) Three-input AND gate (b) Four-input OR gate

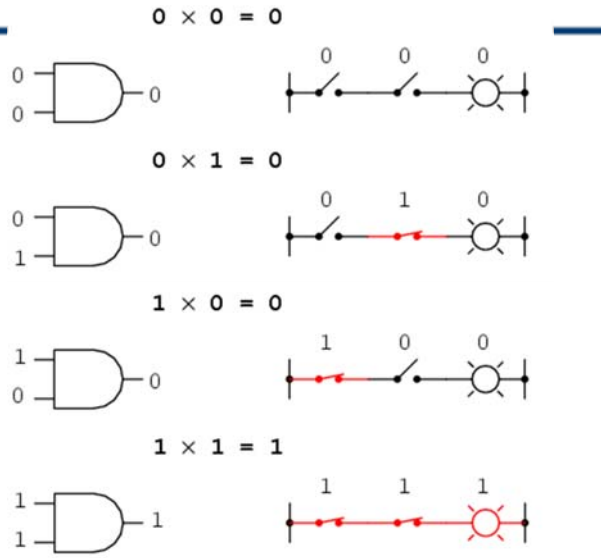
- The three-input AND gate responds with logic 1 output if all three inputs are logic 1, the output produces logic 0 if any input is logic 0.
- The four-input OR gate responds with logic 1 if any input is logic 1; its output becomes logic 0 only when all inputs are logic 0.

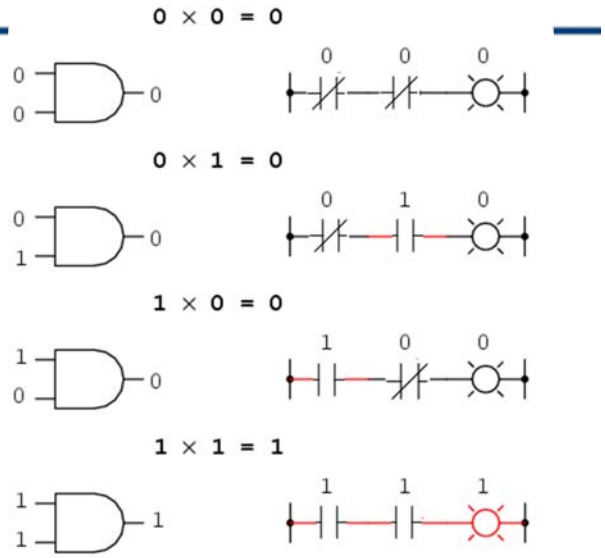
0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 1



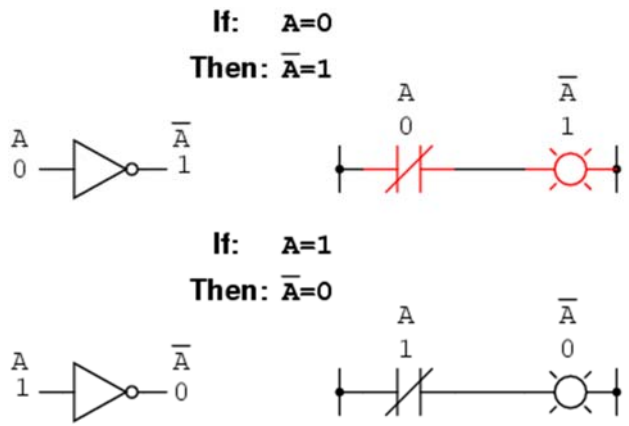


0 × 0 = 0
0 × 1 = 0
1 × 0 = 0
1 × 1 = 1



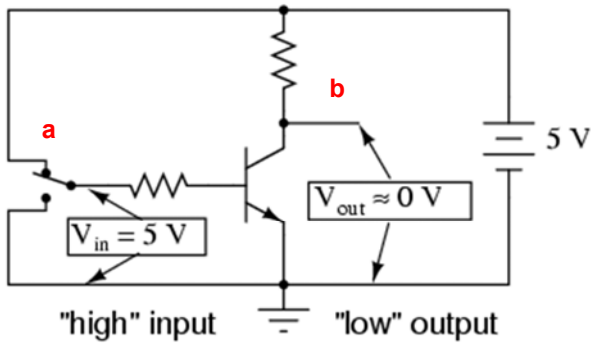


•



From electronics to logic gate

Inverter



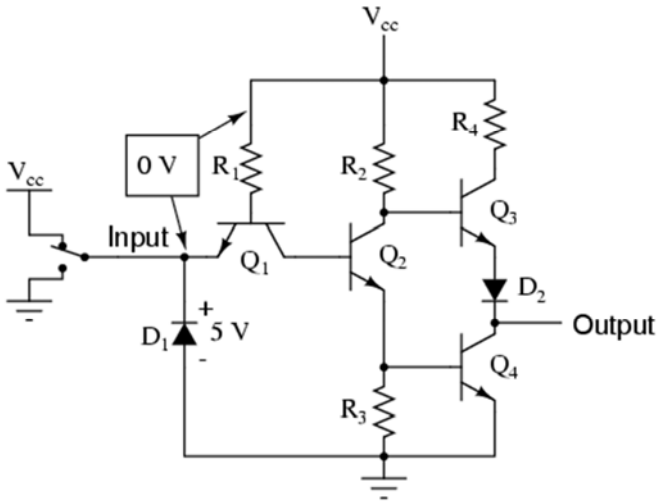
a	b
0	1
1	0

0 V = "low" logic level (0)

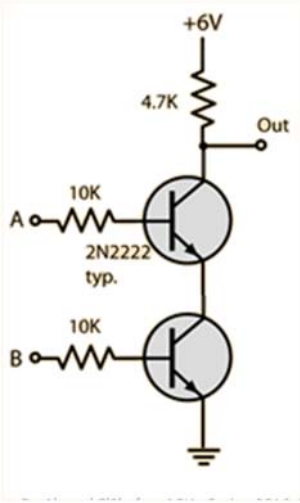
5 V = "high" logic level (1)

Practical Inverter

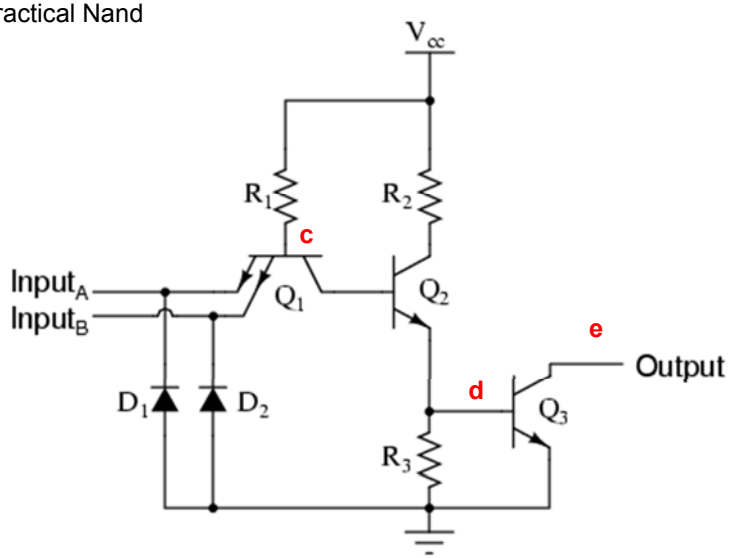
$V_{cc} = 5$ volts



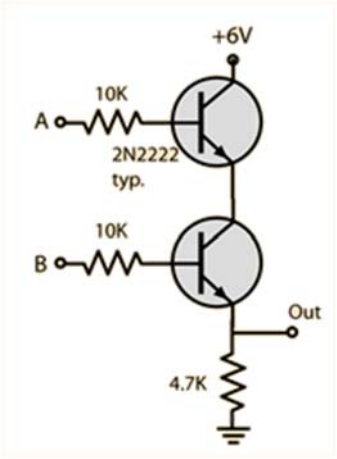
- Nand



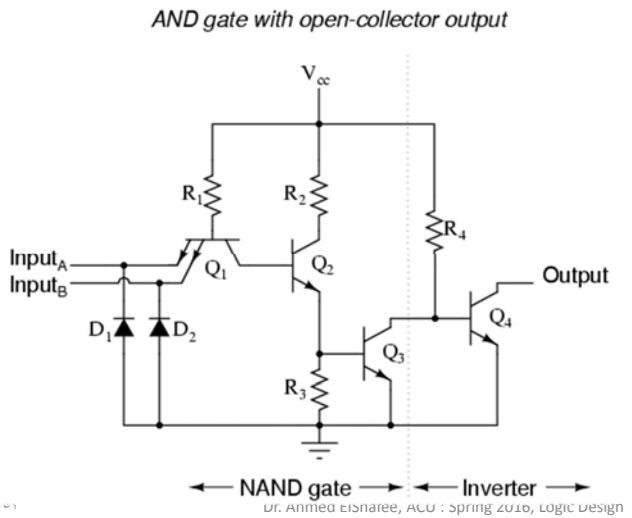
Practical Nand



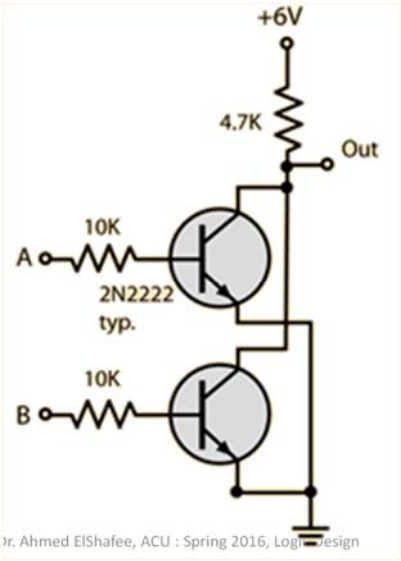
- And gate



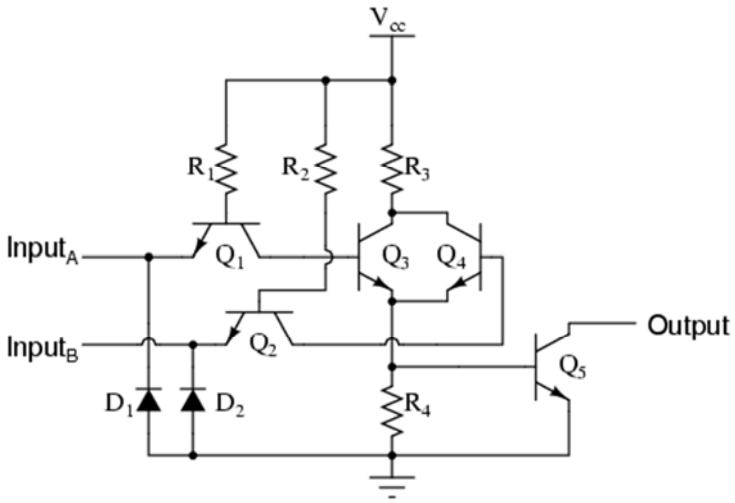
Practical And



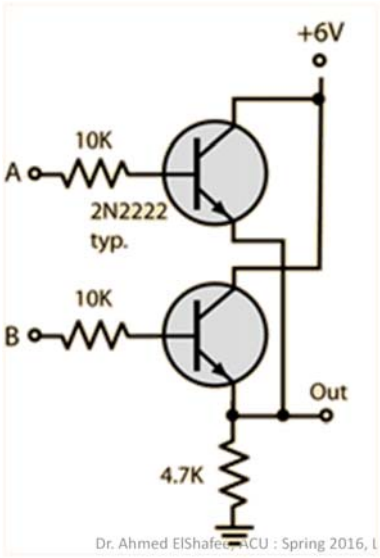
- NOR



Practical NOR



- OR



- Practical OR