Lecture 14: Digital Circuits

ELEC1111 Electrical and Telecommunications Engineering

UNSW AUSTRALIA

Never Stand Still

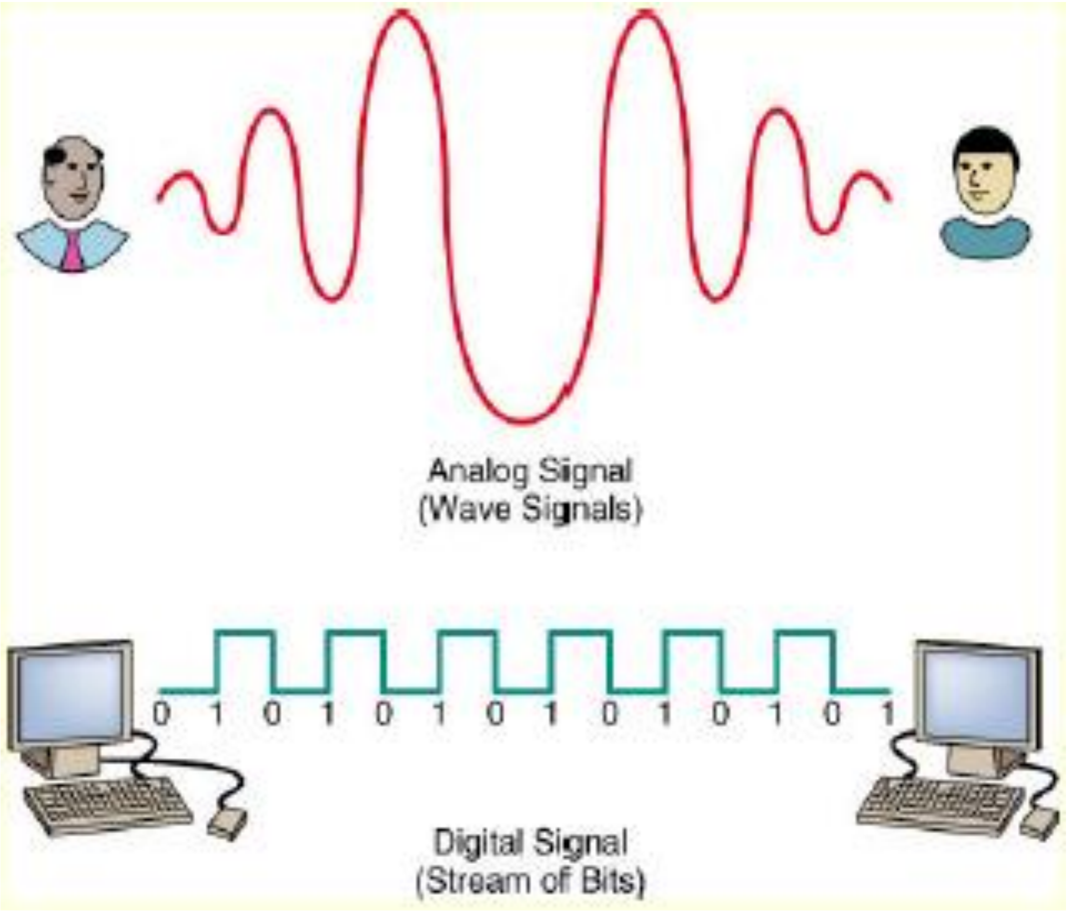
Faculty of Engineering

School of Electrical Engineering and Telecommunications

- Introduction
- Binary numbers
  - Representation
  - Conversion to decimal
  - Computation
- Logic Circuits
  - Logic Variables
  - Truth Table
  - Primitive Gates
  - Combining Primitive Gates
- Boolean Algebra
  - DeMorgan’s Theorem

UNSW THE UNIVERSITY OF NEW SOUTH WALES

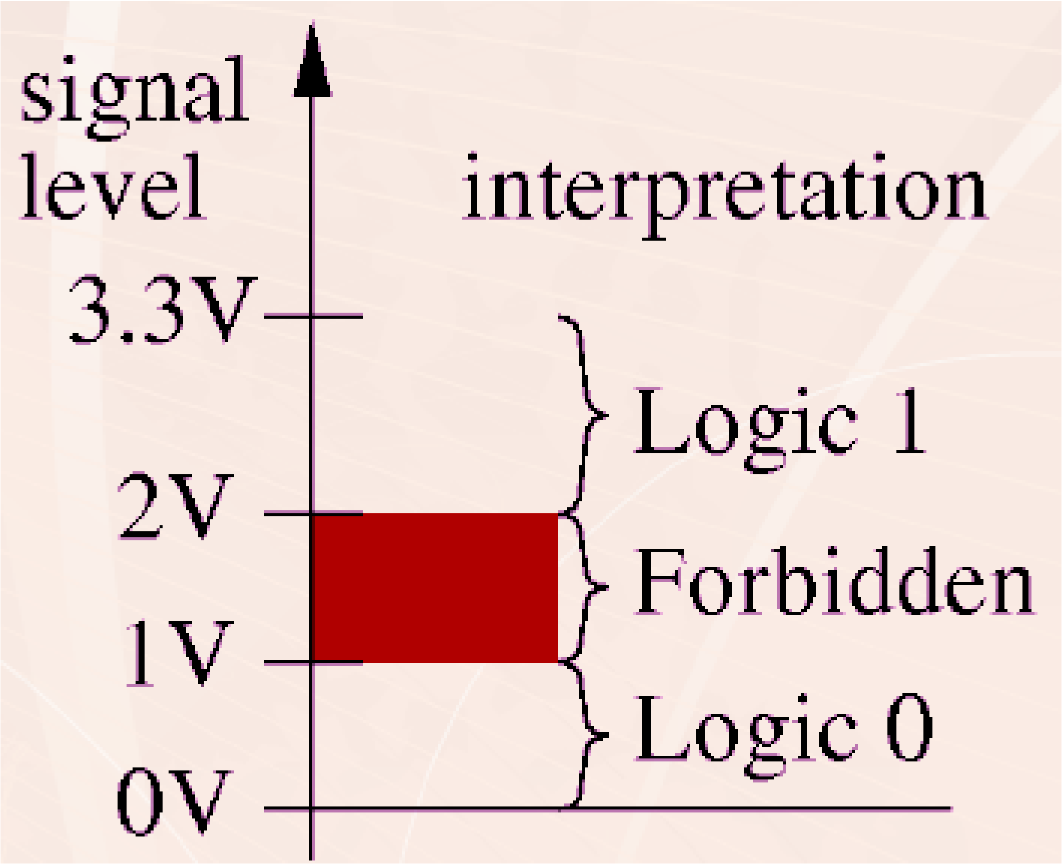
Analogue versus Digital



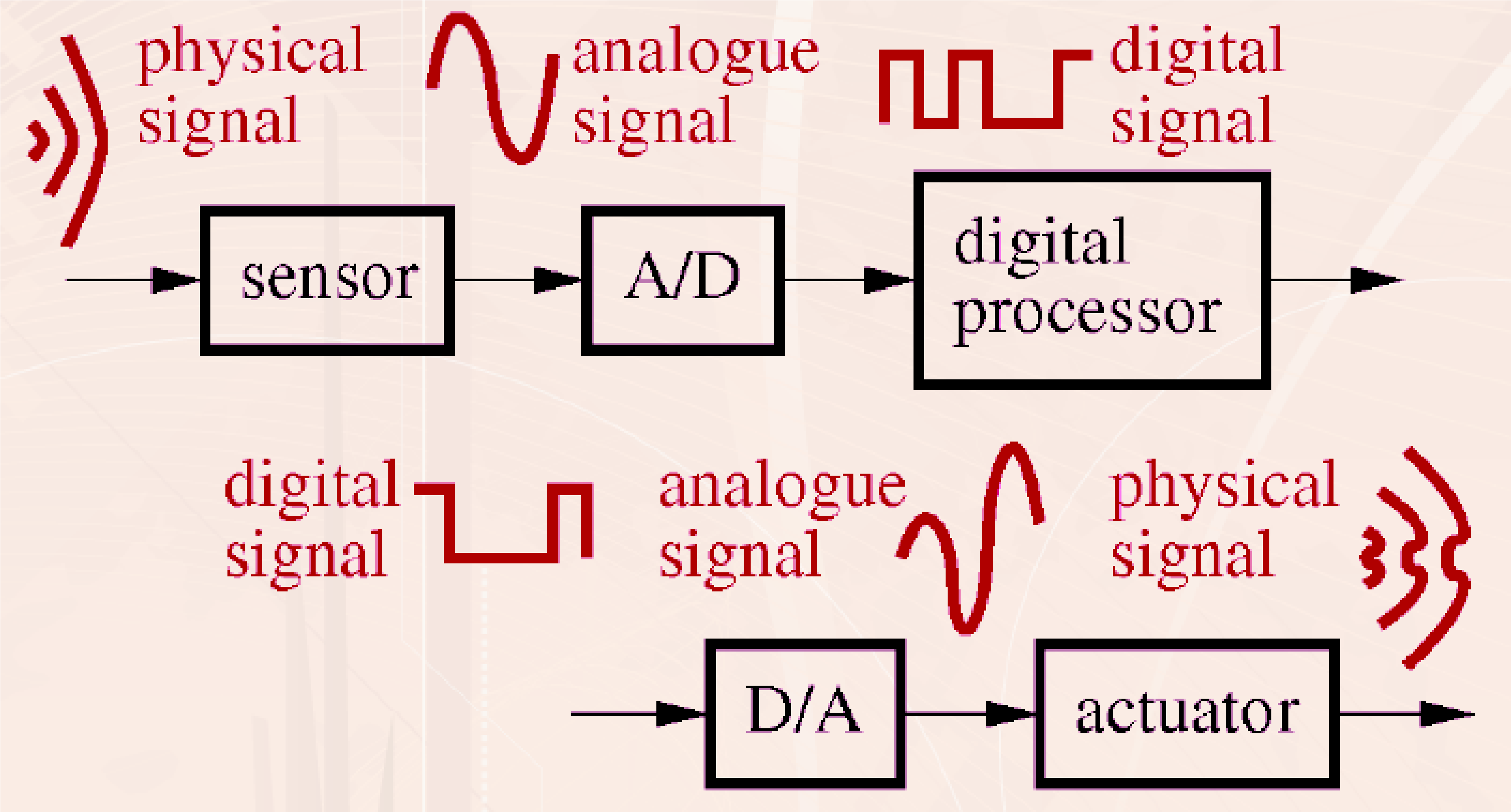
Two-level logic

Almost universal voltage encoding Level values:

- ✓ technology dependent
- ✓ power supply dependent



Typical electronic systems

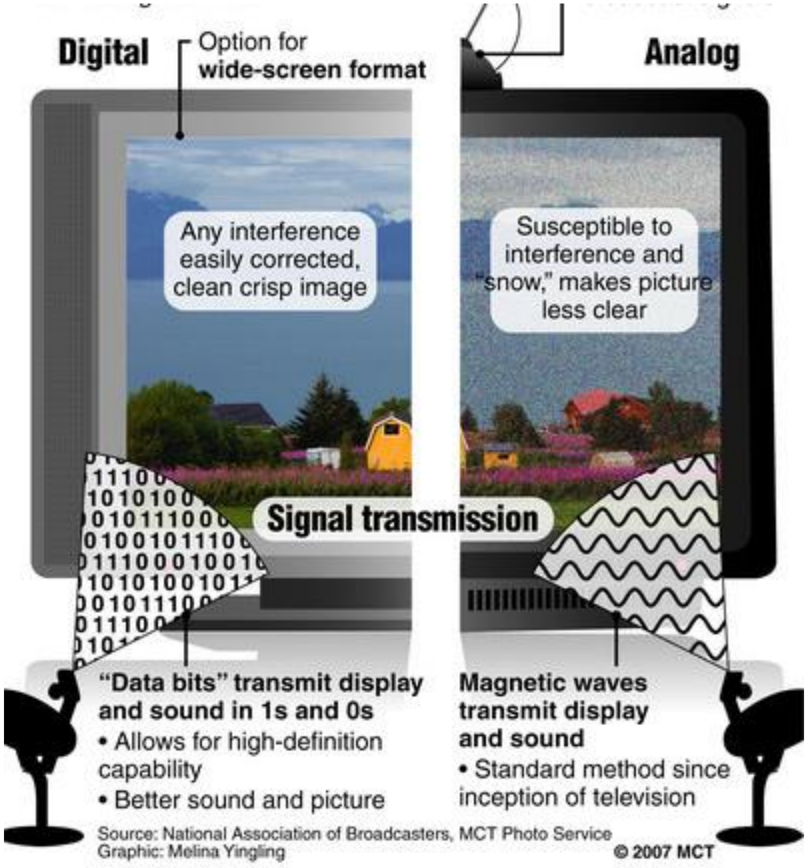


Why use digital

- ✓ Good noise rejection
- ✓ High reliability
- ✓ Low drift
- ✓ High accuracy
- ✓ Predictable
- ✓ Low power
- ✓ Ease of design

Limitations

- ✓ generates noise
- ✓ A/D & D/A interface costly ?



Binary number system

- ✓ Has only two numerals, 0 and 1
- ✓ Require very long strings of 1s and 0s
- ✓ The binary digits are also known as *bits*
- ✓ $(11100110)_2$ is an 8-bit number

'base' or 'radix'

- ✓ Nibble - a group of four bits
- ✓ Byte - a group of eight bits
- ✓ Word - a group of sixteen bits;

(Sometimes used to designate 32 bit or 64 bit)

Binary to decimal

MSB LSB

MSB: Most Significant Bit

LSB: Least Significant Bit

$$(11100110)_2 = 1 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$$

$$= 128 + 64 + 32 + 0 + 0 + 4 + 2 + 0$$

$$= (230)_{10}$$

$(101.101)_2 \quad =$

Decimal to binary

	Quotient	Remainder
$156 \div 2$	78	0
$78 \div 2$	39	0
$39 \div 2$	19	1
$19 \div 2$	9	1
$9 \div 2$	4	1
$4 \div 2$	2	0
$2 \div 2$	1	0
$1 \div 2$	0	1

$$(156)_{10} = (10011100)_2$$

LSB: Least Significant Bit

MSB: Most Significant Bit

Example

Note: Binary addition

0+0=0

1+0=1

0+1=1

1+1=10

18+15=

Binary subtraction

- ✓ A - B = A + (-B)
- ✓ Should have a way to represent negative numbers
- ✓ Usually most significant bit (MSB) is allotted for representing sign
- ✓ 0 is for + sign and 1 is for - sign.
  - ✓ e.g. $(5)_{10} = (101)_2$
  - ✓ (+ 5)10 = (0101)2 and (- 5)10 = (1101)2
- ✓ $(1101)_2$ could be misinterpreted as $(13)_{10}$
- ✓ We must first decide how many bits are going to be needed to represent the largest numbers we'll be dealing with
- ✓ The –ve number should instead be taken as 2's complement number.

Two's complement

- ✓ Complements used so that only addition is required.
- ✓ Universally adopted in today's digital computers
- ✓ For two's complement - invert all the bits of a number, changing all 1's to 0's and vice versa.
- ✓ Then add 1.

$$\begin{cases} 1 \rightarrow 0 \\ 0 \rightarrow 1 \end{cases}$$

$$\boxed{\text{2's Complement} = \text{1's Complement} + 1}$$

- ✓ The result is two's complement of the number.
- ✓ So, is a 4-bit system, $(5)_{10} = (0101)_2$
  - ✓ Taking two's complement – invert and add 1

    $1010 + 1 = (1011)_2$ – 2's complement of $(5)_{10}$

Two’s complement

Decimal	4-bit Two’s complement
7	0111
6	0110
5	0101
4	0100
3	0011
2	0010
1	0001
0	0000

Decimal	4-bit Two’s complement
-1	1111
-2	1110
-3	1101
-4	1100
-5	1011
-6	1010
-7	1001
-8	1000

Two's complement - Example

$7_{10} - 5_{10} \Longrightarrow$ Addition equivalent: $7_{10} + (- 5_{10})$

Logic Circuits

- ✓ Digital logic can be implemented in circuits called digital logic circuits.
- ✓ There are two types of logic circuits.
  1. Combinational logic
  2. Sequential logic
- ✓ These are used in computers and form part of ARTHIMETIC LOGIC UNIT

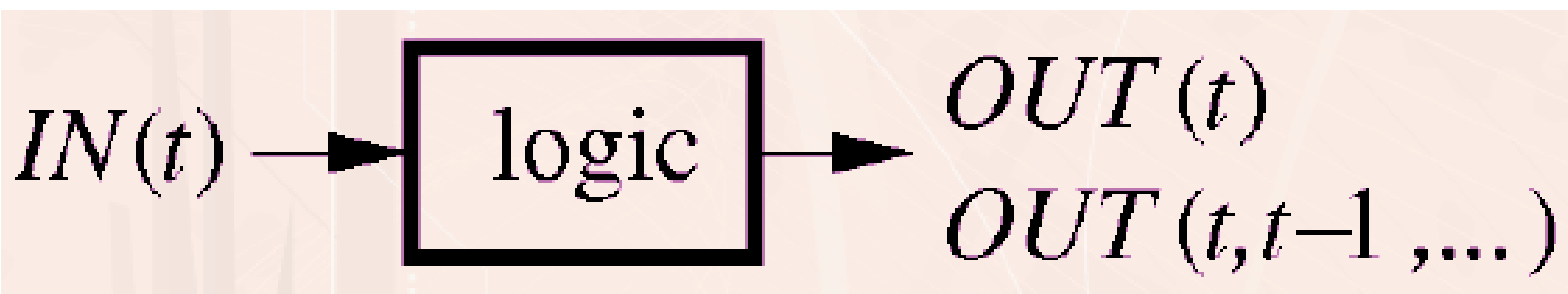
Combinational / Sequential

Combinational

- output is a pure function of present input
- no memory

Sequential

- Output depends not only on the present input but also on the history of the input
- memory



Truth Tables

- ✓ Exhaustive description of output for all possible inputs
- ✓ Example - Phone rings (*R*=1) if power on (*P*=1) and incoming call (*C*=1)

P	C	R
0	0	0
0	1	0
1	0	0
1	1	1

truth table

2^N combinations for N inputs

Assigning Logic Variables

- ✓ Assign logic variable to precise statement
- ✓ Example: David's purchase

  He buys if he wants an item and has cash or if he needs it and has cash or card

  - David needs item (N=1)
  - David wants item (W=1)
  - David has sufficient cash for purchase (C=1)
  - David has brought his EFTPOS card (E=1)
  - David buys the item (B=1)

Assigning Logic Variables

N: needs

W: wants

C: cash

E: eftpos

B: buys

$2^4 = 16$ combinations

	N	W	C	E	B
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	1
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	1
15	1	1	1	1	1

truth table

Primitive Gates

- ✓ A logic gate:
  - ✓ implements a combinational logic function

- ✓ 2^N possible functions of N variables

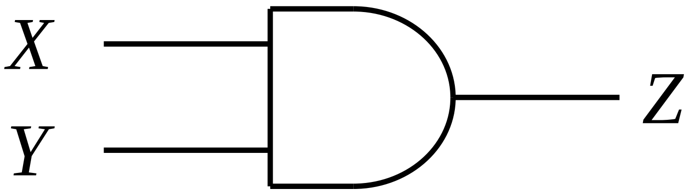
- ✓ All can be described using primitive gates

- ✓ Types of logic gates:

  AND, OR, NOT, XOR,

  NAND, NOR, XNOR

Logical AND



AND gate

$$Z = X \text{ and } Y$$

$$Z = X \cdot Y$$

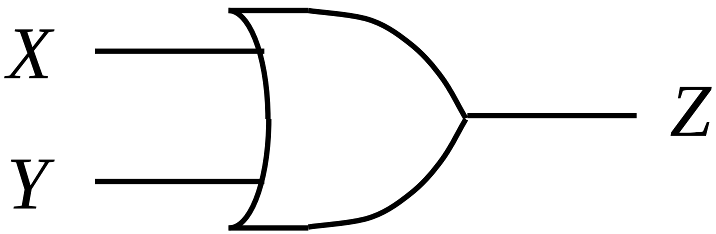
Logical multiplication

X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

AND truth table

Z=1 if both X and Y are 1

Logical OR



OR gate

$$Z = X \text{ or } Y$$

$$Z = X + Y$$

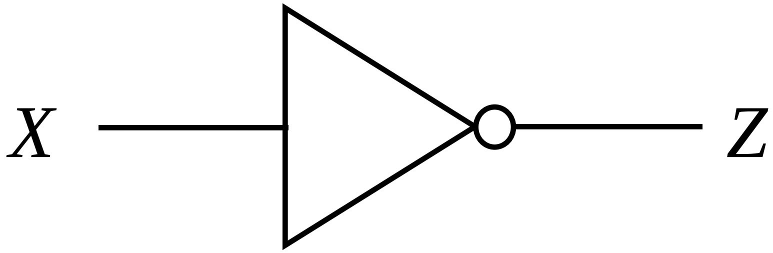
Logical addition

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

OR truth table

Z=1 if X or Y or both are 1

Logical NOT (negation, complement, inversion)



NOT gate

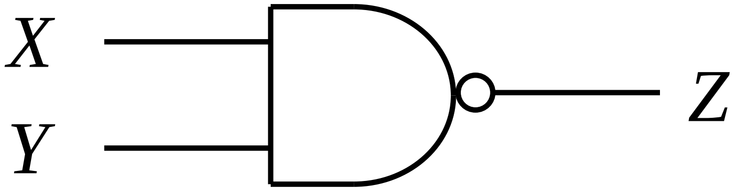
$$Z = \text{not } X$$

$$Z = \overline{X}$$

X	Z
0	1
1	0

NOT truth table

Logical NAND (NOT AND)



NAND gate

$$Z = \overline{X.Y}$$

$$Z = \overline{XY}$$

$$Z = (XY)'$$

X	Y	Z
0	0	1
0	1	1
1	0	1
1	1	0

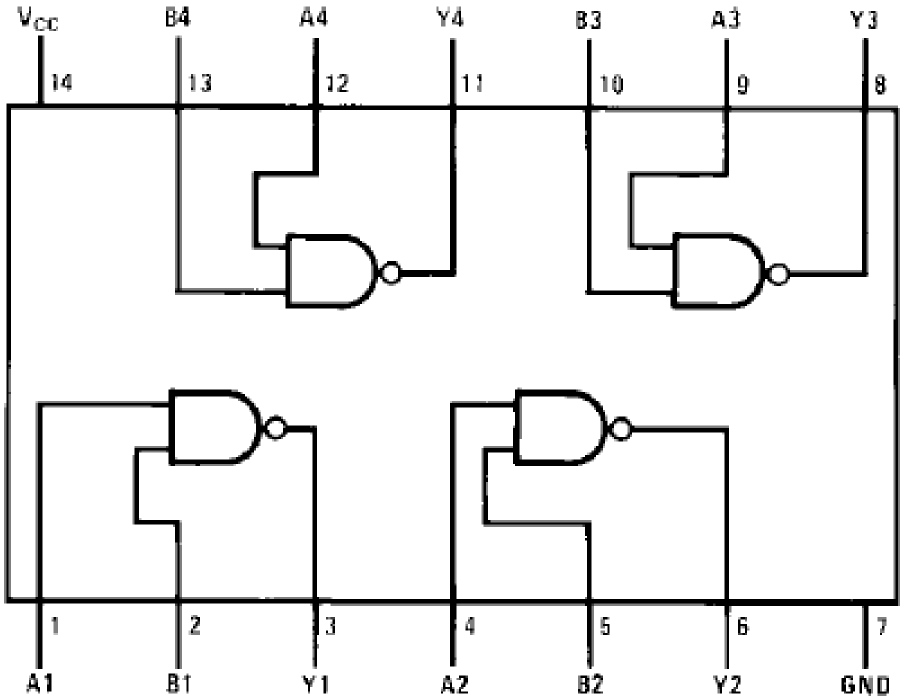
NAND truth table

Z=1 if X is 0 or Y is 0

Example Physical Implementation

74LS00 Quad NAND gate

Connection Diagram



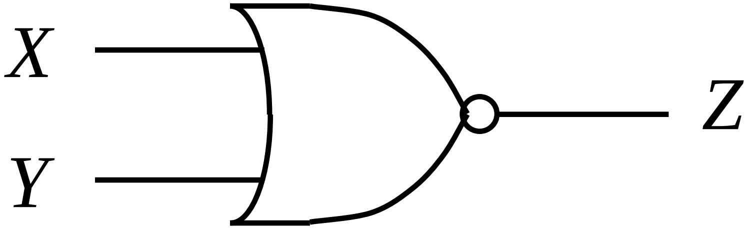
Function Table

$$Y = \overline{AB}$$

Inputs		Output
A	B	Y
L	L	H
L	H	H
H	L	H
H	H	L

H = HIGH Logic Level
L = LOW Logic Level

Logical NOR (NOT OR)



NOR gate

X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	0

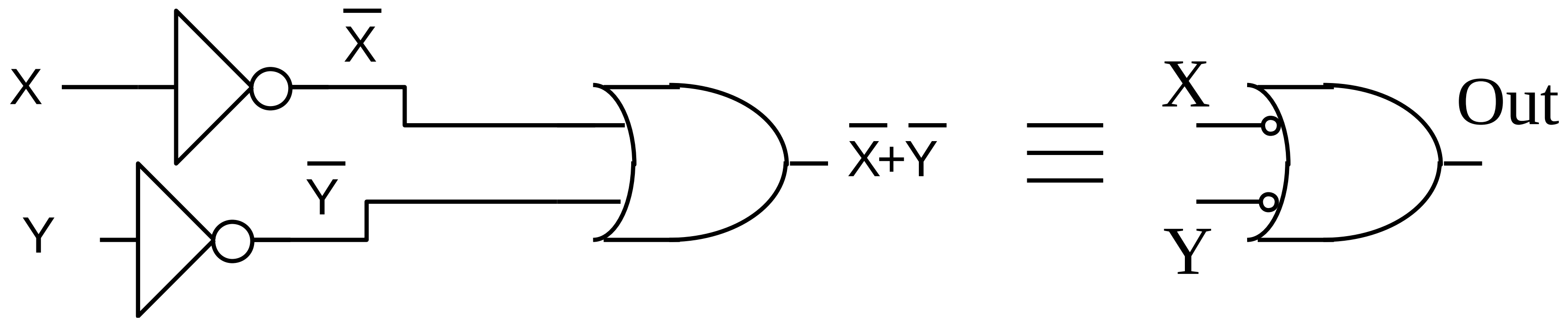
NOR truth table

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

$$Z = (X + Y)'$$

Z=1 if both X and Y are 0

Example

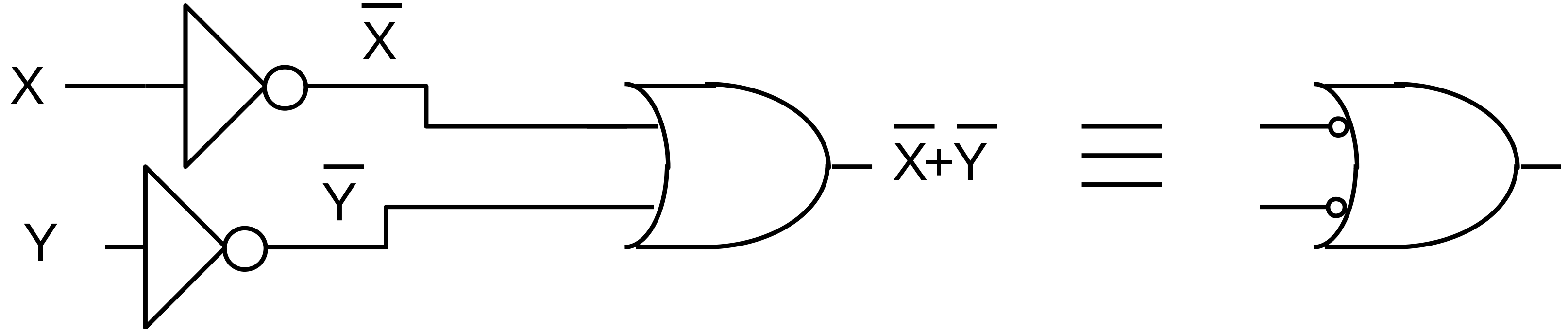


X	Y			Out

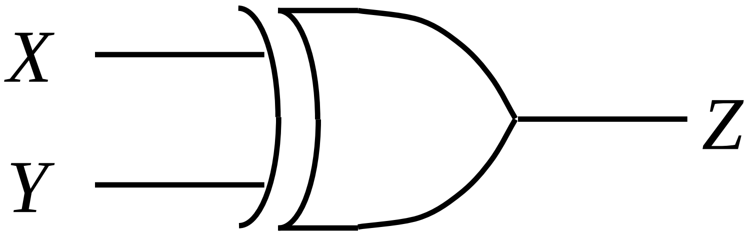
Drawing Conventions

Bubble - invert signal

Inverted input signified by a bubble on the input….eg:



Logical XOR (exclusive OR)



XOR gate

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

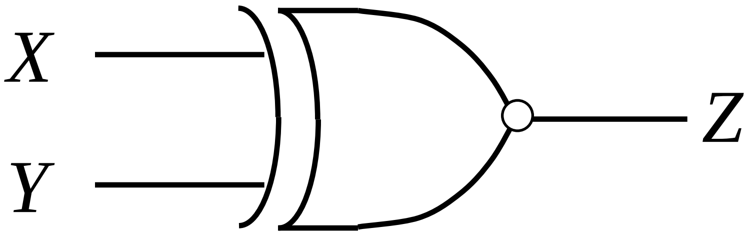
XOR truth table

$$Z = X \text{ xor } Y$$

$$Z = X \oplus Y$$

Z=1 if X has a different value from Y

Logical XNOR (NOT exclusive OR)



XNOR gate

$$Z = \overline{\mathrm{X\ xnor}\ Y}$$

$$Z = \overline{X \oplus Y}$$

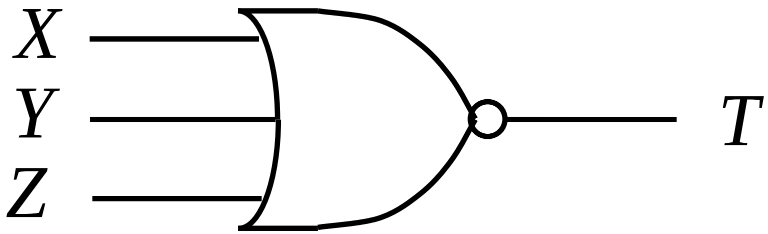
$$Z = (X \oplus Y)'$$

X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	1

XNOR truth table

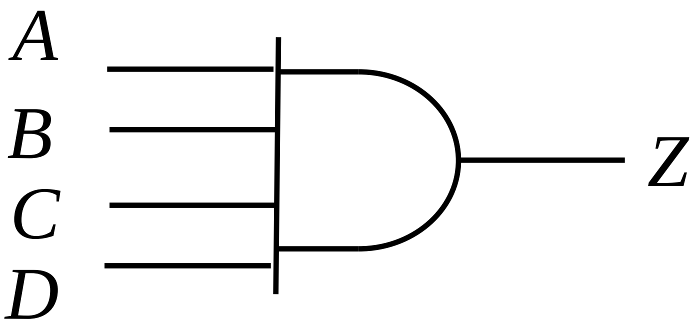
Z=1 if X and Y are the same

Gates with more than 2 inputs



$$T = \overline{X + Y + Z}$$

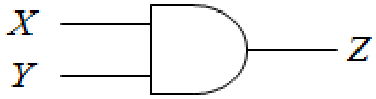
3-input NOR gate



$$Z = A \cdot B \cdot C \cdot D$$

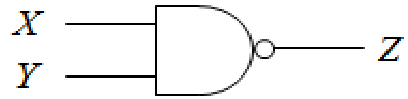
4-input AND gate

Logical Gates Summary



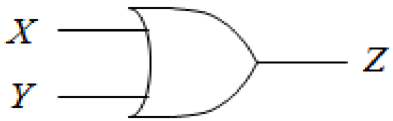
AND gate

$$Z = X.Y$$



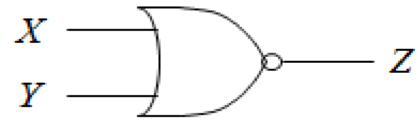
NAND gate

$$Z = \overline{X.Y}$$



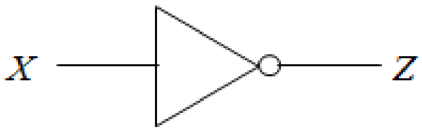
OR gate

$$Z = X + Y$$



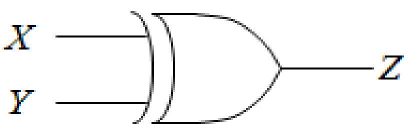
NOR gate

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



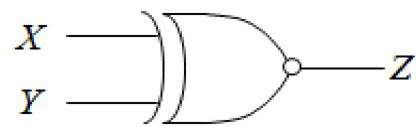
NOT gate

$$Z = \overline{X}$$



XOR gate

$$Z = X \oplus Y$$



XNOR gate

$$Z = \overline{X \oplus Y}$$

Combining Primitive Gates

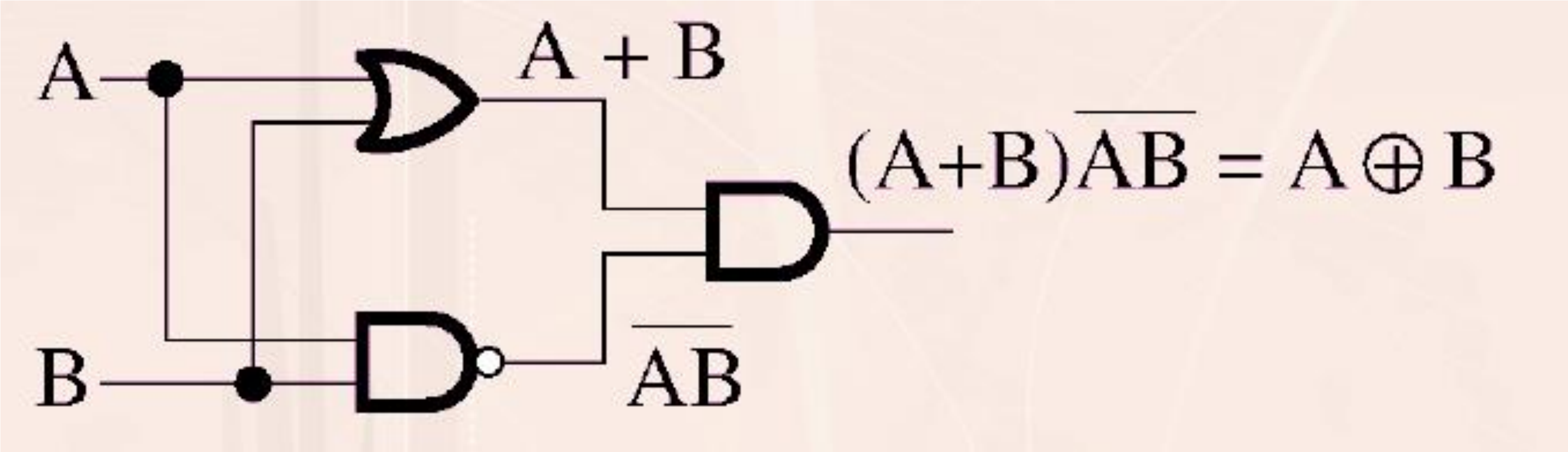
Example XOR

$$A \oplus B = (A+B).\overline{AB}$$

– as truth tables are identical

A	B	A+B	$\overline{AB}$	$(A+B)\ \overline{AB}$	$A \oplus B$
0	0	0	1	0	0
0	1	1	1	1	1
1	0	1	1	1	1
1	1	1	0	0	0

Combining Primitive Gates



David’s Purchasing revisited

$B = N(C+E)+WC$

	N	W	C	E	B
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	1
7	0	1	1	1	1

	N	W	C	E	B
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	1
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	1
15	1	1	1	1	1

truth table

Logic circuit diagram?

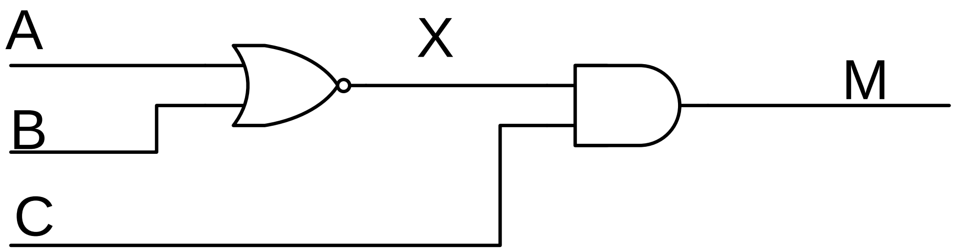
Example

- Draw the logic diagram represented by:

$$Z = A + B.\overline{C}$$

Example

Draw a truth table for the logic:



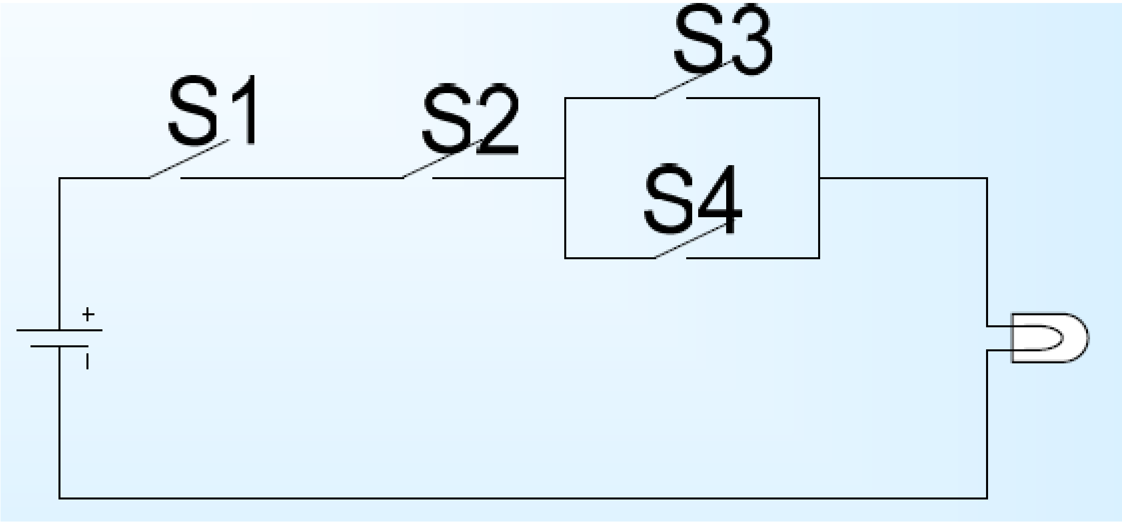
A	B	C	X	M

Boolean Algebra

- ✓ We encounter situations where the choice is binary

  Move – Stop, On – Off, Yes – No
- ✓ An intended action takes place or does not take place
- ✓ Signals with two possible states are called switching signals
- ✓ We need to work with a large number of such signals
- ✓ There is a need for formal methods of handling such signals

Boolean Algebra



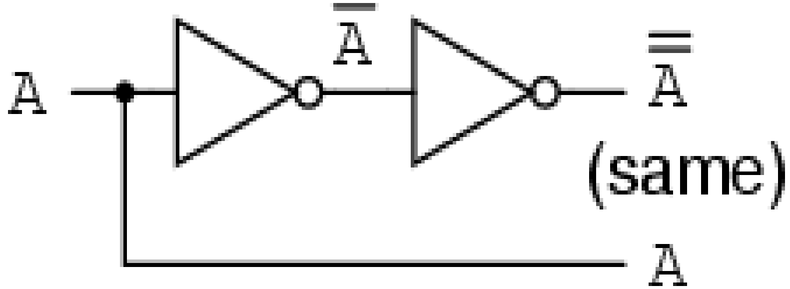
Four switches control the operation of the bulb. The bulb is switched on if the switches S1 and S2 are closed, and S3 or S4 is also closed, otherwise the bulb will not be switched on.

Single Variable Theorems

- $X \cdot 0 = 0$
- $X \cdot 1 = X$
- $X \cdot X = X$
- $X \cdot \overline{X} = 0$

$\overline{\overline{A}} = A$

- $X + 0 = X$
- $X + 1 = 1$
- $X + X = X$
- $X + \overline{X} = 1$
- $\overline{\overline{X}} = X$

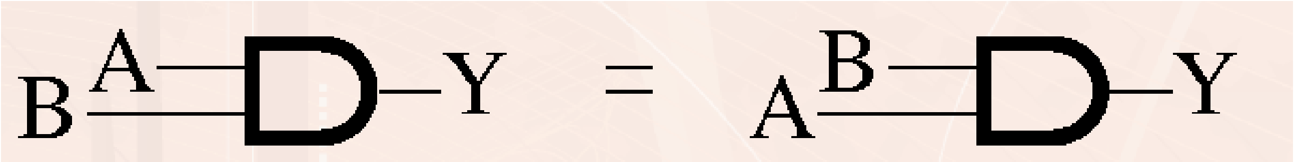


Commutative Laws

$A \cdot B = B \cdot A$

$A + B = B + A$

- Gate: input order doesn't matter

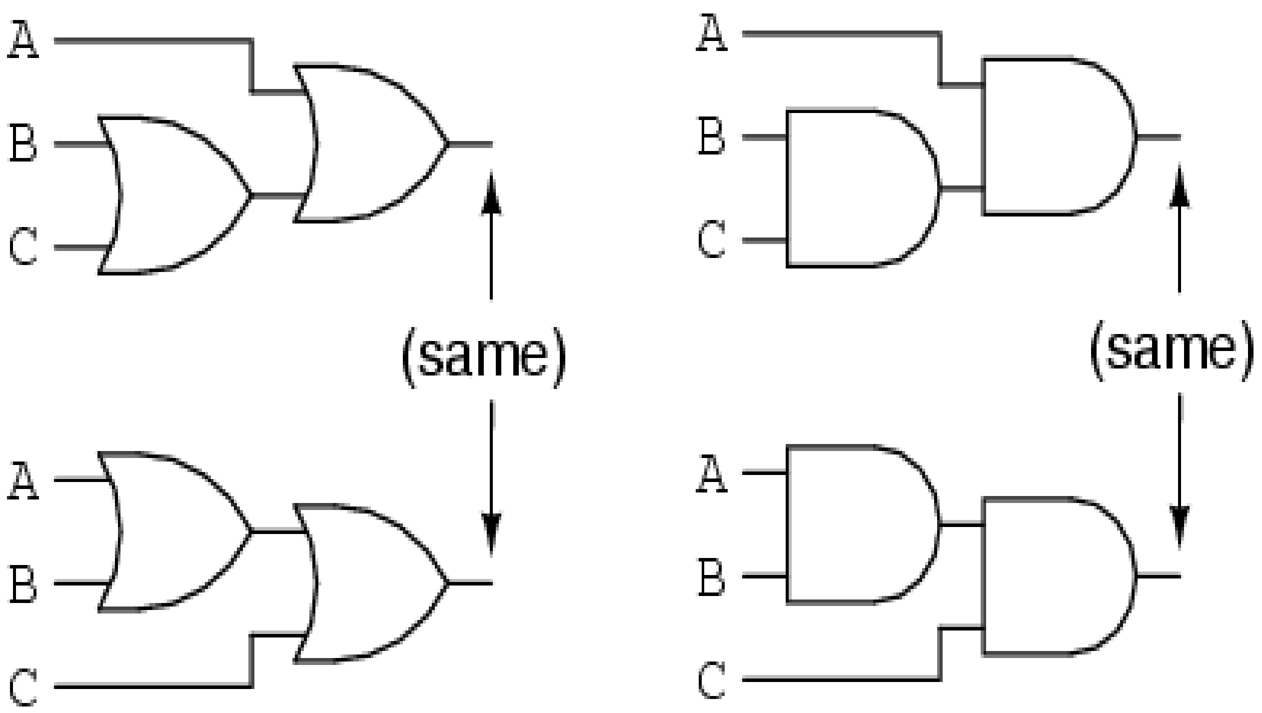


Associative Laws

$$(A \cdot B) \cdot C = A \cdot (B \cdot C) = A \cdot B \cdot C$$

$$(A + B) + C = A + (B + C) = A + B + C$$

- Gate: multiple input gates defined



Distributive Laws

$$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$$

$$A + (B \cdot C) = (A + B) \cdot (A + C)$$

A	B	C	B+C	A.B	A.C	A · (B + C)	(A · B) + (A · C)
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

Distributive Laws

$$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$$

$$A + (B \cdot C) = (A + B) \cdot (A + C)$$

A	B	C	B·C	A+B	A+C	A + (B·C)	(A + B)·(A + C)
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

Absorption Laws

- $A + A \cdot B = A$
- $A \cdot (A + B) = A$
- $A \cdot B + A \cdot \overline{B} = A$
- $(A + B) \cdot (A + \overline{B}) = A$

Example: Prove

$$(A + B)(A + C) = A + BC$$

Examples

Prove:

$(A + B)(A + C) = A + BC$

$A + \overline{A}B = A + B$

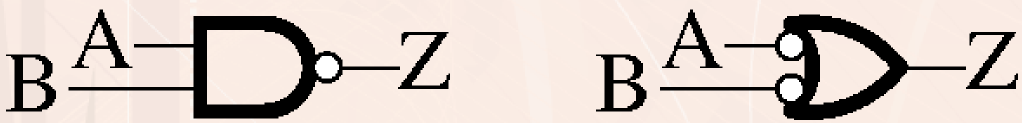
$AB + BC(B+C) = B(A+C)$

$A + B(A+C) + AC = A+BC$

DeMorgan’s Theorem

- $\overline{A + B} = \overline{A} \cdot \overline{B}$
- $\overline{A \cdot B} = \overline{A} + \overline{B}$

$$Z = \overline{A \cdot B} = \overline{A} + \overline{B}$$



✓ Proof – truth tables

DeMorgan’s Theorem - Examples

Reduce the following:

$$\overline{A + \overline{BC}}$$

$$\overline{\overline{A + BC} + \overline{A\overline{B}}}$$

Example 1

For a system represented by the following equation:

$$Y = AB + \overline{A}BC + A\overline{C}$$

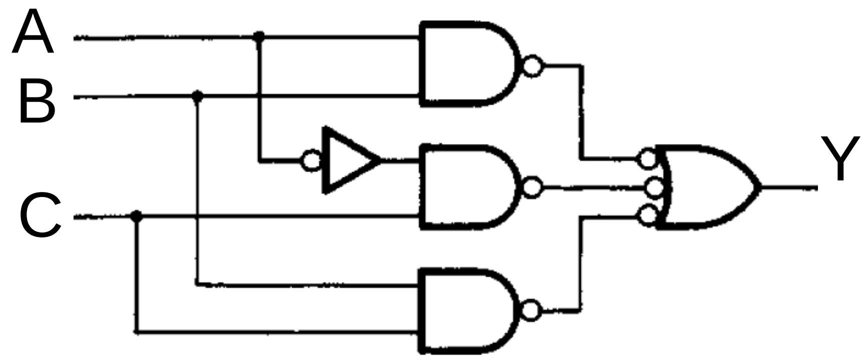
(a) Draw the logic diagrams of the system

(b) Draw up the truth table describing the operation of the circuit

A	B	C				

Example 2

For a system described by the following logic diagram:



(a) Drive the logical expression of the system

(b) Draw up the truth table describing the operation of the circu

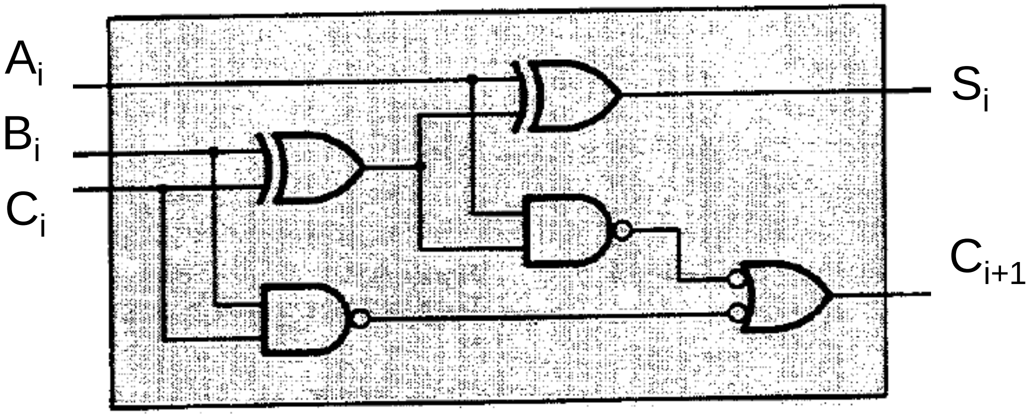
Example 2

Truth table

A	B	C				

Example 3

For a system (1-bit adder) described by the following logic diagram:



(a) Drive the logical expression of the system

(b) Draw up the truth table describing the operation of the circu

Example 3

For a system (1-bit adder) described by the following logic diagram:

A_i	B_i	C_i					

Example 4

Draw a logic diagram for each of the following logic equations using NAND gates only

(a) $Output = \overline{A}B + C\overline{D}$

(b) $Output = (\overline{A + B})(C + \overline{D})$

Example 5

(a) Show that the following logic system can be implemented using only one 2-input OR gate and two NOT gates

$$Output = a\bar{b}\bar{c} + \bar{b}c + \bar{a}\bar{b} + \overline{bd}$$

(b) Reduce the following logic equation

$$\overline{A.(B + C.(B + \bar{A}))}$$