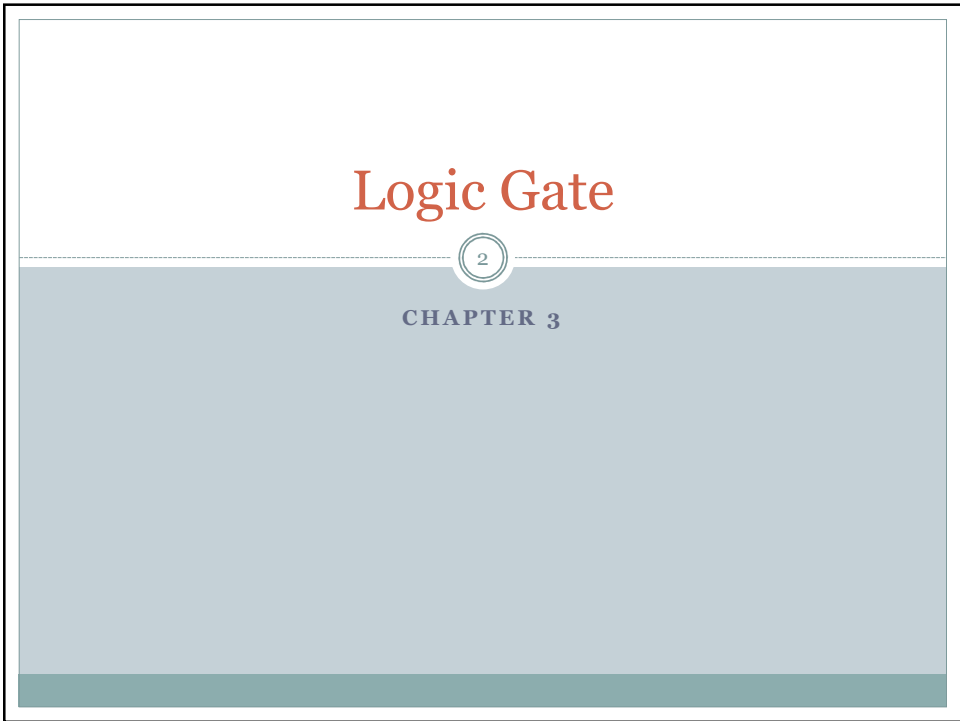


Digital Logic Design

1

This slide features a white background with a light blue horizontal band across the middle and a dark teal band at the bottom. The title "Digital Logic Design" is centered in a red serif font. Below the title, a small white circle with a black border contains the number "1".



Logic Gate

2

CHAPTER 3

This slide features a white background with a light blue horizontal band across the middle and a dark teal band at the bottom. The title "Logic Gate" is centered in a red serif font. Below the title, a small white circle with a black border contains the number "2". Underneath the circle, the text "CHAPTER 3" is centered in a dark blue, all-caps sans-serif font.

Contents

3

- The inverter
- The AND gate
- The OR gate
- The NAND gate
- The NOR gate
- The Exclusive OR, NOR
- Fixed function logic
- Programmable logic
- Applications

Logic Gates

4

Logic gates are electronic circuits that are used to perform logical operations

Usually these are voltage operated and respond to two separate voltage levels that represent logic 0 and logic 1

Consider a digital system with:

Logic 0 = 0 volt

Logic 1 = 5 volts

Symbols for the basic gates:

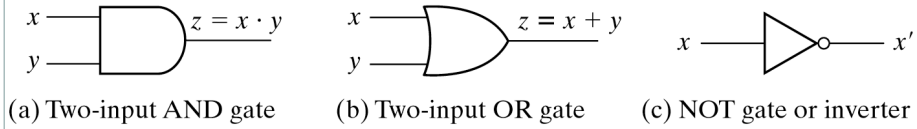
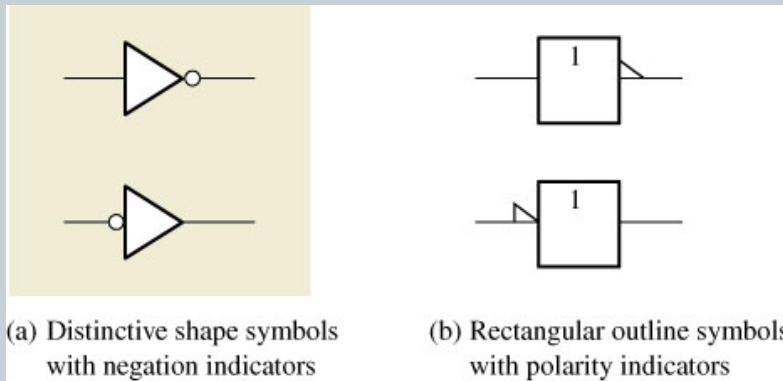


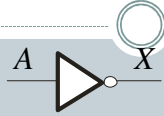
Fig. 1-4 Symbols for digital logic circuits

The Inverter

6



The Inverter

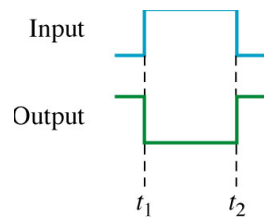
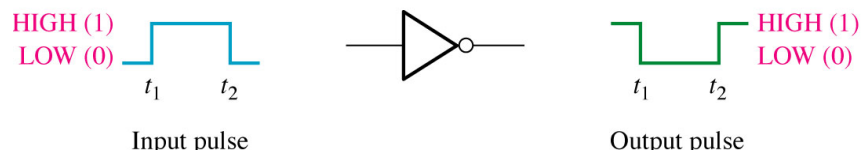


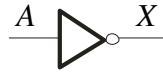
The inverter performs the Boolean **NOT** operation. When the input is **LOW**, the output is **HIGH**; when the input is **HIGH**, the output is **LOW**.

Input	Output
A	X
LOW (0)	HIGH (1)
HIGH (1)	LOW(0)

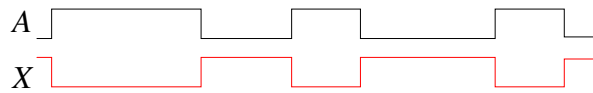
The **NOT** operation (complement) is shown with an overbar. Thus, the Boolean expression for an inverter is $X = \overline{A}$.

Inverter operation

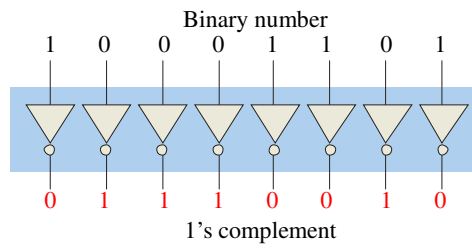




Example waveforms:



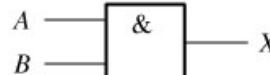
A group of inverters can be used to form the 1's complement of a binary number:



The AND gate



(a) Distinctive shape



(b) Rectangular outline with the AND (&) qualifying symbol

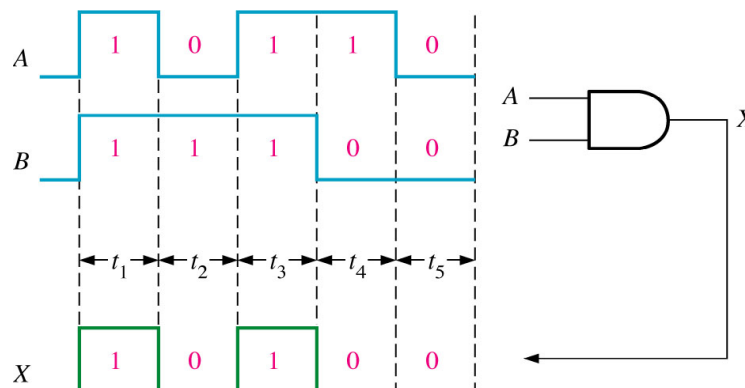
The AND gate

The **AND gate** produces a HIGH output when all inputs are HIGH; otherwise, the output is LOW. For a 2-input gate, the truth table is

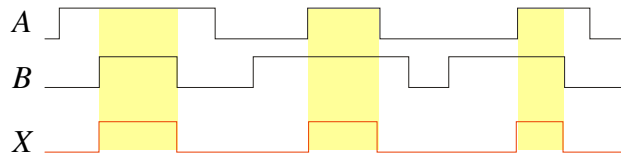
Inputs		Output
A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

The **AND** operation is usually shown with a dot between the variables but it may be implied (no dot). Thus, the AND operation is written as $X = A \cdot B$ or $X = AB$.

Pulsed AND gate Operation



Example waveforms:



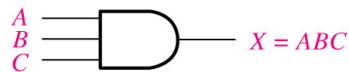
The AND operation is used in computer programming as a selective mask. If you want to retain certain bits of a binary number but reset the other bits to 0, you could set a mask with 1's in the position of the retained bits.

Example If the binary number 10100011 is ANDed with the mask 00001111, what is the result? **00000011**

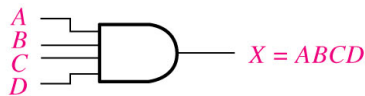
AND gates with various numbers of Inputs



(a)

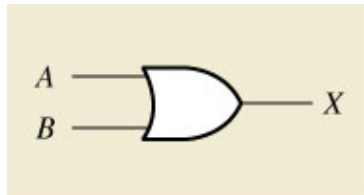


(b)

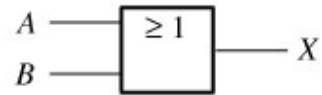


(c)

The OR gate



(a) Distinctive shape



(b) Rectangular outline with the OR (≥ 1) qualifying symbol

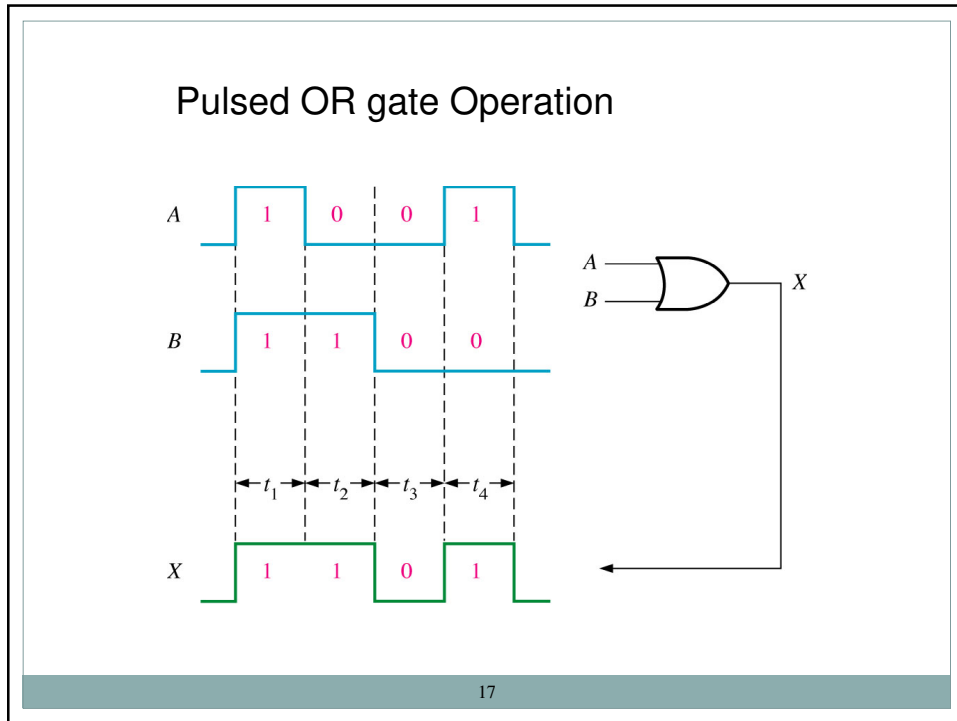
15

The OR gate

The **OR gate** produces a HIGH output if any input is HIGH; if all inputs are LOW, the output is LOW. For a 2-input gate, the truth table is

Inputs		Output
A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

The **OR** operation is shown with a plus sign (+) between the variables. Thus, the OR operation is written as $X = A + B$.



The OR gate

Example waveforms:

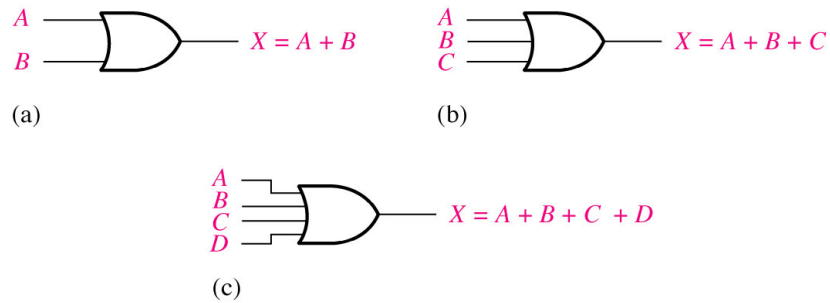
The diagram shows three waveforms: A (yellow), B (yellow), and X (orange). A is 1, 0, 1, 1, 0, 1, 1, 0. B is 0, 1, 1, 1, 0, 1, 1, 0. X is 1, 1, 1, 1, 0, 1, 1, 0.

The OR operation can be used in computer programming to set certain bits of a binary number to 1.

Example ASCII letters have a 1 in the bit 5 position for lower case letters and a 0 in this position for capitals. (Bit positions are numbered from right to left starting with 0.) What will be the result if you OR an ASCII letter with the 8-bit mask 00100000?

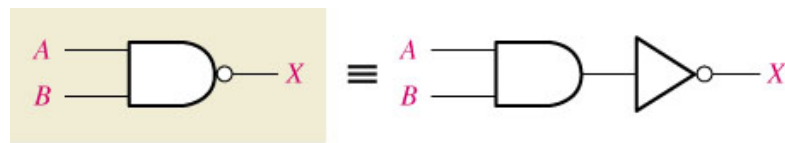
The resulting letter will be lower case.

OR gates with various number of Inputs

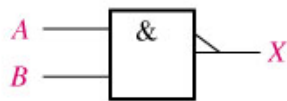


19

The NAND gate



(a) Distinctive shape, 2-input NAND gate and its NOT/AND equivalent



(b) Rectangular outline, 2-input NAND gate with polarity indicator

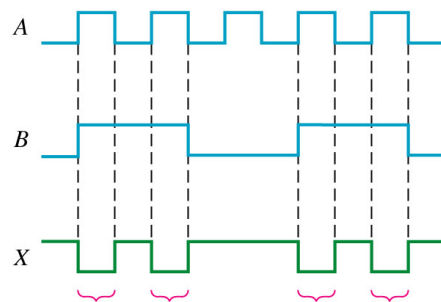
The NAND gate

The **NAND gate** produces a LOW output when all inputs are HIGH; otherwise, the output is HIGH. For a 2-input gate, the truth table is

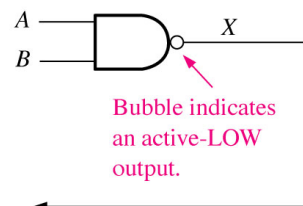
Inputs		Output
A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

The **NAND** operation is shown with a dot between the variables and an overbar covering them. Thus, the NAND operation is written as $X = \overline{A \cdot B}$ (Alternatively, $X = \overline{AB}$.)

Pulsed NAND gate Operation

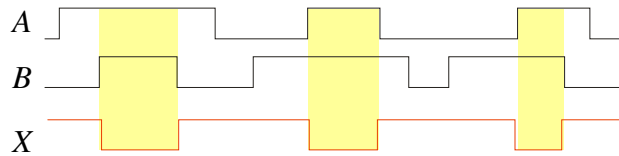


A and B are both HIGH during these four time intervals. Therefore X is LOW.



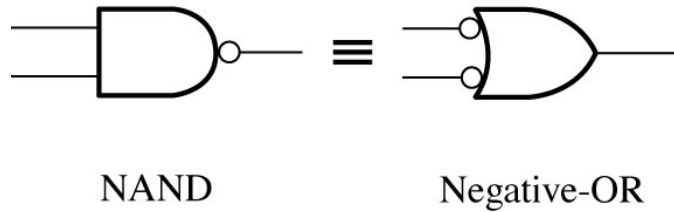
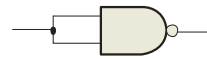
The NAND gate

Example waveforms:

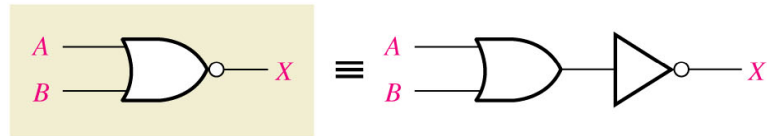


The NAND gate is particularly useful because it is a “universal” gate – all other basic gates can be constructed from NAND gates.

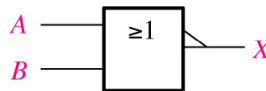
Example How would you connect a 2-input NAND gate to form a basic inverter?



The NOR gate



(a) Distinctive shape, 2-input NOR gate and its NOT/OR



(b) Rectangular outline, 2-input

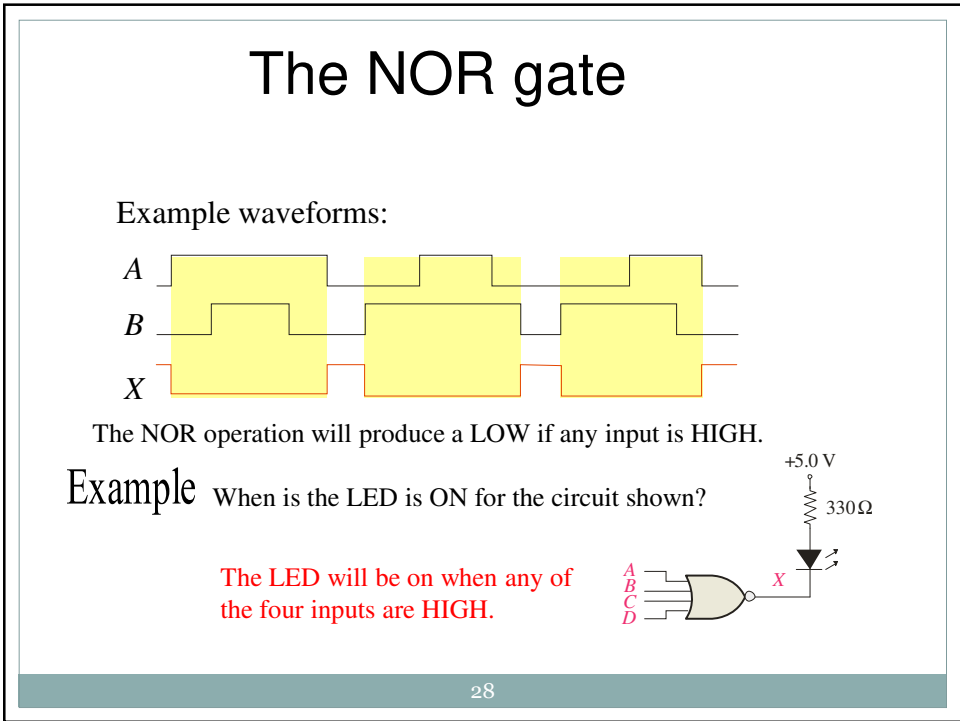
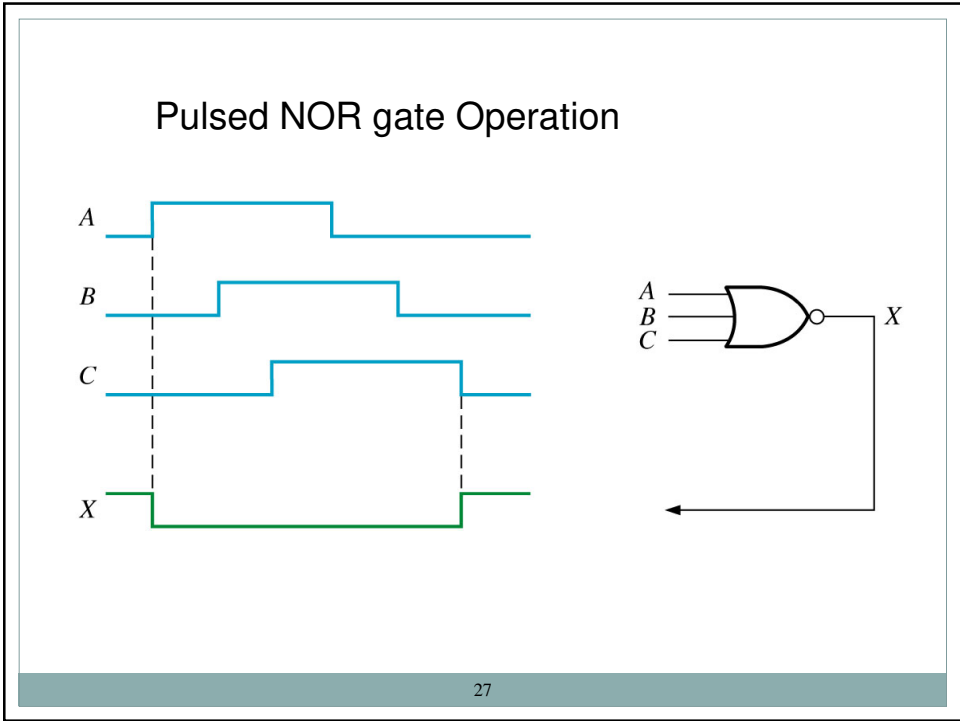
25

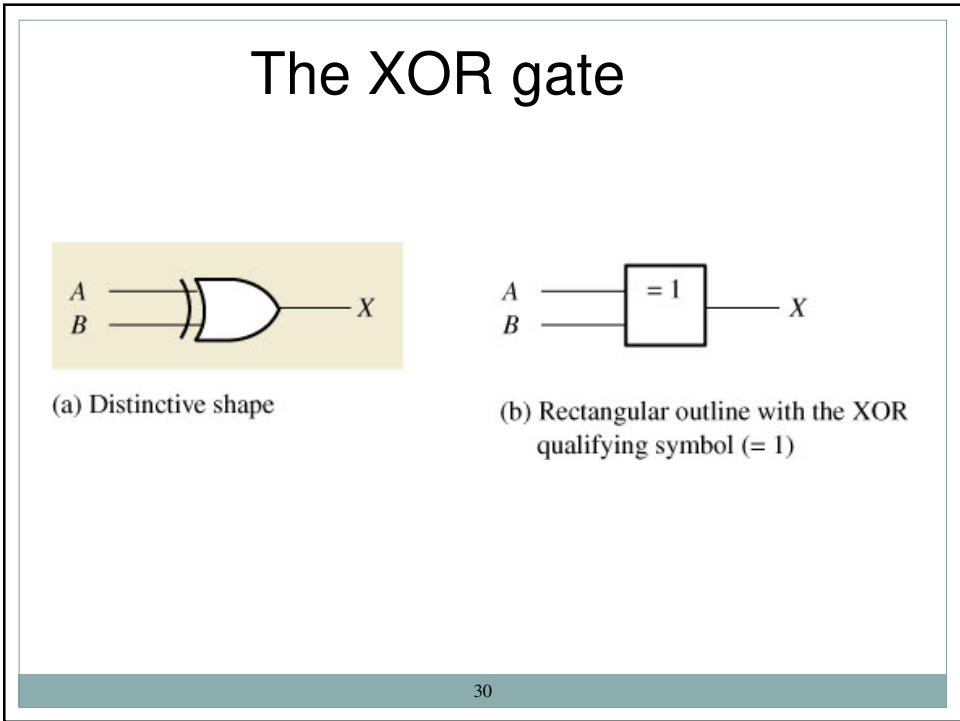
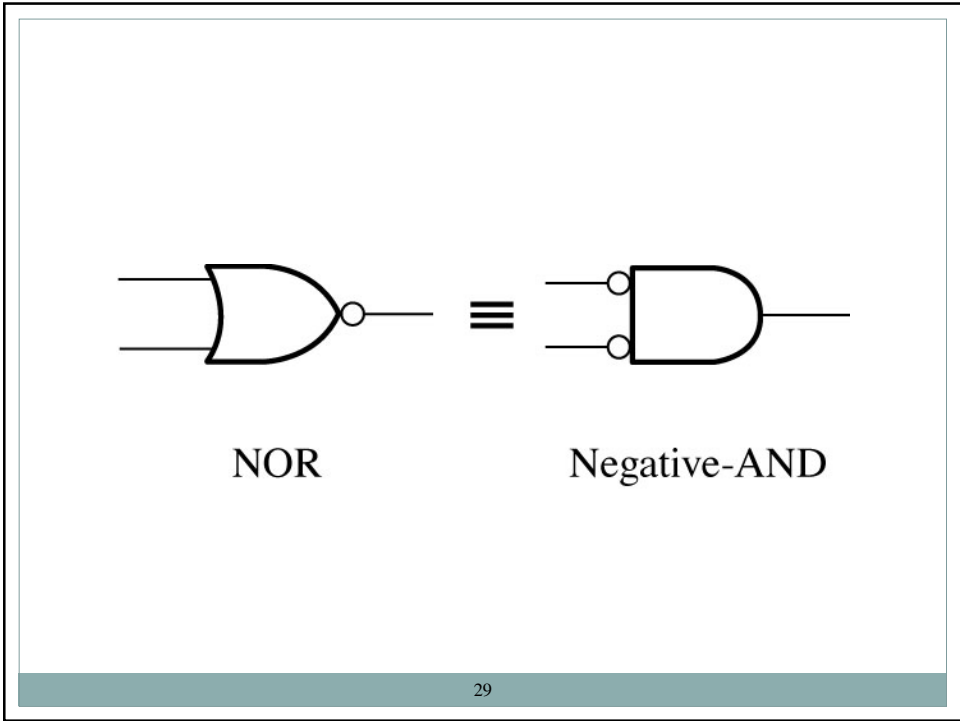
The NOR gate

The **NOR gate** produces a LOW output if any input is HIGH; if all inputs are HIGH, the output is LOW. For a 2-input gate, the truth table is

Inputs		Output
A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

The **NOR** operation is shown with a plus sign (+) between the variables and an overbar covering them. Thus, the NOR operation is written as $X = \overline{A + B}$.





The XOR gate

X

The **XOR gate** produces a HIGH output only when both inputs are at opposite logic levels. The truth table is

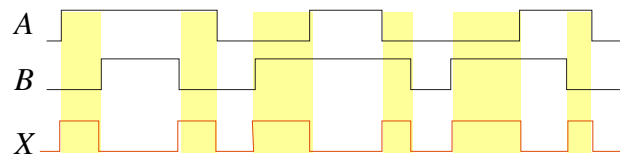
Inputs		Output
A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

The **XOR** operation is written as $X = \bar{A}B + A\bar{B}$.

Alternatively, it can be written with a circled plus sign between the variables as $X = A \oplus B$.

The XOR gate

Example waveforms:

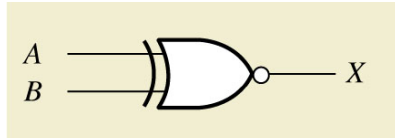


Notice that the XOR gate will produce a HIGH only when exactly one input is HIGH.

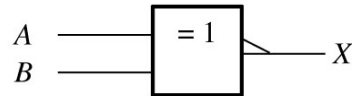
Question If the A and B waveforms are both inverted for the above waveforms, how is the output affected?

There is no change in the output.

The XNOR gate



(a) Distinctive shape



(b) Rectangular outline

33

The XNOR gate

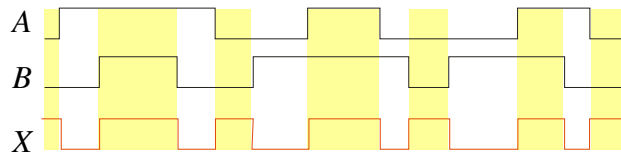
The **XNOR gate** produces a HIGH output only when both inputs are at the same logic level. The truth table is

Inputs		Output
A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

The **XNOR** operation shown as $X = \overline{A}B + A\overline{B}$. Alternatively, the XNOR operation can be shown with a circled dot between the variables. Thus, it can be shown as $X = A \odot B$.

The XNOR gate

Example waveforms:



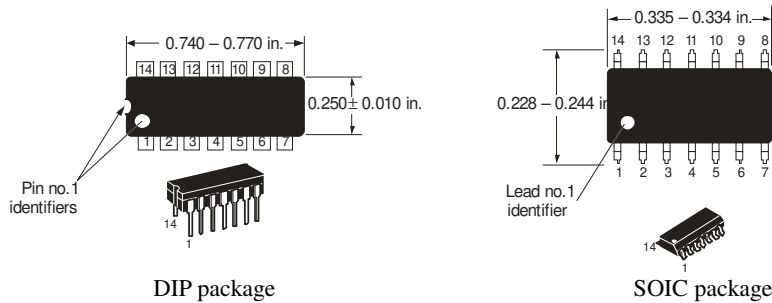
Notice that the XNOR gate will produce a HIGH when both inputs are the same. This makes it useful for comparison functions.

Question If the A waveform is inverted but B remains the same, how is the output affected?

The output will be inverted.

Fixed Function Logic

Two major fixed function logic families are TTL and CMOS. A third technology is BiCMOS, which combines the first two. Packaging for fixed function logic is shown.



Types of Fixed Function Logic Gates

Quad 2-input NAND- 00

Quad 2-input NOR- 02

Hex inverter – 04

Quad 2-input AND- 08

Triple 3-input NAND – 10

Triple 3-input AND – 11

Dual 4-input NAND- 20

Triple 3-input NOR – 27

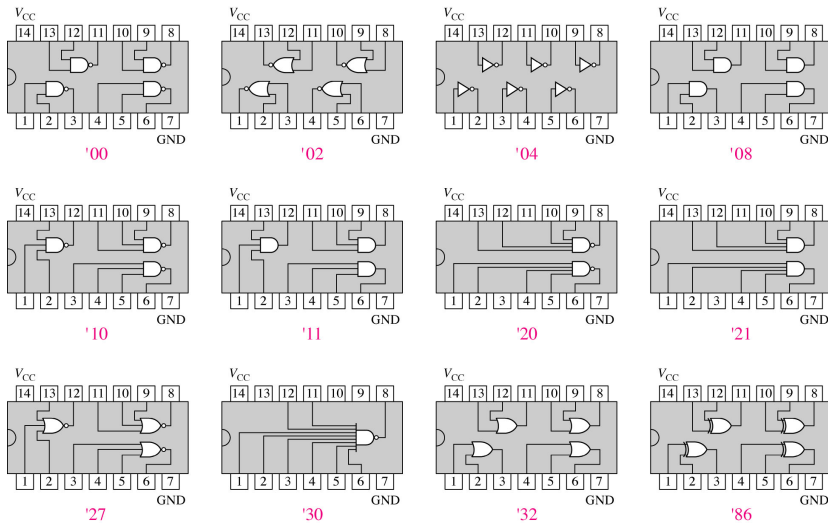
Single 8-input NAND- 30

Quad 2-input OR- 32

Quad 2-input XOR- 86

37

Pin configuration of basic gates



38

Level of Integration

Small-scale integration (SSI): contains fewer than 10 gates in one IC.

Medium-scale integration (MSI): contains approximately 10 to 1,000 gates in a single IC. Such as decoders, adders and multiplexers, etc.

Large Scale Integration (LSI): contains thousands of gates in a single IC. Such as processors, memory chips and programmable logic devices, etc.

Very Large Scale Integration (VLSI): contains hundred of thousands of gates within a single packages. Large memory arrays and complex microcomputer chips, etc.

Fixed Function Logic

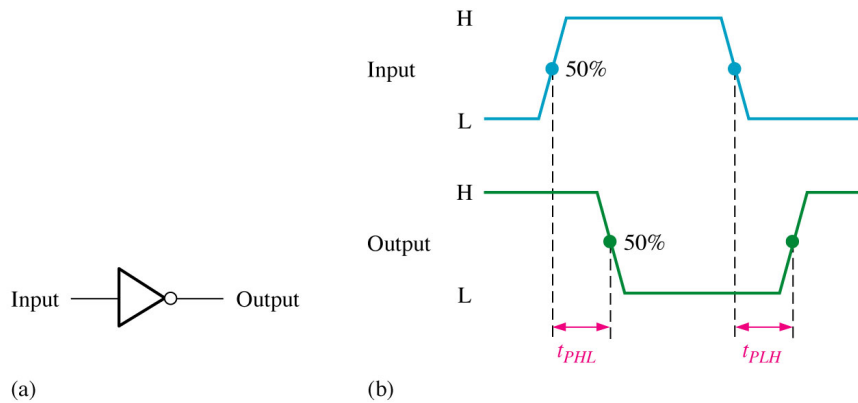
Data sheets include limits and conditions set by the manufacturer as well as DC and AC characteristics. For example, some maximum ratings for a 74HC00A are:

MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_{CC}	DC Supply Voltage (Referenced to GND)	-0.5 to + 7.0 V	V
V_{in}	DC Input Voltage (Referenced to GND)	-0.5 to V_{CC} +0.5 V	V
V_{out}	DC Output Voltage (Referenced to GND)	-0.5 to V_{CC} +0.5 V	V
I_{in}	DC Input Current, per pin	±20	mA
I_{out}	DC Output Current, per pin	±25	mA
I_{CC}	DC Supply Current, V_{CC} and GND pins	±50	mA
P_d	Power Dissipation in Still Air, Plastic or Ceramic DIP † SOIC Package † TSSOP Package †	750 500 450	mW
T_{stg}	Storage Temperature	-65 to + 150	°C
T_L	Lead Temperature, 1 mm from Case for 10 Seconds Plastic DIP, SOIC, or TSSOP Package Ceramic DIP	260 300	°C

Performance Characteristics and Parameters

Propagation delay: is the average transition delay time for a signal to propagate from input to output when the input changes its value.



41

DC Supply Voltage:

CMOS 5 V (2 to 6 V) or 3.3 V (2 V to 3.6 V).

TTL 5 V (4.5 V to 5.5 V)

Power dissipation: supplied power required to operate the gate, which is a product of the dc supply voltage and the average supply current.

Input and Output Logic Levels:

CMOS 5 V ($V_{IL} = 1.5V$; $V_{IH} = 3.5V$)

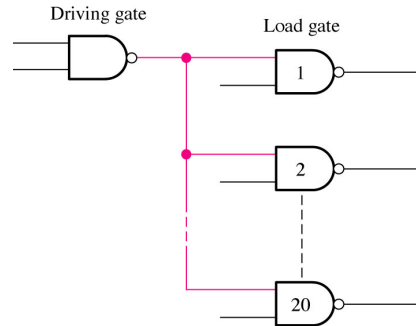
TTL ($V_{IL} = 0.8V$; $V_{IH} = 2.0V$)

CMOS 5 V ($V_{OL} = 0.33V$; $V_{OH} = 4.4V$)

TTL ($V_{OL} = 0.4V$; $V_{OH} = 2.4V$)

42

Fan-out and loading: specifies the number of standard loads that the output of a gate can drive without effecting its normal operation.

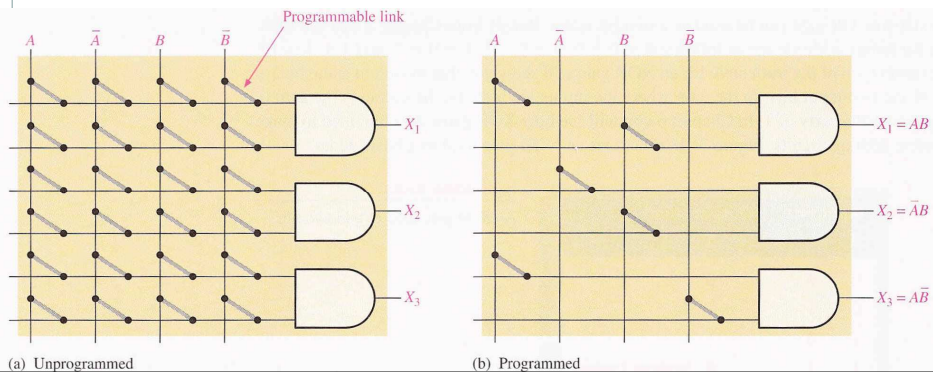


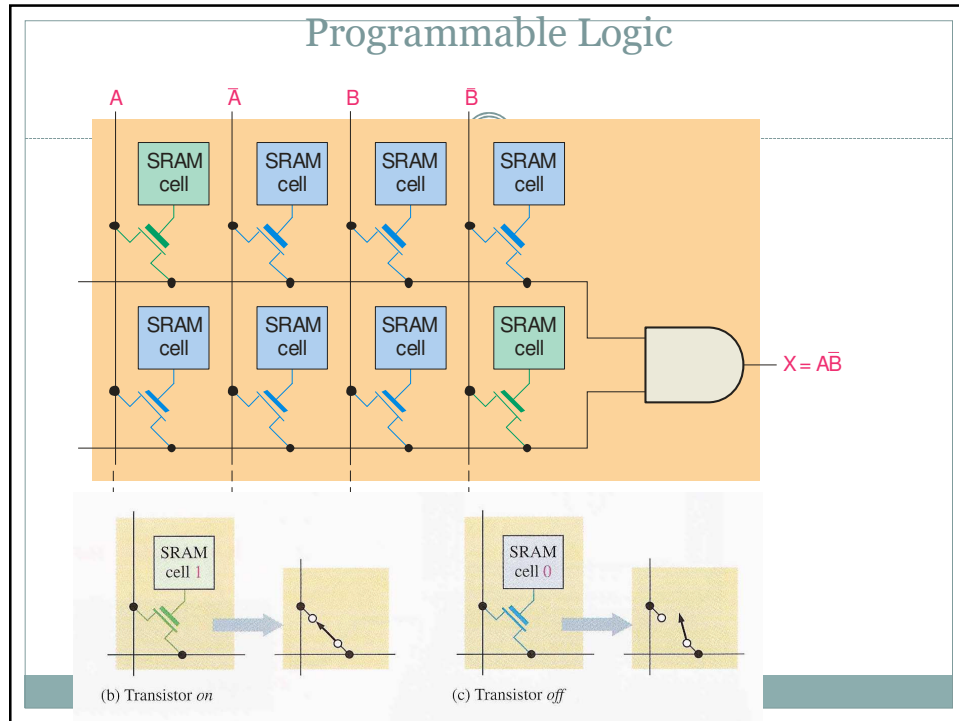
43

Programmable Logic

44

A Programmable Logic Device (PLD) can be programmed to implement logic. There are various technologies available for PLDs. Many use an internal array of AND gates to form logic terms. Many PLDs can be programmed multiple times.





46

- In general, the required logic for a PLD is developed with the aid of a computer. The logic can be entered using a Hardware Description Language (HDL) such as VHDL. Logic can be specified to the HDL as a text file, a schematic diagram, or a state diagram.
- A text entry for programming a PLD in VHDL as a 2-input NAND gate is shown for reference in the following slide. In this case, the inputs and outputs are first specified. Then the signals are described. Although you are probably not familiar with VHDL, you can see that the program is simple to read.

```

entity NandGate is
    port(A, B: in bit;
        LED: out bit);
end entity NandGate;

architecture GateBehavior of NandGate is
    signal A, B: bit;
    begin
        X <= A nand B;
        LED <= X;
    end architecture GateBehavior;
    
```

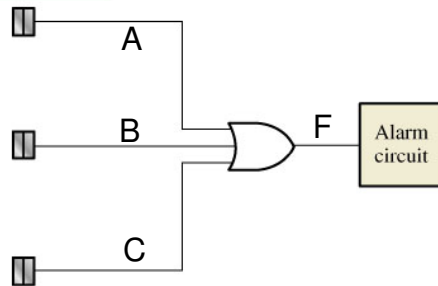
47

Simple Applications of Logic Gates

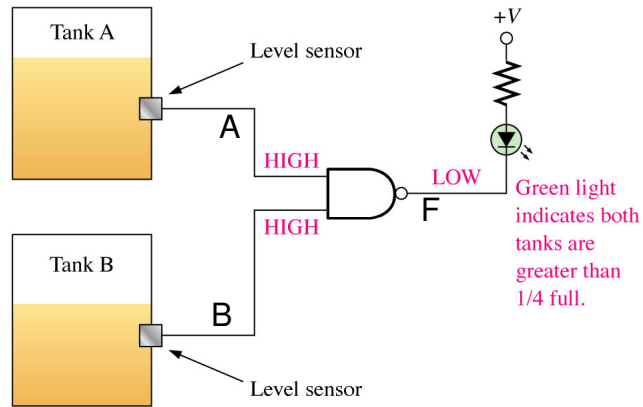
Application-1: Intruder alarm system

Open door/window sensors

HIGH = Open
LOW = Closed

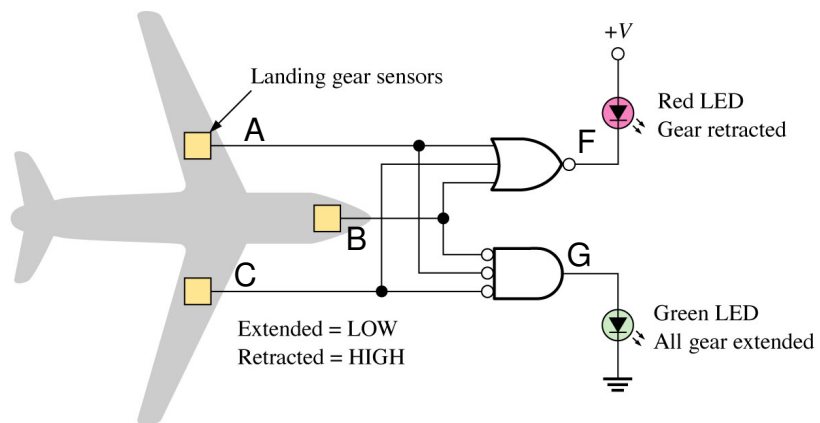


Application-2: Tank fluid level sensor



49

Application3: Aircraft landing gear



50

Thank you for your attention

Any question?

References

1. T. Floyd, "Digital Fundamental", 10th Ed., USA: PrenticeHall, 2008
2. R.J. Tocci, "Digital Systems: Principles and Applications", 10th Ed., USA: Prentice-Hall, 2006
3. W. Kleitz, "Digital Electronics: A Practical Approach", 8th Ed., USA: Prentice-Hall, 2007
4. Begnell and Donovan, "Digital Electronics", 5th Ed., USA: Delmar Thomson Learning, 2006