# SE311 Digital Systems Design

## Lecture 6: Logic Functions

Dr. Samir Al-Amer
(Term 041)

# Conversion between Canonical Forms

**Two binary variables x,y**

$$F = \sum(0,3) \qquad = \qquad \prod(1,2)$$

**Three variables x,y,z**

$$F = \sum(1,2,4) \qquad = \qquad \prod(0,3,5,6,7)$$

**all terms missing in one form appears in the other form.**

# Other Logic Functions

| | |
|---|---|
| $AND$ | $F(x,y) = xy$ |
| $OR$ | $F(x,y) = x + y$ |
| $Complement$ | $F(x,y) = x'$ |
| | $F(x,y) = y'$ |
| $Null$ | $F(x,y) = 0$ |
| $Identity$ | $F(x,y) = 1$ |

# Logic Functions

With two binary variables $2^{2n} = 16$ *possible functions*

* Two constants :

$Null \quad F(x, y) = 0 \qquad Identity \quad F(x, y) = 1$

* Four unary functions (functions of one variable)

$Complement \quad F(x, y) = x' \qquad F(x, y) = y'$

$Transfer \quad F(x, y) = x \qquad F(x, y) = y$

* Ten binary functions

$AND \quad F(x, y) = xy$

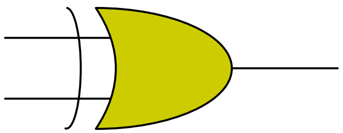
$OR \quad F(x, y) = x + y$

# Logic Functions

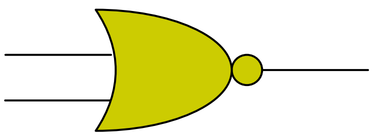
* Ten binary functions

| | | |
|---|---|---|
| *AND* | $F(x,y) = xy$ | |
| *OR* | $F(x,y) = x + y$ | |
| Inhabitation | $F(x,y) = xy'$ | $F(x,y) = x'y$ |
| Implications | $F(x,y) = x + y'$ | $F(x,y) = x' + y$ |
| *XOR* | $F(x,y) = x \oplus y = xy' + x'y$ | |
| *NOR* | $F(x,y) = x \downarrow y = (x + y)'$ | |
| *NAND* | $F(x,y) = x \uparrow y = (xy)'$ | |
| *Equivalence* | $F(x,y) = xy + x'y'$ | |

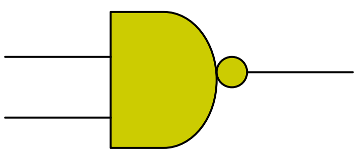
# Digital Logic Gates



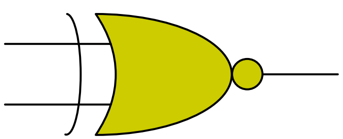
*XOR* $F(x,y) = x \oplus y$



*NOR* $F(x,y) = x \downarrow y$

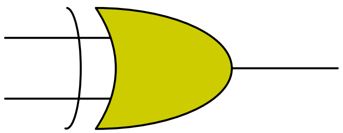


*NAND* $F(x,y) = x \uparrow y$



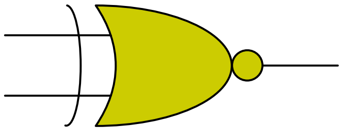
*Equivalence* $F(x,y) = xy + x'y'$

# XOR & Equivalence



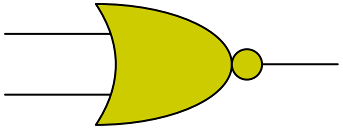
$XOR \qquad F(x,y) = x \oplus y$

$Equivalence \qquad F(x,y) = xy + x'y'$

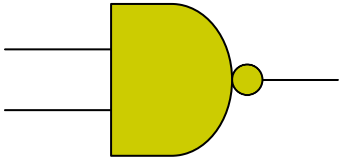


| x | y | XOR(x,y) | XNOR(x,y) |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

# NOR & NAND



NOR

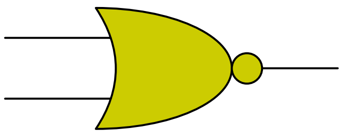


NAND

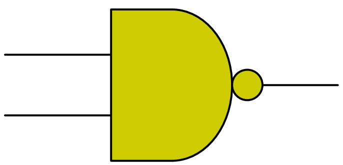
| x | y | NOR(x,y) | NAND(x,y) |
|---|---|---|---|
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 |

# NOR & NAND

## Commutative but not associative



NOR



NAND

$$NOR$$

$$x \uparrow y = y \uparrow x$$

$$(x \uparrow y) \uparrow z \neq x \uparrow (y \uparrow z)$$

$$NAND$$

$$x \downarrow y = y \downarrow x$$

$$(x \downarrow y) \downarrow z \neq x \downarrow (y \downarrow z)$$

# Extensions to Multiple Inputs

Some of the functions are not commutative or not associative which create some difficulties or become impractical

NOR and NAND are commutative but not associative

To overcome the difficulty NAND and NOR are redefined

$$x \downarrow y \downarrow z = (x + y + z)'$$

$$x \uparrow y \uparrow z = (xyz)'$$

# Extensions to Multiple Inputs

**XOR and equivalence are commutative and associative but XOR and equivalence with more than two inputs are not common logic gates.**

For two inputs, the output of XOR =1 if the inputs are different

For two inputs or more:

the output of XOR =1 if the number of 1's in the input is odd

# Positive and Negative Logic

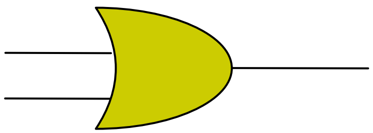
Two logic values 0 and 1

Two signal levels High (H) : the high level of the signal

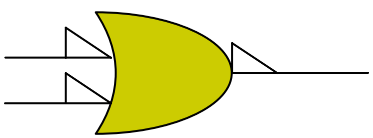
Low (L) : the low level of the signal

| | | |
|---|---|---|
| Positive Logic | HIGH corresponds to | 1 |
| | LOW corresponds to | 0 |
| Negative Logic | HIGH corresponds to | 0 |
| | LOW corresponds to | 1 |

# Digital Logic Gates

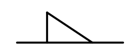


OR Gate

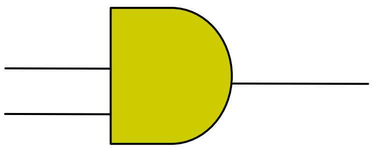


Negative Logic OR Gate

Physical gates can work in both positive or negative logic

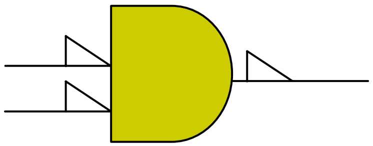
The symbol  is known as polarity indicator and is used to indicate that negative logic is used

# Digital Logic Gates



AND Gate

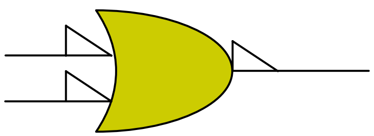
| x | y | z |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

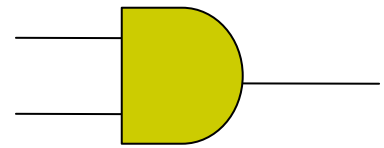


Negative Logic AND Gate

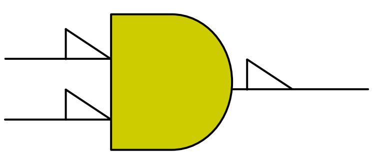
| x | y | z |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 0 | 0 | 0 |

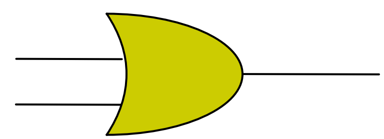
# Positive and Negative Logic





Negative Logic OR Gate is equivalent to AND gate





Negative Logic AND Gate is equivalent to OR gate

# Integrated Circuits

## Level of integration

| | Family | Remark |
|---|---|---|
| SSI | Small Scale Integration | Less than 10 gates per chip<br>Simple logic circuits |
| MSI | Medium Scale Integration | 10-100 gates per chip<br>**Elementary digital operations: Adder, decoder** |
| LSI | Large Scale Integration | Thousands of gates per chip<br>Processors, memory chips, |
| VLSI | Very Large Scale Integration | Hundreds of thousands of gates per chip<br>Large memory arrays, microcomputer chips |

# Digital Logic Families

Digital Logic Family:

A collection of different integrated circuits that do different functions but **they have similar input, output and internal circuit characteristics**.

ICs from the same family can be easily used together.

Examples: TTL family, CMOS family

Within each family there are “series” of IC with similar features

# Digital Logic Families

## Examples

**Standard TTL family:**

Voltage range [0 to 5]

Input

ON voltage >2 V

OFF voltage <0.8 V

**CMOS family** (4000series):

Voltage range [0 to 10]

Input

ON voltage >7 V

OFF voltage <3 V

# Digital Logic Families

| | Family | Remark |
|---|---|---|
| TTL | Transistor-Transistor Logic | Has been in operation for long time<br>Almost obsolete |
| ECL | Emitter –Coupled Logic | High speed |
| MOS | Metal Oxide Semiconductor | Low power consumption , Low speed<br>Used for high component density |
| CMOS | Complementary Metal Oxide Semiconductor | Low power consumption<br>More dominant |

# Important Chip Characteristics

| | |
|---|---|
| Fan-in | The number of inputs available to the gate |
| Fan-out | The number of standard load that the output of a gate can derive without affecting the normal operations |
| Power dissipation | The amount of power consumed by the gate. |
| Propagation delay | Average transition delay for the signal to travel between the input and the output |
| Noise Margin | Maximum external noise that can be added to the input signal without affecting the output. |

# Design and Implementation

- In the design phase of a logic  one can delay consideration of issues on the characteristic of integrated circuits or the selection of a logic family to a later stage.
- In the implementation phase these issues will be significant and some refinements to the design may be needed.