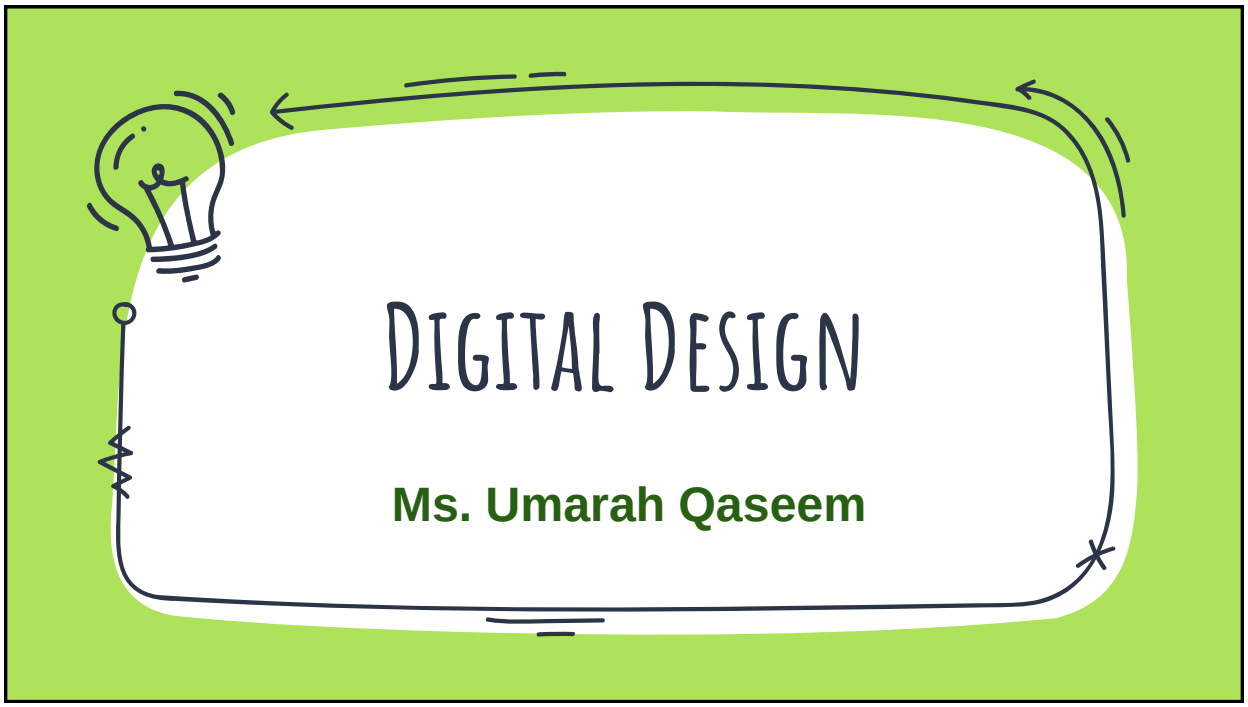
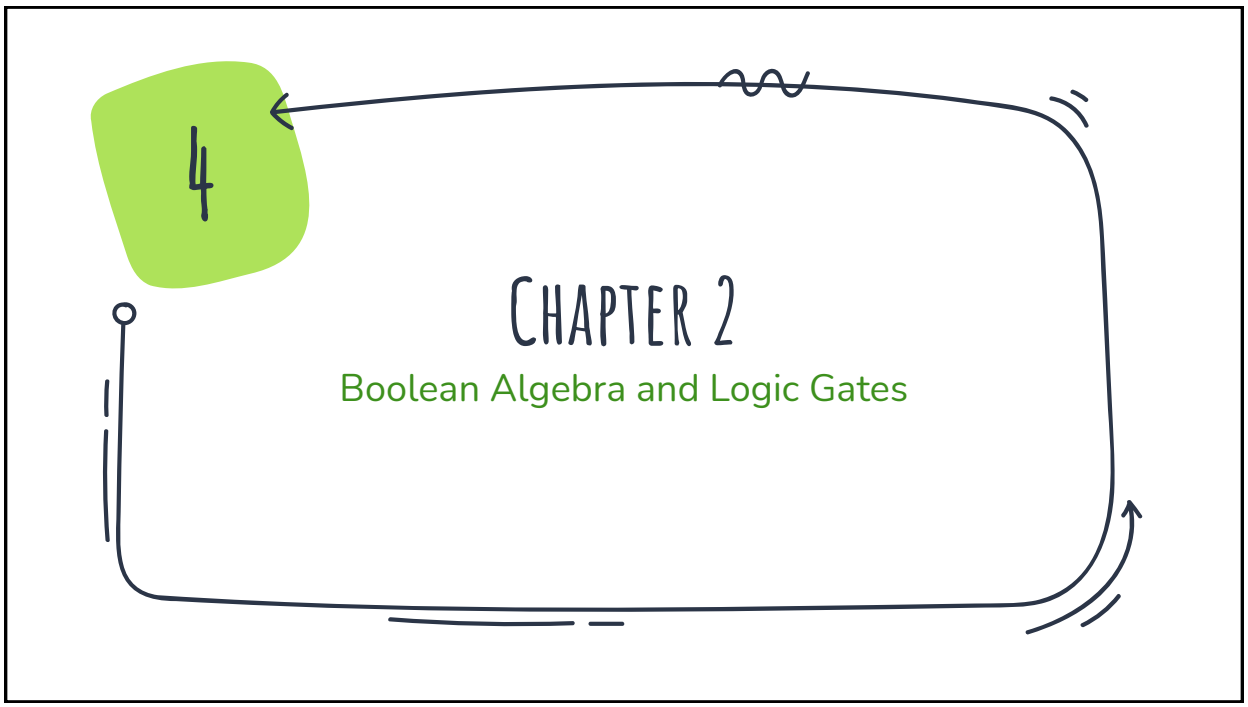

DIGITAL DESIGN
Ms. Umarah Qaseem

1


4
CHAPTER 2
Boolean Algebra and Logic Gates

3

9

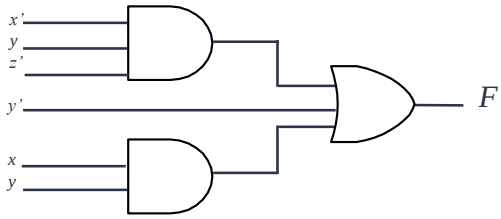
Standard Forms

- Sum of Products (SOP)
  - F= *y' + xy+ x'yz'*
  - Sum denotes OR of these terms
- Product of Sums (POS)
  - F= *x(y'+z)(x'+y+z')*
  - Product denotes AND of these terms

10

Sum of Products

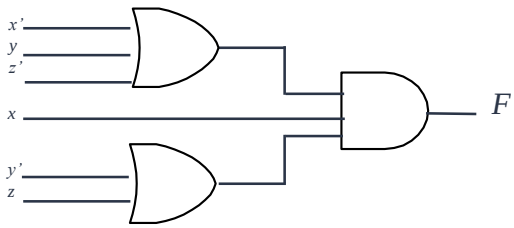
F= y' + xy+ x'yz'



11

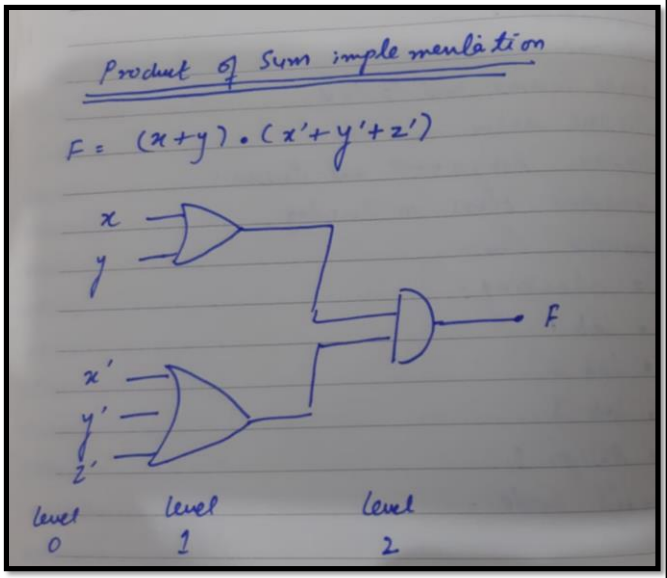
Product of Sums

- F= x(y'+z)(x'+y+z')



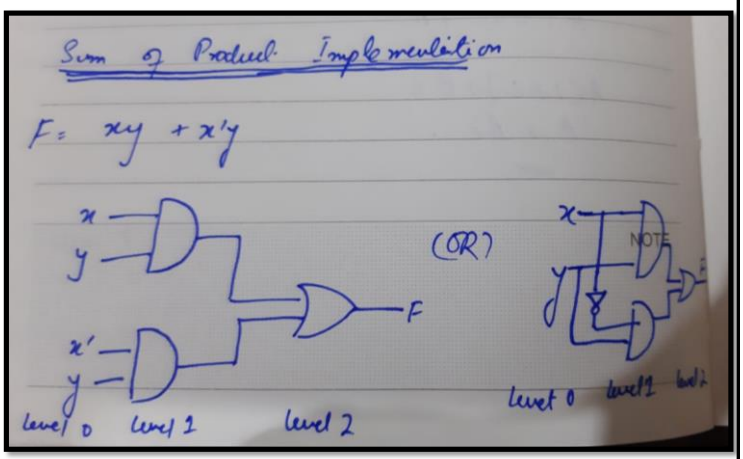
12

2-LEVEL IMPLEMENTATION POS



13

2-LEVEL IMPLEMENTATION SOP



14

Two-level implementations

- Product of sums expressions can be implemented with two-level circuits as following
  - literals and their complements at the “0th” level
  - **OR gates at the first level**
  - a single **AND gate at the second level**

15

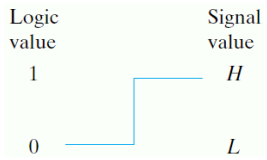
Two-level implementations

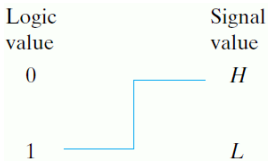
- Sum of product expressions can be implemented with two-level circuits as following
  - literals and their complements at the “0th” level
  - **AND gates at the first level**
  - a single **OR gate at the second level**

16

Positive and Negative Logic

- Signal levels as high (H) or low (L)
- Positive logic
  - H → 1
  - L → 0
- Negative logic
  - H → 0
  - L → 1





17

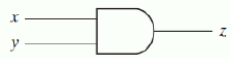
Positive and Negative Logic

- Truth Table for H and L

x	y	F
L	L	L
L	H	L
H	L	L
H	H	H

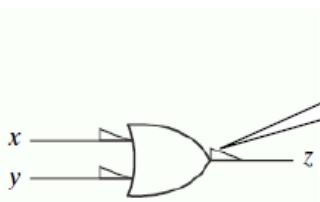
- In positive logic: AND

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



- In negative logic: OR

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



Arrows indicate negative logic

18

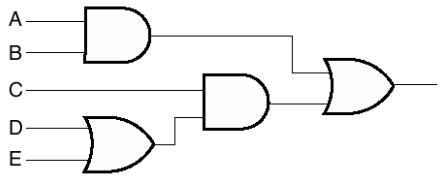
Conversions

19

Non-Standard Boolean Function to Standard Func

- Boolean Expression in non-standard form
  F=AB+C(D+E)

- Neither SOP or POS
- Implemented in 3 levels

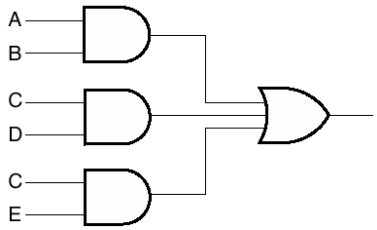


(a) AB + C (D + E)

20

Non-Standard Boolean Function to Standard Func

- F=AB+C(D+E)
- Convert to Standard Form by distributive law
- F=AB+C(D+E) = AB+CD+CE
- Now it becomes a 2 Level Standard SOP Circuit



(b) AB + CD + CE

21

Conversion b/w Canonical Forms

- Complement of a function expressed as sum of minterms equals sum of minterms **missing from the original function**

$F(A,B,C) = \sum(1,4,5,6,7) = m_1+m_4+m_5+m_6+m_7$

- Complement is

$F'(A,B,C) = \sum(0,2,3) = m_0+m_2+m_3$

22

Example: Conversion from standard to Canonical Form: Sum of Minterms

Express the Boolean function $F = x + y'z$ in a sum of minterms.

AIM: Convert it to 3 variable AND Terms and take their SUM

$$x = x(y + y') = xy + xy'$$

$$xy = xy(z + z') = xyz + xyz'$$

$$xy' = xy'(z + z') = xy'z + xy'z'$$

$$y'z = y'z(x + x') = xy'z + x'y'z$$

Adding all terms and excluding recurring terms:

$$F(x, y, z) = x'y'z + xy'z' + xy'z + xyz' + xyz$$

$$F(x, y, z) = m_1 + m_4 + m_5 + m_6 + m_7 = \sum(1,4,5,6,7)$$

24

Example: Conversion from standard to Canonical Form: Product of Maxterms

Convert the standard equation to canonical form.

F = (x'+y) (x+z) (y+z)

$$x' + y + zz' = (x' + y + z)(x' + y + z')$$

$$x + z + yy' = (x + z + y)(x + z + y')$$

$$y + z + xx' = (y + z + x)(y + z + x')$$ (Remove any recurring terms)

$$F = (x + y + z)(x + y' + z)(x' + y + z)(x' + y + z')$$

$M_0M_2M_4M_5 = F(x,y,z) = \Pi(0,2,4,5)$

25

Example: Conversion from standard to Canonical Form: Product of Maxterms

Express the Boolean function $F = xy + x'z$ in a Product of Maxterms.

AIM: Convert it to 3 variable OR Terms and take their Product

$F = xy + x'z = (xy + x')(xy + z)$ (Distributive law)

$(xy + x') = (x + x')(y + x')$

$(xy + z) = (x + z)(y + z)$

$x'+y+zz' = (x'+y+z)(x'+y+z')$

$x+z+yy' = (x+z+y)(x+z+y')$

$y+z+xx' = (y+z+x)(y+z+x')$ (Remove any recurring terms)

$F = (x+y+z)(x+y'+z)(x'+y+z)(x'+y+z')$

$M_0M_2M_4M_5 = F(x,y,z) = \Pi(0,2,4,5)$

26

SUMMARY OF CONVERSION

- To convert into canonical form:
- SOP: multiply each term with (x+x') (x is missing literal)
- POS: Add with each term (zz') (z is missing literal)

- To convert from SOP to POS:
- Use distributive law
  - **a) *x + (y.z) = (x+y).(x+z)***
  - **b) x.(y + z) = (x .y) + (x.z)**

27

27

SOP ⟺ POS Using Truth Table

- Find Sum of Minterms and Product of MaxTerms of *F= xy+x'z*
- Find Sum of Minterms & Product of Maxterms from truth table

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

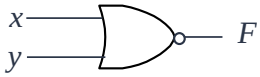
- *F(x,y,z)= ∑(1,3,6,7)*
- *F(x,y,z)= Π (0,2,4,5)*

28

35

NOR Gate

- 2-input NOR logic gate:



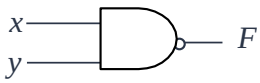
X	Y	F
0	0	1
0	1	0
1	0	0
1	1	0

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

36

NAND Gate

- 2-input NAND logic gate:



X	Y	F
0	0	1
0	1	1
1	0	1
1	1	0

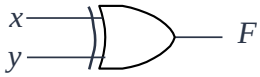
$$F = (X.Y)'$$

37

Exclusive OR Gate

- 2-input exclusive-OR (XOR) logic gate:

$$F = X \oplus Y$$



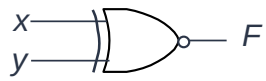
X	Y	F
0	0	0
0	1	1
1	0	1
1	1	0

38

Exclusive NOR Gate

- 2-input exclusive-NOR logic gate:

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



X	Y	F
0	0	1
0	1	0
1	0	0
1	1	1

39

39

Summary

Inputs		Outputs
x	y	xy
0	0	0
0	1	0
1	0	0
1	1	1

xy = x AND y = x * y
AND is true only if **both** inputs are true

Inputs		Outputs
x	y	x + y
0	0	0
0	1	1
1	0	1
1	1	1

x + y = x OR y
OR is true if **either** inputs are true

Inputs	Outputs
x	$\overline{x}$
0	1
1	0

x bar = NOT x
NOT inverts the bit
We will denote x bar as ~X

NOR is NOT of OR

x	y	x NOR y
0	0	1
0	1	0
1	0	0
1	1	0

NAND is NOT of AND

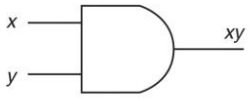
x	y	x NAND y
0	0	1
0	1	1
1	0	1
1	1	0

XOR is true if both inputs **differ**

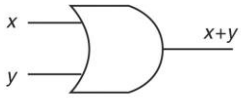
x	y	x XOR y
0	0	0
0	1	1
1	0	1
1	1	0

40

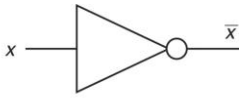
Summary



AND Gate



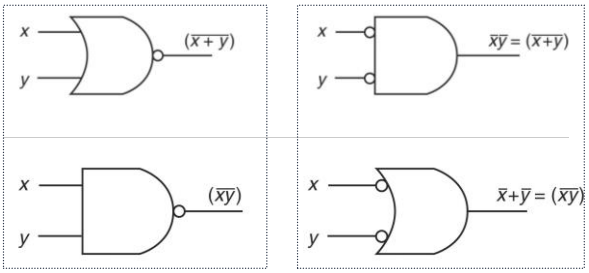
OR Gate



NOT Gate

Here we see the logic gates that represent the boolean operations previously discussed

XOR looks like OR but with the added curved line



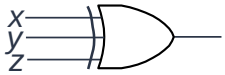
We typically represent NOR and NAND by the two on the left, but the two on the right are also correct

41

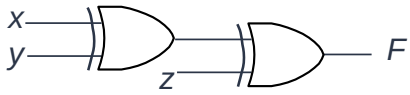
3 Input XOR

3-input exclusive-OR (XOR) logic gate:

$F = X \oplus Y \oplus Z$



Implemented by 3 Input Gates



Implemented by 2 Input Gates

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

42

NAND and NOR Conversion

43

We can make every gate using NAND or NOR gate

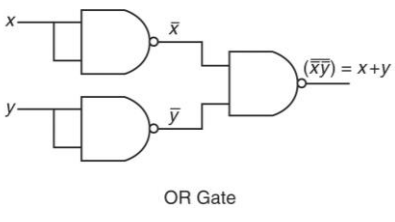
Universal Gates!

44

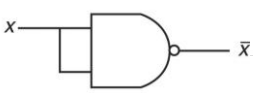
44

Using only NAND

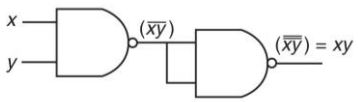
- NAND and NOR gates are called **Universal Gates**
- Using them we can create circuits that compute AND, OR and NOT



OR Gate



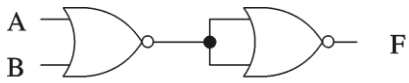
NOT Gate



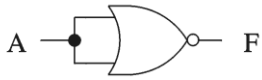
AND Gate

45

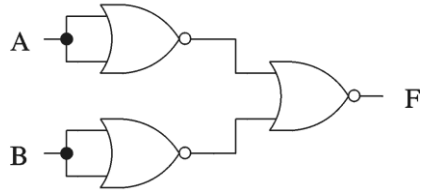
Using only NOR



OR gate



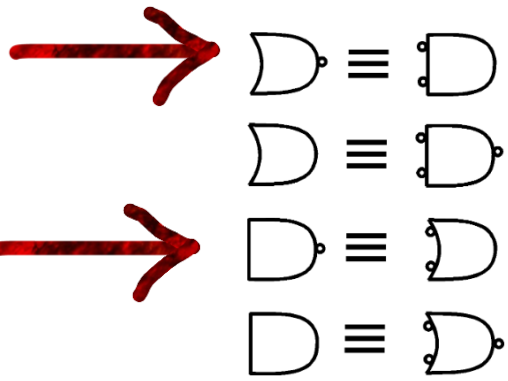
NOT gate



AND gate

46

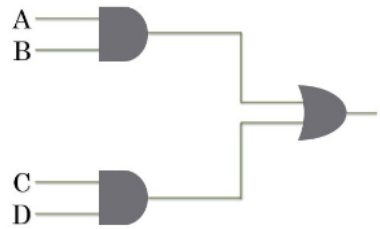
Main Key



AND Gate and OR Gate Equivalents

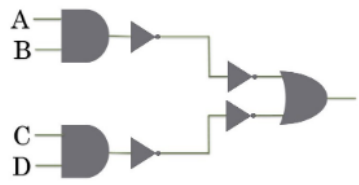
47

IMPLEMENT USING ONLY NAND: EXAMPLE 1



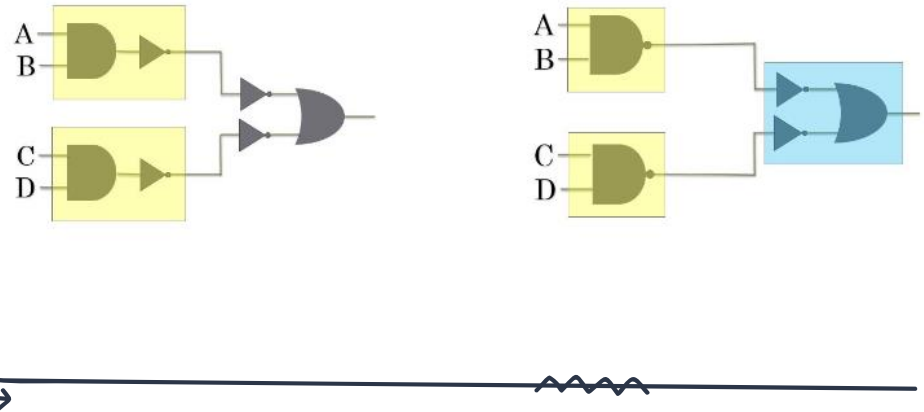
50

IMPLEMENT USING ONLY NAND



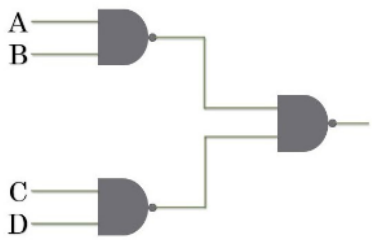
51

IMPLEMENT USING ONLY NAND



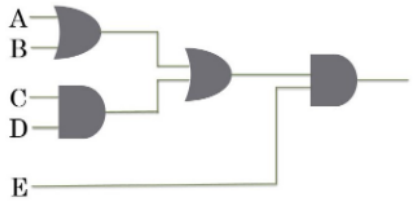
52

IMPLEMENT USING ONLY NAND



53

Example 2: Implement Using Only NAND



(A+B + CD)E

54

Example 2:

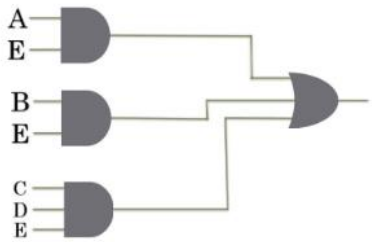
- The original circuit implements the boolean function :
  (A+B+CD)E
- Convert to Sum of Products(SOP).
- In this case, simply open parenthesis.

- We get: AE + BE + CDE.
- Since this Boolean equation is in SOP form, the circuit for this equation will be in a standard two-level implementation

62

Example 2: Implement Using Only NAND – Method 2

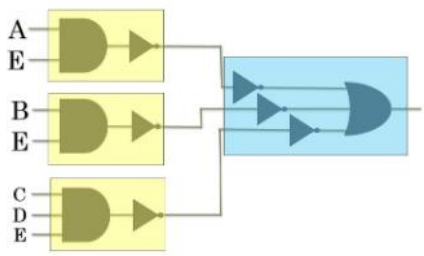
- AE + BE + CDE.



63

Example 2: Implement Using Only NAND – Method 2

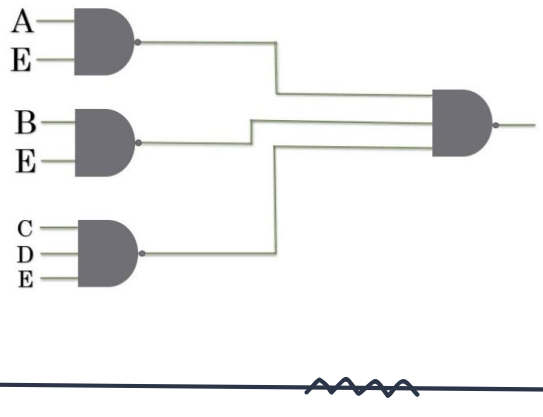
- AE + BE + CDE.



64

Example 2: Implement Using Only NAND – Method 2

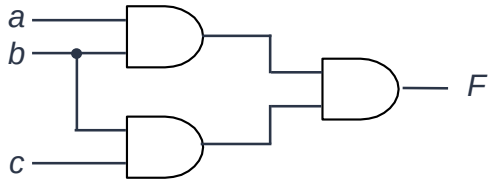
X AE + BE + CDE.



65

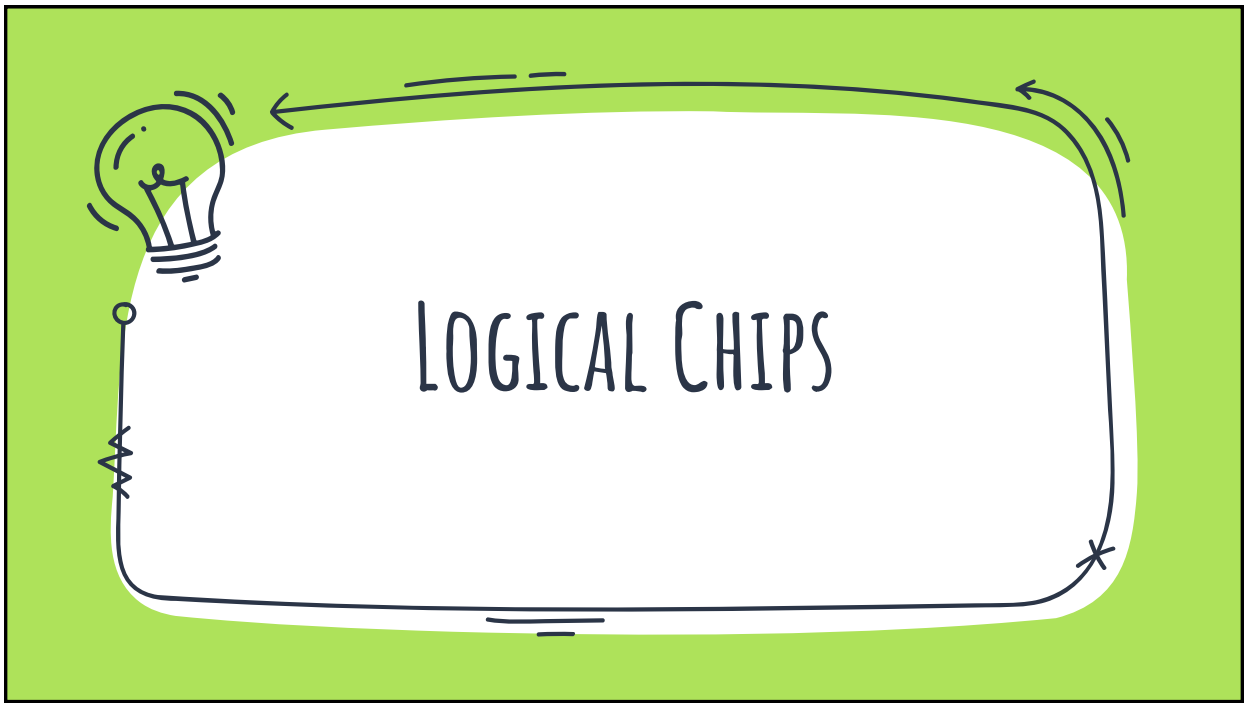
Home Task

Using ONLY NAND gates, draw a schematic for the following function: $F = (a.b)+(b.c)$



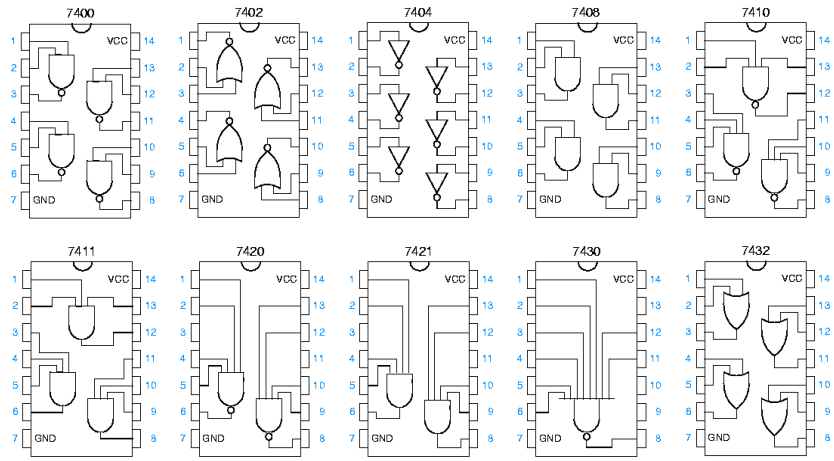
68

68



69

Logic Chips



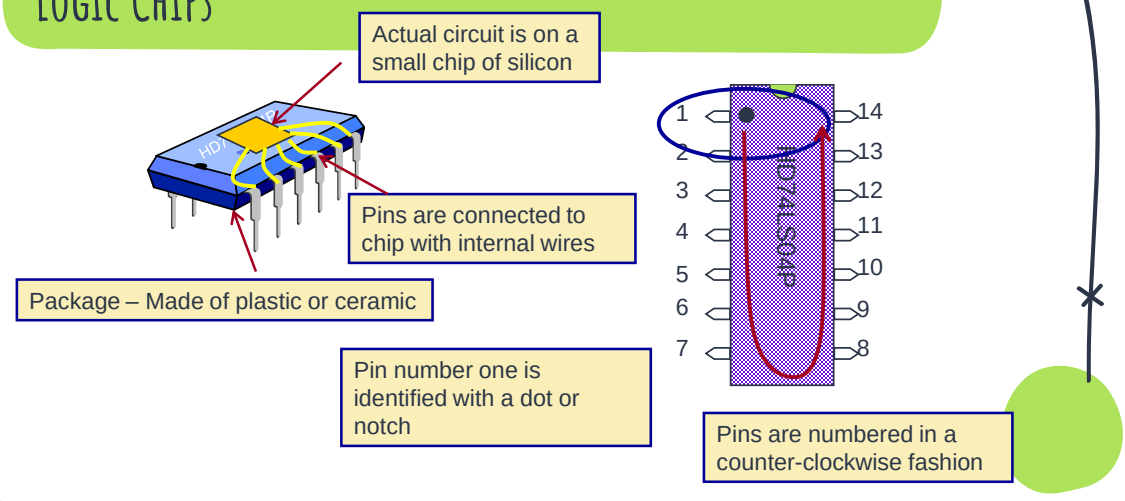
70

Logic Chips

- Digital Electronics devices are usually in a chip format.
- The chip is identified with a part number or a model number.
- A standard series starts with numbers 74, 4, or 14.
  - 7404 is an inverter
  - 7408 is an AND
  - 7432 is an OR
  - 4011B is a NAND

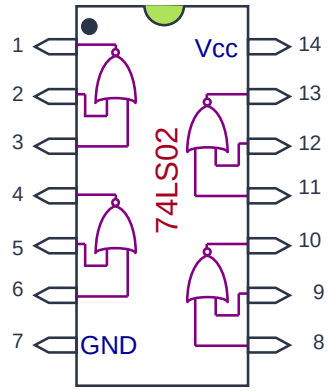
71

Logic Chips

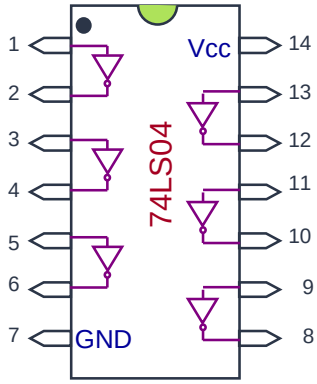


72

Logic Chips: Examples



NOR

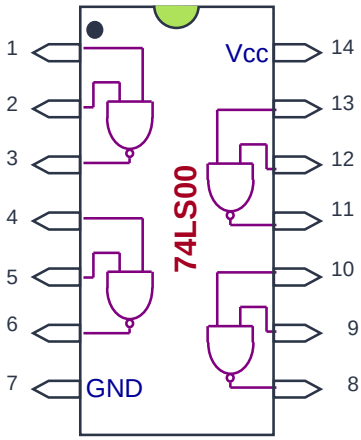


Inverter

73

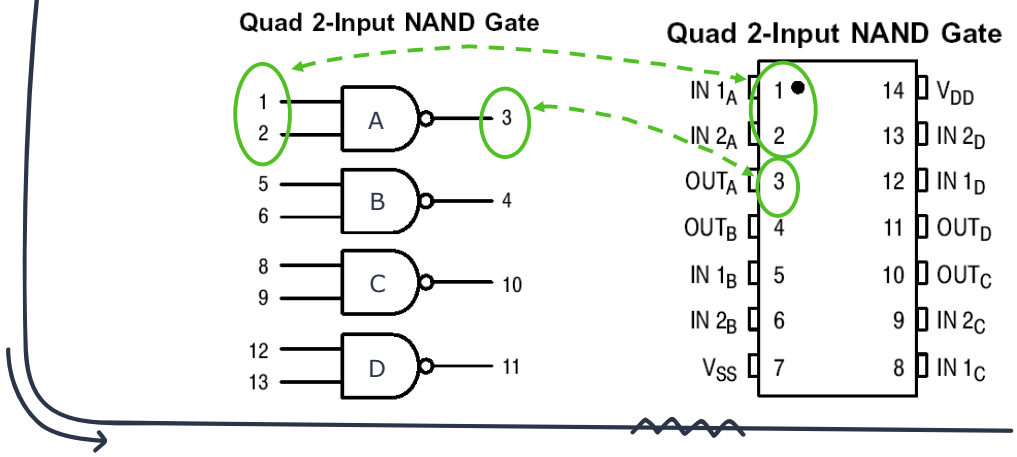
Logic Chips: Examples

NAND Gates



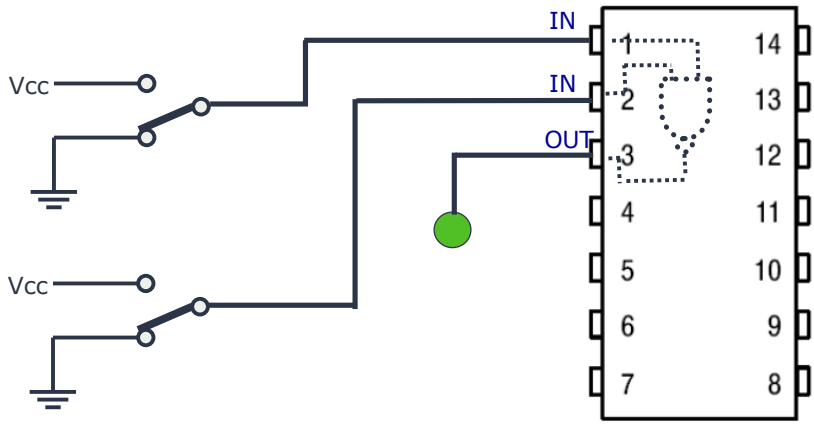
74

Logic Chips



75

Logic Chips



76

Integrated Circuits (IC)

- Small Scale Integration (SSI)
  - 10s of logic gates (previous examples) (late 1960s)
- Medium Scale Integration (MSI)
  - 10s to1000 of logic gates (Introduced in late 1960s)
- Large Scale Integration (LSI)
  - 10000s of logic gates (Introduced in early 1970s)
- Very Large Scale Integration (VLSI)
  - Millions of logic gates (Introduced in early 1970s)
- Ultra Large Scale Integration (ULSI)

77

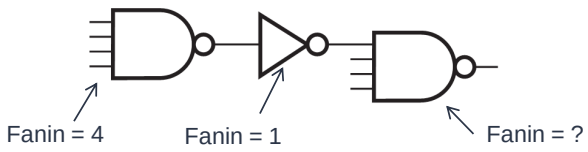
Digital Logic Families

- TTL
  - Transistor-Transistor Logic
- ECL
  - Emitter Coupled Logic
- MOS
  - Metal Oxide Semi Conductor
- CMOS
  - Complementary Metal Oxide Semi Conductor

78

Fan-In

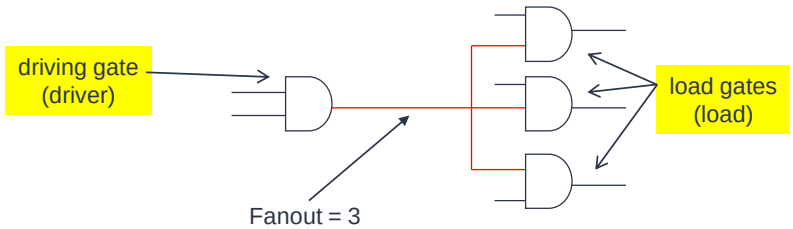
- **Fan in** of a gate is the number of inputs to the gate
- A 3-input OR gate has a fan-in = 3



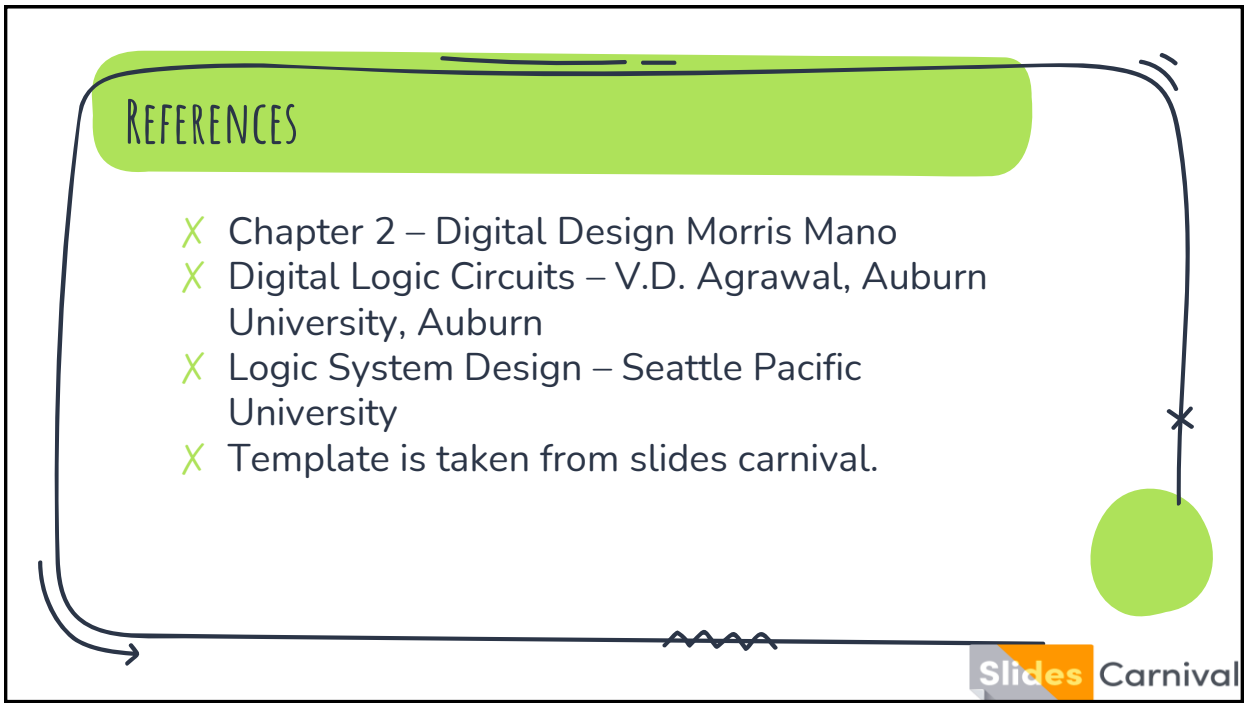
79

Fan-Out

- **Fan out** of a gate is the number of gates that it can drive
- The driven gate is called a load



80


References
Chapter 2 – Digital Design Morris Mano
Digital Logic Circuits – V.D. Agrawal, Auburn University, Auburn
Logic System Design – Seattle Pacific University
Template is taken from slides carnival.
Slides Carnival

81