

CpE358/CS381

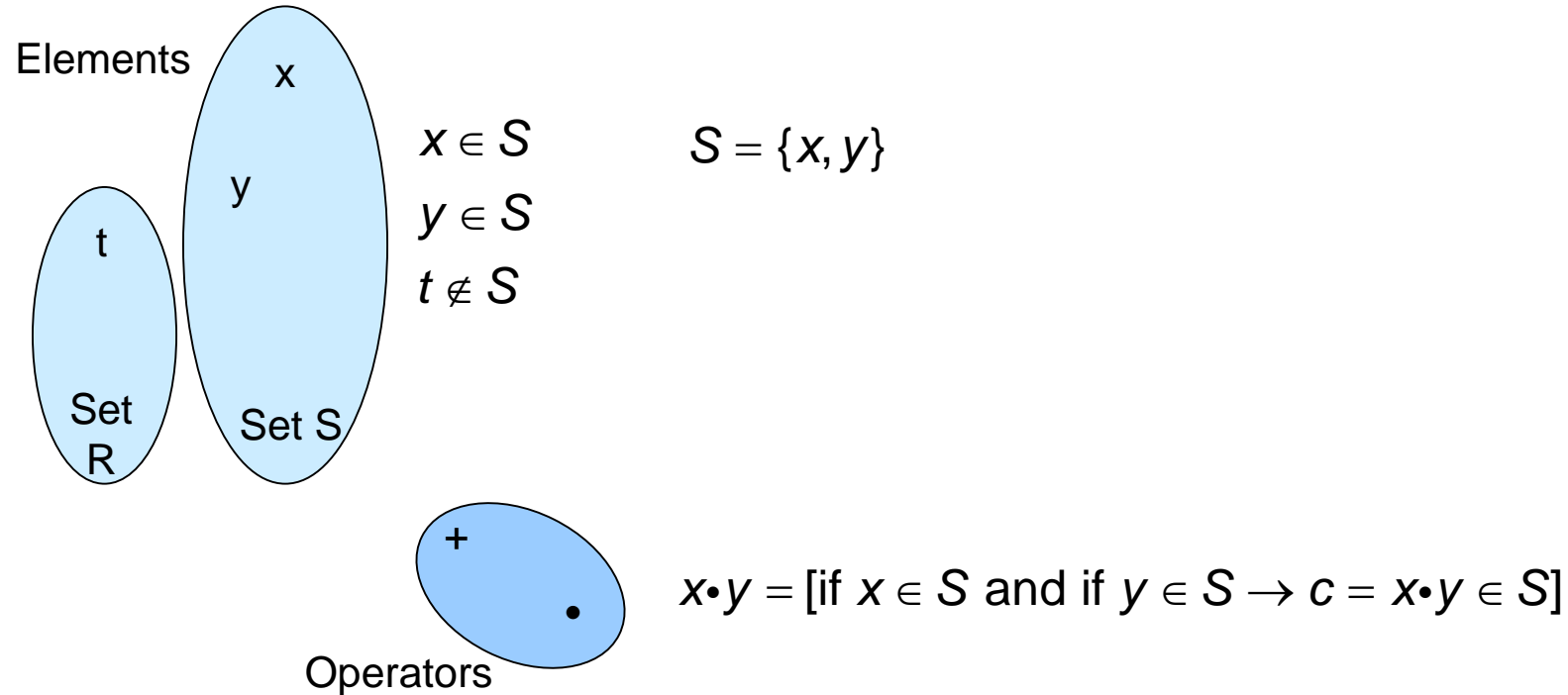
**Switching Theory and
Logical Design**

Class 2

Today's Material

- Fundamental concepts of digital systems (Mano Chapter 1)
- Binary codes, number systems, and arithmetic (Ch 1)
- **Boolean algebra (Ch 2)**
- Simplification of switching equations (Ch 3)
- Digital device characteristics (e.g., TTL, CMOS)/design considerations (Ch 10)
- Combinatoric logical design including LSI implementation (Chapter 4)
- Hazards, Races, and time related issues in digital design (Ch 9)
- Flip-flops and state memory elements (Ch 5)
- Sequential logic analysis and design (Ch 5)
- Synchronous vs. asynchronous design (Ch 9)
- Counters, shift register circuits (Ch 6)
- Memory and Programmable logic (Ch 7)
- Minimization of sequential systems
- Introduction to Finite Automata

Basic Concepts in Set Theory



Properties

1. Closure

- S is closed with respect to operator \bullet if
$$\forall a, b \in S, a \bullet b \in S$$

2. Associative

- A binary operator \bullet on set S is associative if
$$\forall a, b, c \in S, (a \bullet b) \bullet c = a \bullet (b \bullet c)$$

3. Commutative

- A binary operator \bullet on set S is commutative if
$$\forall a, b \in S, a \bullet b = b \bullet a$$

4. Identity

- A set S has an identity element e with respect to operator \bullet if
$$\forall x \in S, \exists e \in S: x \bullet e = e \bullet x = x$$

5. Inverse

- A set S with identity element e with respect to operator \bullet has an inverse if
$$\forall x \in S, \exists y \in S: x \bullet y = e$$

6. Distributive

- For a set S with operators \bullet and $+$, $+$ is distributive over \bullet if
$$\forall x, y, z \in S, x * (y \bullet z) = (x * y) \bullet (x * z) \in S$$

Properties

1. Closure

- S is closed with respect to operator \bullet if
$$\forall a, b \in S, a \bullet b \in S$$

2. Associative

- A binary operator \bullet on set S is associative if
$$\forall a, b, c \in S, (a \bullet b) \bullet c = a \bullet (b \bullet c)$$

3. Commutative

- A binary operator \bullet on set S is commutative if
$$\forall a, b \in S, a \bullet b = b \bullet a$$

4. Identity

- A set S has an identity element e with respect to operator \bullet if
$$\forall x \in S, \exists e \in S: x \bullet e = e \bullet x = x$$

5. Inverse

- A set S with identity element e with respect to operator \bullet has an inverse if
$$\forall x \in S, \exists y \in S: x \bullet y = e$$

6. Distributive

- For a set S with operators \bullet and $+$, $+$ is distributive over \bullet if
$$\forall x, y, z \in S, x * (y \bullet z) = (x * y) \bullet (x * z) \in S$$

Properties

1. Closure

- S is closed with respect to operator \bullet if
$$\forall a, b \in S, a \bullet b \in S$$

2. Associative

- A binary operator \bullet on set S is associative if
$$\forall a, b, c \in S, (a \bullet b) \bullet c = a \bullet (b \bullet c)$$

3. Commutative

- A binary operator \bullet on set S is commutative if
$$\forall a, b \in S, a \bullet b = b \bullet a$$

4. Identity

- A set S has an identity element e with respect to operator \bullet if
$$\forall x \in S, \exists e \in S: x \bullet e = e \bullet x = x$$

5. Inverse

- A set S with identity element e with respect to operator \bullet has an inverse if
$$\forall x \in S, \exists y \in S: x \bullet y = e$$

6. Distributive

- For a set S with operators \bullet and $+$, $+$ is distributive over \bullet if
$$\forall x, y, z \in S, x * (y \bullet z) = (x * y) \bullet (x * z) \in S$$

Properties

1. Closure

- S is closed with respect to operator \bullet if
$$\forall a, b \in S, a \bullet b \in S$$

2. Associative

- A binary operator \bullet on set S is associative if
$$\forall a, b, c \in S, (a \bullet b) \bullet c = a \bullet (b \bullet c)$$

3. Commutative

- A binary operator \bullet on set S is commutative if
$$\forall a, b \in S, a \bullet b = b \bullet a$$

4. Identity

- A set S has an identity element e with respect to operator \bullet if
$$\forall x \in S, \exists e \in S: x \bullet e = e \bullet x = x$$

5. Inverse

- A set S with identity element e with respect to operator \bullet has an inverse if
$$\forall x \in S, \exists y \in S: x \bullet y = e$$

6. Distributive

- For a set S with operators \bullet and $+$, $+$ is distributive over \bullet if
$$\forall x, y, z \in S, x * (y \bullet z) = (x * y) \bullet (x * z) \in S$$

Properties

1. Closure

- S is closed with respect to operator \bullet if
$$\forall a, b \in S, a \bullet b \in S$$

2. Associative

- A binary operator \bullet on set S is associative if
$$\forall a, b, c \in S, (a \bullet b) \bullet c = a \bullet (b \bullet c)$$

3. Commutative

- A binary operator \bullet on set S is commutative if
$$\forall a, b \in S, a \bullet b = b \bullet a$$

4. Identity

- A set S has an identity element e with respect to operator \bullet if
$$\forall x \in S, \exists e \in S: x \bullet e = e \bullet x = x$$

5. Inverse

- A set S with identity element e with respect to operator \bullet has an inverse if
$$\forall x \in S, \exists y \in S: x \bullet y = e$$

6. Distributive

- For a set S with operators \bullet and $+$, $+$ is distributive over \bullet if
$$\forall x, y, z \in S, x * (y \bullet z) = (x * y) \bullet (x * z) \in S$$

Properties

1. Closure

- S is closed with respect to operator \bullet if
$$\forall a, b \in S, a \bullet b \in S$$

2. Associative

- A binary operator \bullet on set S is associative if
$$\forall a, b, c \in S, (a \bullet b) \bullet c = a \bullet (b \bullet c)$$

3. Commutative

- A binary operator \bullet on set S is commutative if
$$\forall a, b \in S, a \bullet b = b \bullet a$$

4. Identity

- A set S has an identity element e with respect to operator \bullet if
$$\forall x \in S, \exists e \in S: x \bullet e = e \bullet x = x$$

5. Inverse

- A set S with identity element e with respect to operator \bullet has an inverse if
$$\forall x \in S, \exists y \in S: x \bullet y = e$$

6. Distributive

- For a set S with operators \bullet and $*$, $*$ is distributive over \bullet if
$$\forall x, y, z \in S, x * (y \bullet z) = (x * y) \bullet (x * z) \in S$$

Properties

1. Closure

- S is closed with respect to operator \bullet if
$$\forall a, b \in S, a \bullet b \in S$$

2. Associative

- A binary operator \bullet on set S is associative if
$$\forall a, b, c \in S, (a \bullet b) \bullet c = a \bullet (b \bullet c)$$

3. Commutative

- A binary operator \bullet on set S is commutative if
$$\forall a, b \in S, a \bullet b = b \bullet a$$

4. Identity

- A set S has an identity element e with respect to operator \bullet if
$$\forall x \in S, \exists e \in S: x \bullet e = e \bullet x = x$$

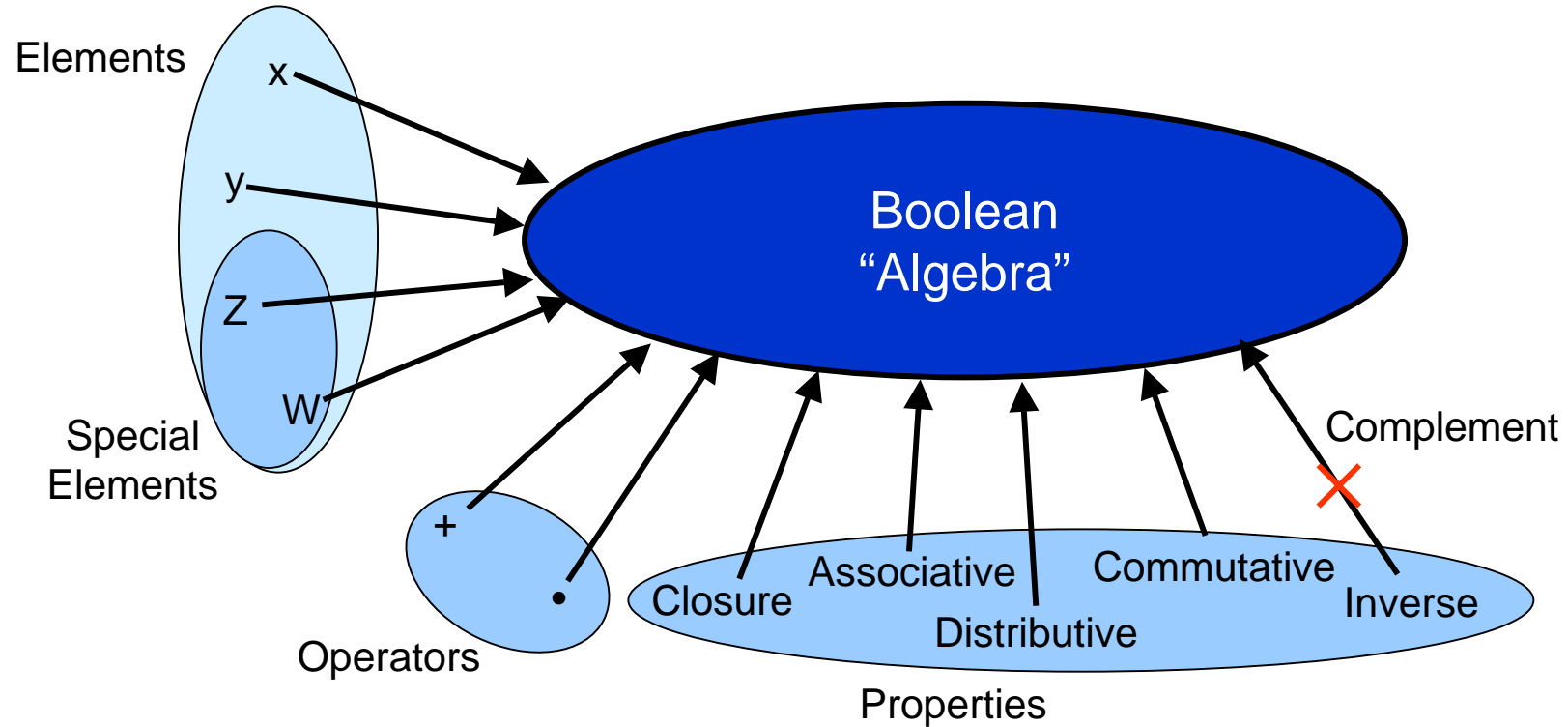
5. Inverse

- A set S with identity element e with respect to operator \bullet has an inverse if
$$\forall x \in S, \exists y \in S: x \bullet y = e$$

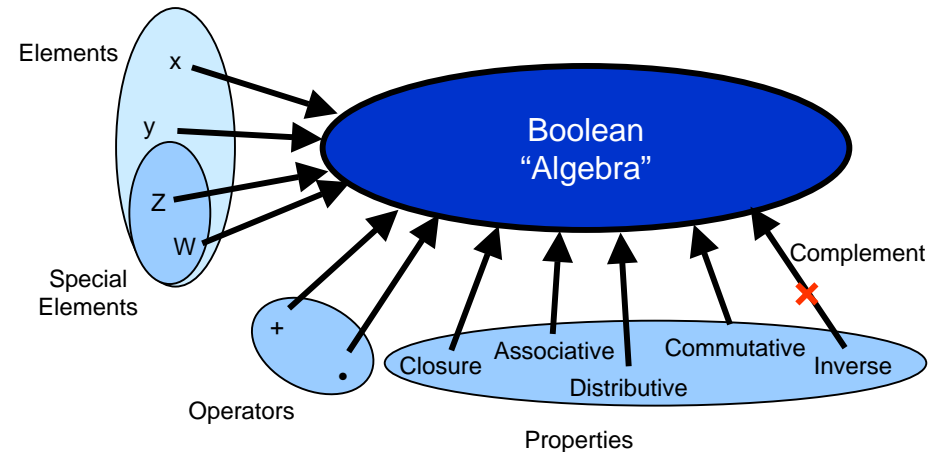
6. Distributive

- For a set S with operators \bullet and $*$, $*$ is distributive over \bullet if
$$\forall x, y, z \in S, x * (y \bullet z) = (x * y) \bullet (x * z) \in S$$

Mathematical Systems



Boolean Algebra

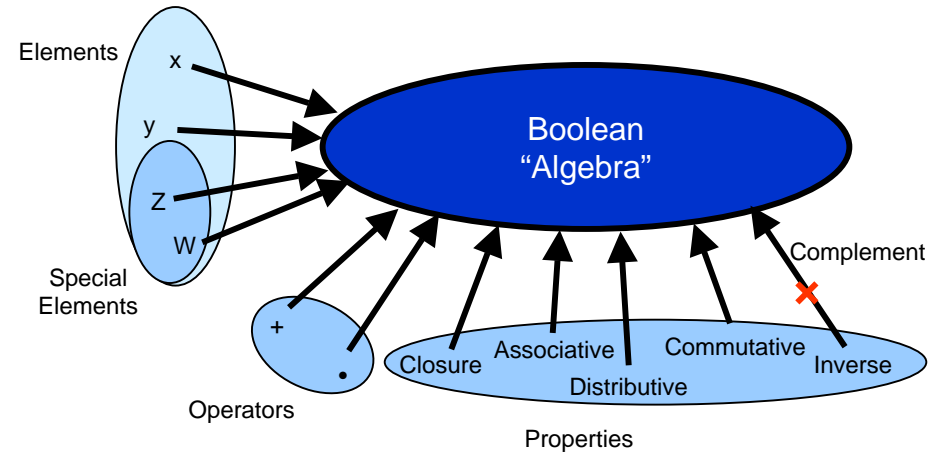


There are two operators:
AND (\bullet) and OR ($+$)

- NOTE: I am using symbols that look similar and act similarly to $+$ (plus) and \bullet (times) would act for normal arithmetic. They are **not** PLUS and TIMES!!!

Boolean Algebra

There are at least two elements (0 and 1)



There are two operators:
AND (\bullet) and OR ($+$)

Boolean Algebra

There are at least two elements (0 and 1)

		x	
	•	0	1
y	0	0	0
	1	0	1

		x	
	+	0	1
y	0	0	1
	1	1	1

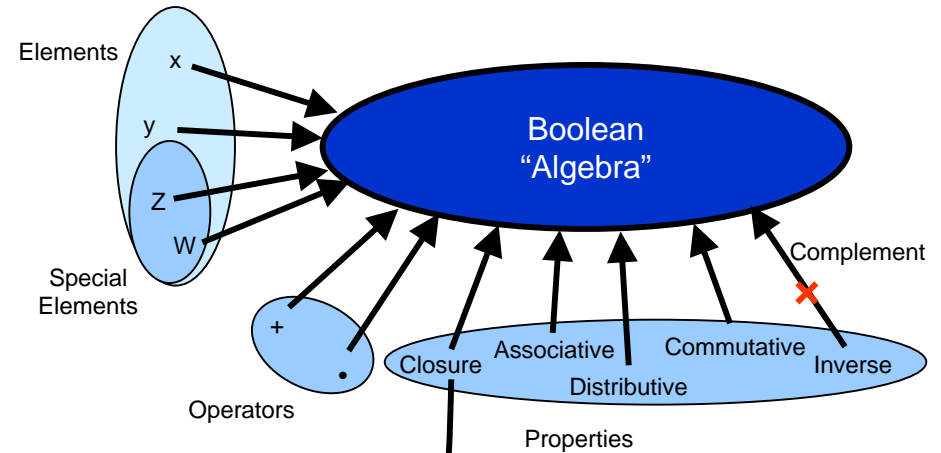
There are two operators:
AND (•) and OR (+)

Closure with respect to AND (•)

$$\forall x, y \in \{0, 1\}, x \cdot y \in \{0, 1\}$$

Closure with respect to OR (+)

$$\forall x, y \in \{0, 1\}, x + y \in \{0, 1\}$$



Boolean Algebra

There are at least two elements (0 and 1)

There are two identity elements:

Z=0 is the identity with respect to OR

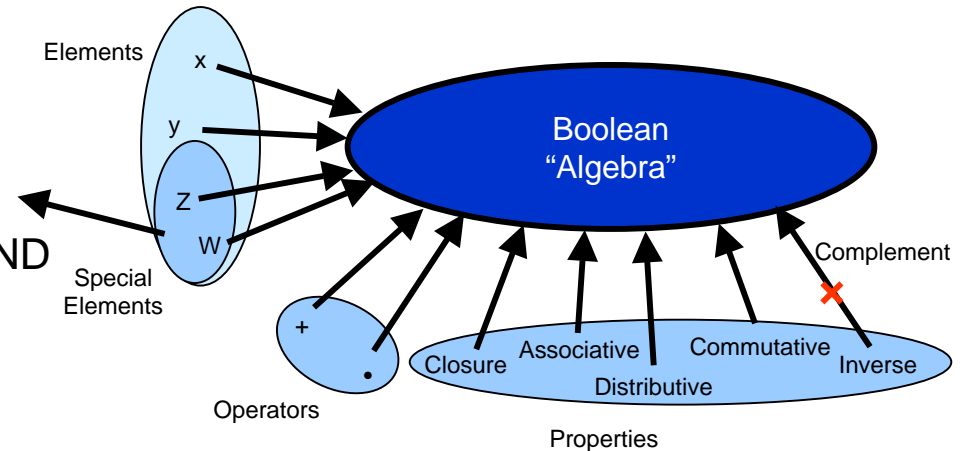
W=1 is the identity element with respect to AND

		x	
	•	0	1
y	0	0	0
	1	0	1

$$\forall x \in \{0,1\}, x \cdot 1 = x$$

		x	
	+	0	1
y	0	0	1
	1	1	1

$$\forall x \in \{0,1\}, x + 0 = x$$



There are two operators:
AND (•) and OR (+)

Boolean Algebra

There are at least two elements (0 and 1)

		x	
	•	0	1
y	0	0	0
	1	0	1

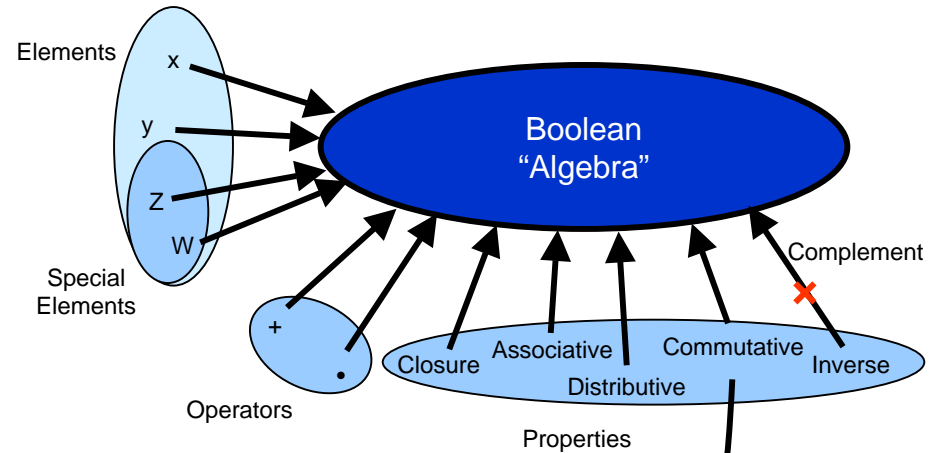
		x	
	+	0	1
y	0	0	1
	1	1	1

Commutative with respect to AND (•)

$$\forall x, y \in \{0, 1\}, x \cdot y = y \cdot x$$

Commutative with respect to OR (+)

$$\forall x, y \in \{0, 1\}, x + y = y + x$$



There are two operators:
AND (•) and OR (+)

Boolean Algebra

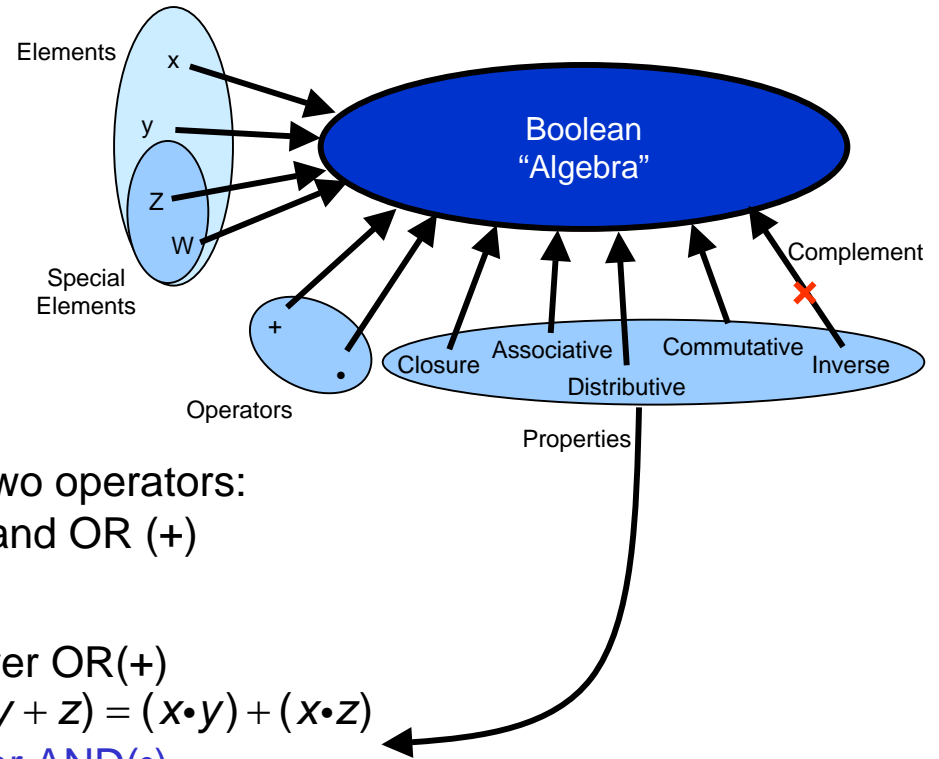
There are at least two elements (0 and 1)

		x	
	•	0	1
y	0	0	0
	1	0	1

		x	
	+	0	1
y	0	0	1
	1	1	1

There are two operators:
AND (•) and OR (+)

AND (•) is distributive over OR (+)
 $\forall x, y, z \in \{0, 1\}, x \cdot (y + z) = (x \cdot y) + (x \cdot z)$
 OR (+) is distributive over AND (•)
 $\forall x, y, z \in \{0, 1\}, x + (y \cdot z) = (x + y) \cdot (x + z)$



Boolean Algebra

There are at least two elements (0 and 1)

		x	
	•	0	1
y	0	0	0
	1	0	1

		x	
	+	0	1
y	0	0	1
	1	1	1

There are two operators:
AND (•) and OR (+)

There is a complement element
with respect to AND and OR

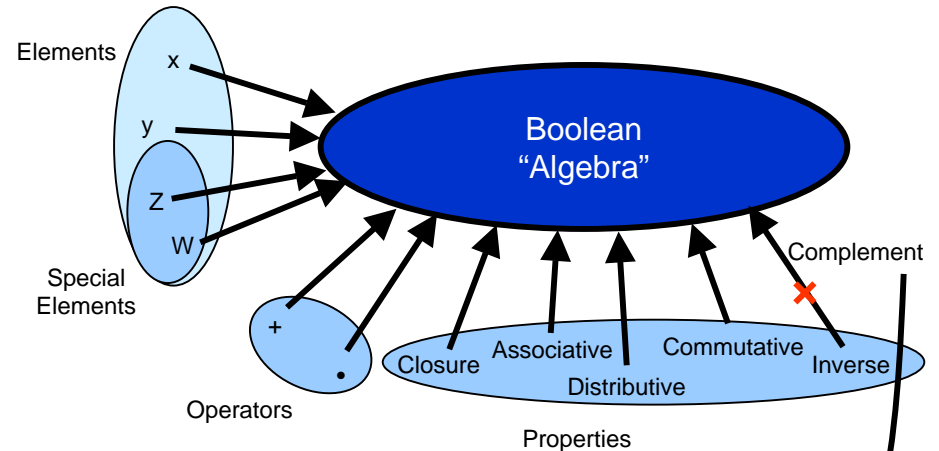
$$\forall x \in \{0,1\}, \exists x' \in \{0,1\} :$$

$$x + x' = 1$$

$$x \cdot x' = 0$$

• Alternate ways to express complement:

$$x' = \overline{x} = \neg x$$



Boolean Algebra

There are at least two elements (0 and 1)

		x	
	•	0	1
y	0	0	0
	1	0	1

		x	
	+	0	1
y	0	0	1
	1	1	1

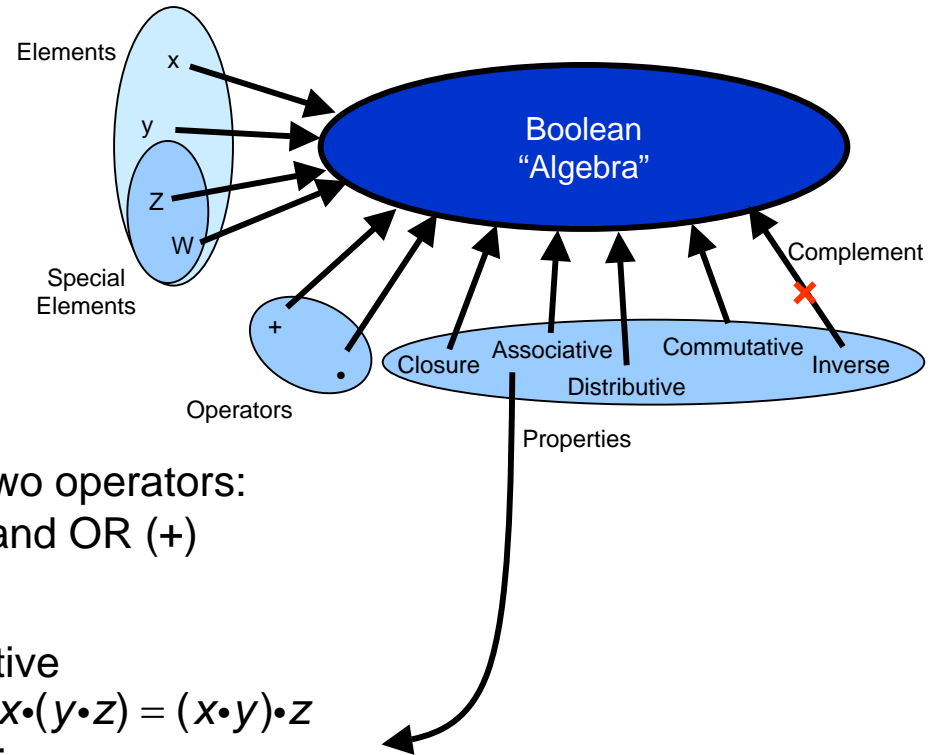
There are two operators:
AND (•) and OR (+)

AND (•) is associative

$$\forall x, y, z \in \{0, 1\}, x \cdot (y \cdot z) = (x \cdot y) \cdot z$$

OR (+) is associative

$$\forall x, y, z \in \{0, 1\}, x + (y + z) = (x + y) + z$$

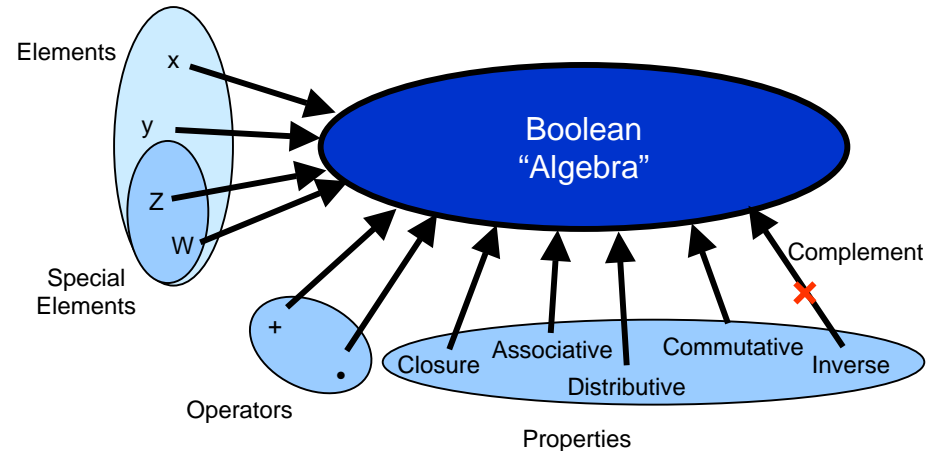


Boolean Algebra

There are at least two elements

\cdot	0	α	β	1
0	0	0	0	0
α	0	α	0	α
β	0	0	β	β
1	0	α	β	1

$+$	0	α	β	1
0	0	α	β	1
α	α	α	1	1
β	β	1	β	1
1	1	1	1	1



There are two operators:
AND (\cdot) and OR ($+$)

- It is possible to define a Boolean Algebra with more than two elements, e.g., $\{0, \alpha, \beta, 1\}$.
- All the properties defined above can be shown to be valid.

Proof by Truth Tables

- Show that the Boolean Algebra defined above is distributive:

a	b	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

b+c	a •(b+c)
0	0
1	0
1	0
1	0
0	0
1	1
1	1
1	1

a •b	a •c	(a •b)+(a •c)
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	1	1
1	0	1
1	1	1

•	0	1
0	0	0
1	0	1

+	0	1
0	0	1
1	1	1



$$\forall x, y, z \in \{0, 1\}, x \cdot (y + z) = (x \cdot y) + (x \cdot z)$$

Addition is also distributive

$$\forall x, y, z \in \{0, 1\}, x + (y \cdot z) = (x + y) \cdot (x + z)$$

Multiplication is not

Theorems and Postulates of Boolean Algebra

$$x + 0 = x$$

$$x \cdot 1 = x$$

$$x + x' = 1$$

$$x \cdot x' = 0$$

$$x + x = x$$

$$x \cdot x = x$$

$$x + 1 = 1$$

$$x \cdot 0 = 0$$

$$(x')' = x$$

$$x + y = y + x$$

$$x \cdot y = y \cdot x$$

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

$$x \cdot (y \cdot z) = (x \cdot y) \cdot z$$

$$x \cdot (y + z) = x \cdot y + x \cdot z$$

$$x + (y \cdot z) = (x + y) \cdot (x + z)$$

$$(x + y)' = x' \cdot y'$$

$$(x \cdot y)' = x' + y'$$

DeMorgan's Law

$$x + x \cdot y = x$$

$$x \cdot x + y = x$$

Absorption

Duality:
Interchange identity
Interchange operator

Operator Precedence

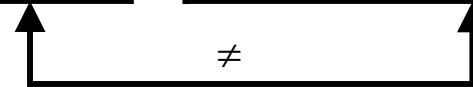
1. Parenthesis
2. NOT
3. AND
4. OR

$$x \cdot y + z = (x \cdot y) + z \neq x \cdot (y + z)$$

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

$x \cdot y$	$(x \cdot y) + z$
0	0
0	1
0	0
0	1
0	0
0	1
1	1
1	1
1	1

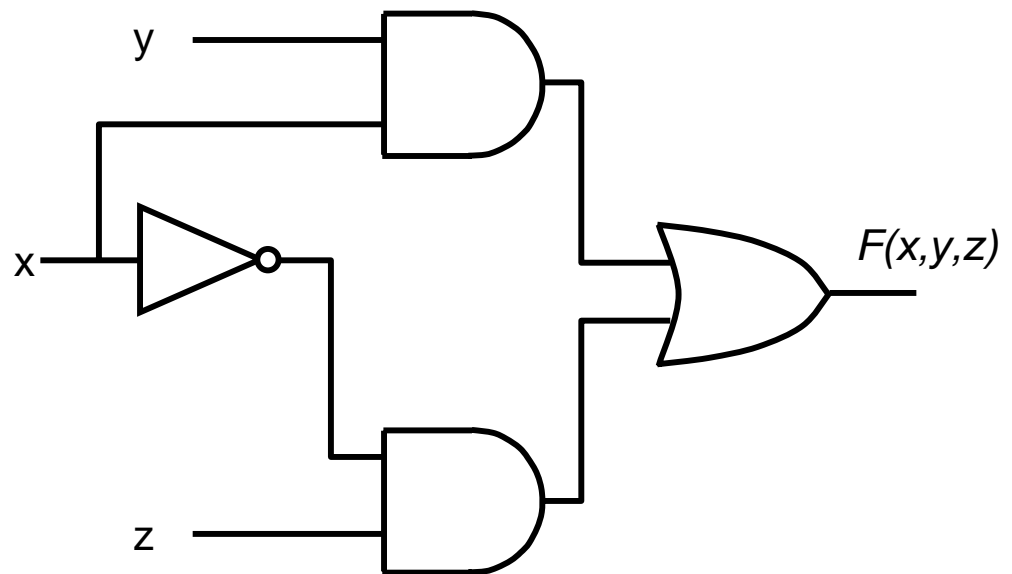
$(y+z)$	$x \cdot (y+z)$
0	0
1	0
1	0
1	0
0	0
1	1
1	1
1	1



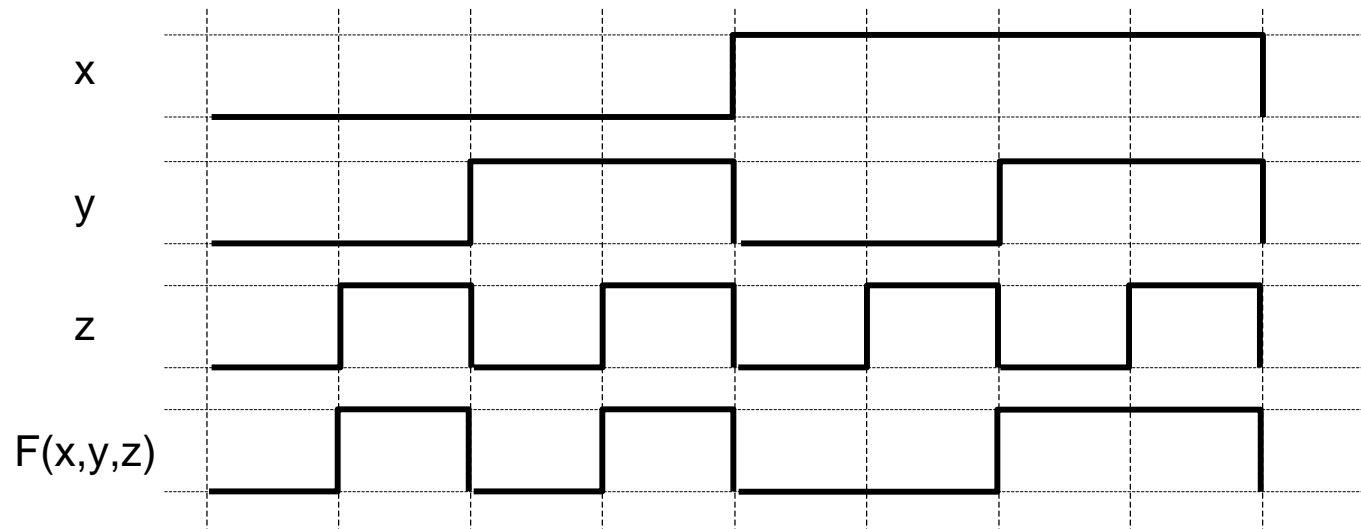
Boolean Functions

$$F(x, y, z) = x \cdot y + x' \cdot z$$

x	y	z	F(x,y,z)
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



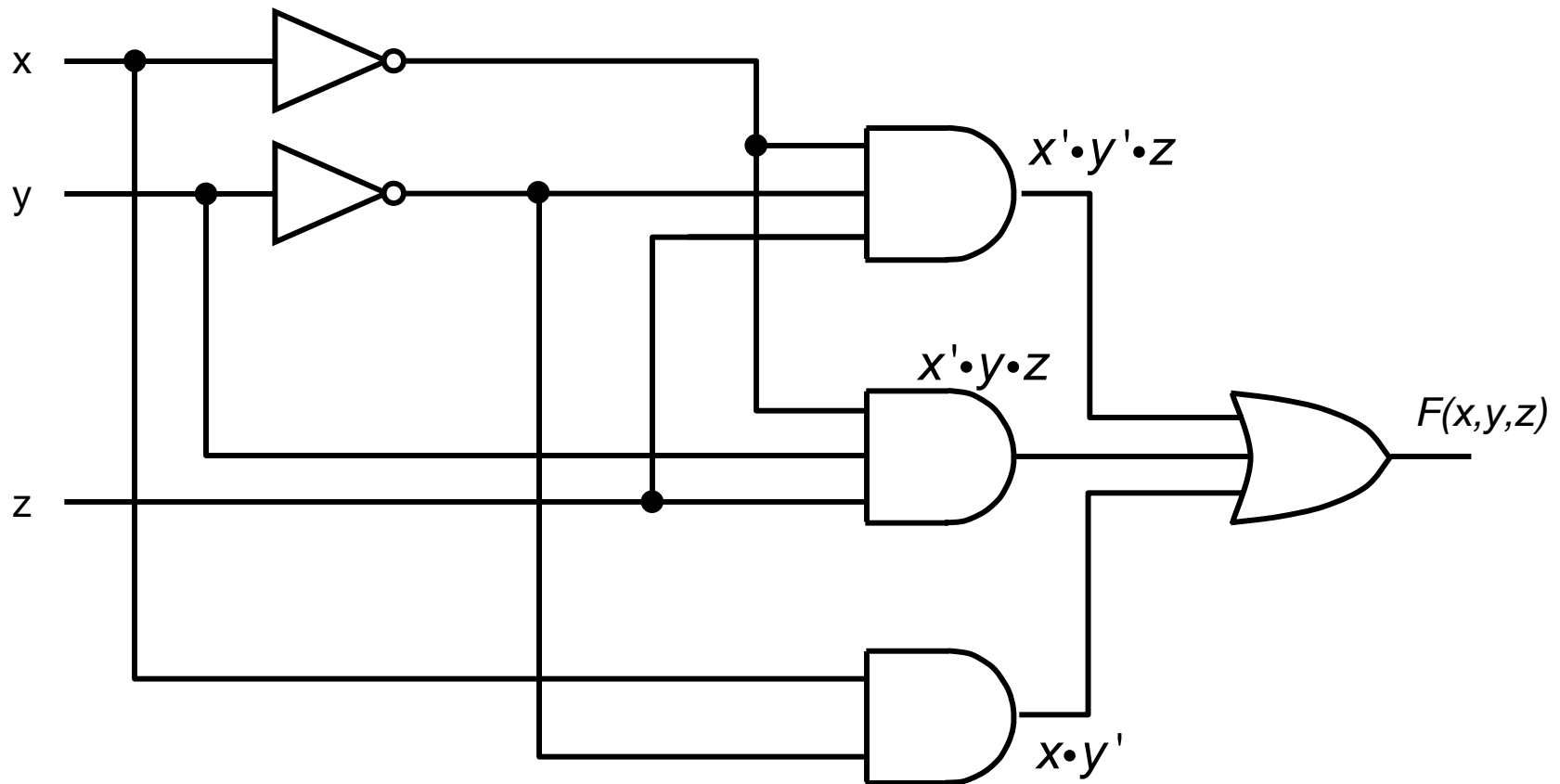
Boolean Functions Expressed as Timing Diagrams



- This is an idealized timing diagram:
 - There are no delays through gates, all events occur at instants of time.

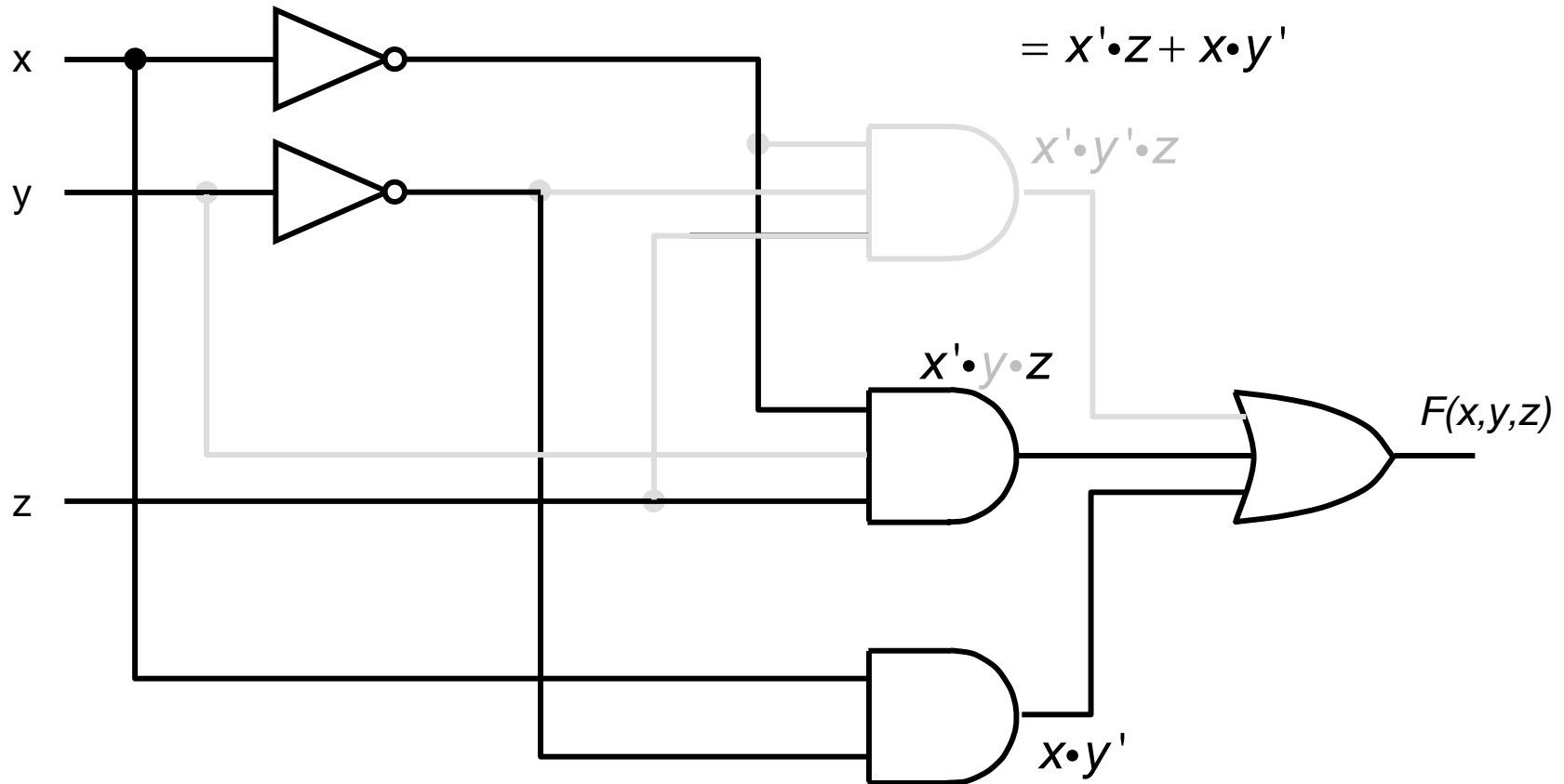
Simplifying Boolean Functions

$$F_a(x, y, z) = x' \cdot y' \cdot z + x' \cdot y \cdot z + x \cdot y'$$



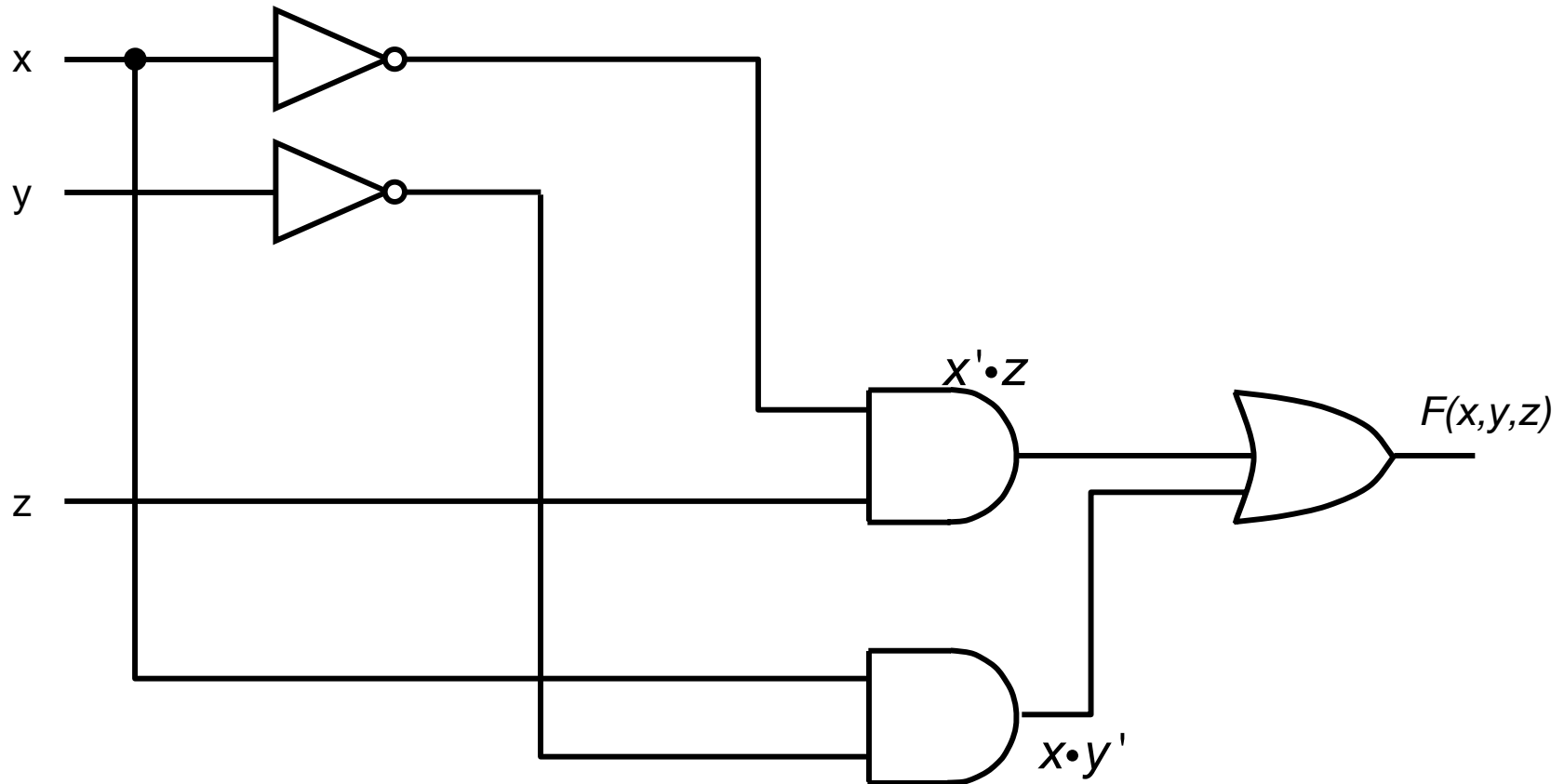
Simplifying Boolean Functions

$$\begin{aligned}F_a(x, y, z) &= x' \cdot y' \cdot z + x' \cdot y \cdot z + x \cdot y' \\ &= x' \cdot z \cdot (y' + y) + x \cdot y' \\ &= x' \cdot z + x \cdot y'\end{aligned}$$



Simplifying Boolean Functions

$$F_b(x, y, z) = x' \cdot z + x \cdot y'$$



Verifying Simplification

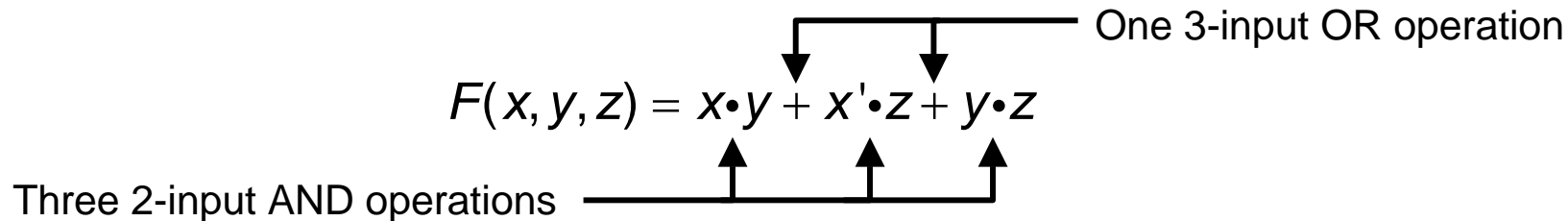
$$F_a(x, y, z) = x' \cdot y' \cdot z + x' \cdot y \cdot z + x \cdot y'$$

$$F_b(x, y, z) = x' \cdot z + x \cdot y'$$

x	y	z	$x' \cdot y' \cdot z$	$x' \cdot y \cdot z$	$x \cdot y'$	$x' \cdot z$	$F_a(x, y, z)$	$F_b(x, y, z)$
0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	1	1	1
0	1	0	0	0	0	0	0	0
0	1	1	0	1	0	1	1	1
1	0	0	0	0	1	0	1	1
1	0	1	0	0	1	0	1	1
1	1	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0

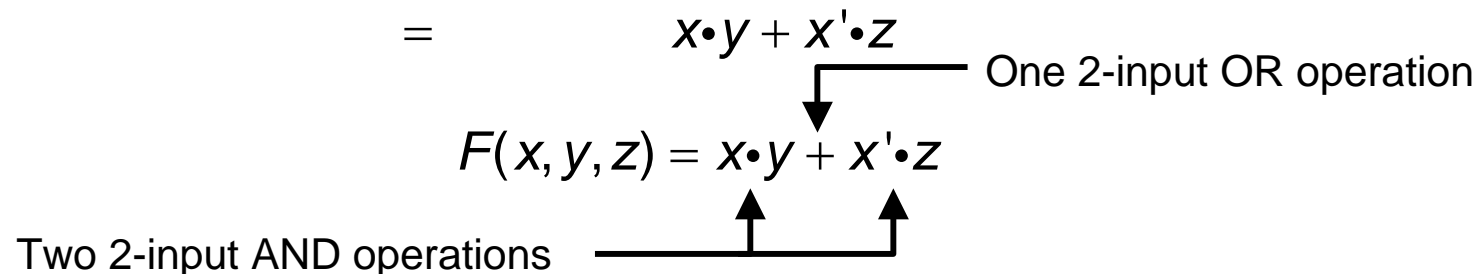
Gate Complexity of Boolean Expression

- Gate requirements can be estimated directly from expression for Boolean function



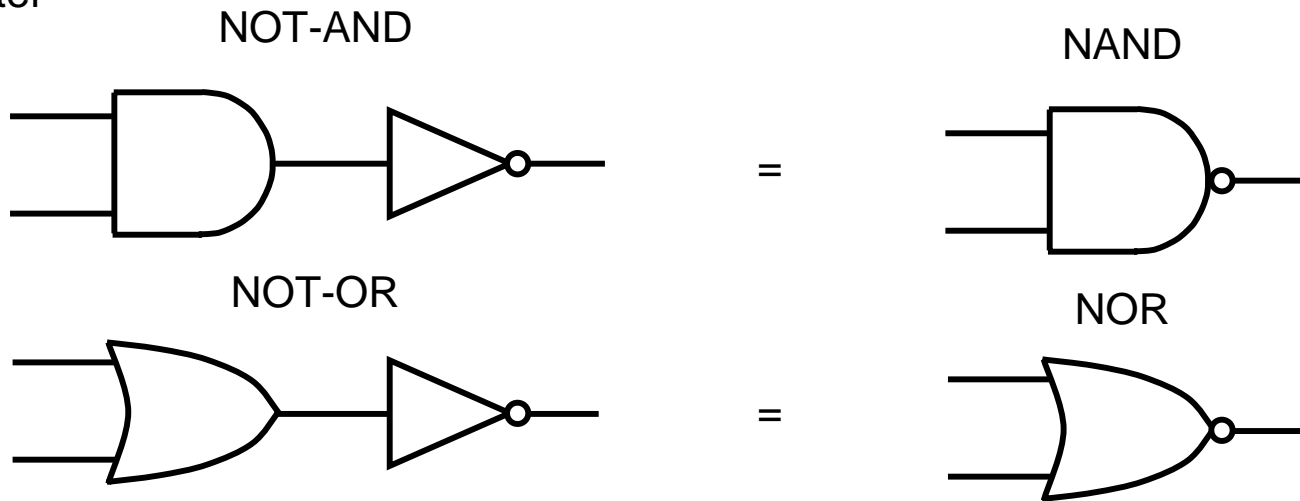
- Expression can be simplified algebraically:

$$\begin{aligned}
 x \cdot y + x' \cdot z + y \cdot z &= x \cdot y + x' \cdot z + y \cdot z(x + x') \\
 &= x \cdot y + x' \cdot z + x \cdot y \cdot z + x' \cdot y \cdot z \\
 &= x \cdot y \cdot (1 + z) + x' \cdot z \cdot (1 + y) \\
 &= x \cdot y + x' \cdot z
 \end{aligned}$$



More Logic Functions

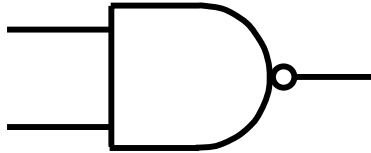
- AND and OR function are not often seen as such in real logic designs.
- NOT-AND and NOT-OR are easier to implement in hardware and are generally faster



- The bubble can be placed on any lead to indicate inversion. Two other common logic functions are:

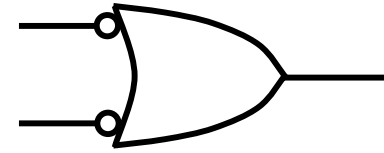


More on DeMorgan's Law

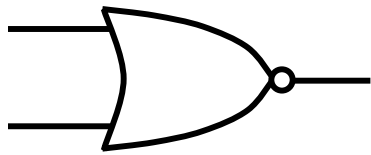


NAND

$$(x \cdot y)' = x' + y'$$

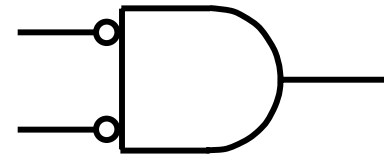


NAND



NOR

$$(x + y)' = x' \cdot y'$$



NOR

Functions of Two Inputs

- There are 16 possible functions of two inputs

x	y	F ₀	F ₁	F ₂	F ₃	F ₄	F ₅	F ₆	F ₇	F ₈	F ₉	F ₁₀	F ₁₁	F ₁₂	F ₁₃	F ₁₄	F ₁₅
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
Name of F _n	Z e r o	A N D	i n h i b	x	i n h i b	y	X O R	O R	N O R	=	y'	y -> x	x'	x -> y	N A N D	O n e	

Generic Boolean Functions

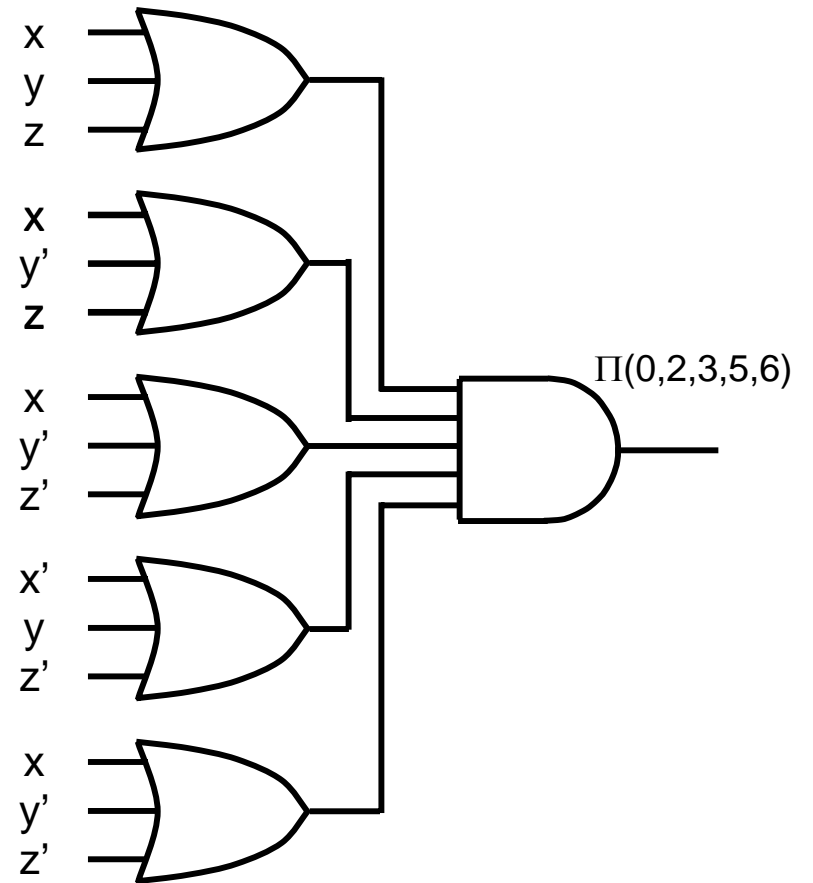
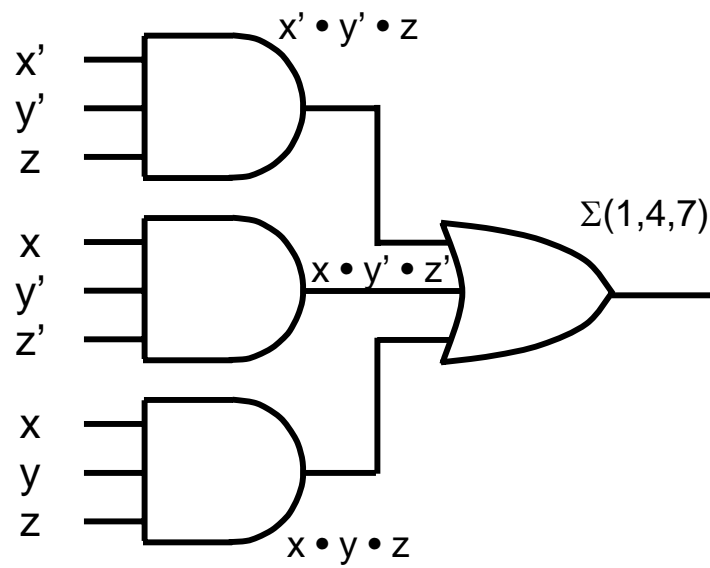
- A function can be expressed in terms of “min-terms” or “max-terms”

x	y	z	Minterm	Name	Maxterm	Name
0	0	0	$x' \cdot y' \cdot z'$	m_0	$x'+y'+z'$	M_0
0	0	1	$x' \cdot y' \cdot z$	m_1	$x'+y'+z$	M_1
0	1	0	$x' \cdot y \cdot z'$	m_2	$x'+y+z'$	M_2
0	1	1	$x' \cdot y \cdot z$	m_3	$x'+y+z$	M_3
1	0	0	$x \cdot y' \cdot z'$	m_4	$x+y'+z'$	M_4
1	0	1	$x \cdot y' \cdot z$	m_5	$x+y'+z$	M_5
1	1	0	$x \cdot y \cdot z'$	m_6	$x+y+z'$	M_6
1	1	1	$x \cdot y \cdot z$	m_7	$x+y+z$	M_7

- An alternative description of a function is to only specify the minterms

$$F(x, y, z) = x' \cdot y' \cdot z + x \cdot y' \cdot z' + x \cdot y \cdot z = m_1 + m_4 + m_7 = \sum (1, 4, 7)$$

Sum of Products vs. Product of Sums



Conversion between $\Sigma(\Pi)$ and $\Pi(\Sigma)$

$$m = \{m_0, m_1, m_2, \dots, m_7\}$$

$$F(x, y, z) = \Sigma(a, b, c, d)$$

$$F(x, y, z) = m_a + m_b + m_c + m_d$$

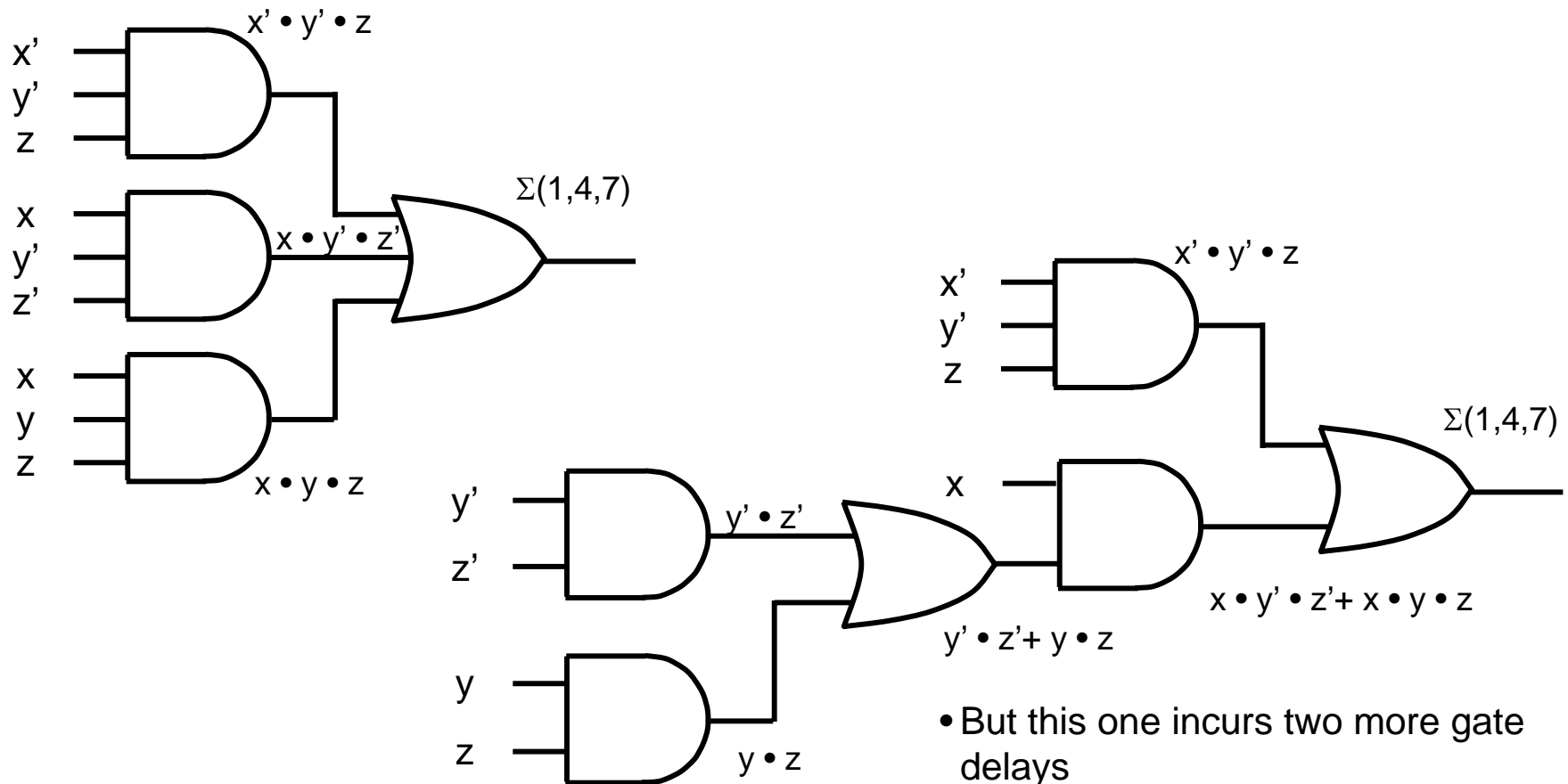
$$F'(x, y, z) = \Sigma(m - \{m_a, m_b, m_c, m_d\}) = \Sigma(m_e, m_f, m_g, m_h) = m_e + m_f + m_g + m_h$$

$$F(x, y, z) = (m_e + m_f + m_g + m_h)' = m_e' \cdot m_f' \cdot m_g' \cdot m_h' = M_e \cdot M_f \cdot M_g \cdot M_h$$

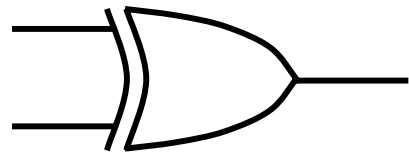
$$F(x, y, z) = \Pi(e, f, g, h)$$

Design Considerations

- These two designs are logically equivalent:



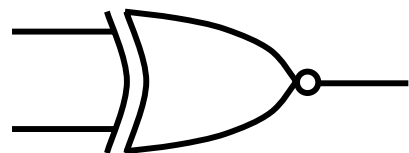
Other Basic Logic Gates



Exclusive OR
(XOR)

$$x \oplus y = x' \cdot y + x \cdot y'$$

		x	
\oplus		0	1
	0	0	1
	1	1	0



Exclusive NOR
(equivalence)

$$(x \triangleq y) = x' \cdot y' + x \cdot y$$

		x	
=		0	1
	0	1	0
	1	0	1

Digital Logic Families

- RTL – Resistor Transistor Logic
 - Original logic family used in ICs
 - VERY slow and power hungry by recent standards
 - Mostly replaced by TTL in 1960s-70s
- TTL – Transistor-Transistor Logic
 - 74xx series - workhorse of logic designs
 - Mostly replaced by CMOS with 74xCxx equivalents
- ECL – Emitter Coupled Logic
 - Popular in 1970s
 - Very high speed
 - High power consumption
- MOS – Metal Oxide Silicon
 - High density, introduced for memory applications
- CMOS – Complementary Metal Oxide Silicon
 - Extremely low quiescent power consumption
 - Wide range of speed/power tradeoffs
 - Industry standard today

Summary

- Fundamental concepts of digital systems (Mano Chapter 1)
- Binary codes, number systems, and arithmetic (Ch 1)
- **Boolean algebra (Ch 2)**
- Simplification of switching equations (Ch 3)
- Digital device characteristics (e.g., TTL, CMOS)/design considerations (Ch 10)
- Combinatoric logical design including LSI implementation (Chapter 4)
- Hazards, Races, and time related issues in digital design (Ch 9)
- Flip-flops and state memory elements (Ch 5)
- Sequential logic analysis and design (Ch 5)
- Synchronous vs. asynchronous design (Ch 9)
- Counters, shift register circuits (Ch 6)
- Memory and Programmable logic (Ch 7)
- Minimization of sequential systems
- Introduction to Finite Automata

Homework 2 – due in Class 4

- Problems 2-1, 2-5, 2-15, 2-19. Show all work