Cairo University

Faculty of Graduate Studies for Statistical Research

Computer Science Department

Cairo University

CS504

Digital Logic & Computer Organization

Lecture 3

Lecture Outline (Chapter 2)

- ★ **Canonical Forms (Section 2.6)**
  - **Converting To Sum-Of-Minterms Form**
  - **Converting To Product-Of-Maxterms Form**
  - **Conversion Between Canonical Forms**
  - **Algebraic Conversion Between Canonical Forms**
  - **Function Complements**
  - **Standard Forms**
  - **AND/OR Two-Level Implementation**
- ★ **Other Logic Operations (Section 2.7)**
- ★ **Digital Logic Gates (Section 2.8)**
  - **Extension To Multiple Inputs**
  - **Positive And Negative Logic**

Converting To Sum-Of-Minterms Form

- A function that is not in the Sum-of-Minterms form can be converted to that form by means of a truth table
- Consider $F = \overline{y} + \overline{x}\,\overline{z}$

x	y	z	F	Minterm
0	0	0	**1**	$m_0 = \overline{x}\,\overline{y}\,\overline{z}$
0	0	1	**1**	$m_1 = \overline{x}\,\overline{y}\,z$
0	1	0	**1**	$m_2 = \overline{x}\,y\,\overline{z}$
0	1	1	0	
1	0	0	**1**	$m_4 = x\,\overline{y}\,\overline{z}$
1	0	1	**1**	$m_5 = x\,\overline{y}\,z$
1	1	0	0	
1	1	1	0	

$$F = \sum(0, 1, 2, 4, 5) =$$

$$m_0 + m_1 + m_2 + m_4 + m_5 =$$

$$\overline{x}\,\overline{y}\,\overline{z} + \overline{x}\,\overline{y}\,z + \overline{x}\,y\,\overline{z} +$$

$$x\,\overline{y}\,\overline{z} + x\,\overline{y}\,z$$

Converting To Product-Of-Maxterms Form

- A function that is not in the Product-of-Maxterms form can be converted to that form by means of a truth table
- Consider again: $F = \overline{y} + \overline{x}\,\overline{z}$

x	y	z	F	Maxterm
0	0	0	1	
0	0	1	1	
0	1	0	1	
0	1	1	**0**	$M_3 = (x+\overline{y}+\overline{z})$
1	0	0	1	
1	0	1	1	
1	1	0	**0**	$M_6 = (\overline{x}+\overline{y}+z)$
1	1	1	**0**	$M_7 = (\overline{x}+\overline{y}+\overline{z})$

$$F = \prod(3, 6, 7) =$$

$$M_3 \cdot M_6 \cdot M_7 =$$

$$(x+\overline{y}+\overline{z})\ (\overline{x}+\overline{y}+z)\ (\overline{x}+\overline{y}+\overline{z})$$

Conversions Between Canonical Forms

x	y	z	F	Minterm	Maxterm
0	0	0	**0**		$M_0 = (x + y + z)$
0	0	1	**1**	$m_1 = \overline{x}\,\overline{y}\,z$	
0	1	0	**1**	$m_2 = \overline{x}\,y\,\overline{z}$	
0	1	1	**1**	$m_3 = \overline{x}\,y\,z$	
1	0	0	**0**		$M_4 = (\overline{x} + y + z)$
1	0	1	**1**	$m_5 = x\,\overline{y}\,z$	
1	1	0	**0**		$M_6 = (\overline{x} + \overline{y} + z)$
1	1	1	**1**	$m_7 = x\,y\,z$	

$$F = m_1+m_2+m_3+m_5+m_7 = \sum(1, 2, 3, 5, 7) =$$

$$\overline{x}\,\overline{y}\,z + \overline{x}\,y\,\overline{z} + \overline{x}\,y\,z + x\,\overline{y}\,z + x\,y\,z$$

$$F = M_0 \cdot M_4 \cdot M_6 = \prod(0, 4, 6) = (x+y+z)(\overline{x}+y+z)(\overline{x}+\overline{y}+z)$$

Algebraic Conversion To Sum-Of-Minterms

- Expand all terms first to explicitly list all minterms
- AND any term missing a variable v with $(v + \overline{v})$
- Example 1: $f = x + \overline{x}\,\overline{y}$ (2 variables)
  $f = x\,(y + \overline{y}) + \overline{x}\,\overline{y}$
  $f = x\,y + x\,\overline{y} + \overline{x}\,\overline{y}$
  $\mathrm{f} = \mathrm{m}_3 + \mathrm{m}_2 + \mathrm{m}_0 = \sum(0, 2, 3)$
- Example 2: $g = a + \overline{b}\,c$ (3 variables)
  $g = a\,(b + \overline{b})(c + \overline{c}) + (a + \overline{a})\,\overline{b}\,c$
  $g = a\,b\,c + a\,b\,\overline{c} + a\,\overline{b}\,c + a\,\overline{b}\,\overline{c} + a\,\overline{b}\,c + \overline{a}\,\overline{b}\,c$
  $g = \overline{a}\,\overline{b}\,c + a\,\overline{b}\,\overline{c} + a\,\overline{b}\,c + a\,b\,\overline{c} + a\,b\,c$
  $g = \mathrm{m}_1 + \mathrm{m}_4 + \mathrm{m}_5 + \mathrm{m}_6 + \mathrm{m}_7 = \sum(1, 4, 5, 6, 7)$

Algebraic Conversion To Product-Of-Maxterms

- Expand all terms first to explicitly list all Maxterms
- OR any term missing a variable v with $v \cdot \overline{v}$
- Example 1: $f = x + \overline{x}\,\overline{y}$ (2 variables)

  Apply 2nd distributive law:

  $f = (x + \overline{x})\,(x + \overline{y}) = 1 \cdot (x + \overline{y}) = (x + \overline{y}) = M_1$
- Example 2: $g = a\,\overline{c} + b\,c + \overline{a}\,\overline{b}$ (3 variables)

  $g = (a\,\overline{c} + b\,c + \overline{a})\,(a\,\overline{c} + b\,c + \overline{b})$ (distributive)

  $g = (\overline{c} + b\,c + \overline{a})\,(a\,\overline{c} + c + \overline{b})$ $(x + \overline{x}\,y = x + y)$

  $g = (\overline{c} + b + \overline{a})\,(a + c + \overline{b})$ $(x + \overline{x}\,y = x + y)$

  $g = (\overline{a} + b + \overline{c})\,(a + \overline{b} + c) = M_5 \,.\, M_2 = \prod(2, 5)$

Function Complements

- The complement of a function expressed as a sum of minterms is constructed by selecting the minterms missing in the sum-of-minterms canonical form
- Alternatively, the complement of a function expressed by a Sum of Minterms form is simply the Product of Maxterms with the same indices
- Example: Given $F(x, y, z) = \sum (1, 3, 5, 7)$

$$\overline{F}(x, y, z) = \sum (0, 2, 4, 6)$$

$$\overline{F}(x, y, z) = \prod (1, 3, 5, 7)$$

Summary of Minterms And Maxterms

- There are 2^n minterms and maxterms for Boolean functions with n variables.
- Minterms and maxterms are indexed from 0 to $2^n - 1$
- Any Boolean function can be expressed as a logical sum of minterms and as a logical product of maxterms
- The complement of a function contains those minterms not included in the original function
- The complement of a sum-of-minterms is a product-of-maxterms with the same indices

Standard Forms

- **<u>Standard Sum-of-Products (SOP) form:</u>** equations are written as an OR of AND terms
- **<u>Standard Product-of-Sums (POS) form:</u>** equations are written as an AND of OR terms
- **Examples:**
  - SOP: $A\,B\,C + \overline{A}\,\overline{B}\,C + B$
  - POS: $(A + B) \cdot (A + \overline{B} + \overline{C}) \cdot C$
- These "mixed" forms are <u>neither SOP nor POS</u>
  - $(A\,B + C)\,(A + C)$
  - $A\,B\,\overline{C} + A\,C\,(A + B)$

Standard Sum-Of-Products (SOP)

- **A sum of minterms form for *n* variables can be written down directly from a truth table.**
  - **Implementation of this form is a two-level network of gates such that:**
  - **The first level consists of *n*-input AND gates**
  - **The second level is a single OR gate**
- **This form often can be simplified so that the corresponding circuit is simpler.**

Standard Sum-Of-Products (SOP)

- **A Simplification Example:**

$$F(A,B,C) = \sum(1,4,5,6,7)$$

- **Writing the minterm expression:**

$$F = \overline{A}\,\overline{B}\,C + A\,\overline{B}\,\overline{C} + A\,\overline{B}\,C + AB\overline{C} + ABC$$

- **Simplifying:**

$$F = \overline{A}\,\overline{B}\,C + A\,(\overline{B}\,\overline{C} + \overline{B}\,C + B\,\overline{C} + B\,C)$$

$$F = \overline{A}\,\overline{B}\,C + A\,(\overline{B}\,(\overline{C} + C) + B\,(\overline{C} + C))$$

$$F = \overline{A}\,\overline{B}\,C + A\,(\overline{B} + B)$$

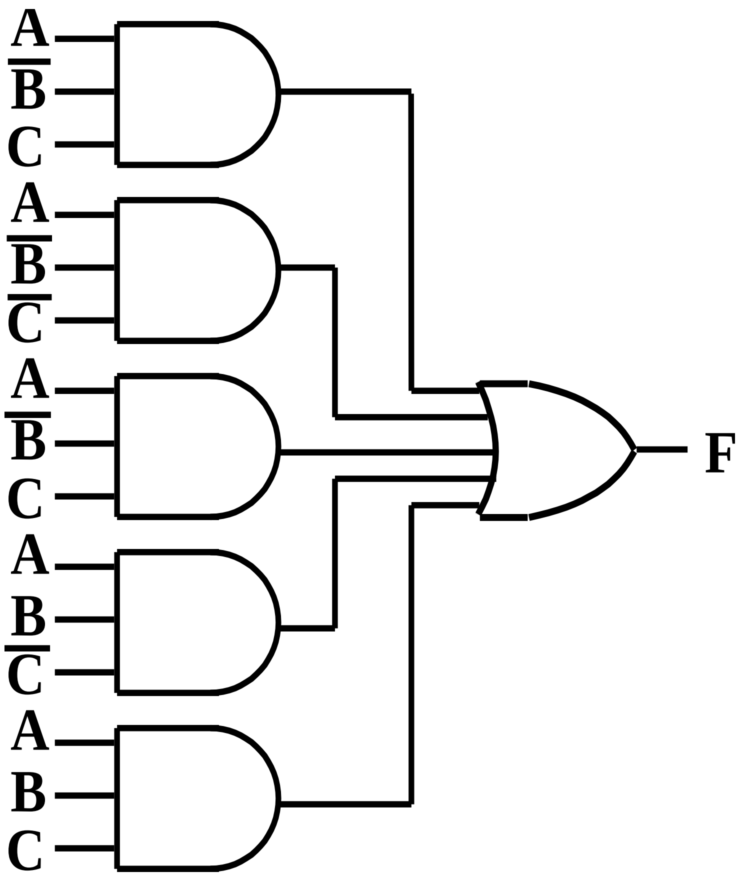
$$F = \overline{A}\,\overline{B}\,C + A$$

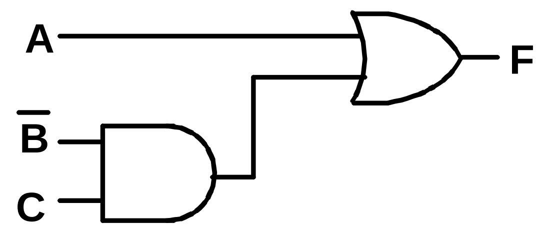
$$F = \overline{B}\,C + A$$

- **Simplified F contains 3 literals compared to 15**

AND/OR Two-Level Implementation

- The two implementations for F are shown below





It is quite apparent which is simpler!

SOP and POS Observations

- **The previous examples show that:**
  - Canonical Forms (Sum-of-minterms, Product-of-Maxterms), or other standard forms (SOP, POS) differ in complexity
  - Boolean algebra can be used to manipulate equations into simpler forms
  - Simpler equations lead to simpler implementations

Other Logic Operations

- 2^n rows in the truth table of n binary variables
- 2^{2^n} functions for n binary variables
- 16 functions of two binary variables

Table 2.7

Truth Tables for the 16 Functions of Two Binary Variables

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

Other Logic Operations (2)

Table 2.8
Boolean Expressions for the 16 Functions of Two Variables

Boolean Functions	Operator Symbol	Name	Comments
$F_0 = 0$		Null	Binary constant 0
$F_1 = xy$	$x \cdot y$	AND	x and y
$F_2 = xy'$	x/y	Inhibition	x, but not y
$F_3 = x$		Transfer	x
$F_4 = x'y$	y/x	Inhibition	y, but not x
$F_5 = y$		Transfer	y
$F_6 = xy' + x'y$	$x \oplus y$	Exclusive-OR	x or y, but not both
$F_7 = x + y$	$x + y$	OR	x or y
$F_8 = (x + y)'$	$x \downarrow y$	NOR	Not-OR
$F_9 = xy + x'y'$	$(x \oplus y)'$	Equivalence	x equals y
$F_{10} = y'$	y'	Complement	Not y
$F_{11} = x + y'$	$x \subset y$	Implication	If y, then x
$F_{12} = x'$	x'	Complement	Not x
$F_{13} = x' + y$	$x \supset y$	Implication	If x, then y
$F_{14} = (xy)'$	$x \uparrow y$	NAND	Not-AND
$F_{15} = 1$		Identity	Binary constant 1

Other Logic Operations (3)

- Consider the 16 functions
  - two are equal to a constant
  - four are repeated twice
  - inhibition and implication are not commutative or associative
  - the other eight: complement, transfer, AND, OR, NAND, NOR, XOR, and equivalence are used as standard gates
  - complement: inverter
  - transfer: buffer (used for power amplification)
  - equivalence: XNOR

Digital Logic Gates

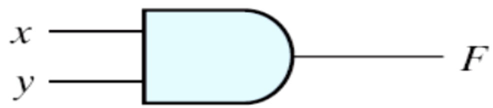
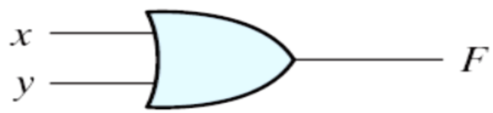
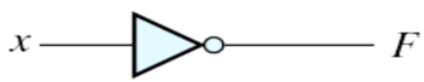
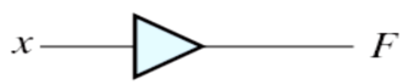
Name	Graphic symbol	Algebraic function	Truth table
AND		$F = xy$	x y | F 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1
OR		$F = x + y$	x y | F 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1
Inverter		$F = x'$	x | F 0 | 1 1 | 0
Buffer		$F = x$	x | F 0 | 0 1 | 1

Figure 2.5 Digital Logic Gates

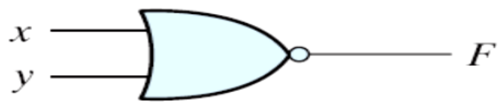
Digital Logic Gates (2)

NAND

$F = (xy)'$

x	y	F
0	0	1
0	1	1
1	0	1
1	1	0

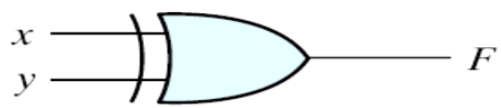
NOR



$F = (x + y)'$

x	y	F
0	0	1
0	1	0
1	0	0
1	1	0

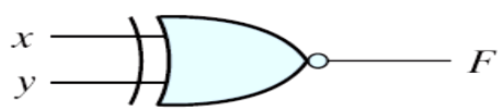
Exclusive-OR (XOR)



$$F = xy' + x'y = x \oplus y$$

x	y	F
0	0	0
0	1	1
1	0	1
1	1	0

Exclusive-NOR or equivalence



$$F = xy + x'y' = (x \oplus y)'$$

x	y	F
0	0	1
0	1	0
1	0	0
1	1	1

Figure 2.5 Digital Logic Gates (Continued)

Extension To Multiple Inputs

- All gates -except for the inverter and buffer- can be extended to have more than two inputs
- A gate can be extended to multiple inputs if the binary operation it represents is commutative & associative
- The AND and OR operations, defined in Boolean algebra, possess these two properties.
- $x+y = y+x$ (commutative)
- $(x+y)+z = x+(y+z) = x+y+z$ (associative)

Extension To Multiple Inputs (2)

- NAND and NOR are commutative but not associative => they are not extendable

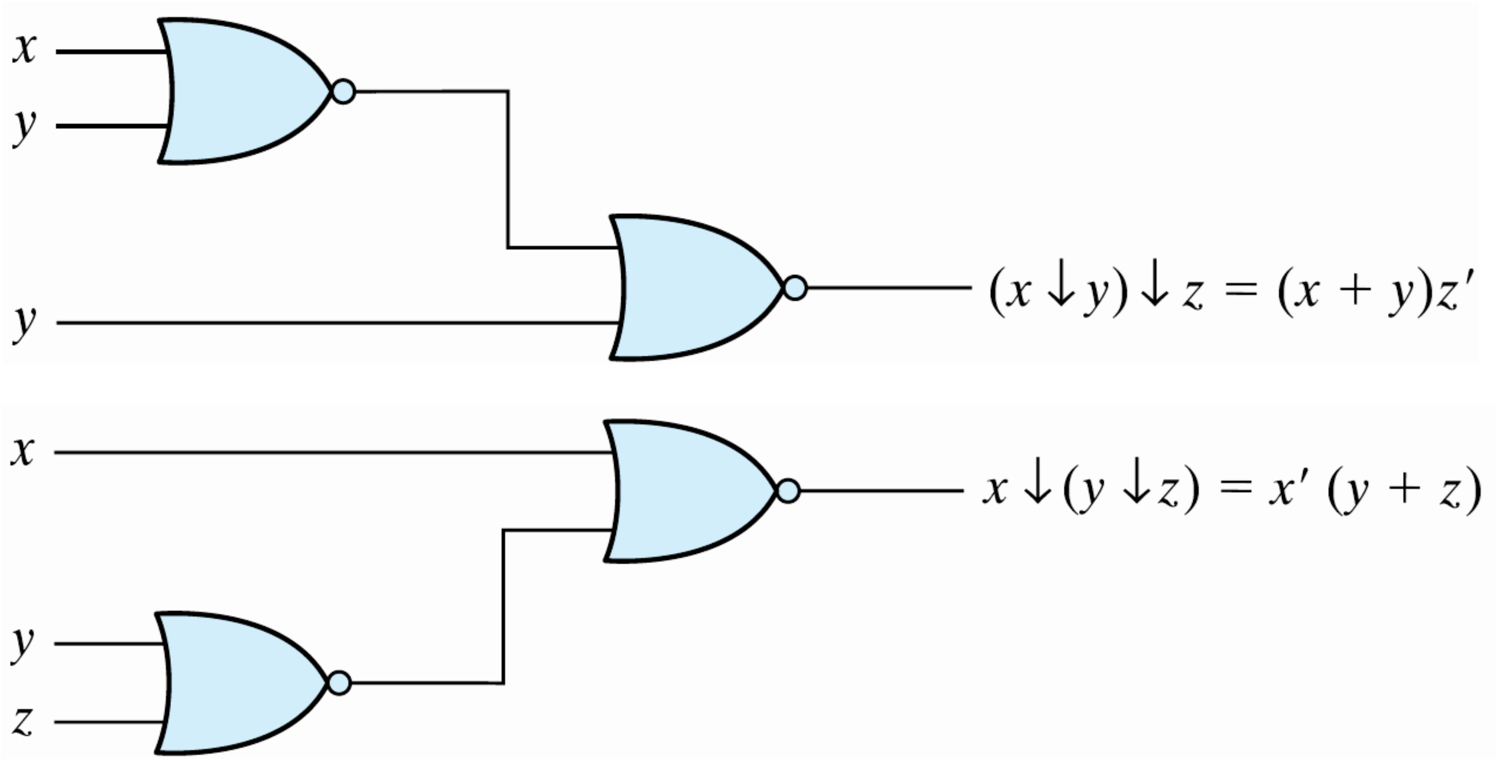


Fig. 2-6 Demonstrating the nonassociativity of the NOR operator; $(x \downarrow y) \downarrow z \neq x \downarrow (y \downarrow z)$

Extension To Multiple Inputs (3)

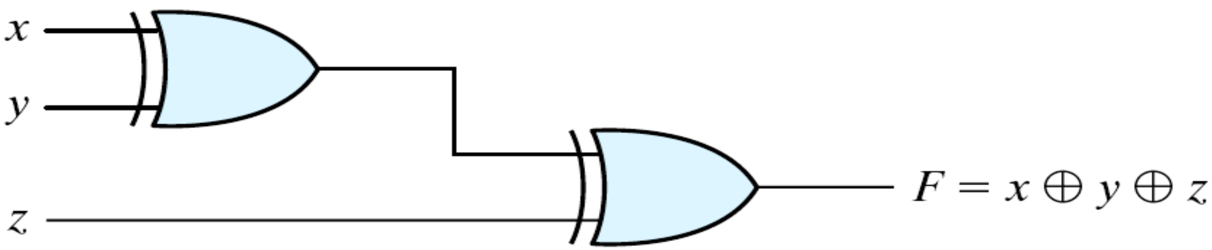
- Multiple NOR = a complement of OR gate
- Multiple NAND = a complement of AND gate

(a) 3-input NOR gate (b) 3-input NAND gate

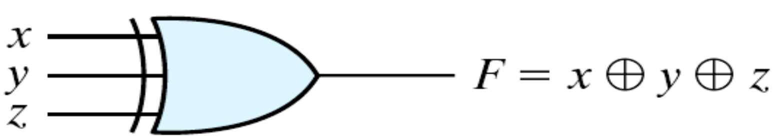
Fig. 2-7 Multiple-input and cascated NOR and NAND gates

Extension To Multiple Inputs (4)

- The XOR and XNOR gates are commutative and associative
- XOR is an odd function: it is equal to 1 if the inputs variables have an odd number of 1's



(a) Using 2-input gates



(b) 3-input gate

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

(c) Truth table

Fig. 2-8 3-input exclusive-OR gate

Positive And Negative Logic

- The same physical gate has different logical meanings depending on interpretation of the signal levels.
- *Positive Logic*
  - HIGH signal levels represent Logic 1
  - LOW signal levels represent Logic 0
- *Negative Logic*
  - LOW signal levels represent Logic 1
  - HIGH signal levels represent Logic 0
- A gate that implements a Positive Logic AND function will implement a Negative Logic OR function, and vice-versa.

Positive And Negative Logic (2)

- Given this signal level table:

Input X Y	Output
L L	L
L H	H
H L	H
H H	H

- What logic function is implemented?

Positive Logic	(H = 1) (L = 0)
0 0	0
0 1	1
1 0	1
1 1	1

Negative Logic	(H = 0) (L = 1)
1 1	1
1 0	0
0 1	0
0 0	0

Positive And Negative Logic (3)

- Rearranging the negative logic terms to the standard function table order:

Positive Logic	(H = 1) (L = 0)
0 0	0
0 1	1
1 0	1
1 1	1

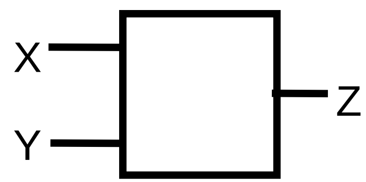
OR

Negative Logic	(H = 0) (L = 1)
0 0	0
0 1	0
1 0	0
1 1	1

AND

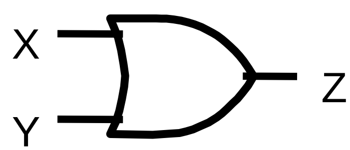
Logic Symbol Conventions

- Use of *polarity indicator* to represent use of negative logic convention on gate inputs or outputs

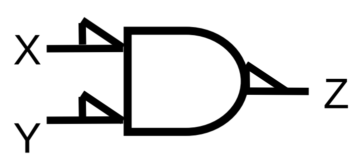


Logic Circuit

X	Y	Z
L	L	L
L	H	H
H	L	H
H	H	H



Positive Logic



Negative Logic

The End

Questions?