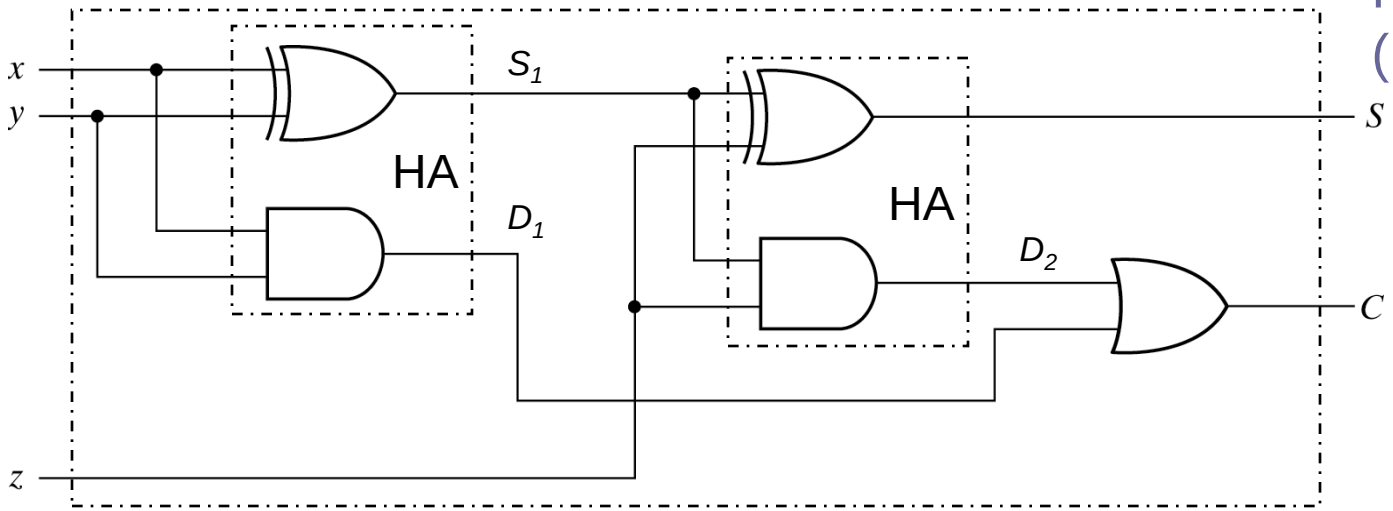
YORK
UNIVERSITÉ
UNIVERSITY

EECS 3201: Digital Logic Design Lecture 5

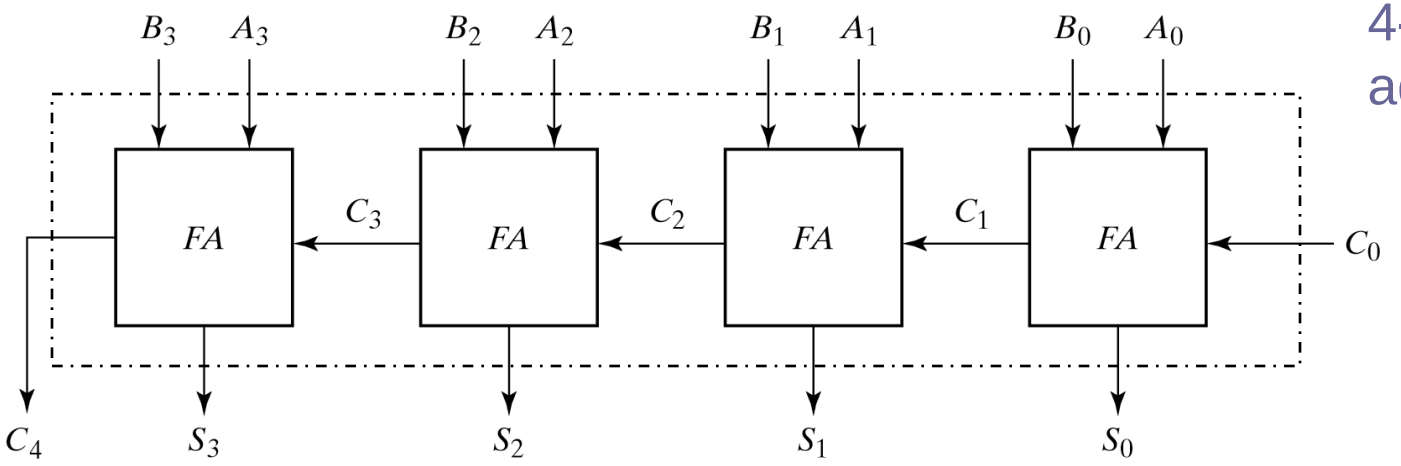
Ihab Amer, PhD, SMIEEE, P.Eng.

Hierarchical Design

Full Adder (FA)



4-bit binary adder



4-bit Adder – GL (Hierarchical)

```
module halfadder (S,C,x,y);
   input x,y;
   output S,C;
//Instantiate primitive gates
   xor (S,x,y);
   and (C,x,y);
endmodule
```

```
module fulladder (S,C,x,y,z);
   input x,y,z;
   output S,C;
   wire S1,D1,D2; //Outputs of
//first XOR and two AND gates
//Instantiate the halfadder
    halfadder HA1 (S1,D1,x,y),
                  HA2 (S,D2,S1,z);
    or g1(C,D2,D1);
endmodule
```

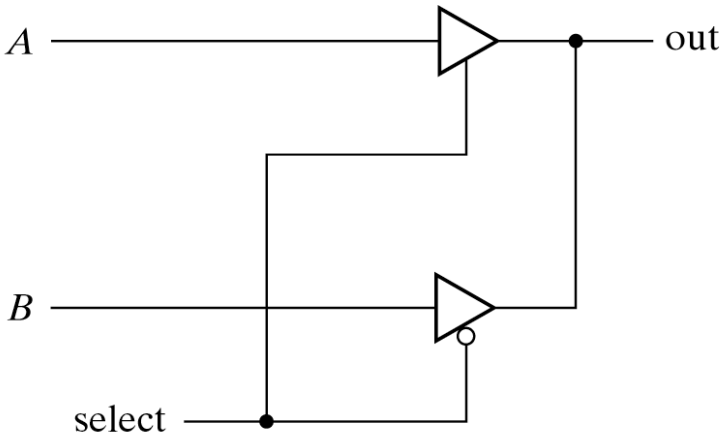
```
module _4bit_adder (S,C4,A,B,C0);
   input [3:0] A,B;
   input C0;
   output [3:0] S;
   output C4;
   wire C1,C2,C3;  //Intermediate carries
//Instantiate the fulladder
   fulladder  FA0 (S[0],C1,A[0],B[0],C0),
              FA1 (S[1],C2,A[1],B[1],C1),
              FA2 (S[2],C3,A[2],B[2],C2),
              FA3 (S[3],C4,A[3],B[3],C3);
endmodule
```

A vector of 4 bits

4-bit Adder – DF

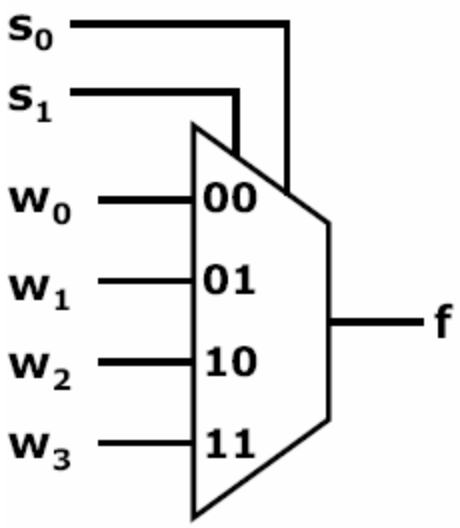
```
module binary_adder (A, B, Cin, SUM, Cout);

        input [3:0] A, B;
        input Cin;
        output [3:0] SUM;
        output Cout;

        assign    {Cout, SUM} = A + B + Cin;

endmodule
```

Muxtri – GL




```
module muxtri (A,B,select,OUT);
  input  A,B,select;
  output OUT;
  tri OUT;
  bufif1(OUT,A,select);
  bufif0(OUT,B,select);
endmodule
```

4:1 MUX – BH



s_1	s_0	f
0	0	w_0
0	1	w_1
1	0	w_2
1	1	w_3

```
module mux4x1_bh (i0,i1,i2,i3,select,y);
  input i0,i1,i2,i3;
  input [1:0] select;
  output y;
  reg y;
  always @ (i0 or i1 or i2 or i3 or select)
        case (select)
          2'b00: y = i0;
          2'b01: y = i1;
          2'b10: y = i2;
          2'b11: y = i3;
        endcase
endmodule
```

$$f = s_1's_0'w_0 + s_1's_0w_1 + s_1s_0'w_2 + s_1s_0w_3$$

YORK UNIVERSITÉ UNIVERSITY

Agenda

- Boolean Axioms and Theorems
- Basic application to Boolean equations
- A simple synthesis example
- Basic definition
- Sum-of-Products
- Product-of-Sums
- Other gates (more transistor stuff)
- Bubble pushing

Boolean Algebra

- Boolean Equations: Expressions relating variables that are true or false
  - perfect for describing digital logic circuits
  - should be able to write directly from truth table
    - but this could lead to pretty complex looking expressions
    - can we simplify?

- Boolean Algebra: A bunch of rules to manipulate and simplify Boolean equations (i.e. stuff describing our circuits) to make it easier to realize complex functions with simple logic gates

Axioms

	Axiom		Dual	Name
A1	$B = 0$ if $B \neq 1$	A1′	$B = 1$ if $B \neq 0$	Binary field
A2	$\overline{0} = 1$	A2′	$\overline{1} = 0$	NOT
A3	$0 \bullet 0 = 0$	A3′	$1 + 1 = 1$	AND/OR
A4	$1 \bullet 1 = 1$	A4′	$0 + 0 = 0$	AND/OR
A5	$0 \bullet 1 = 1 \bullet 0 = 0$	A5′	$1 + 0 = 0 + 1 = 1$	AND/OR

- Duality in axioms (and theorems):
  - ANDs and ORs, 0's and 1's interchanged

Theorems

- From axioms
- Rules for dealing with single variables

	Theorem		Dual	Name
T1	$B \bullet 1 = B$	T1′	$B + 0 = B$	Identity
T2	$B \bullet 0 = 0$	T2′	$B + 1 = 1$	Null Element
T3	$B \bullet B = B$	T3′	$B + B = B$	Idempotency
T4		$\overline{\overline{B}} = B$		Involution
T5	$B \bullet \overline{B} = 0$	T5′	$B + \overline{B} = 1$	Complements

Theorems of Several Variables

	Theorem		Dual	Name
T6	$B \bullet C = C \bullet B$	T6′	$B + C = C + B$	Commutativity
T7	$(B \bullet C) \bullet D = B \bullet (C \bullet D)$	T7′	$(B + C) + D = B + (C + D)$	Associativity
T8	$(B \bullet C) + B \bullet D = B \bullet (C + D)$	T8′	$(B + C) \bullet (B + D) = B + (C \bullet D)$	Distributivity
T9	$B \bullet (B + C) = B$	T9′	$B + (B \bullet C) = B$	Covering
T10	$(B \bullet C) + (B \bullet \overline{C}) = B$	T10′	$(B + C) \bullet (B + \overline{C}) = B$	Combining
T11	$(B \bullet C) + (\overline{B} \bullet D) + (C \bullet D)$ $= B \bullet C + \overline{B} \bullet D$	T11′	$(B + C) \bullet (\overline{B} + D) \bullet (C + D)$ $= (B + C) \bullet (\overline{B} + D)$	Consensus
T12	$\overline{B_0 \bullet B_1 \bullet B_2 \ldots}$ $= (\overline{B_0} + \overline{B_1} + \overline{B_2} \ldots)$	T12′	$\overline{B_0 + B_1 + B_2 \ldots}$ $= (\overline{B_0} \bullet \overline{B_1} \bullet \overline{B_2})$	De Morgan's Theorem

Simplifying Boolean Equations

- Example 1
- $Y = AB + \overline{A}B$

- Example 2
- $Y = A(AB + ABC)$

Synthesis Example

x_1	x_2	$f(x_1 \cdot x_2)$
0	**0**	1
0	1	1
1	**0**	**0**
1	1	1

YORK UNIVERSITÉ UNIVERSITY

Some Definition

- Complement: variable with a bar over it
  $\bar{A}, \bar{B}, \bar{C}$
- Literal: variable or its complement
  $A, \bar{A}, B, \bar{B}, C, \bar{C}$
- Implicant: product of literals
  $AB\bar{C}, \bar{A}C, BC$
- Minterm: product that includes all input variables
  $AB\bar{C}, A\bar{B}\bar{C}, ABC$
- Maxterm: sum that includes all input variables
  $(A+\bar{B}+C), (\bar{A}+B+\bar{C}), (\bar{A}+\bar{B}+C)$

Sum-of-Products (SOP) Form

- All equations can be written in SOP form
- Each row has a minterm
- A minterm is a product (AND) of literals
- Each minterm is TRUE for that row (and only that row)
- Form function by ORing minterms where the output is TRUE
- Thus, a sum (OR) of products (AND terms)

A	B	Y	minterm	minterm name
0	0	0	$\overline{A}\,\overline{B}$	m_0
0	1	1	$\overline{A}\,B$	m_1
1	0	0	$A\,\overline{B}$	m_2
1	1	1	$A\,B$	m_3

$$Y = F(A, B) =$$

Product-of-Sums (POS) Form

- All Boolean equations can be written in POS form
- Each row has a maxterm
- A maxterm is a sum (OR) of literals
- Each maxterm is FALSE for that row (and only that row)
- Form function by ANDing the maxterms for which the output is FALSE
- Thus, a product (AND) of sums (OR terms)

A	B	Y	maxterm	maxterm name
0	0	0	$A + B$	M_0
0	1	1	$A + \overline{B}$	M_1
1	0	0	$\overline{A} + B$	M_2
1	1	1	$\overline{A} + \overline{B}$	M_3

$$Y = F(A, B) = (A + B)(A + \overline{B}) = \Pi(0, 2)$$

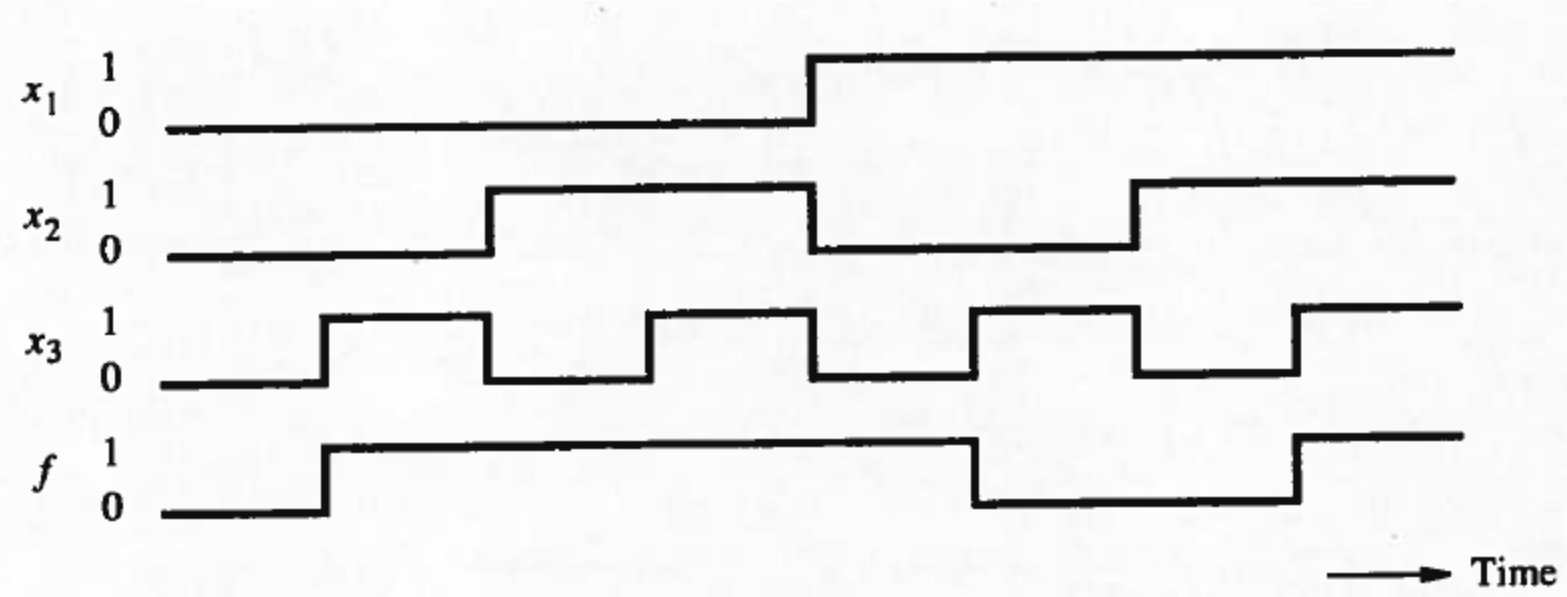
Boolean Equations Example

- You are going to the cafeteria for lunch
  - You won't eat lunch ($\overline{E}$)
  - If it's not open ($\overline{O}$) or
  - If they only serve corndogs (C)
- Write a truth table for determining if you will eat lunch (E)

O	C	E
0	0	
0	1	
1	0	
1	1	

Basic Synthesis Example

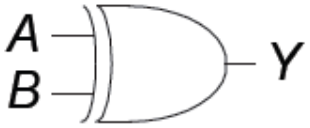
- Synthesize



Other Circuits

- We have more basic logic functions to synthesize with

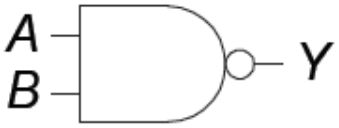
XOR



$Y = A \oplus B$

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

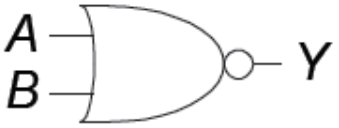
NAND



$Y = \overline{AB}$

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

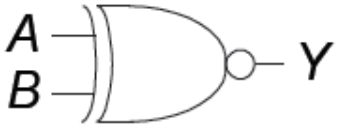
NOR



$Y = \overline{A + B}$

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

XNOR

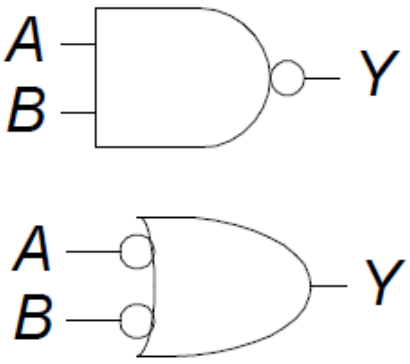


$Y = \overline{A \oplus B}$

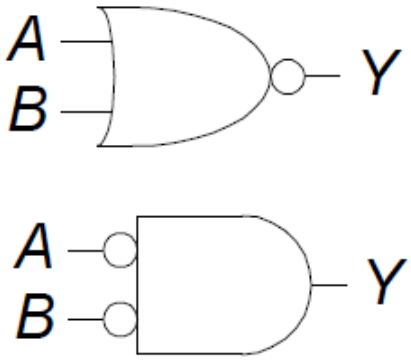
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

DeMorgan's Theorem

- $Y = \overline{AB} = \overline{A} + \overline{B}$

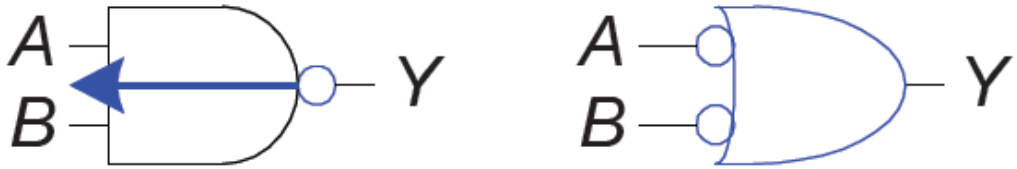


- $Y = \overline{A + B} = \overline{A} \bullet \overline{B}$

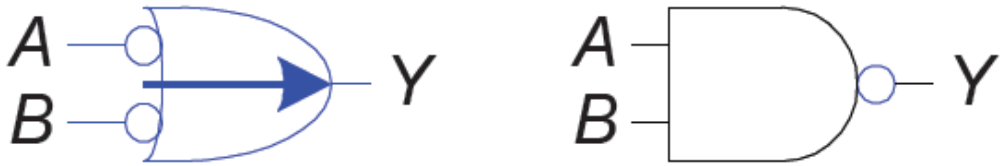


Bubble Pushing

- Backward:
  - Body changes
  - Adds bubbles to inputs

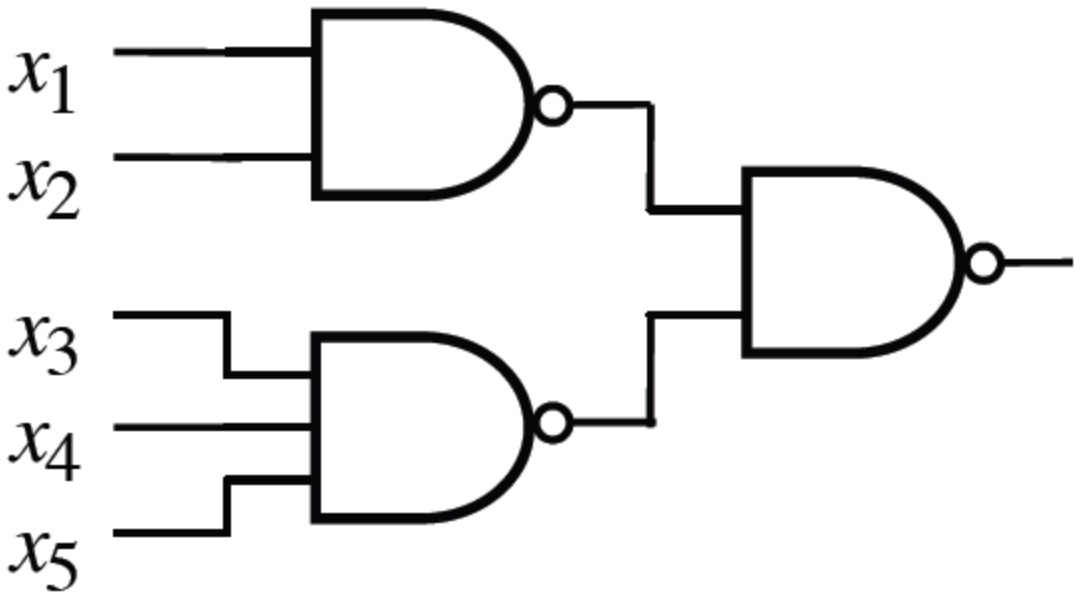


- Forward:
  - Body changes
  - Adds bubble to output



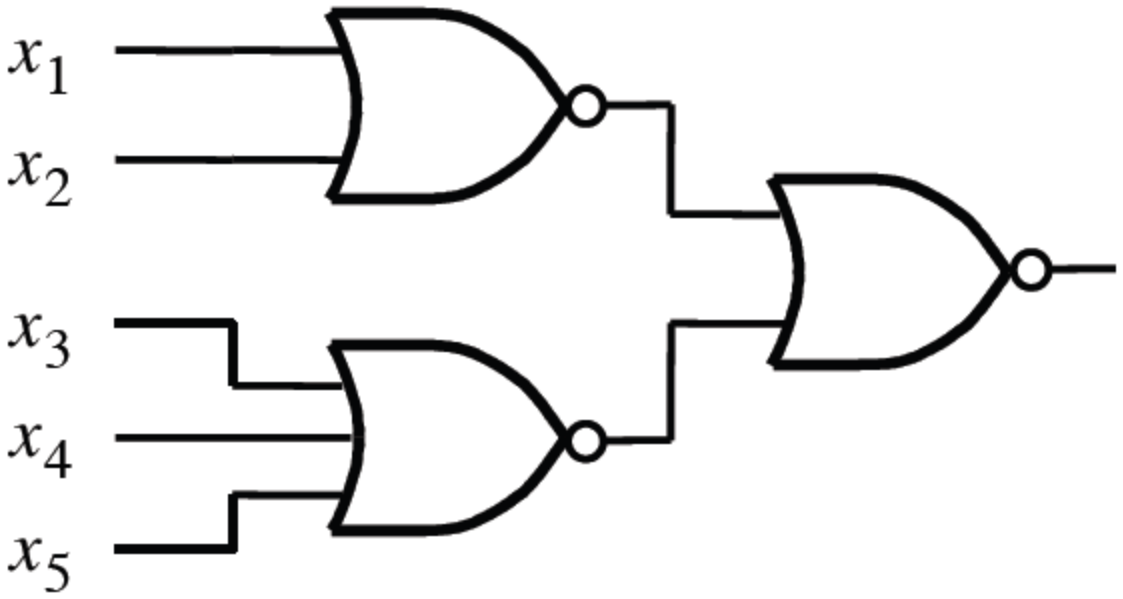
Bubble Pushing and SOP

- What is the Boolean expression for this circuit?



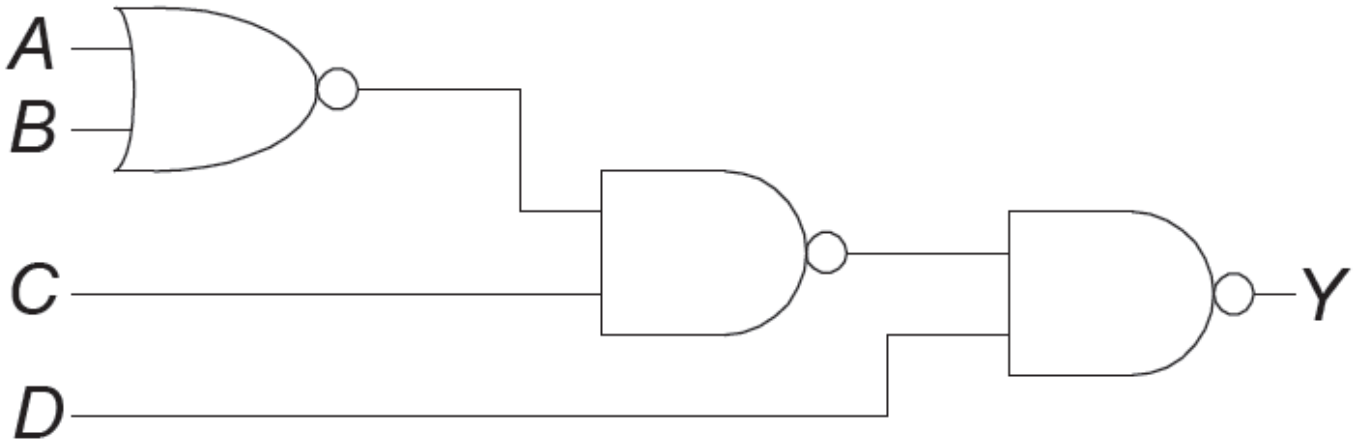
Bubble Pushing and POS

- What is the Boolean expression for this circuit?



Bubble Pushing Rules

- Begin at output, then work toward inputs
- Push bubbles on final output back
- Draw gates in a form so bubbles cancel



References

- Lecture Notes of Dr. Sebastian Magierowski – Fall 2013