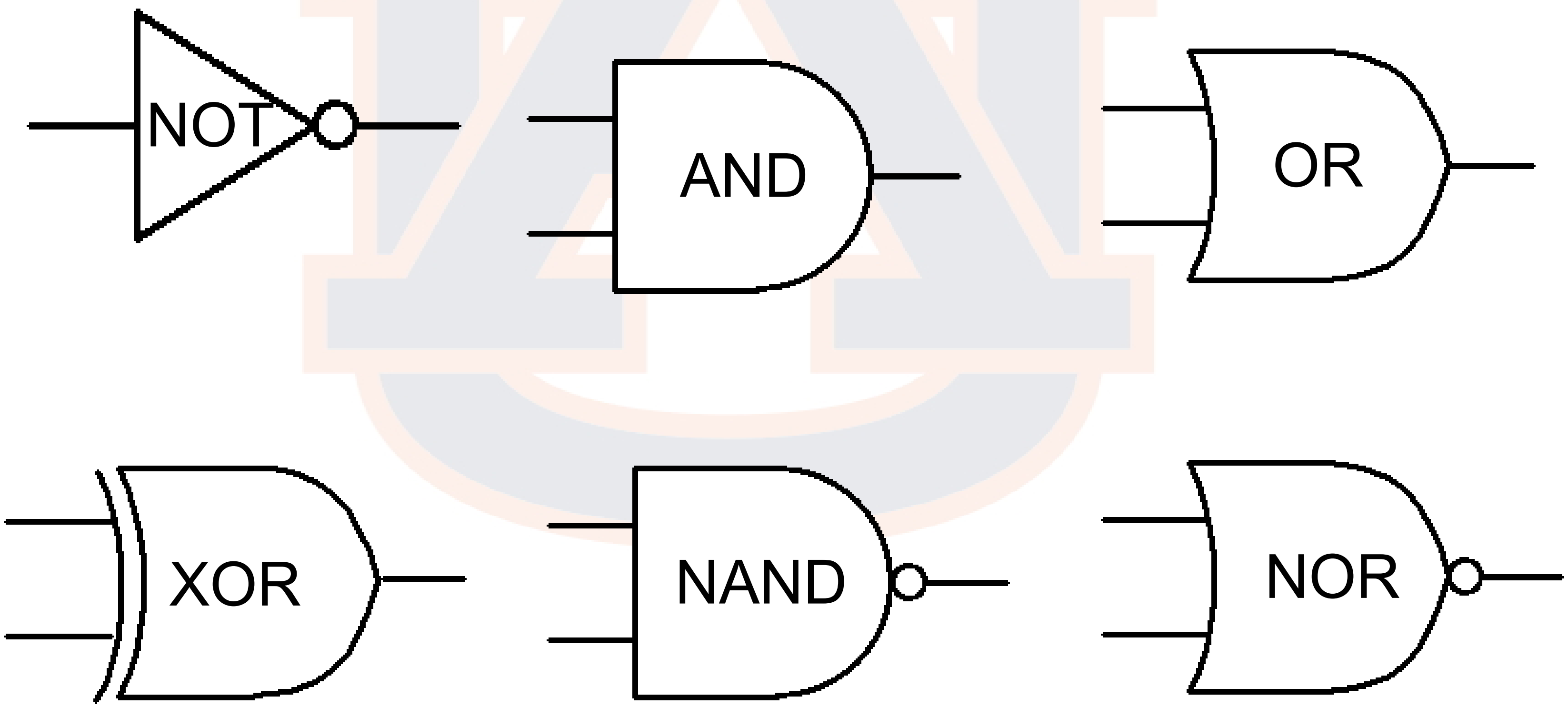
Digital Logic Circuits
'Circuit Diagrams'

ELEC2200
Summer 2009

David J. Broderick
brodedj@auburn.edu
http://www.auburn.edu/~brodedj
Office: Broun 360

Circuit Diagrams

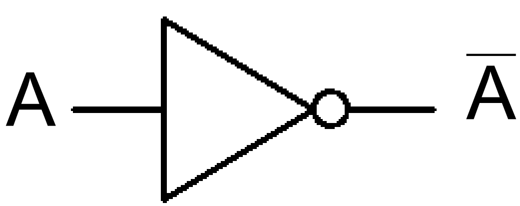
- Let's employ our design techniques and draw the corresponding circuit



Elementary Gates

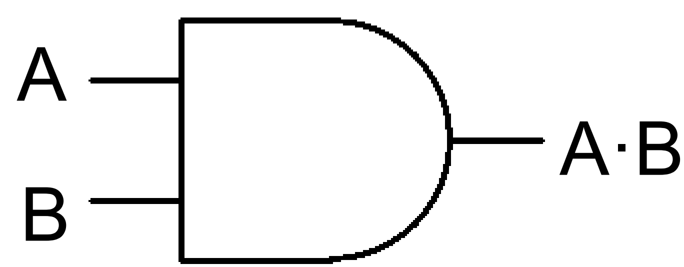
- Complement, Invert, Not
- Two notations: A' and $\overline{A}$

A	$\overline{A}$
0	1
1	0



- And
- Notated as A·B

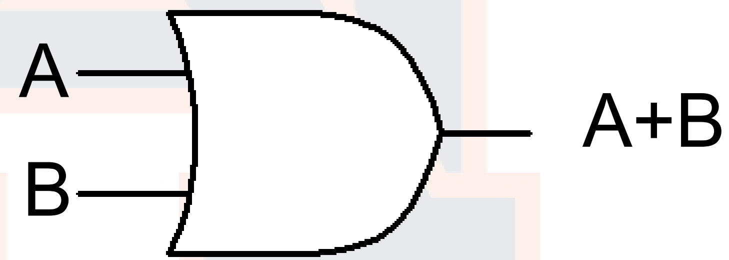
A	B	A·B
0	0	0
0	1	0
1	0	0
1	1	1



Elementary Gates

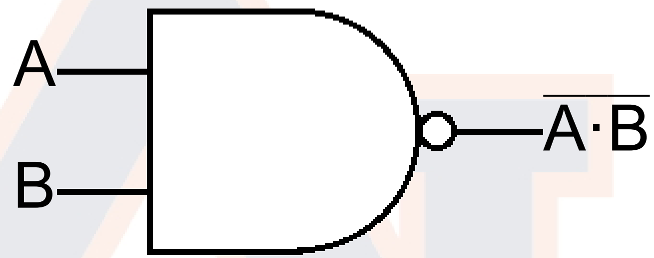
- Or
- Notated as A+B

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1



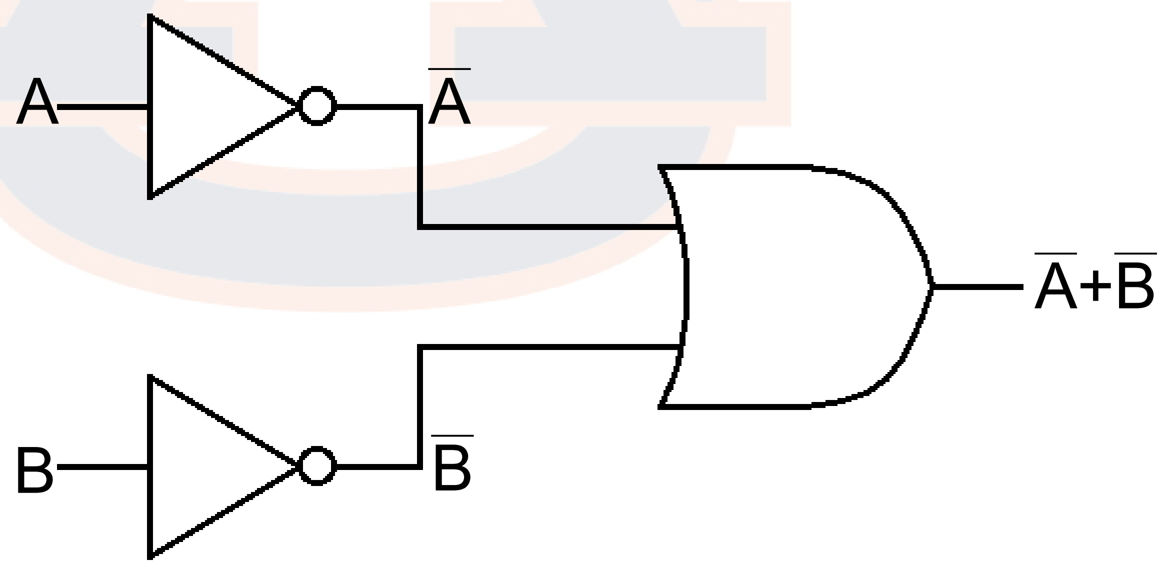
Elementary Gates

- NAND
  - $\overline{A \cdot B}$



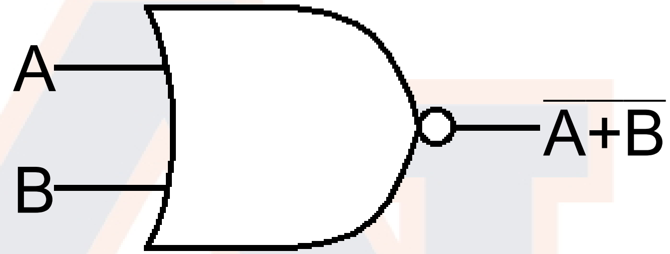
- DeMorgan's gives us $\overline{A \cdot B} = \overline{A} + \overline{B}$

A	B	$\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0



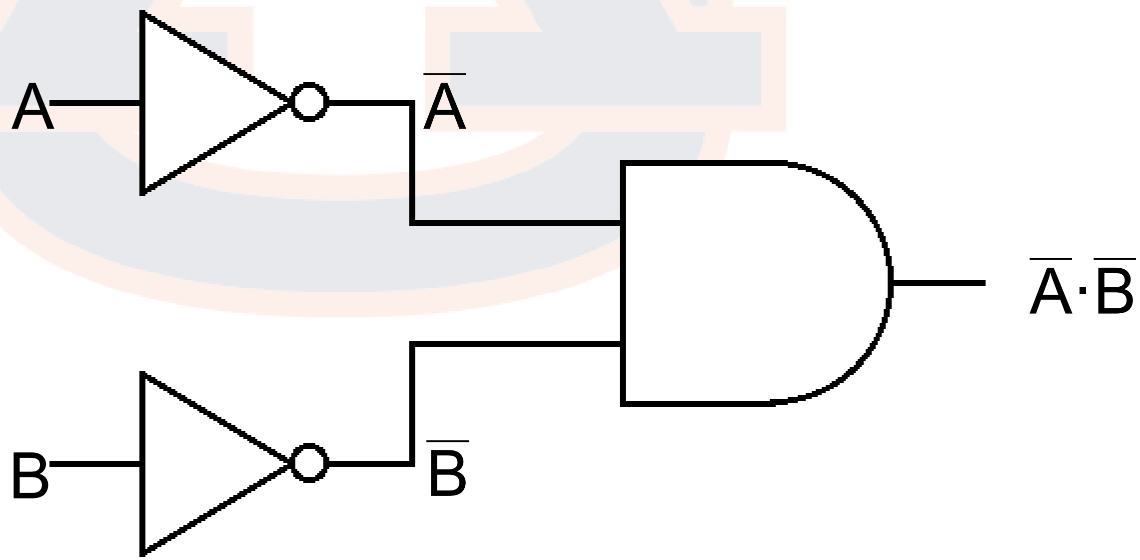
Elementary Gates

- NOR
  - $\overline{A+B}$



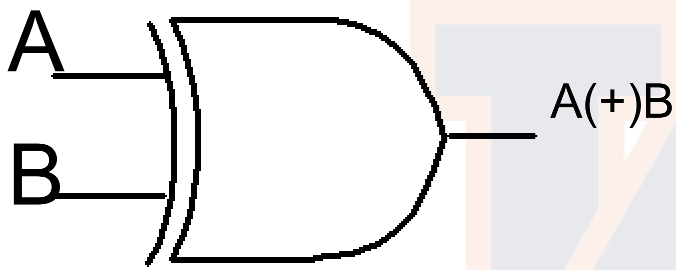
- DeMorgan's gives us $\overline{A+B}=\overline{A}\cdot\overline{B}$

A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0



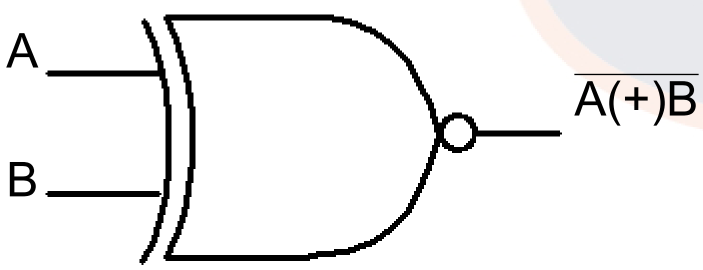
Elementary Gates

- XOR



A	B	A(+)B
0	0	0
0	1	1
1	0	1
1	1	0

- XNOR



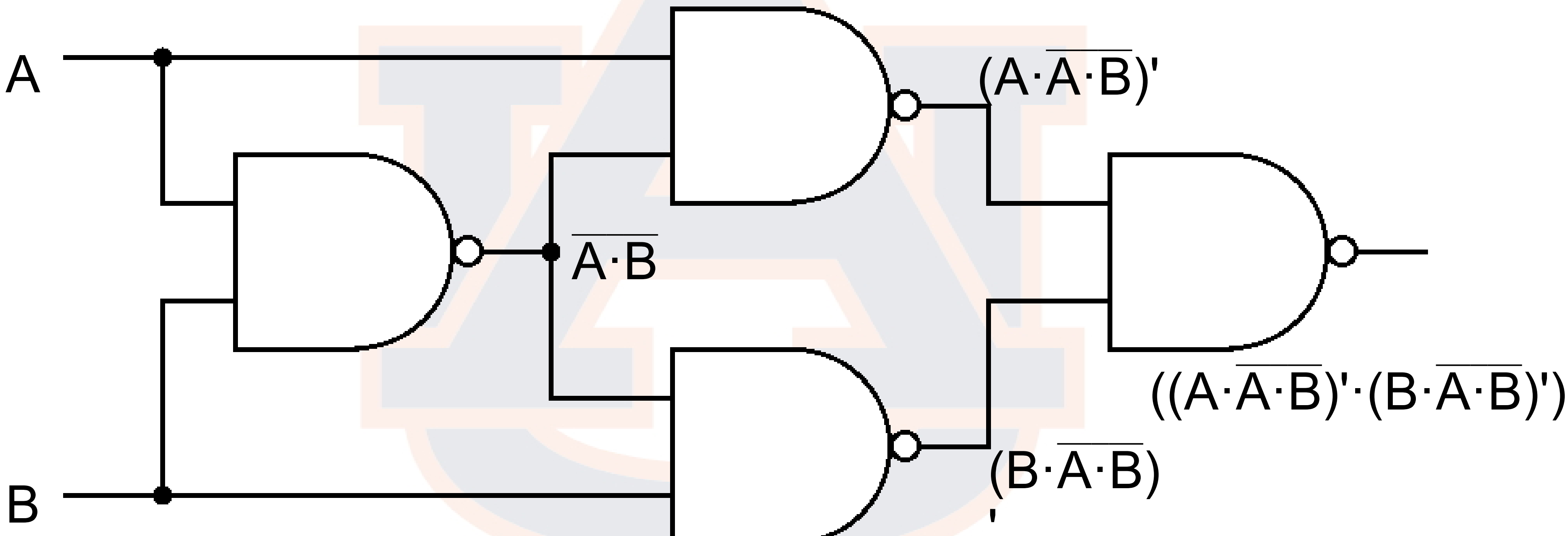
A	B	$\overline{A(+)B}$
0	0	1
0	1	0
1	0	0
1	1	1

XOR Properties

- XOR has some unique properties
  - Controlled Inverter
    - $A(+)0=A$
    - $A(+)1=\overline{A}$
  - Invert one XOR input->XNOR
    - $A(+)\overline{B}=\overline{A}(+)B=\overline{A(+)B}$
  - Invert one XOR input->XNOR
    - $(A(+)\overline{B})'=(\overline{A}(+)B)'=A(+)B$
  - Constant output
    - $A(+)A=0$
    - $A(+)\overline{A}=1$
- Multiple implementations of XOR possible

XOR Implementations

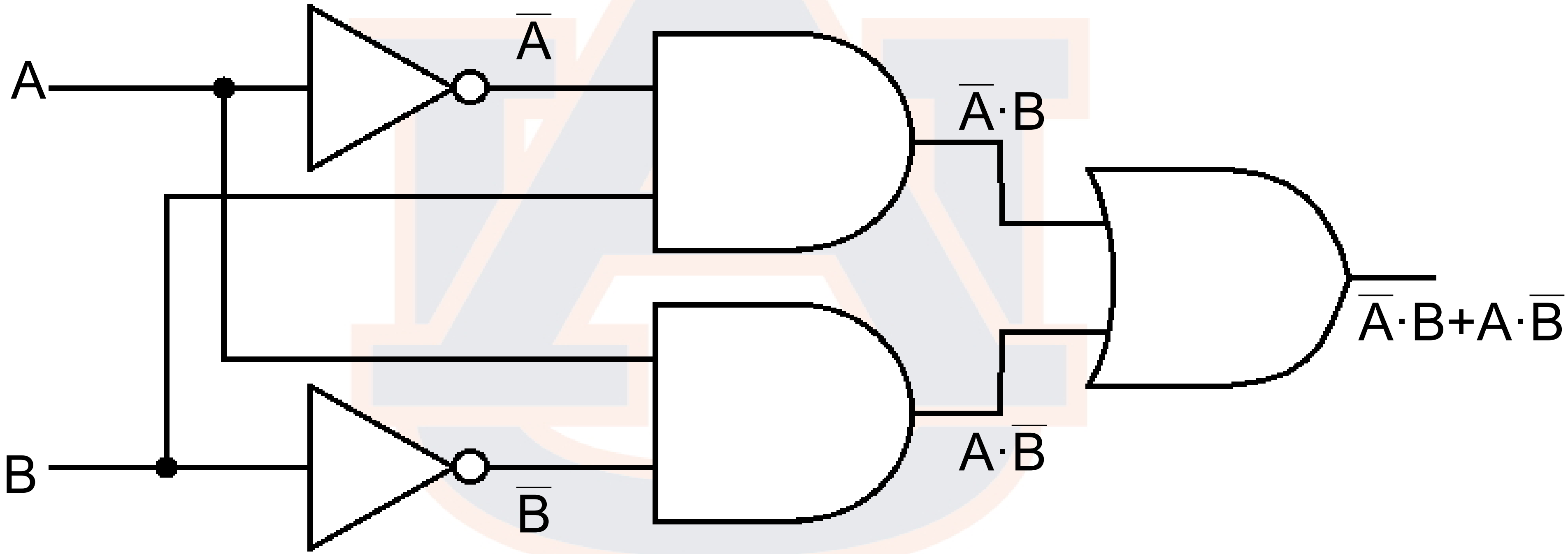
- 4 Gates



$$((A\cdot\overline{A\cdot B})'\cdot(B\cdot\overline{A\cdot B})')'=A(+)B$$

XOR Implementations

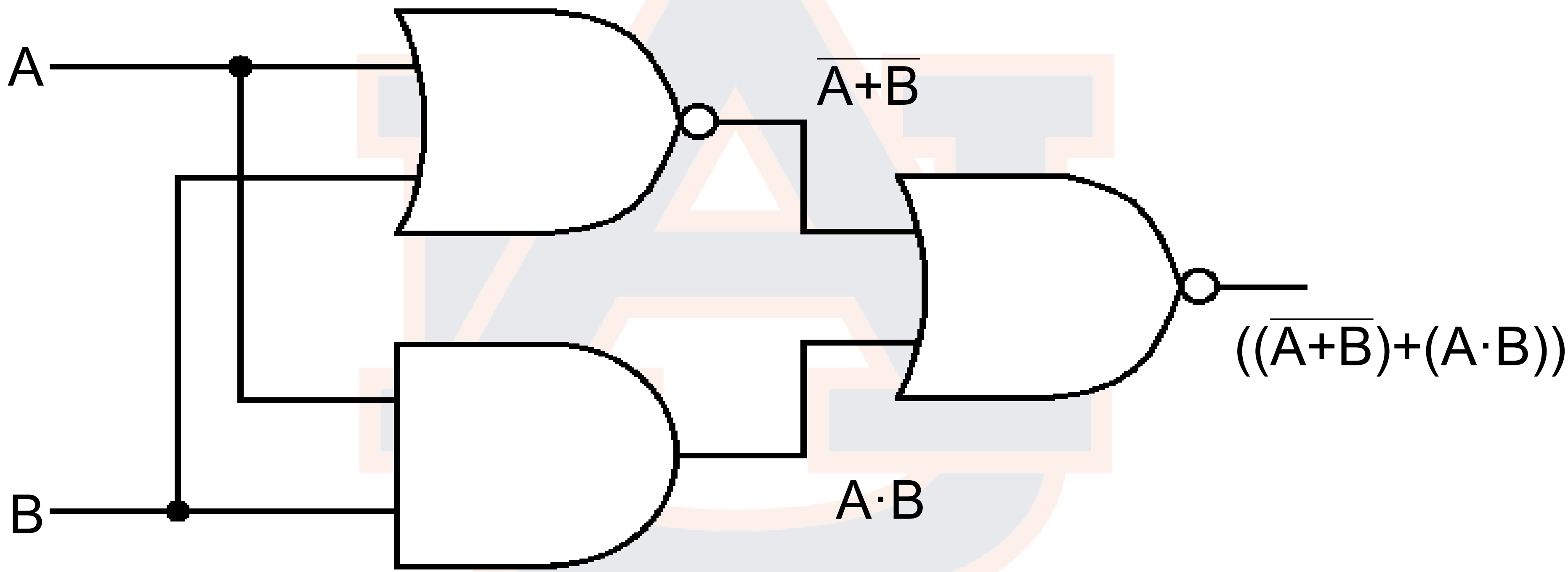
- 5 Gates



$$\overline{A}\cdot B+A\cdot\overline{B}=A(+)B$$

XOR Implementations

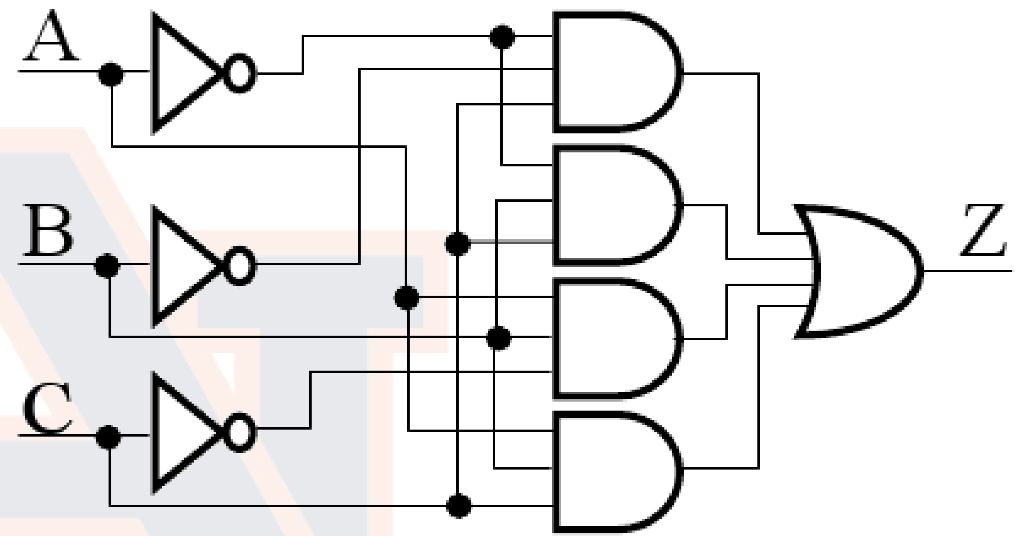
- 3 Gates



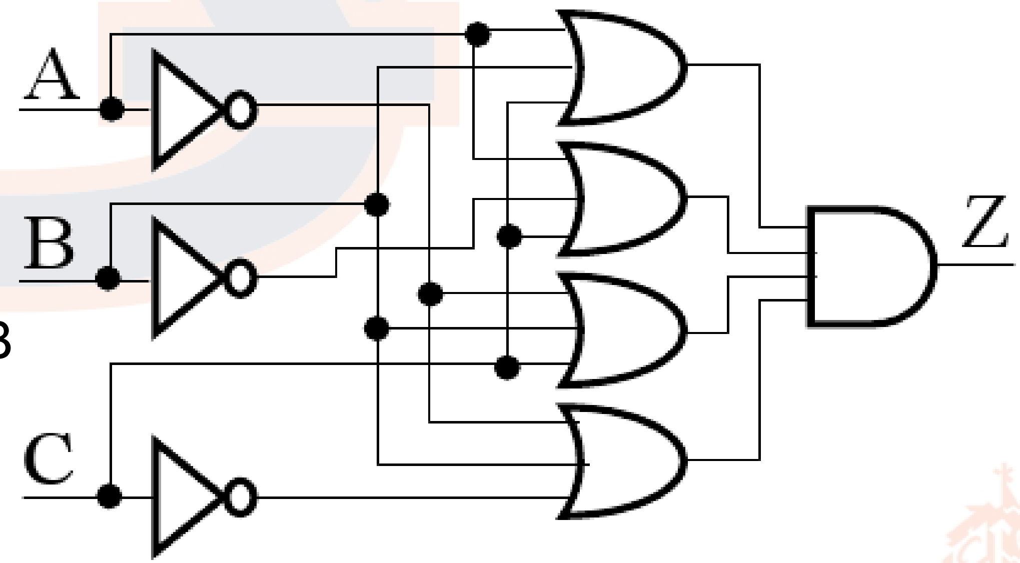
$$((\overline{A+B})+(A\cdot B))'=A(+)B$$

SOP/POS Circuits

- SOP expressions
  - AND-OR
  - Inverters for complemented literals
  - $Z=\overline{A}\cdot\overline{B}\cdot C+\overline{A}\cdot B\cdot C+A\cdot B\cdot\overline{C}+A\cdot B\cdot C$
  - 2-level AND-OR logic

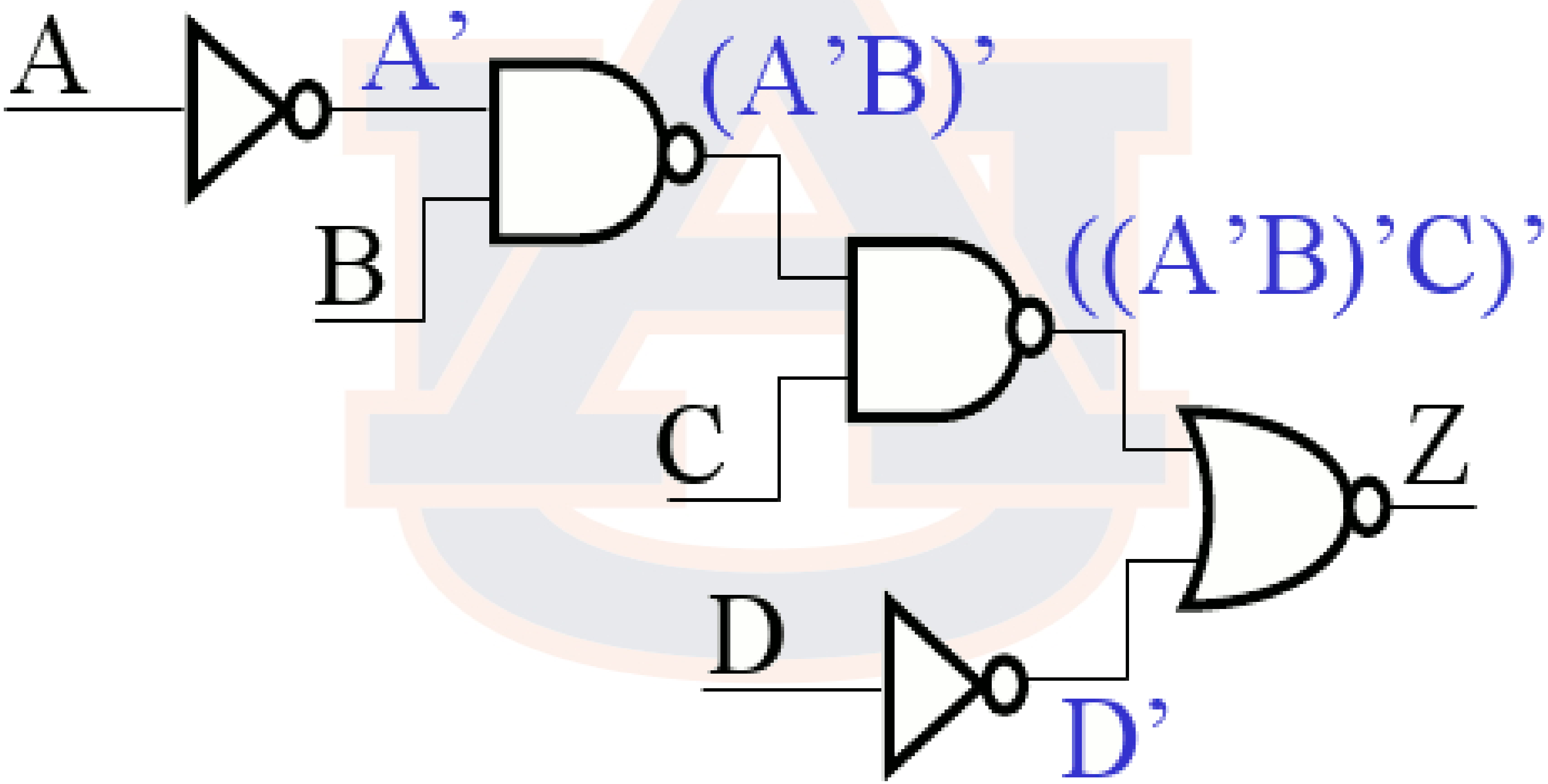


- POS expressions
  - OR-AND
  - Inverters for complemented literals
  - $Z=(A+B+C)\cdot(A+\overline{B}+C)\cdot(\overline{A}+B+C)\cdot(\overline{A}+B+\overline{C})$
  - 2-level OR-AND logic



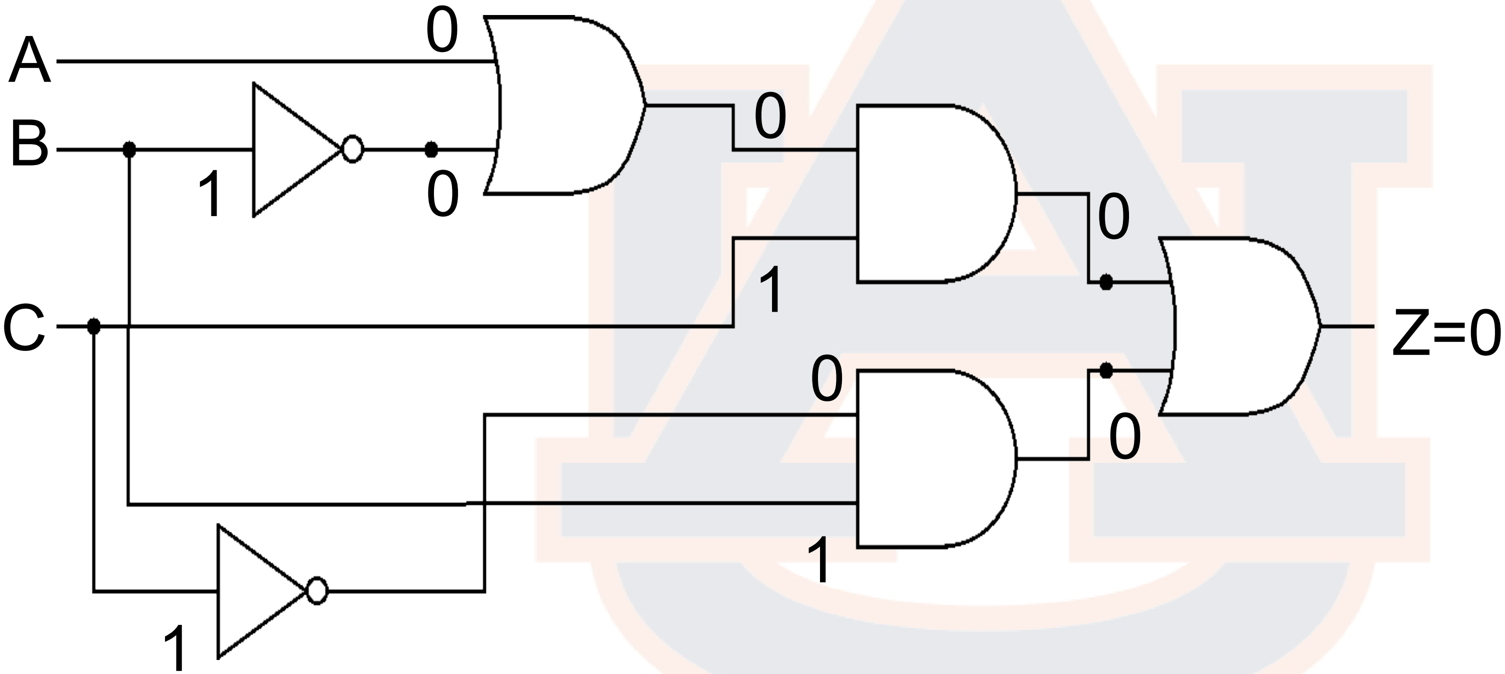
Gate Level Representation

- From $Z=\overline{(((A'\cdot B)'\cdot C)'+D')}$



Circuit Analysis

- Recovering truth table from design

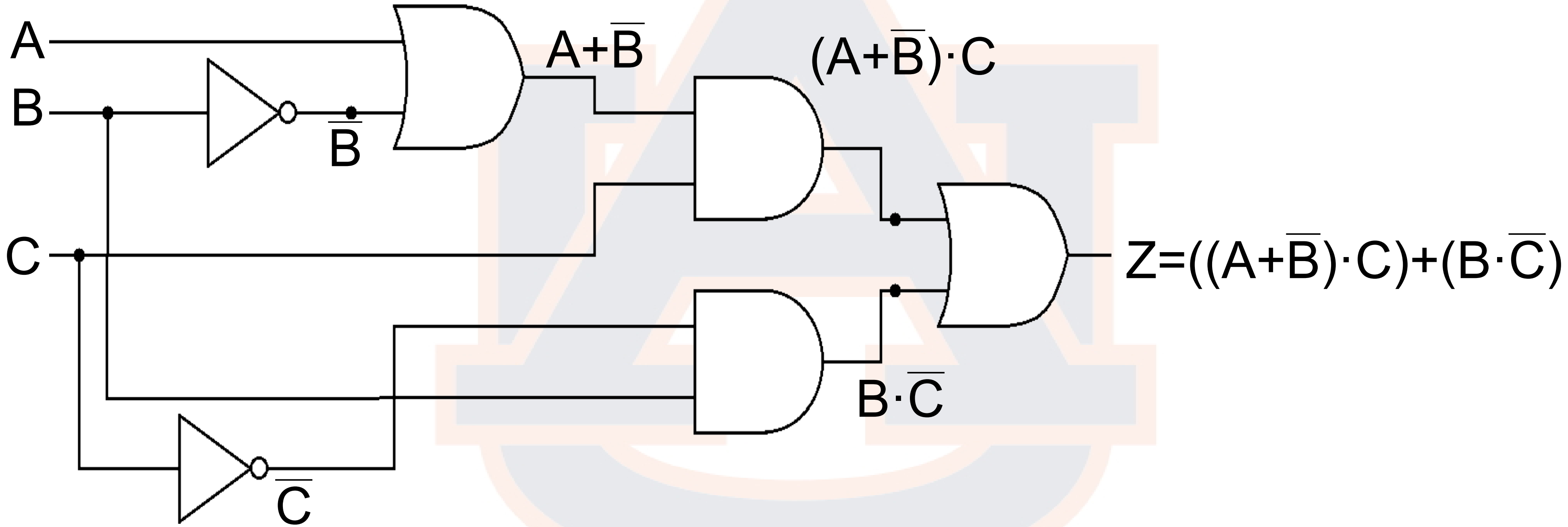


A	B	C	Z
0	0	0	
0	0	1	
0	1	0	
0	1	1	0
1	0	0	
1	0	1	
1	1	0	
1	1	1	

- Apply inputs for each row and calculate output

Circuit Analysis

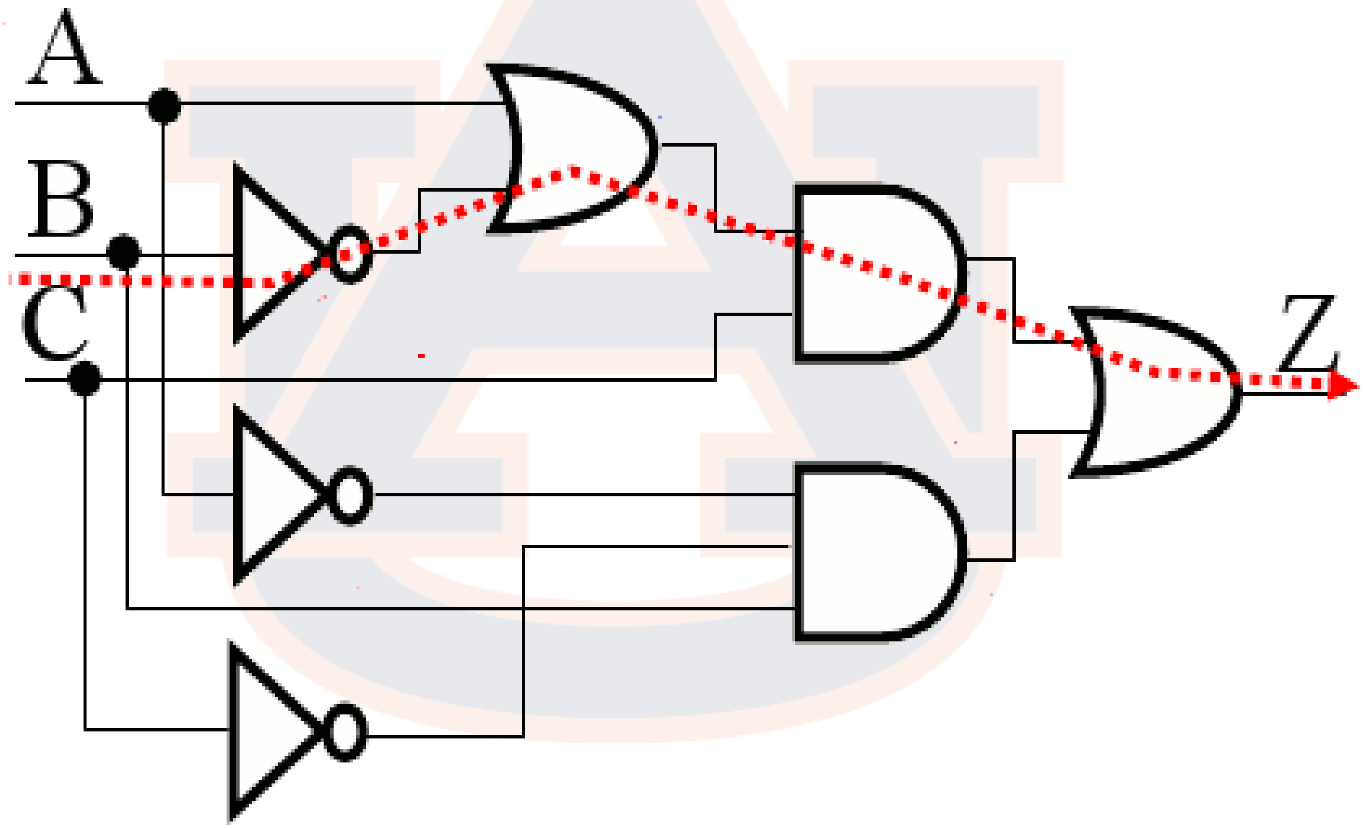
- Recovering the expression



- Propagate the expressions from input to output

Circuit Analysis

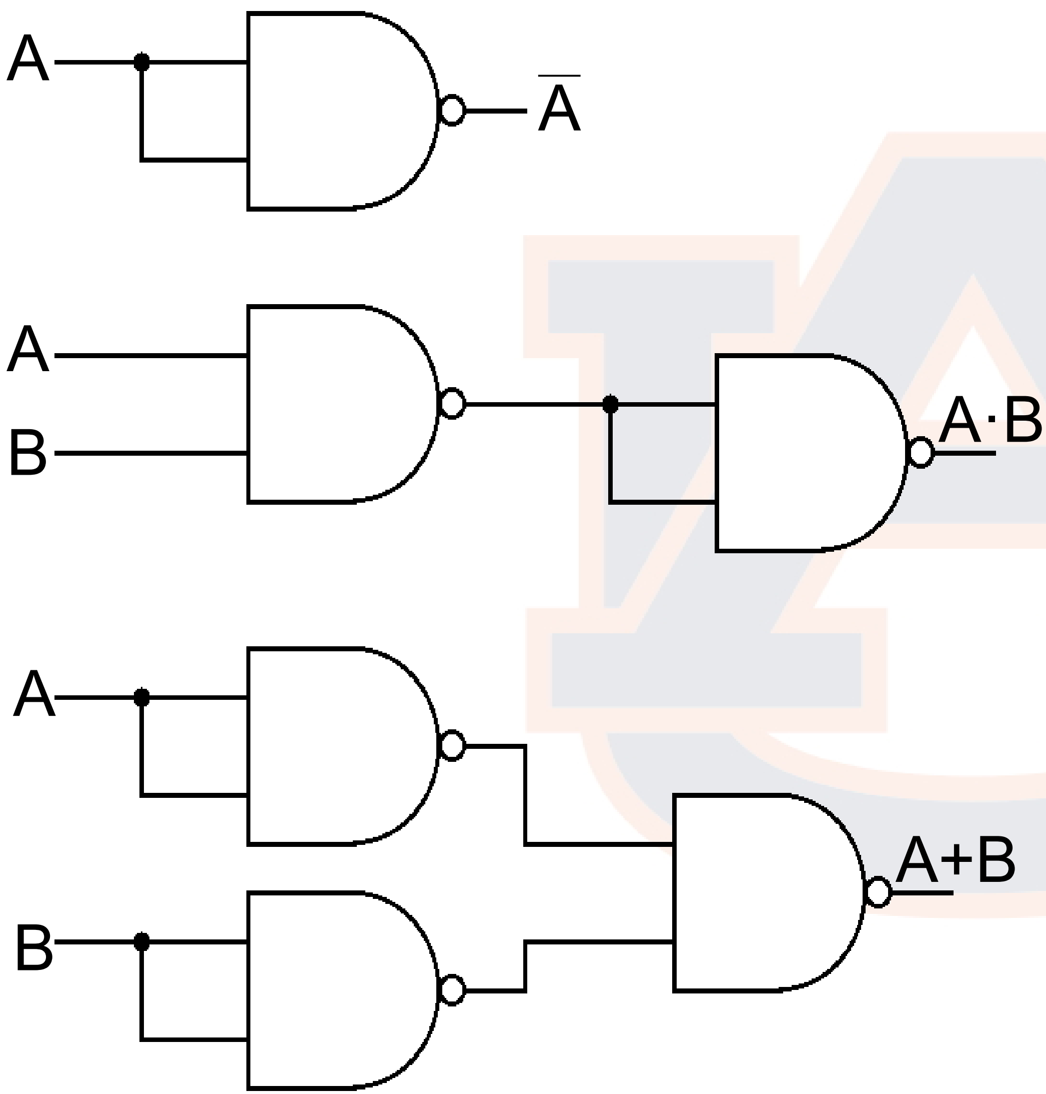
- Find the expression and truth table



Functionally Complete Sets

- If any digital circuit can be built from a set of gates (types), that set is said to be functionally complete
    - NOT, AND, OR
    - NAND
    - NOR
    - Multiplexers
- If a set can be used to construct NOT, AND, and OR functions it is functionally complete

NAND Logic



- The NAND gate is a functionally complete set
- DeMorgan's needed for OR function
- Same approach can be taken with NOR
- Can build any circuit from NAND gates alone