Logic Expressions and their Properties

- Precedence Rules (decreasing precedence): Not -> AND/NAND -> OR/XOR -> IMPL -> IFF
- Associativity:
    - Note that this is purely syntactic, does not change the output of the expression
    - In general for associativity, when we ask if an operator is associative, we mean:
    - Does (x op y) op z = x op (y op z)? For example, (1 + 2) + 3 = 1 + (2 + 3) so ‘plus’ is associative
    - AND and OR are associative
    - NAND, NOR, XOR, IMPL, IFF are *not*.
    - How to show why? Construct a truth table for x NAND (y NAND z) and show that its output is not the same as (x NAND y) NAND z
- Commutativity:
    - Does x op y = y op x? Put in words, “Can we switch the order of the *operands”* and get the same result?
    - 1 + 2 = 2 + 1, plus is commutative.
    - AND, OR, NAND, NOR, XOR, IFF are commutative (NOT doesn’t make sense, commutativity only applies to operations with more than 1 input)
    - IMPL is not commutative (A => B is not the same as B=>A). If IMPL *were* commutative, then the logical fallacy “Affirming the Consequent” would not be a fallacy. In fact, that would also make IMPL and IFF the same thing.

Diagrams for Other Gates

- See scans
- We can extend binary gates to n-ary gates, with n input lines and one output
- Input counts don’t go very high. This is because gates are not ideal devices.

Gates to Logic Formulas:

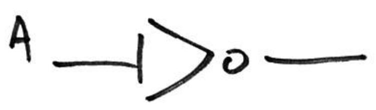
- Follow the inputs to gates, and work your way up to compute the output for each gate. Use the output as an input to the next gate and construct your formula that way.
- Disjunctive normal form: a propositional formula that corresponds to a truth table is in this format! It is the disjunction (the OR) of terms, where each term is the conjunction (AND) of the input variables
- Example: ~XY + ~X~Y + X~Y is in disjunctive normal form (DNF) but (X + Y)Z is not
- An expression is in **Full DNF** if each term (each conjunction) *includes all input variables*. None of the above are in full DNF
- But we can force them into DNF by introducing terms. For example XY + YZ is not in full DNF
- We write XY(Z + ~Z) + YZ(X + ~X). We then distribute
- XYZ + XY~Z + XYZ + ~XYZ. Notice we have one redundant term. Kill it => XYZ + XY~Z

+ ~XYZ. This is now full DNF.
- In general, if you have N rows where the output is 1 in your truth table, you will have N terms in your Full DNF formula
- I can represent any logical formula of N inputs in DNF

Logic Formulas to Gates:

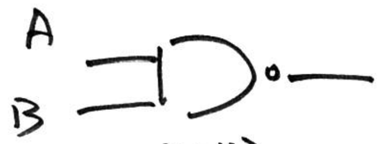
- Each term in the logic formula corresponds to a “situation(s)” where the output in the truth table will be 1
- If the formula is in Full DNF, then each term only corresponds to “one situation” (one set of inputs). If it’s not in Full DNF, then a term may correspond to more than “one situation.” For example for three inputs, ~Y~Z + X + ~XY~Z is in DNF (but not full). ~Y~Z will be true if we get input 0 for Y and Z (so ~Y and ~Z will both be true). But it doesn’t matter what X is. X~Y~Z and ~X~Y~Z will both be true, so this term corresponds to two “situations,” or two rows of the truth table. Can you guess how many rows correspond to the second term? (X). This means that as long as X is true, it doesn’t matter what the other inputs are, it will be true. This means there are 4 rows in the truth table corresponding to this term: X~Y~Z, X~YZ, XY~Z, and XYZ
- For example ~XY~Z in a formula means that the output row corresponding to 0 1 0 in my truth table will have a 1
- Once we have a truth table, we can use every term where this is the case as a conjunction (AND gate) and then we use an OR gate to tie them all together (this is the disjunction).
- Using full DNF expressions here makes it easy to go straight to only AND gates connected by a single OR gate.
- This is not done in practice (by algorithms in CAD software) because the DNF-based circuit won’t be the most efficient

NOT

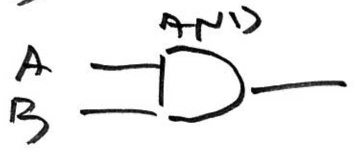


$\bar{A}$ ~A ¬A

NAND

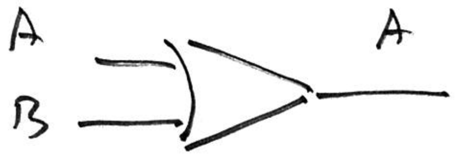


A NAND B

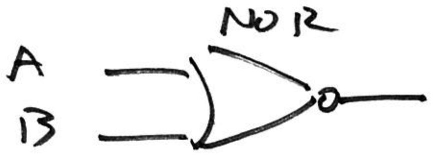


A AND B

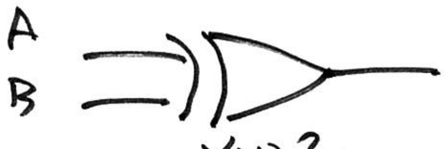
OR



A OR B

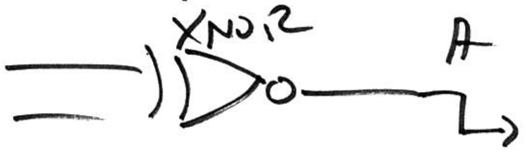


XOR



A XOR B

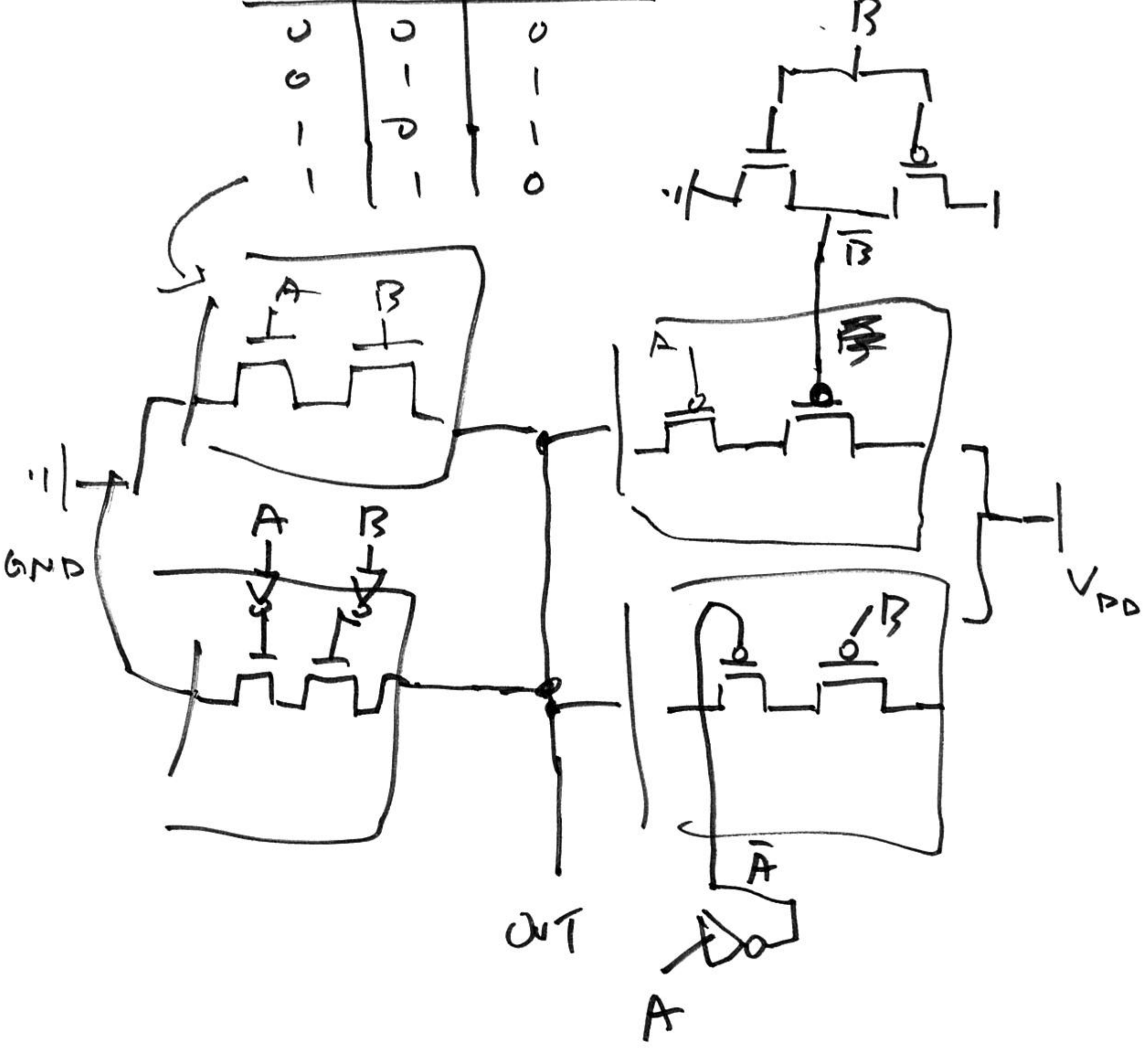
$A \oplus B$



$\sim(A \oplus B)$

XOR

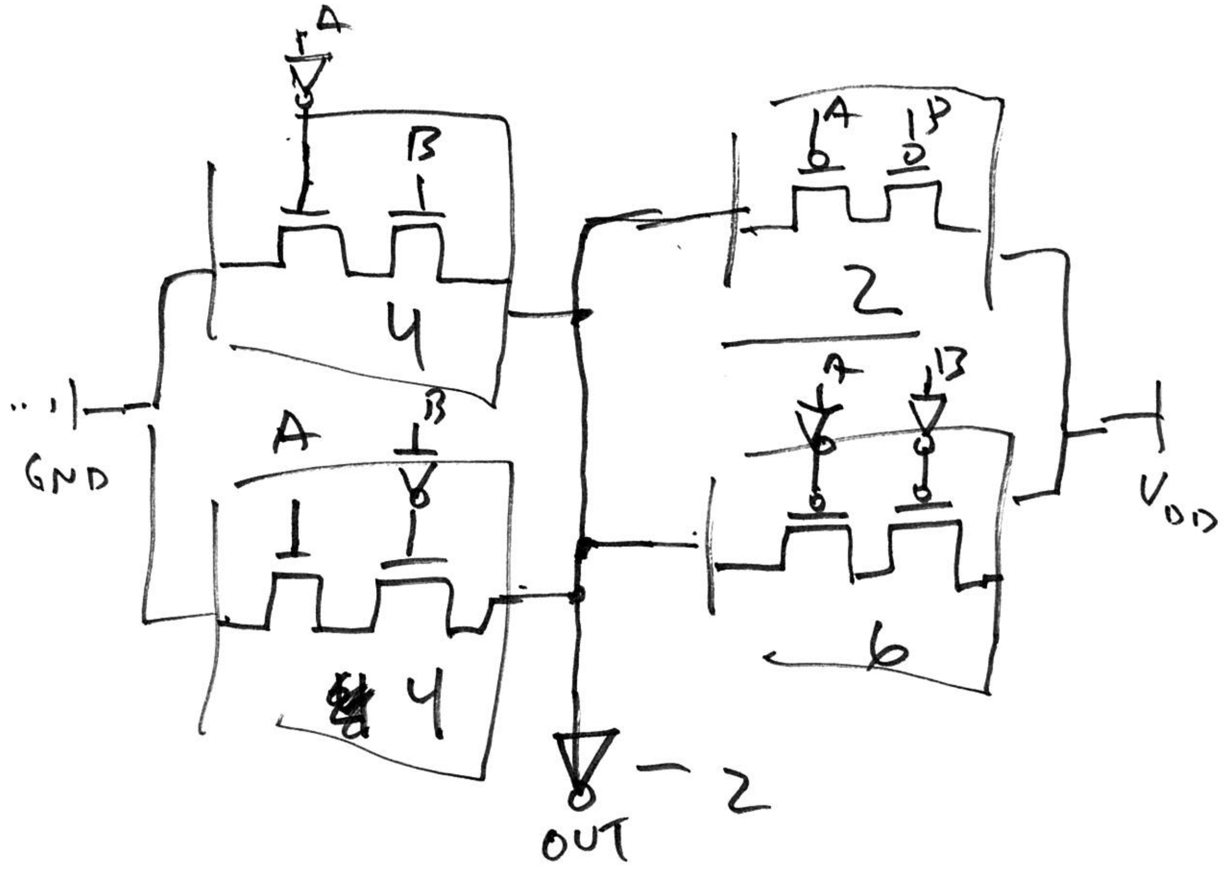
A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0



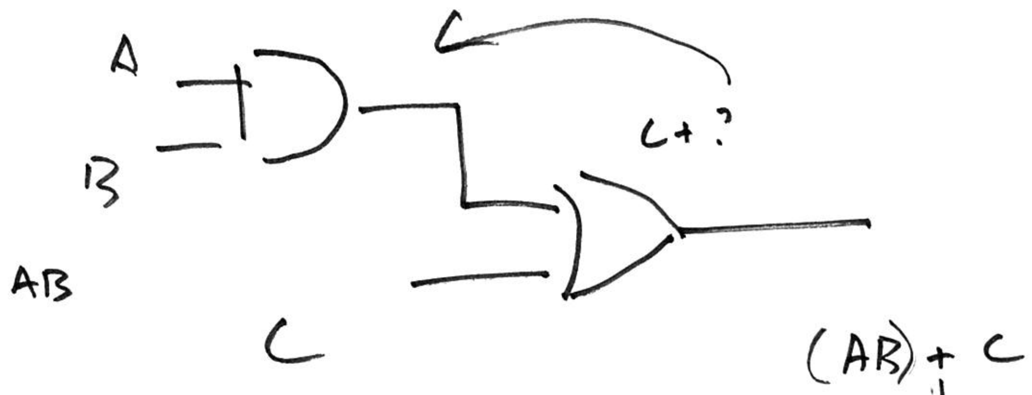
A XNOR B

	A	B	
→	0	0	1
	0	1	0
	1	0	0
→	1	1	1

$\sim(A \text{ XNOR } B) = A \text{ XOR}$



GATES



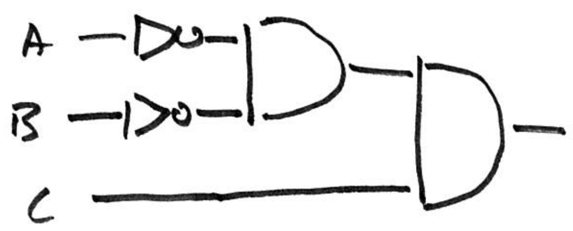
A	B	C	??
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

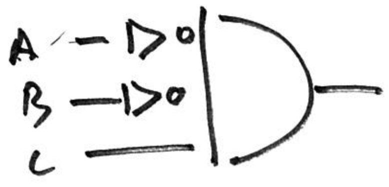
$\text{OUT} = \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$

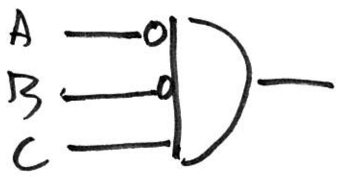
DISJUNCTIVE NORMAL FORM (DNF)

$\sim(ABC)$ FOR FIRST. ROW

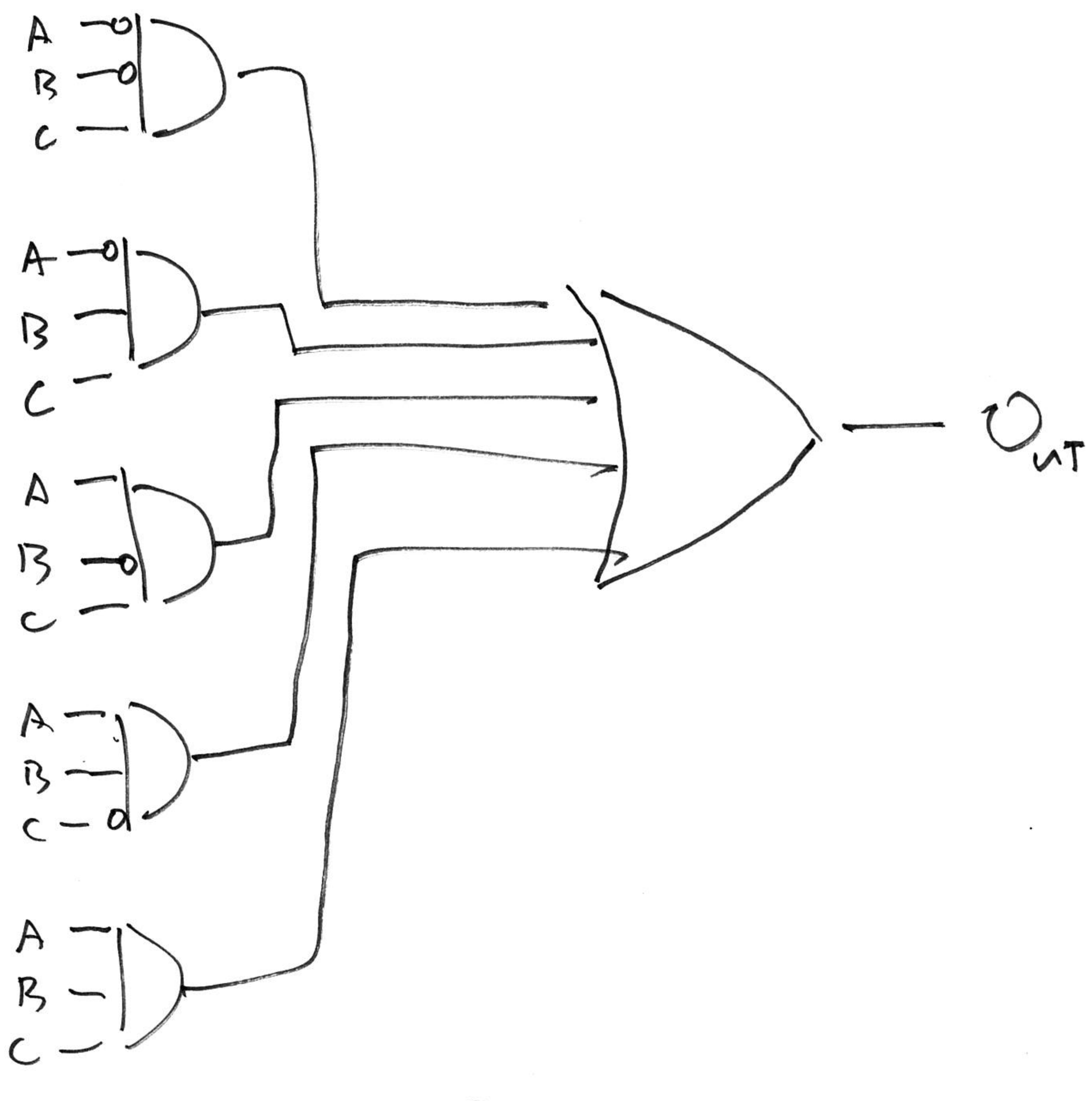
$\bar{A} + \bar{B} + \bar{C}$







[illegible] T

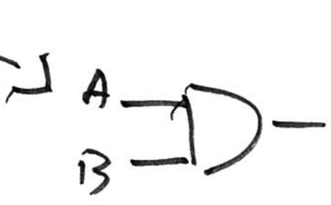

A
B
C
A
B
C
A
B
C
A
B
C
A
B
C
Out

DE MORGAN'S LAW

$$\sim(AB) = \sim A \sim B$$

$$= \sim A + \sim B$$

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



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