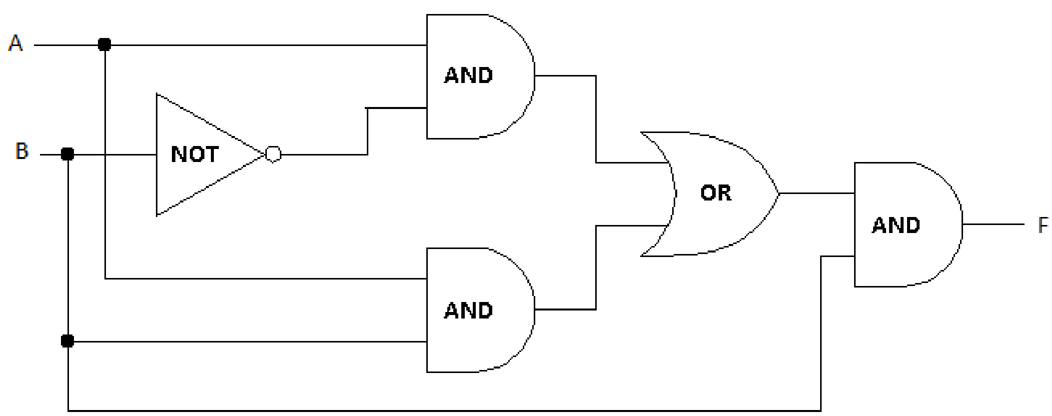
1. More logical equivalence examples

Laws of Logical Equivalence		
Name	**OR version**	**AND version**
Commutative	A + B = B + A	A * B = B * A
Associative	(A + B) + C = A + (B + C)	(A * B) * C = A * (B * C)
Distributive	A + (B * C) = (A + B) * (A + C)	A * (B + C) = (A * B) + (A * C)
Idempotent	A + A = A	A * A = A
Identity	A + 0 = A	A * 1 = A
	A + 1 = 1	A * 0 = 0
Complement	A +~A = 1	A * ~A = 0
	~1 = 0	~0 = 1
Double Negative	~(~A) = A	
De Morgan's	~(A + B) = ~A * ~B	~(A * B) = ~A + ~B
Absorption	A + (A * B) = A	A * (A + B) = A

a. Prove the OR version of the Absorption Law, A + A * B = A.

b. Simplify the following digital logic circuit using propositional algebra.



f = ((A * ~B) + (A * B)) * B Initial circuit logic

2. More on gates
    a. Functionally complete sets
        i. AND, OR, NOT
        ii. AND, NOT
        iii. OR, NOT
        iv. NAND - $\uparrow$ is the NAND operator.
        v. NOR - $\downarrow$ is the NOR operator.
3. Truth tables
    b. Minterms
    c. Maxterms
    d. Example: 3-variable Boolean function true when A is true, B is true, and C is false

4. Synthesizing using gates
    e. Consider a Boolean function of three variables
        i. True when either, but not both, of the first two variables is true

Index	A	B	C	f(A, B, C)	Minterm	Maxterm
0	0	0	0			
1	0	0	1			
2	0	1	0			
3	0	1	1			
4	1	0	0			
5	1	0	1			
6	1	1	0			
7	1	1	1			

    f. Sum-of-products
        i. Can simplify using equivalence laws to reduce number of gates
    g. Product-of-sums
        i. Can also simplify using laws of equivalence