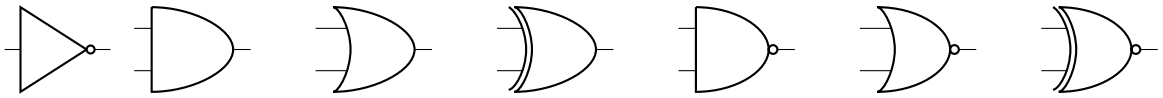
CS61C Spring 18 Discussion 5 – Logic, SDS, & FSM

Logic Gates

1. Label the following logic gates:



Solution: not, and, or, xor, nand, nor, xnor

2. Convert the following to boolean expressions:
   (a) NAND

   Solution: $\bar{A}\bar{B} + \bar{A}B + A\bar{B}$

   (b) XOR

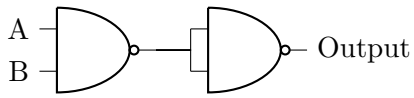
   Solution: $\bar{A}B + A\bar{B}$

   (c) XNOR

   Solution: $\bar{A}\bar{B} + AB$

3. Create an AND gate using only NAND gates.

Solution:



4. How many different two-input logic gates can there be? How many n-input logic gates?

Solution: A truth table with n inputs has 2^n rows. Each logic gate has a 0 or a 1 at each of these rows. Imagining a function as a 2^n-bit number, we count 2^{2^n} total functions, or 16 in the case of n = 2.

Boolean Logic

$1 + A = 1$	$A + \bar{A} = 1$	$A + AB = A$	$(A + B)(A + C) = A + BC$
$0B = 0$	$B\bar{B} = 0$	$A + \bar{A}B = A + B$	
DeMorgan's Law:	$\overline{AB} = \bar{A} + \bar{B}$	$\overline{A + B} = \bar{A}\bar{B}$	

1. Minimize the following boolean expressions:
   (a) Standard: $(A + B)(A + \bar{B})C$

Solution:

$$(AA + A\bar{B} + AB + B\bar{B})C = (A + A(\bar{B} + B))C = AC \tag{1}$$

(b) Grouping & Extra Terms: $\bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + AB\bar{C} + A\bar{B}\bar{C} + ABC + A\bar{B}C$

Solution:

$$\begin{aligned} \bar{A}\bar{C}(\bar{B} + B) + A\bar{C}(B + \bar{B}) + AC(B + \bar{B}) &= \bar{A}\bar{C} + A\bar{C} + AC && (2)\\ &= \bar{A}\bar{C} + A\bar{C} + A\bar{C} + AC && (3)\\ &= (\bar{A} + A)\bar{C} + A(\bar{C} + C) && (4)\\ &= A + \bar{C} && (5) \end{aligned}$$

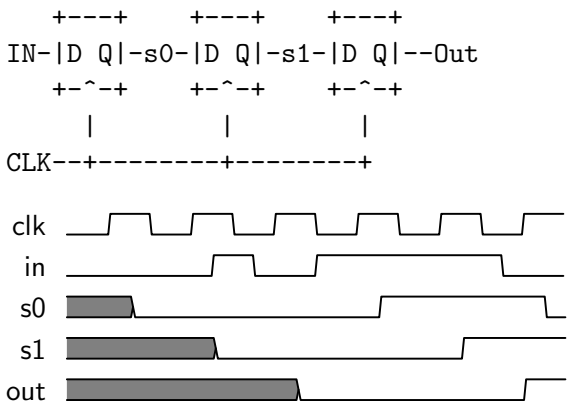
(c) DeMorgan's: $\overline{A(\bar{B}\bar{C} + BC)}$

Solution:

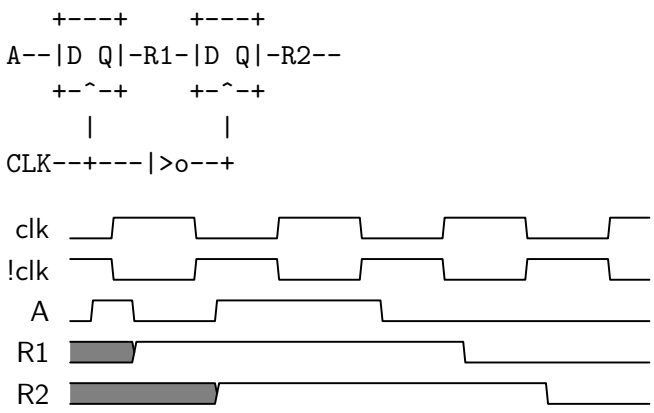
$$\begin{aligned} \overline{A(\bar{B}\bar{C} + BC)} &= \bar{A} + \overline{\bar{B}\bar{C} + BC} && (6)\\ &= \bar{A} + \overline{\bar{B}\bar{C}}\,\overline{BC} && (7)\\ &= \bar{A} + (B + C)(\bar{B} + \bar{C}) && (8)\\ &= \bar{A} + B\bar{C} + \bar{B}C && (9) \end{aligned}$$

State

1. Fill out the timing diagram for the circuit below:



2. Fill out the timing diagram for the circuit below:



FSM

1. Fill in the following FSM for outputting a 1 whenever we have two repeating bits as the most recent bits, and a 0 otherwise. You may not need all states.