

CS 61C
Fall 2018

Logic and State
Discussion 10: October 29, 2018

1 Boolean Logic

In digital electronics, it is often important to get certain outputs based on your inputs, as laid out by a truth table. Truth tables map directly to Boolean expressions, and Boolean expressions map directly to logic gates. However, in order to minimize the number of logic gates needed to implement a circuit, it is often useful to simplify long Boolean expressions.

We can simplify expressions using the nine key laws of Boolean algebra:

Name	AND Form	OR form
Commutative	$AB = BA$	$A + B = B + A$
Associative	$AB(C) = A(BC)$	$A + (B + C) = (A + B) + C$
Identity	$1A = A$	$0 + A = A$
Null	$0A = 0$	$1 + A = 1$
Absorption	$A(A + B) = A$	$\overline{A + AB} = \overline{A}$
Distributive	$(A + B)(A + C) = A + BC$	$A(B + C) = AB + AC$
Idempotent	$A(A) = A$	$A + A = A$
Inverse	$A(\overline{A}) = 0$	$A + \overline{A} = 1$
Demorgan's	$\overline{AB} = \overline{A} + \overline{B}$	$\overline{A + B} = \overline{A}(\overline{B})$

1.1 Simplify the following Boolean expressions:

(a) $(A + B)(A + \overline{B})C$

$\underbrace{(A + B)(A + \overline{B})}_A \Rightarrow AC$

(b) $\overline{A}\overline{B}\overline{C} + \overline{A}B\overline{C} + A\overline{B}\overline{C} + A\overline{B}C + ABC + A\overline{B}C$

Handwritten derivation for (b):

$$\begin{aligned} & (\overline{A} + A)(\overline{B} + B)\overline{C} + A(\overline{B} + B)C \\ & \underbrace{\overline{A}\overline{B}\overline{C} + \overline{A}B\overline{C}}_{\overline{A}\overline{C}} + \underbrace{A\overline{B}\overline{C} + A\overline{B}C}_{A\overline{B}} + \underbrace{ABC + A\overline{B}C}_0 \\ & \Rightarrow \overline{A}\overline{C} + A\overline{B} + 0 \\ & \Rightarrow \overline{A}\overline{C} + A\overline{B} \end{aligned}$$

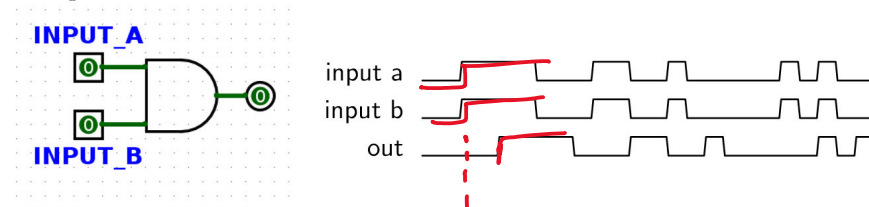
Diagram showing a box labeled 'AC' with an arrow pointing to the final result $\overline{A}\overline{C} + A\overline{B}$.

(c) $\overline{A(\overline{B}C + BC)}$

$$(d) \bar{A}(A+B) + (B+AA)(A+\bar{B}) = \underbrace{\bar{A}A}_{0} + \bar{A}B \quad \underbrace{(B+A)(A+\bar{B})}_A = \bar{A}B + A = B+A$$

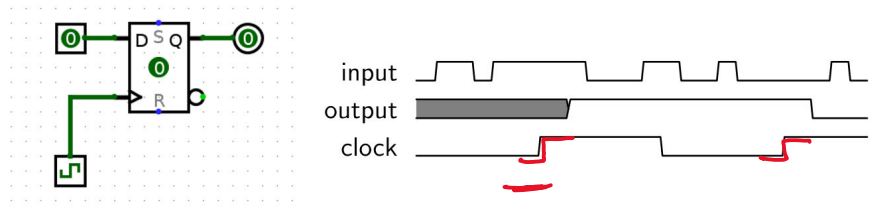
2 State Intro

There are two basic types of circuits: combinational logic circuits and state elements. **Combinational logic** circuits simply change based on their inputs after whatever propagation delay is associated with them. For example, if an AND gate (pictured below) has an associated propagation delay of 5ns, its output will change based on its input as follows:

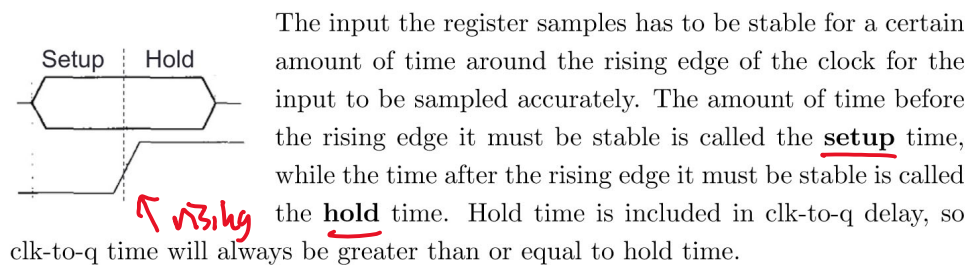


You should notice that the output of this AND gate always changes 5ns after its inputs change.

State elements, on the other hand, can *remember* their inputs even after the inputs change. State elements change value based on a clock signal. A rising edge-triggered register, for example, samples its input at the rising edge of the clock (when the clock signal goes from 0 to 1).

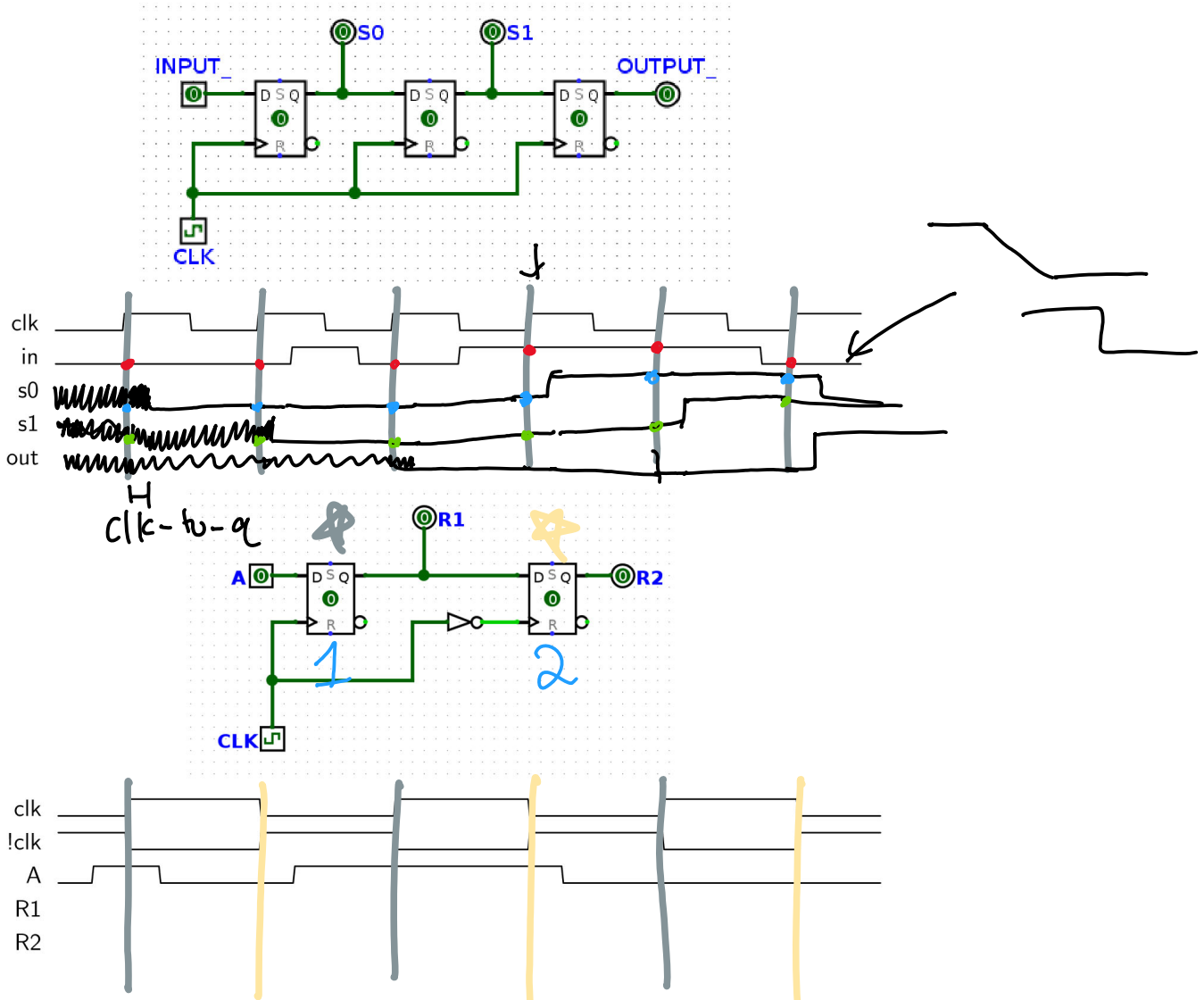


You'll notice that the value of the output in the diagram above doesn't change immediately after the rising edge of the clock. Like logic gates, registers also have a delay associated with them before their output will reflect the input that was sampled. This is called the clk-to-q delay. ("Q" often indicates output).

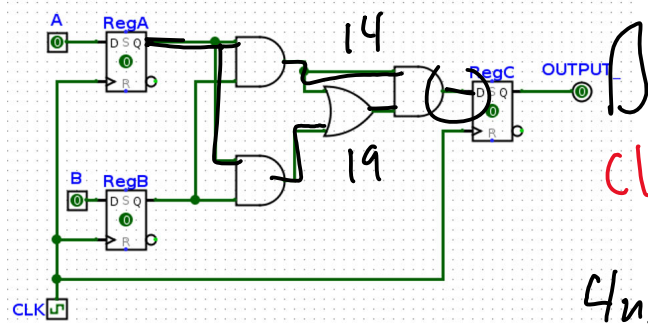


Clock cycle time must be small enough that inputs to registers don't change within the hold time and large enough to account for clk-to-q times, setup times, and combinational logic delays.

- 2.1 For the following 2 circuits, fill out the timing diagram. The clock period (rising edge to rising edge) is 10ns. For every register, clk-to-q delay is 1ns and setup/hold time are 0ns. There is no delay associated with a NOT gate.



- 2.2 In the circuit below, RegA and RegB have setup, hold, and clk-to-q times of 4ns, all logic gates have a delay of 5ns, and RegC has a setup time of 6ns. What is the maximum allowable hold time for RegC? What is the minimum acceptable clock cycle time for this circuit, and clock frequency does it correspond to?



Critical Path: longest time to propagate data

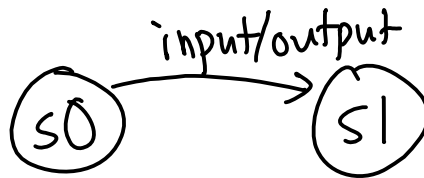
$$clk\text{-to-q} + 3 \cdot \text{delay} + \text{setup} = 4 + 15 + 6 = 25ns$$

$$4ns + 2 \cdot 5 = 14ns$$

clk to q

$$\text{max hold} = 14ns$$

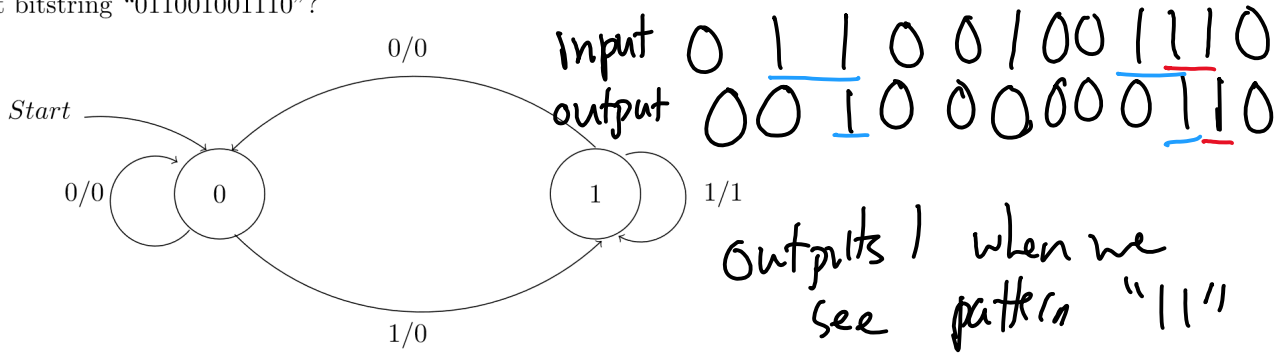
$$\text{freq} = \frac{1}{\text{period}}$$



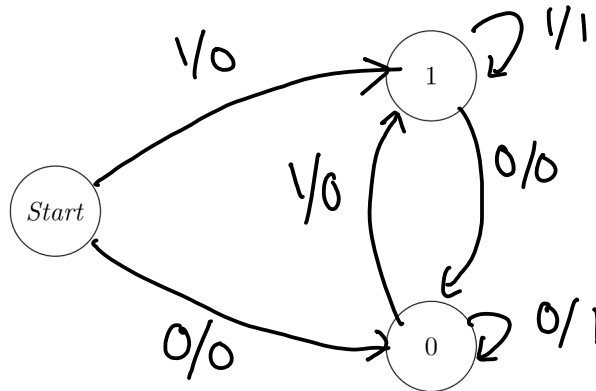
3 Finite State Machines

Automatons are machines that receive input and use various states to produce output. A finite state machine is a type of simple automaton where the next state and output depend only on the current state and input. Each state is represented by a circle, and every proper finite state machine has a starting state, signified either with the label "Start" or a single arrow leading into it. Each transition between states is labeled [input]/[output].

3.1 What pattern in a bitstring does the FSM below detect? What would it output for the input bitstring "011001001110"?



3.2 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.



00 11

Reminder: all states should handle input of 0 & 1

3.3 Write an FSM that will output a 1 if it recognizes the regex pattern {10+1}.