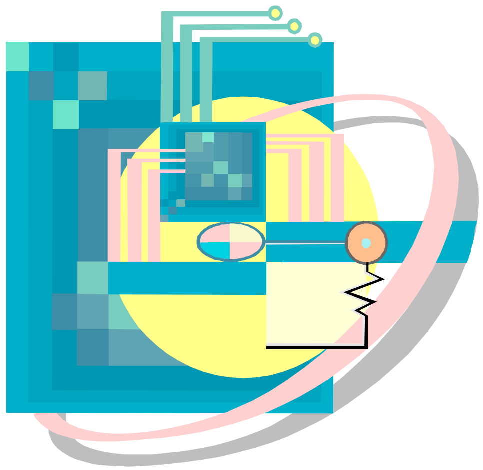
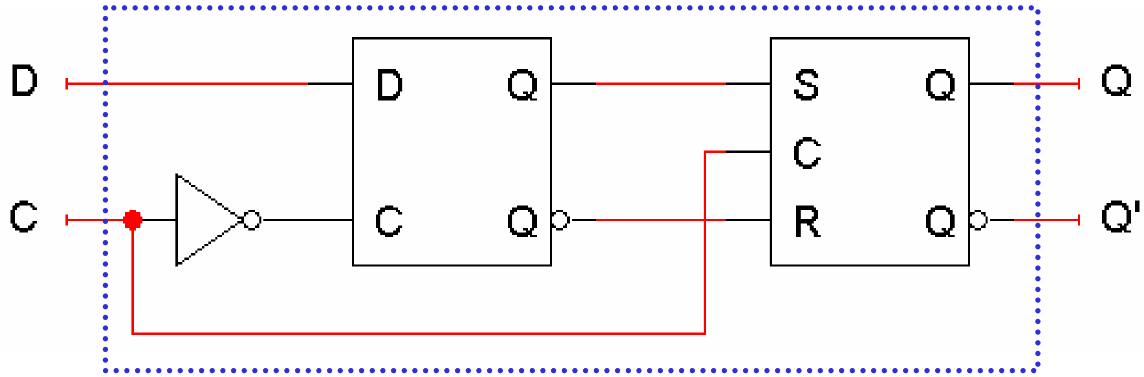
Sequential circuit analysis

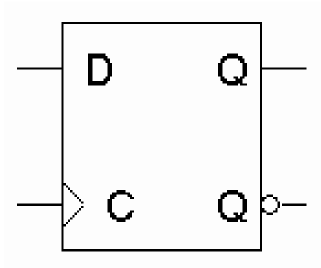


- Last week we started talking about memory.
  - The outputs of a sequential circuit depend on not only the inputs, but also on what's stored in the circuit's memory.
  - Latches and flip-flops are basic one-bit memory units.
- Today we'll finish up our discussion of flip-flops.
  - There are several variations of our basic flip-flop from last week.
  - Understanding the timing of flip-flops is important.
- Then we'll also see some examples of sequential circuits, and learn how to analyze and describe them.

©2000-2003 Howard Huang

A positive edge-triggered D flip-flop



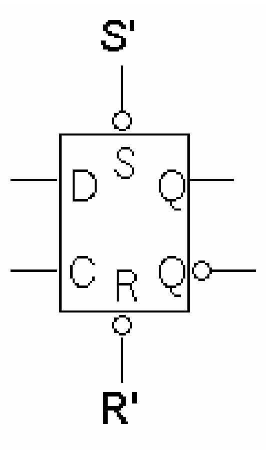


- This positive edge-triggered D flip-flop includes two latches.
  - The flip-flop output Q changes only *after* the positive edge of C.
  - The change is based on the flip-flop input value D that was present at the positive edge of the clock signal.
- A D flip-flop behaves like a D latch except for its positive edge-triggered nature, which is not explicit in the table below.

C	D	Q
0	x	No change
1	0	0 (reset)
1	1	1 (set)

Direct inputs

- Most flip-flops also provide direct inputs, or asynchronous inputs, that let you immediately set or clear the state, regardless of the clock input.
- Such inputs are especially useful for setting the initial state of a flip-flop.
- Here is a LogicWorks D flip-flop with active-low direct inputs.



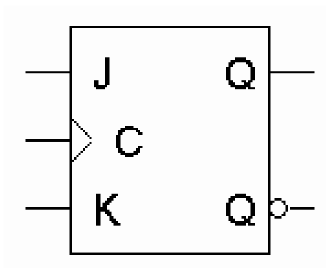
S’	R’	C	D	Q
0	0	x	x	Avoid
0	1	x	x	1 (set)
1	0	x	x	0 (reset)
1	1	0	x	No change
1	1	1	0	0 (reset)
1	1	1	1	1 (set)

Direct inputs set or reset the flip-flop asynchronously

Set S’R’ = 11 for normal operation of the flip-flop

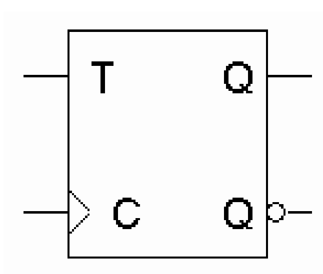
Flip-flop variations

- We can make different versions of flip-flops based on the D flip-flop, just like we made different latches based on the S'R' latch.
- The JK flip-flop has inputs that act like S and R, but JK = 11 *complements* the flip-flop's current state.



C	J	K	Q_{next}
0	x	x	No change
1	0	0	No change
1	0	1	0 (reset)
1	1	0	1 (set)
1	1	1	$Q'_{current}$

- A T flip-flop can only maintain or complement its current state.



C	T	Q_{next}
0	x	No change
1	0	No change
1	1	$Q'_{current}$

Characteristic tables

- The tables that we've made so far are called characteristic tables.
  - They show the next state Q(t+1) in terms of the current state Q(t) and the inputs.
  - For simplicity, the control input C is usually not listed.
  - Again, these tables don't indicate the positive edge-triggered nature of the flip-flops.

D	Q(t+1)	Operation
0	0	Reset
1	1	Set

J	K	Q(t+1)	Operation
0	0	Q(t)	No change
0	1	0	Reset
1	0	1	Set
1	1	Q'(t)	Complement

T	Q(t+1)	Operation
0	Q(t)	No change
1	Q'(t)	Complement

Characteristic equations

- We can also write characteristic equations, where the next state Q(t+1) is defined in terms of the current state Q(t) and the flip-flop inputs.

D	Q(t+1)	Operation
0	0	Reset
1	1	Set

$$Q(t+1) = D$$

J	K	Q(t+1)	Operation
0	0	Q(t)	No change
0	1	0	Reset
1	0	1	Set
1	1	Q'(t)	Complement

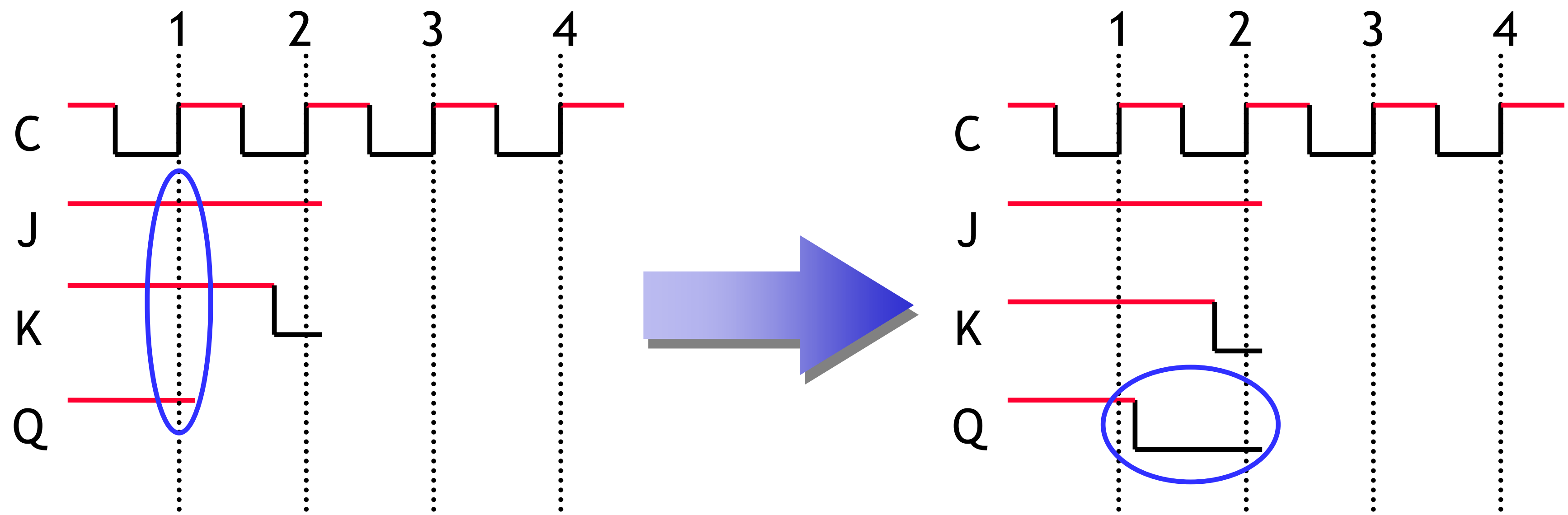
$$Q(t+1) = K'Q(t) + JQ'(t)$$

T	Q(t+1)	Operation
0	Q(t)	No change
1	Q'(t)	Complement

$$\begin{aligned} Q(t+1) &= T'Q(t) + TQ'(t) \\ &= T \oplus Q(t) \end{aligned}$$

Flip-flop timing diagrams

- "Present state" and "next state" are relative terms.
- In the example JK flip-flop timing diagram on the left, you can see that at the first positive clock edge, JK = 11 and Q(1) = 1.
- We can use this information to find the next state, Q(2) = Q(1)'.
- Q(2) changes right after the positive clock edge, as shown to the right. The flip-flop will not change again until the next positive clock edge.

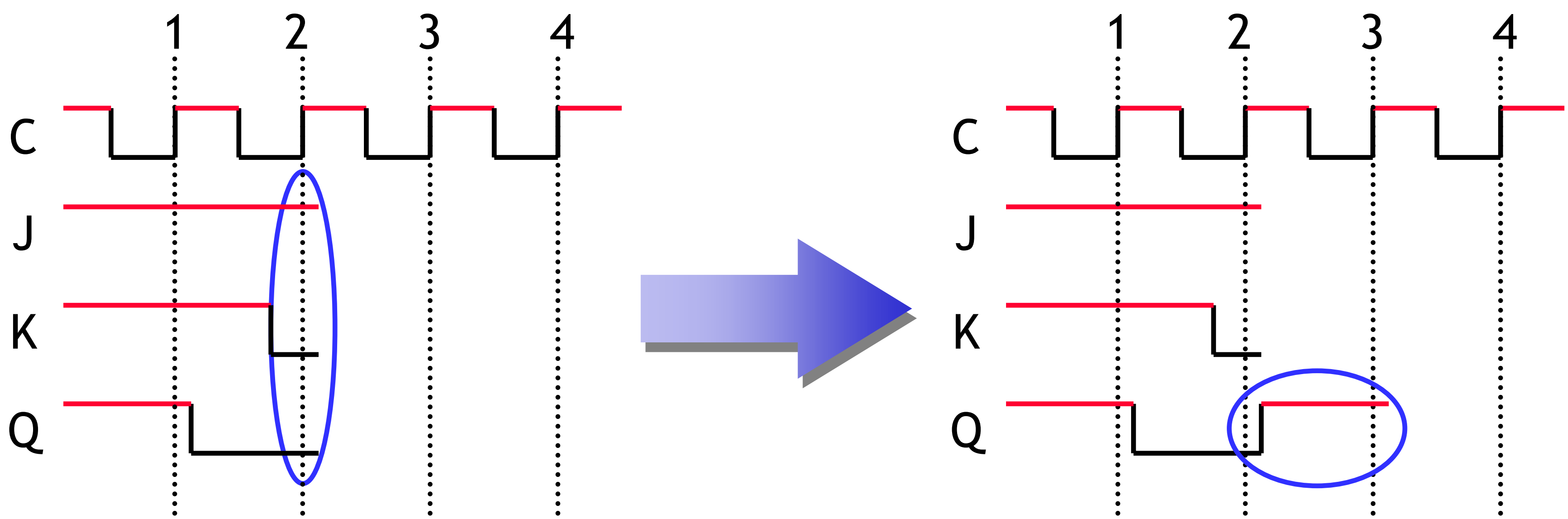


These values at clock cycle 1...

… determine the "next" Q

“Present” and “next” are relative

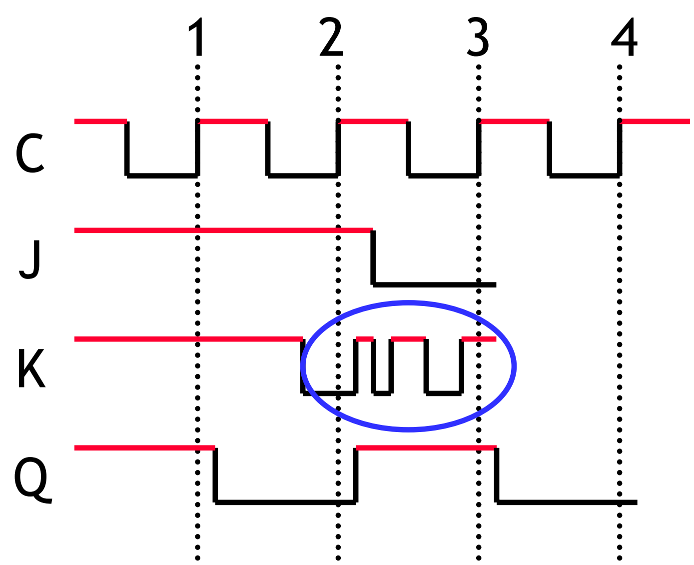
- Similarly, the values of J, K and Q at the second positive clock edge can be used to find the value of Q during the third clock cycle.
- When we do this, Q(2) is now referred to as the “present” state, and Q(3) becomes the “next” state.



These values at clock cycle 2... … determine the “next” Q

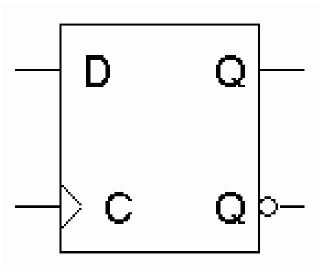
Positive edge triggered

- One final point to repeat again one more time: the flip-flop outputs are affected only by the input values *at the positive clock edge*.
  - In the diagram below, K changes rapidly between the second and third positive edges.
  - But only the inputs at the third positive clock edge (JK = 01 and Q = 1) will affect the next state. Here, this means Q changes to 0.
- This is a fairly simple timing model. In real life there are also "setup" and and "hold" times that account for various propagation delays.



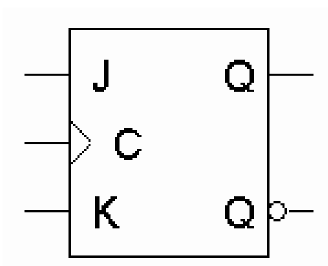
Flip-flop review

Flip-flops	Characteristic tables	Characteristic equations



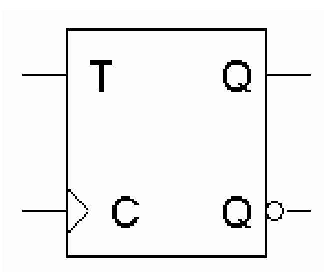
D	Q(t+1)	Operation
0	0	Reset
1	1	Set

$$Q(t+1) = D$$



J	K	Q(t+1)	Operation
0	0	Q(t)	No change
0	1	0	Reset
1	0	1	Set
1	1	Q'(t)	Complement

$$Q(t+1) = K'Q(t) + JQ'(t)$$

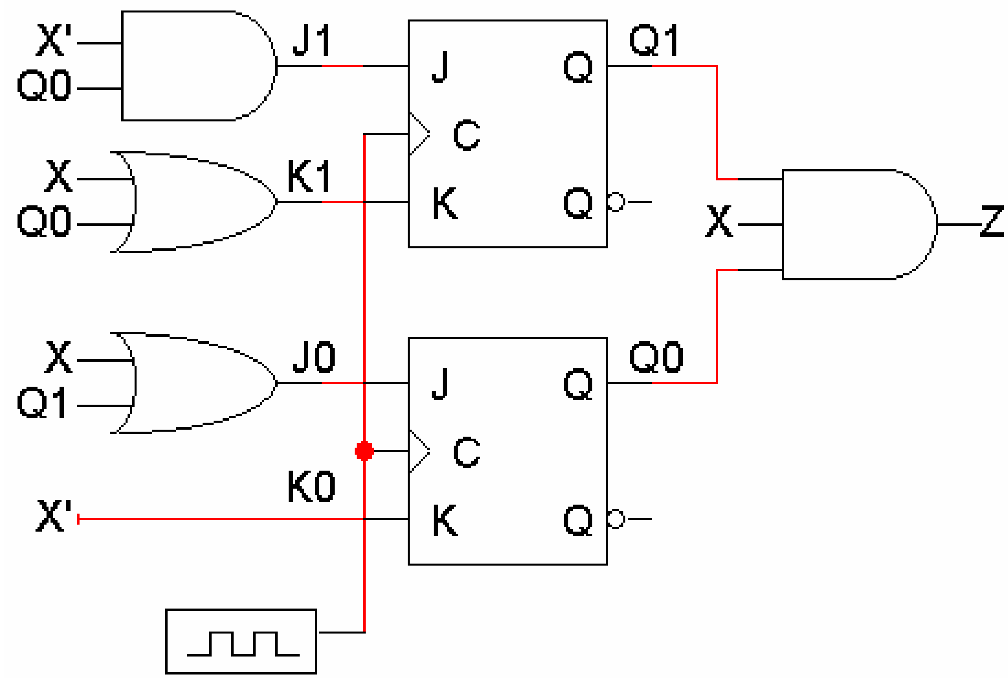


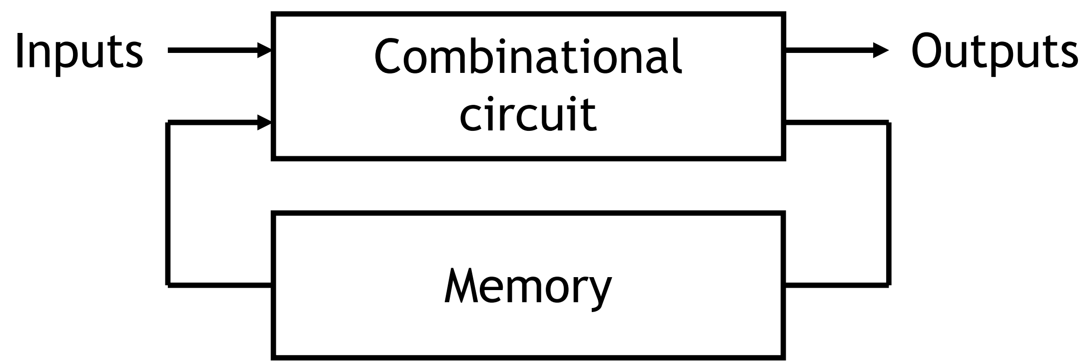
T	Q(t+1)	Operation
0	Q(t)	No change
1	Q'(t)	Complement

$$Q(t+1) = T \oplus Q(t)$$

What do sequential circuits look like?

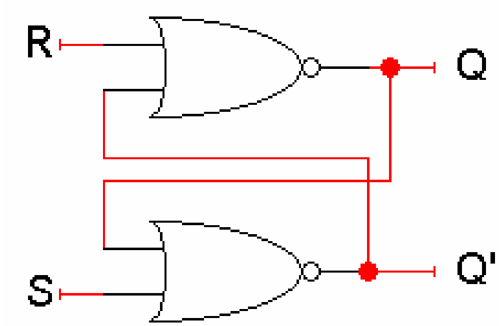
- Here is a sequential circuit with two JK flip-flops. There is one input X and one output Z.
- The values of the flip-flops (Q_1Q_0) form the state, or the memory, of the circuit.
- The flip-flop outputs also go back into the primitive gates on the left. This matches the abstract sequential circuit diagram at the bottom.





How do you analyze a sequential circuit?

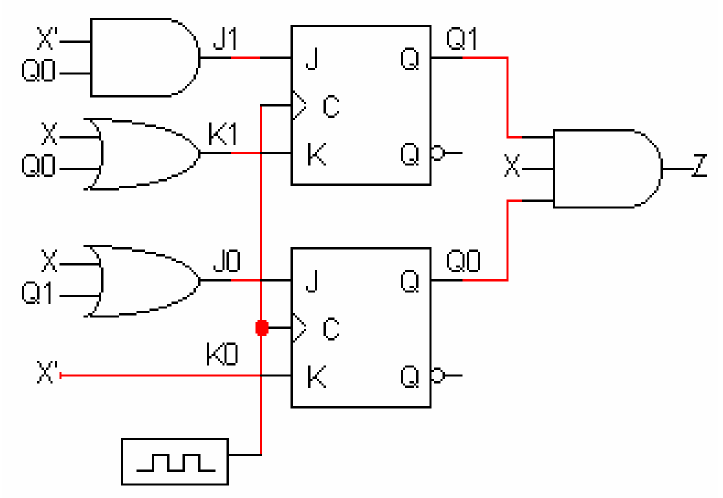
- We can analyze a combinational circuit by deriving a truth table, which shows how the circuit outputs are generated from its inputs.
- But in a sequential circuit, the outputs are dependent upon not only the inputs, but also the current state of the flip-flops. So to understand how a sequential circuit works, we have to know how the memory changes.
- A state table is the sequential analog of a truth table. It shows inputs *and* current states on the left, and outputs *and* next states on the right.
- We saw an example of a state table last week, for an SR latch.



Inputs		Current		Next	
S	R	Q	Q’	Q	Q’
0	0	0	1	0	1
0	0	1	0	1	0
0	1	0	1	0	1
0	1	1	0	0	1
1	0	0	1	1	0
1	0	1	0	1	0

Analyzing our example circuit

- A state table for our example circuit is shown below.
- The present state Q_1Q_0 and the input X will determine the next state Q_1Q_0 and the output Z.



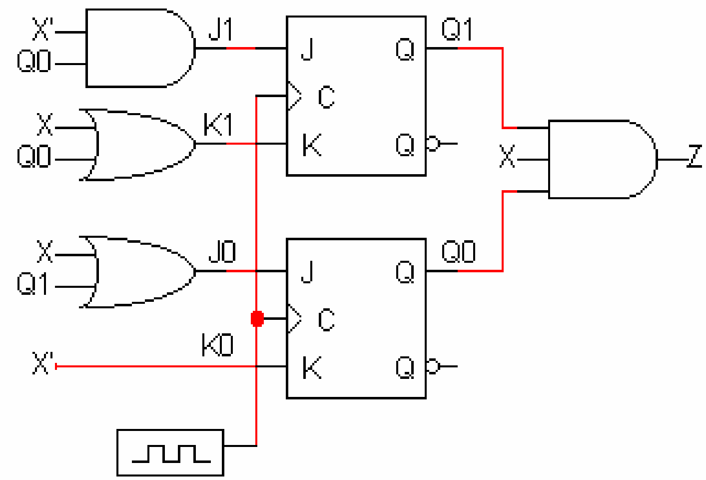
Present State Q_1	Present State Q_0	Inputs X	Next State Q_1	Next State Q_0	Outputs Z
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

The outputs are easy

- From the diagram, you can see that

$$Z = Q_1Q_0X$$

- This is an example of a Mealy machine, where the output depends on both the present state (Q_1Q_0) and the input (X).



Present State		Inputs	Next State		Outputs
Q_1	Q_0	X	Q_1	Q_0	Z
0	0	0			0
0	0	1			0
0	1	0			0
0	1	1			0
1	0	0			0
1	0	1			0
1	1	0			0
1	1	1			1

Flip-flop input equations

- Finding the next states is harder. To do this, we have to figure out how the flip-flops are changing.

1. Find Boolean expressions for the flip-flop inputs.
2. Use these expressions to find the actual flip-flop input values for each possible combination of present states and inputs.
3. Use flip-flop characteristic tables or equations to find the next states, based on the flip-flop input values and the present states.

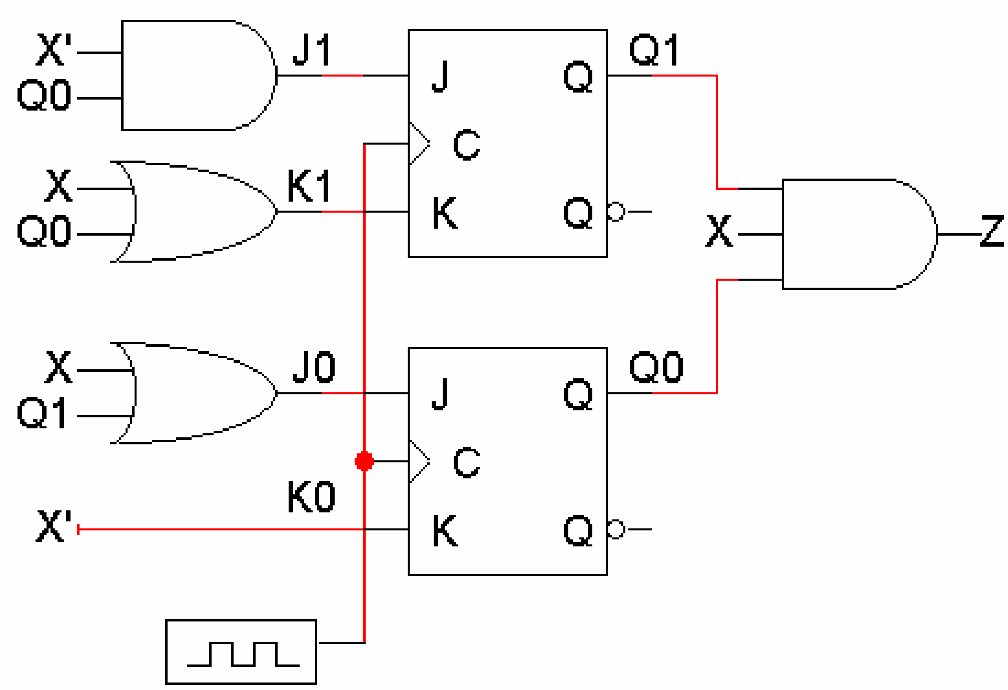
Step 1: Flip-flop input equations

- For our example, the flip-flop input equations are:

$$J_1 = X' \, Q_0$$
$$K_1 = X + Q_0$$

$$J_0 = X + Q_1$$
$$K_0 = X'$$

- JK flip-flops each have *two* inputs, J and K. (D and T flip-flops have one input each.)



Step 2: Flip-flop input values

- With these equations, we can make a table showing J_1, K_1, J_0 and K_0 for the different combinations of present state Q_1Q_0 and input X.

$J_1 = X' Q_0$ $\qquad$ $J_0 = X + Q_1$

$K_1 = X + Q_0$ $\qquad$ $K_0 = X'$

Present State		Inputs	Flip-flop Inputs			
Q_1	Q_0	X	J_1	K_1	J_0	K_0
0	0	0	0	0	0	1
0	0	1	0	1	1	0
0	1	0	1	1	0	1
0	1	1	0	1	1	0
1	0	0	0	0	1	1
1	0	1	0	1	1	0
1	1	0	1	1	1	1
1	1	1	0	1	1	0

Step 3: Find the next states

- Finally, use the JK flip-flop characteristic tables or equations to find the next state of *each* flip-flop, based on its present state and inputs.
- The general JK flip-flop characteristic equation was given earlier today.

$$Q(t+1) = K'Q(t) + JQ'(t)$$

- In our example circuit, we have two JK flip-flops, so we have to apply this equation to *each* of them.

$$Q_1(t+1) = K_1'Q_1(t) + J_1Q_1'(t)$$
$$Q_0(t+1) = K_0'Q_0(t) + J_0Q_0'(t)$$

Step 3 concluded

- Finally, here are the next states for Q_1 and Q_0, using these equations.

$$Q_1(t+1) = K_1'Q_1(t) + J_1Q_1'(t)$$
$$Q_0(t+1) = K_0'Q_0(t) + J_0Q_0'(t)$$

Present State		Inputs	Flip-flop Inputs				Next State	
Q_1	Q_0	X	J_1	K_1	J_0	K_0	Q_1	Q_0
0	0	0	0	0	0	1	0	0
0	0	1	0	1	1	0	0	1
0	1	0	1	1	0	1	1	0
0	1	1	0	1	1	0	0	1
1	0	0	0	0	1	1	1	1
1	0	1	0	1	1	0	0	1
1	1	0	1	1	1	1	0	0
1	1	1	0	1	1	0	0	1

Getting the state table columns straight

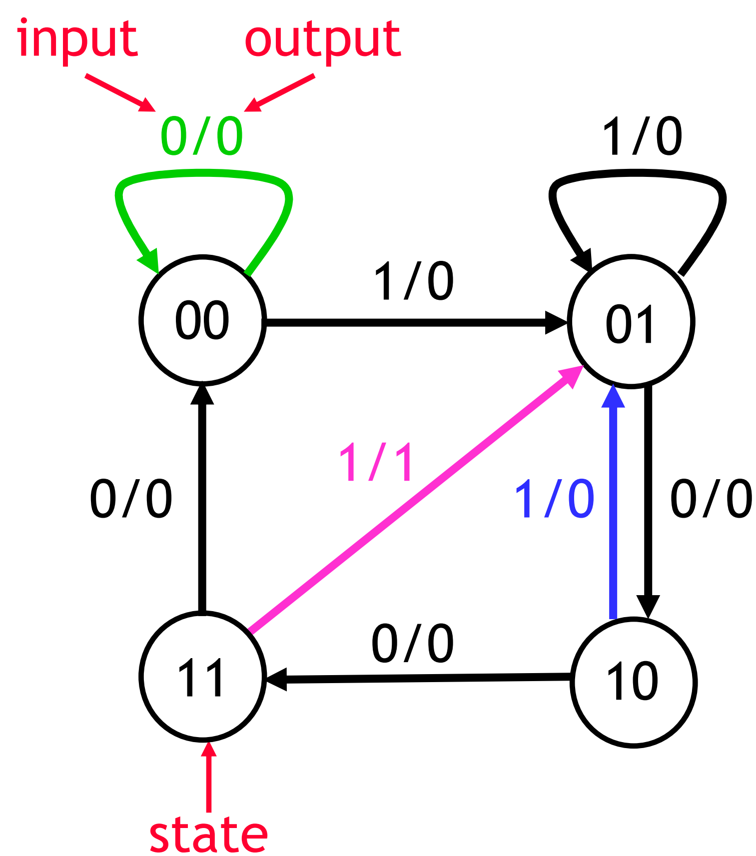
- The table starts with Present State and Inputs.
  - Present State and Inputs yield FF Inputs.
  - Present State and FF Inputs yield Next State, based on the flip-flop characteristic tables.
  - Present State and Inputs yield Output.
- We really only care about FF Inputs in order to find Next State.

Present State		Inputs	Flip-flop Inputs				Next State		Output
Q_1	Q_0	X	J_1	K_1	J_0	K_0	Q_1	Q_0	Z
0	0	0	0	0	0	1	0	0	0
0	0	1	0	1	1	0	0	1	0
0	1	0	1	1	0	1	1	0	0
0	1	1	0	1	1	0	0	1	0
1	0	0	0	0	1	1	1	1	0
1	0	1	0	1	1	0	0	1	0
1	1	0	1	1	1	1	0	0	0
1	1	1	0	1	1	0	0	1	1

State diagrams

- We can also represent the state table graphically with a state diagram.
  - The diagram has one node for each possible state.
  - Arrows in the diagram connect present states to next states, and are labelled with "input/output."
- A diagram corresponding to our example state table is shown below.

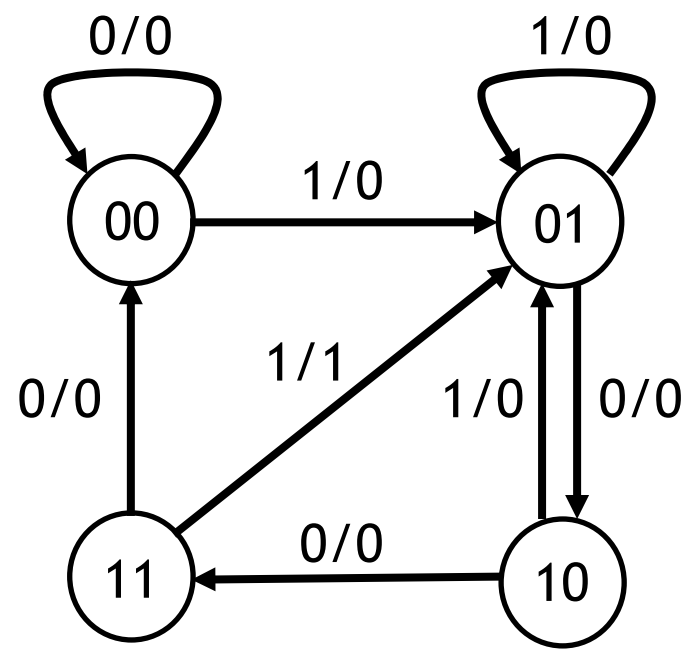
Present State Q_1	Present State Q_0	Inputs X	Next State Q_1	Next State Q_0	Output Z
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	0	1	0
1	1	0	0	0	0
1	1	1	0	1	1



Size of the state diagram

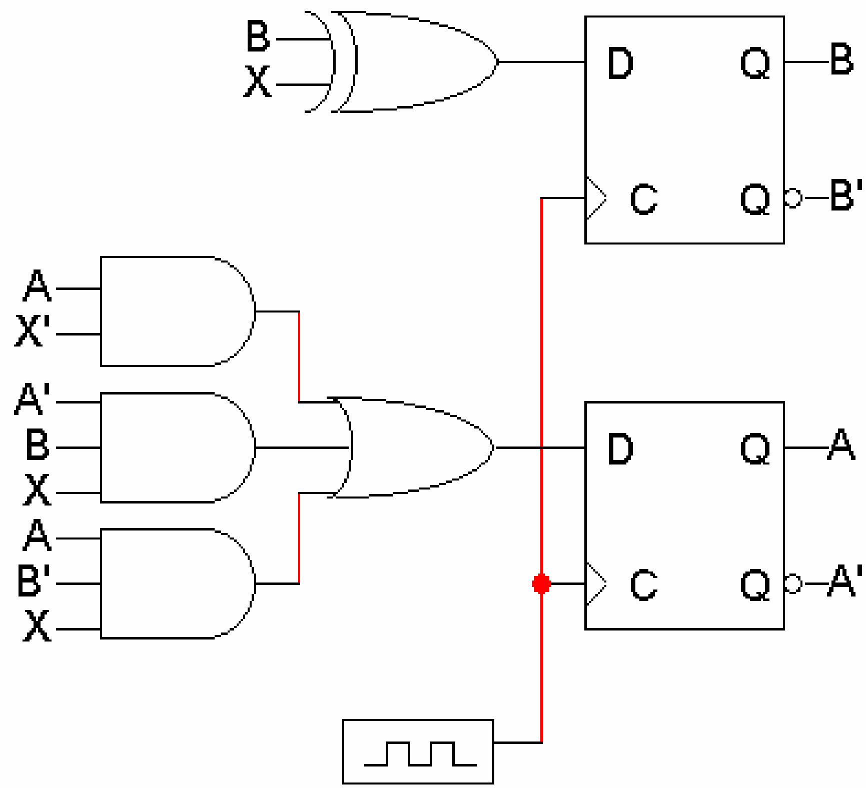
- Always check the size of your state diagrams.
  - If there are n flip-flops, there should be 2^n nodes in the diagram.
  - If there are m inputs, then each node will have 2^m outgoing arrows.
- Our example circuit has two flip-flops and one input, so the state diagram should have four nodes, each with two outgoing arrows.

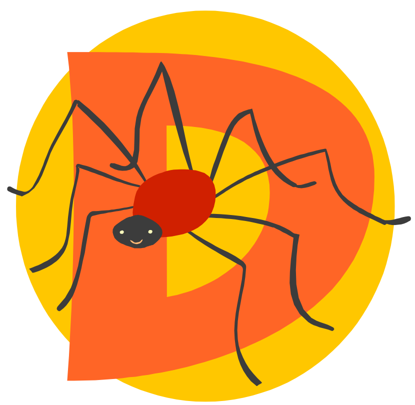
Present State Q_1	Q_0	Inputs X	Next State Q_1	Q_0	Output Z
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	0	1	0
1	1	0	0	0	0
1	1	1	0	1	1



A D flip-flop example

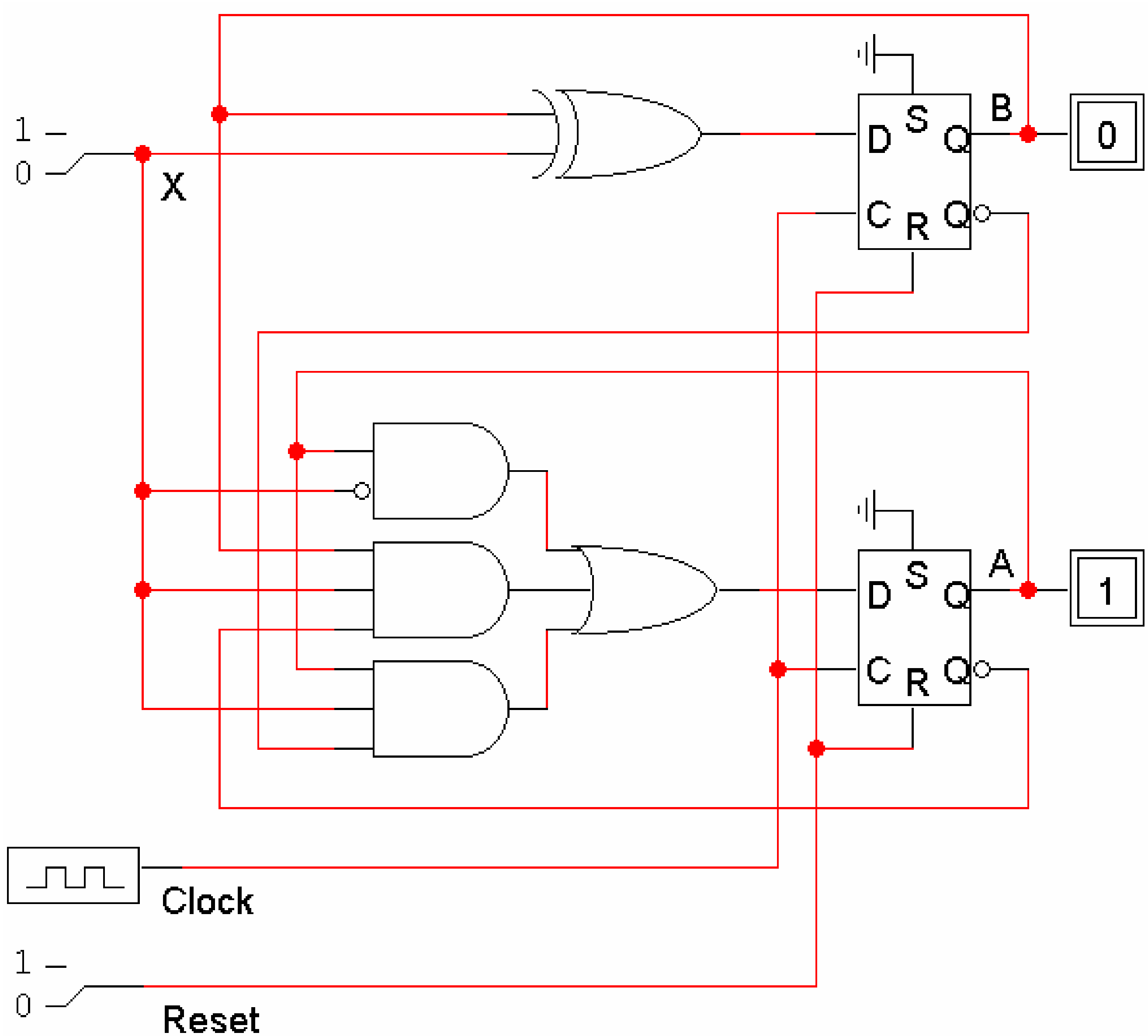
- Here are two D flip-flops, with values labelled A and B.
- There is one input, X.
- There are no explicit outputs.
  - In this case, the outputs are assumed to be the flip-flop values A and B themselves.
  - This is an example of a Moore machine, where the outputs depend on only the present state.





Making this in LogicWorks

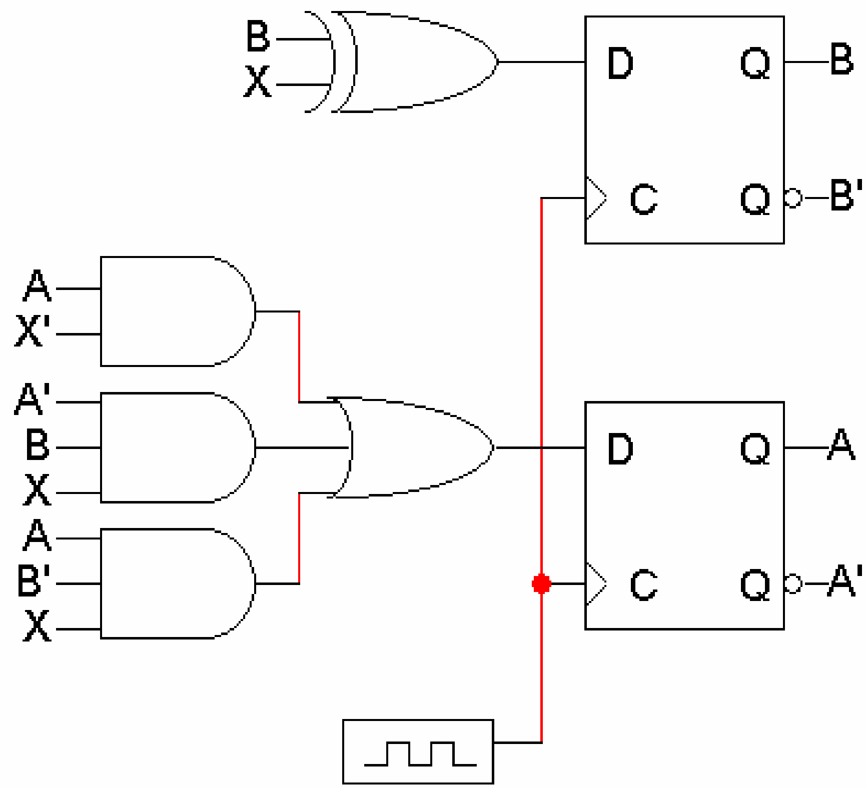
- You'll need to set the direct inputs R and S. Here we've added a Reset switch; when it is 1, both flip-flops are cleared to 0.
- The flip-flops have complemented outputs already, so you won't need to generate A' and B' yourself.



Analyzing the example circuit

- The basic state table is below.
- Again, you can see that the present states are being used to generate the next states.
- For this example, remember that the present state also serves as the output.

Present State A	Present State B	Inputs X	Next State A	Next State B
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

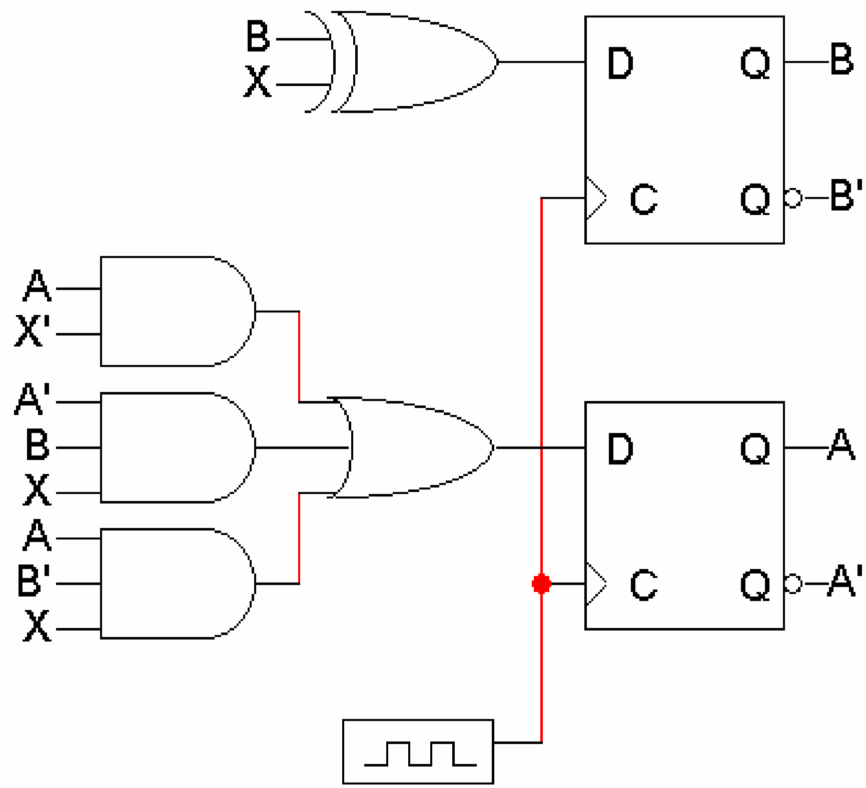


Step 1: Flip-flop input equations

- There are two input equations, one for each flip-flop.

$$D_A = AX' + A'BX + AB'X$$
$$D_B = B \oplus X$$

- "D_A" indicates a D-type flip-flop, whose output is A.



Step 2: Flip-flop input values

- Now that we have the equations for D_A and D_B, we can fill in actual values for each combination of present state and inputs.

$$D_A = AX' + A'BX + AB'X$$

$$D_B = B \oplus X$$

Present State		Inputs	Flip-flop Inputs	
A	B	X	D_A	D_B
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	1	0
1	0	1	1	1
1	1	0	1	1
1	1	1	0	0

Step 3: Find the next states

- Finally, use the D flip-flop characteristic equation to find the next state of each flip-flop, based on its present state and its inputs.
- D flip-flops are simple because the next state is the same as the D input, *regardless* of the present state.

$$Q(t+1) = D$$

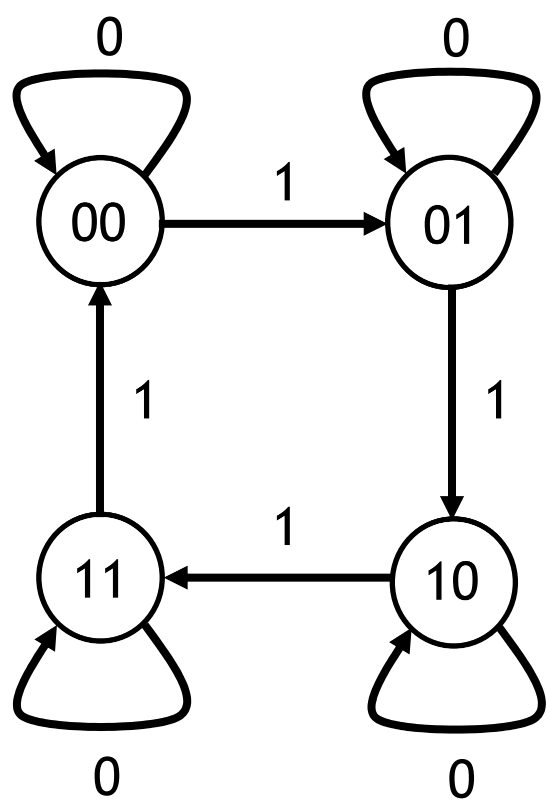
- People often don't even bother writing the "Flip-flop Inputs" columns.

Present State		Inputs	Flip-flop Inputs		Next State	
A	B	X	D_A	D_B	A	B
0	0	0	0	0	0	0
0	0	1	0	1	0	1
0	1	0	0	1	0	1
0	1	1	1	0	1	0
1	0	0	1	0	1	0
1	0	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	0	0	0	0

So what does this circuit do?

- When X = 0, the next state is the same as the present state.
- When X = 1, the next state is "one more" than the present state.
- This is a basic two-bit counter with an enable input, X. It's also called a modulo-4 counter, since it counts from 0 to 3 repeatedly.

Present State		Inputs	Next State	
A	B	X	A	B
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	1	0
1	0	1	1	1
1	1	0	1	1
1	1	1	0	0



Summary

- To analyze sequential circuits, you have to understand how the flip-flops change on each clock cycle, according to their current values and inputs.
- A state table show all the possible ways that the outputs and state of a sequential circuit can change, based on the its inputs and present state.
- State diagrams are an alternative way of showing the same information.
- Next time we'll look at designing sequential circuits. This is the opposite process—you make a state table and/or diagram first, and then turn that into a sequential circuit.