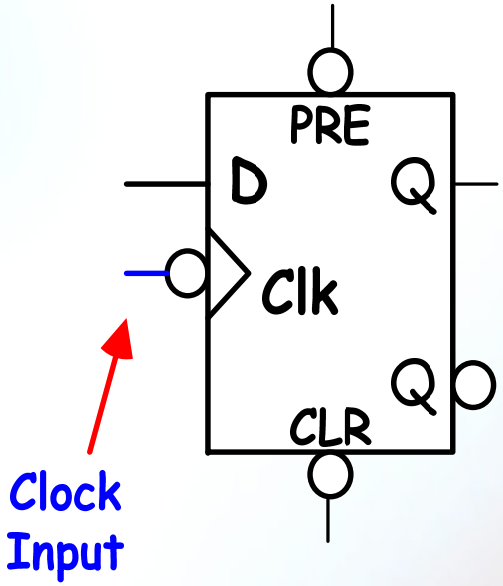
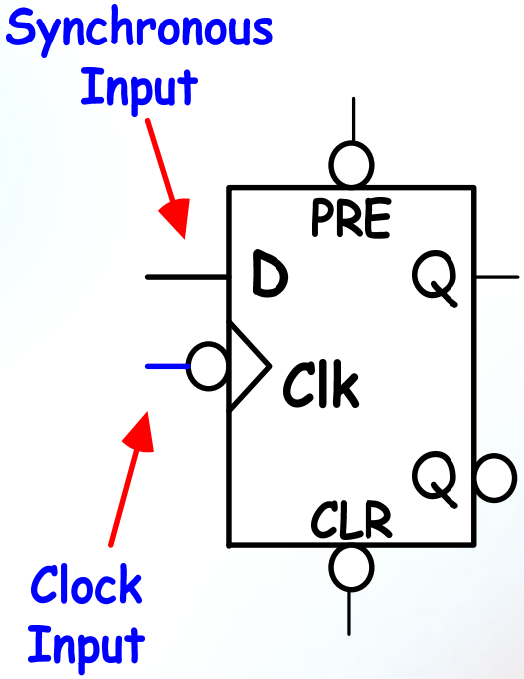
# EET 310
# Flip-Flops

# FF's and some Definitions

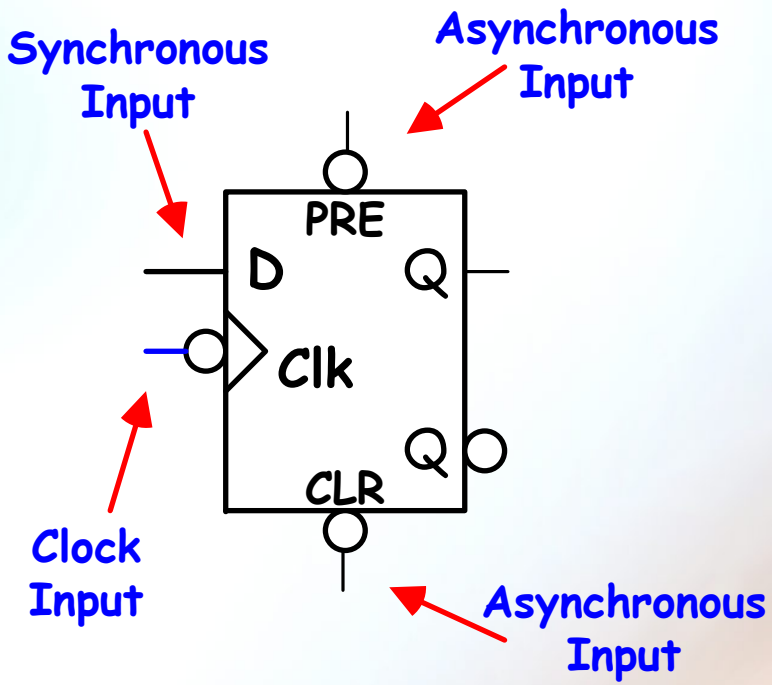



**Clock Input:** FF's are controlled by a **trigger** or **Clock signal**. All **FF**'s have a clock input. If a device which attempts to do a **FF**'s task does not have a Clock then it is a LATCH, not a FF.

# FF's and some Definitions




**Synchronous Input:** Controls the device **based on the timing from a clock signal.** When the clocking signal occurs, the **synchronous input** is allowed to affect the output. **If the signal does not occur the synchronous input can not affect anything.**




**Asynchronous Input:** Affects the output without waiting for a clocking signal. It IGNORES the clock.

# Pre and Clr Inputs

**'PRE' Input**

The 'PRE' input will cause the Q output to SET (go HIGH) when a '0' is applied to the PRE input. This occurs ASYNCHRONOUSLY (without the clock).

# Pre and Clr Inputs

**'PRE' Input**

The 'PRE' input will cause the Q output to SET (go HIGH) when a '0' is applied to the PRE input. This occurs ASYNCHRONOUSLY (without the clock).

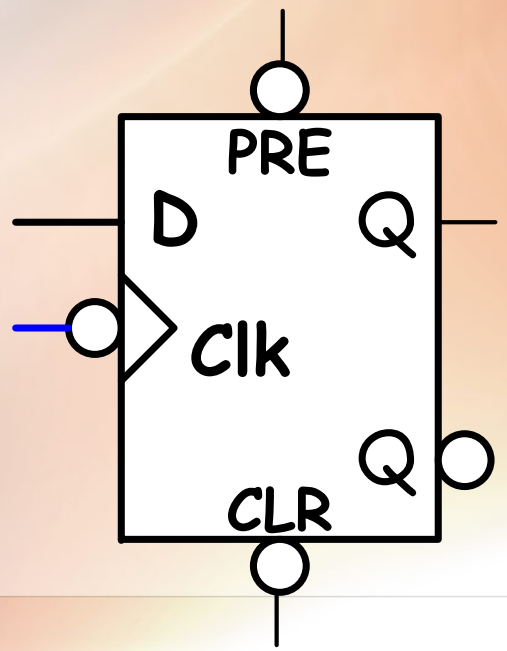
**'CLR' Input**

The 'CLR' input will cause the Q output to RESET (go LOW) when a '0' is applied to the 'CLR' input. This occurs ASYNCHRONOUSLY (without the clock).

# The PRE and CLR inputs

**ACTIVE HIGH or LOW?**

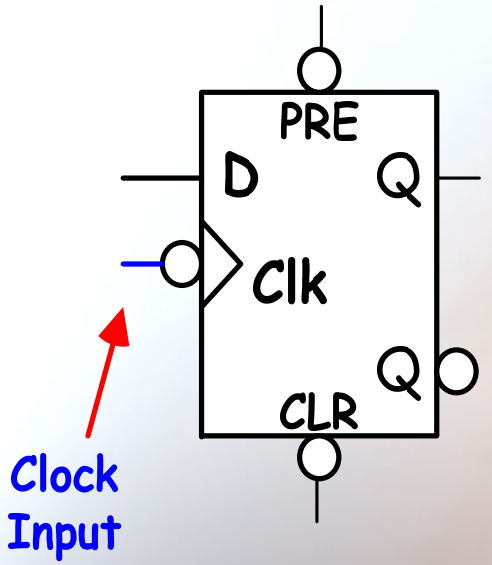
The activity level of an input or output is determined by the existence of an inverting bubble on the terminal or not. Both the **PRE** and the **CLR** inputs below are **ACTIVE LOW** inputs. This means that a '0' is required on the input in order for the input to affect the output.



# The CLOCK

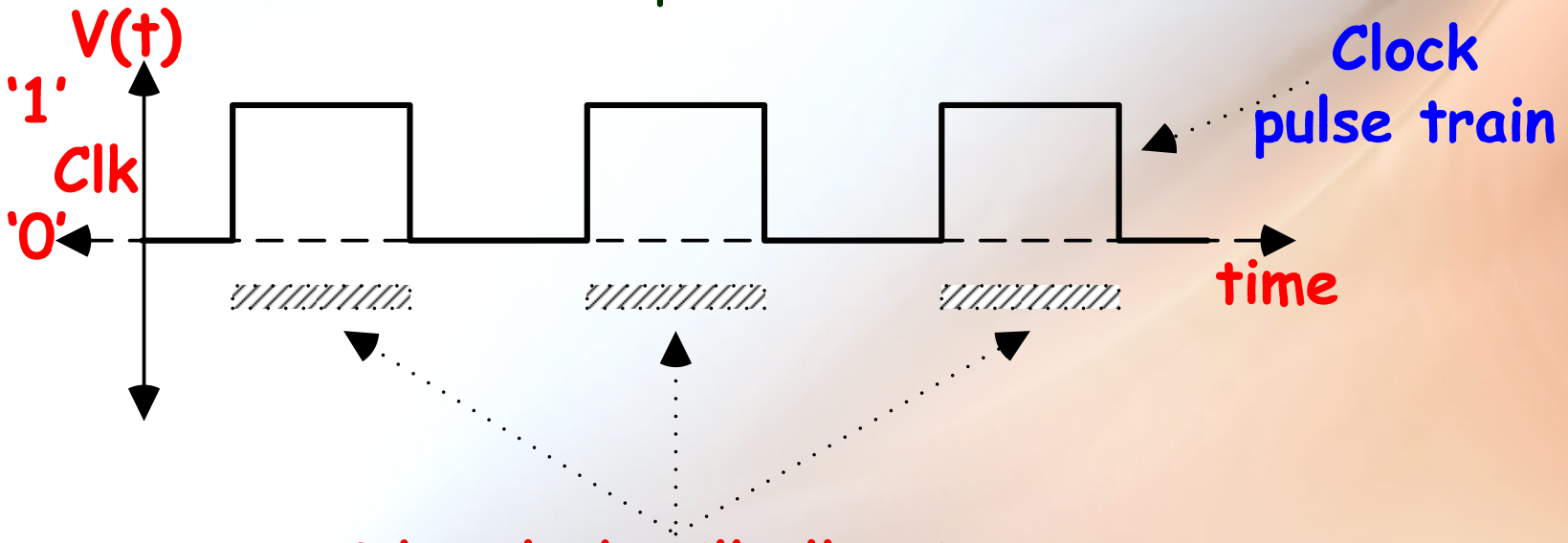
There are three types of clock inputs:

- Level Triggered (not used often)
- Leading Edge Triggered
- Trailing Edge Triggered (pictured below)



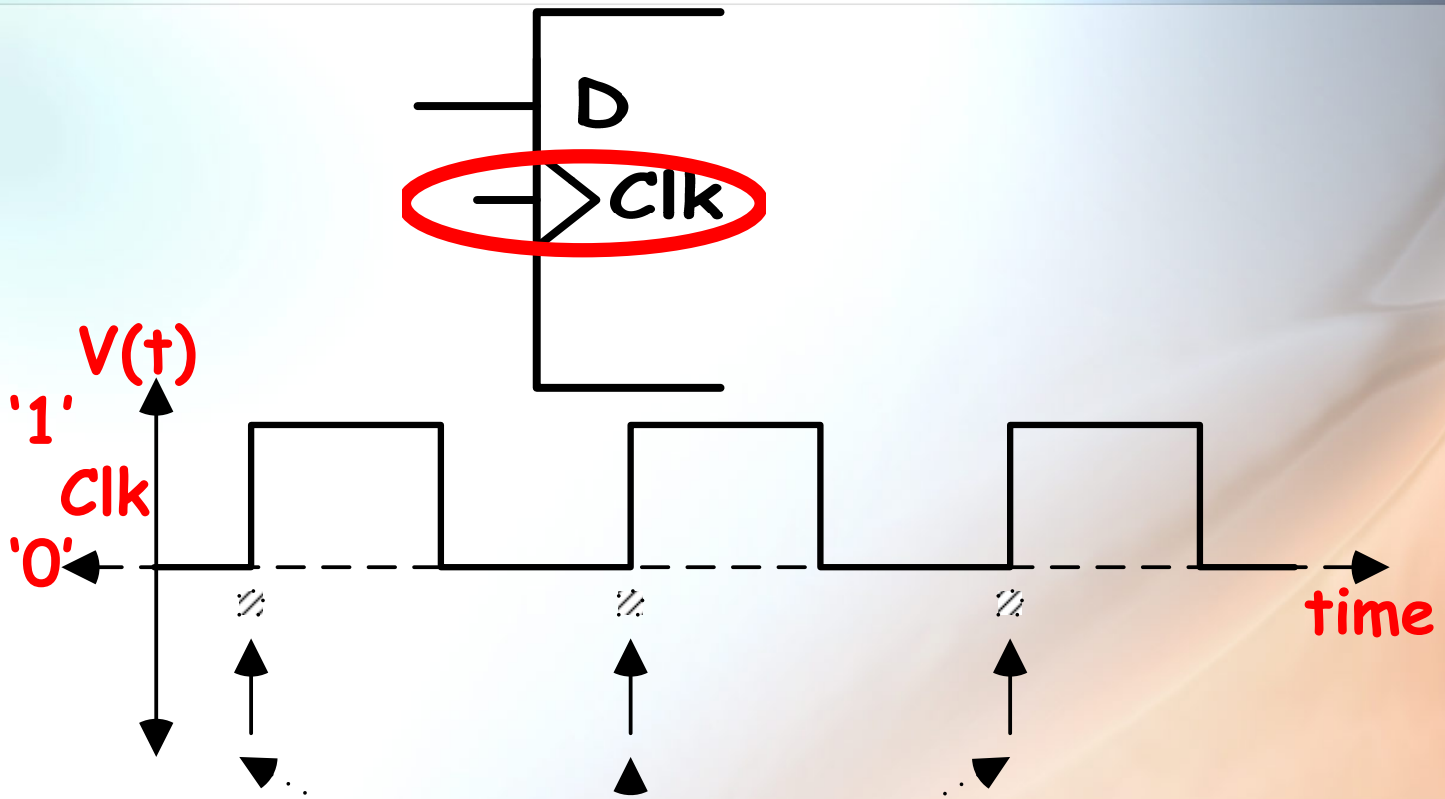
# Level Triggering


The problem with this type of triggering is that the input could change millions of times during each 'Level" period, thus affecting the output millions of times.



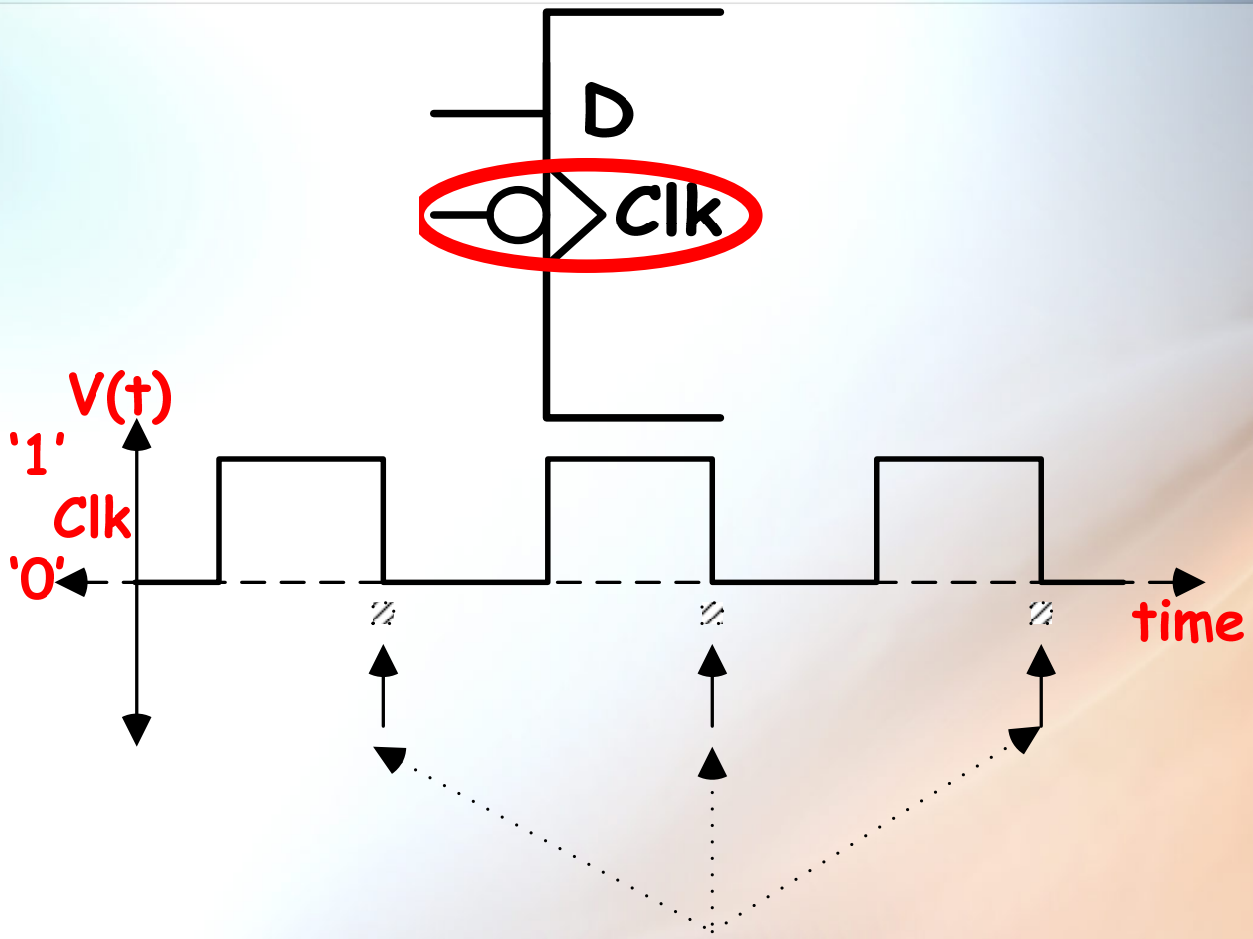

The clock will allow input signals to affect the output during these time periods.

# Leading Edge Triggering



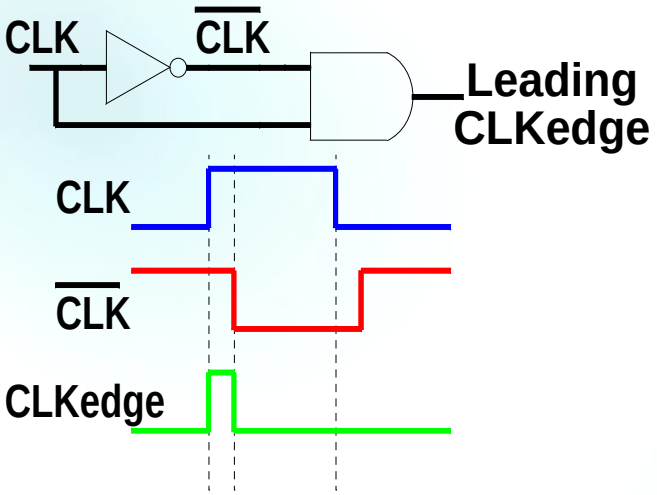

The "Leading" edge of each pulse will allow "one and only one" input event for each edge to affect the output.

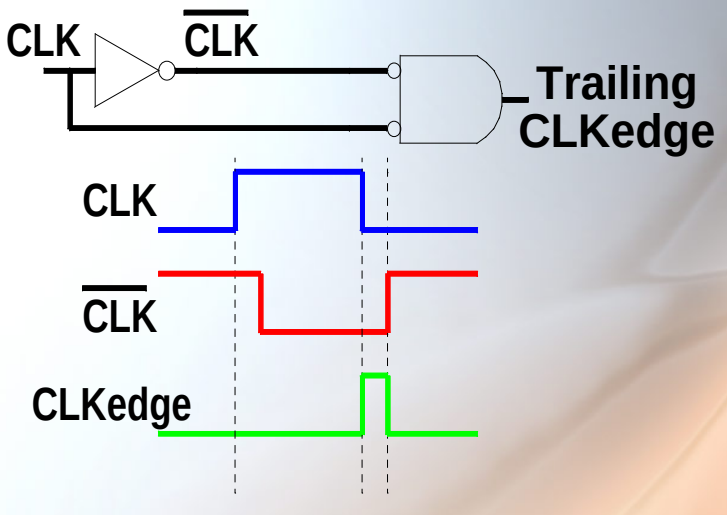
# Trailing Edge Triggering




The "Trailing" edge of each pulse will allow "one and only one" input event for each edge to affect the output.

# How the edge trigger is accomplished

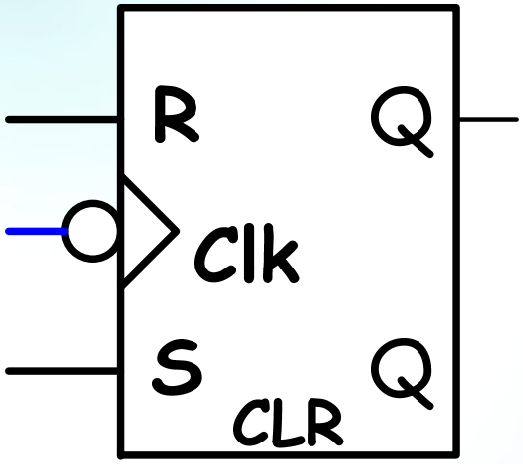



These circuits work by using the fact that there is no such thing as an "ideal" gate. Instead, there exists a delay from the input and the output. The gate on the far right is a "**negative logic**" representation of a **NOR** gate.

# The RS FF

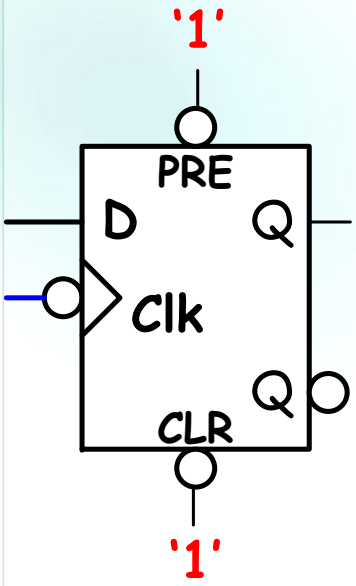
- The RS FF is not seen much anymore except on tests and in textbooks.




- The S=1,R=1 condition should never be allowed to happen.

| S | R | Q | $Q^+$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | Never |
| 1 | 1 | 1 | Never |

# The 'D' Flip-flop




**The 'D' Flip-flop output will take on the value of the 'D' input.** So, if there is a logic '1' on the 'D' input when the trigger event occurs, the '1' will be transferred to the output until the next trigger event. At that point, if 'D' changes to a logic '0', so will the Q output.

# Present State and Next State

- **$Q_P$ = Present state:**

  The state of the output before the clock signal.

- **$Q_N$ = $Q^+$ = Next state:**

  The state the output will attain based on the flip-flop synchronous inputs **after the clock signal occurs. It is what will happen in the FUTURE!**

**Some books use $Q_n$ for present state and $Q_{n+1}$ for next state.**

# D FF Transitions

| D | $Q_p$ | $Q_N$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | |
| 1 | 0 | |
| 1 | 1 | |

When D is '0' and the present state, $Q_p$, is '0', then the next state , $Q_n$, is '0'.

# D FF Transitions

| D | $Q_p$ | $Q_N$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | |
| 1 | 1 | |

When D is '0' and $Q_p$, is '1' then $Q_n$ is '0'.

# D FF Transitions

| D | $Q_p$ | $Q_N$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | |

When D is '1', and $Q_p$ is '0', then $Q_n$, is '1'.

# D FF Transitions

| D | $Q_p$ | $Q_N$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

When D is '1' and $Q_p$, is '1', then $Q_n$, is '1'.

# D FF wrap up

We note that whenever the **D input** is a '**0**', the output will be '**Reset**' no matter what the present state is.

| D | $Q_p$ | $Q_N$ | |
|---|---|---|---|
| 0 | 0 | 0 | } Reset |
| 0 | 1 | 0 | |
| 1 | 0 | 1 | |
| 1 | 1 | 1 | |

# D FF wrap up

Whenever the **D input** is a '**1**', the output will be '**Set**', no matter what the present state is.

| D | $Q_p$ | $Q_N$ | |
|---|---|---|---|
| 0 | 0 | 0 | Reset |
| 0 | 1 | 0 | |
| 1 | 0 | 1 | Set |
| 1 | 1 | 1 | |

# D FF wrap up

| D | $Q_p$ | $Q_N$ | |
|---|---|---|---|
| 0 | 0 | 0 | Reset |
| 0 | 1 | 0 | |
| 1 | 0 | 1 | Set |
| 1 | 1 | 1 | |

The main thing to note here is that the NEXT STATE column is always the same as the D column in a D-FF

# D FF wrap up

| D | $Q_p$ | $Q_N$ | |
|---|---|---|---|
| 0 | 0 | 0 | } Reset |
| 0 | 1 | 0 | |
| 1 | 0 | 1 | } Set |
| 1 | 1 | 1 | |

The D FF performs two basic named functions:
**Data:** The **D-ff** is one of the most basic memory cells. Data placed on the input "**D**" is transferred to the output "**Q**" and "stored" there until it is replaced during the next clock active period.

# D FF wrap up

| D | $Q_p$ | $Q_N$ | |
|---|---|---|---|
| 0 | 0 | 0 | } Reset |
| 0 | 1 | 0 | |
| 1 | 0 | 1 | } Set |
| 1 | 1 | 1 | |

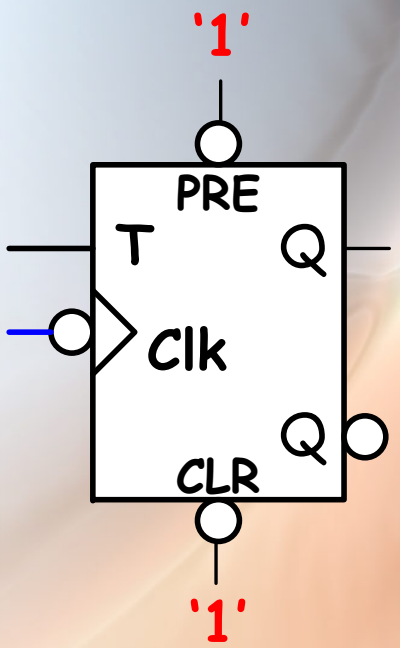
The D FF performs two basic named functions:

**Data:** The **D-ff** is one of the most basic memory cells. Data placed on the input "**D**" is transferred to the output "**Q**" and "stored" there until it is replaced during the next clock active period.

**Delay:** Data placed on the input "**D**" is delayed in its transition to the output "**Q**" until the clocks active period occurs.

# The 'T' Flip-flop Transitions

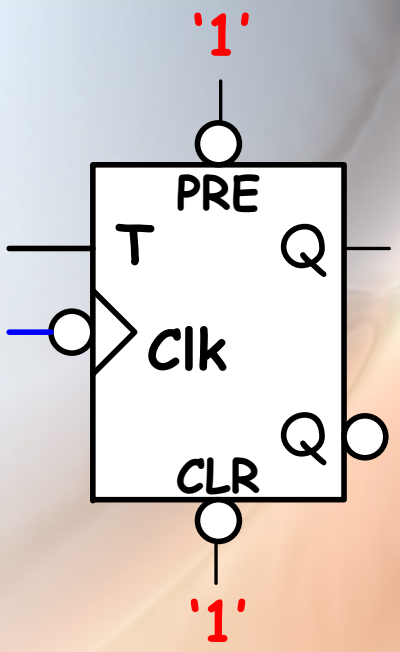
| T | $Q_p$ | $Q_N$ | |
|---|---|---|---|
| 0 | 0 | 0 | } Hold |
| 0 | 1 | 1 | |
| 1 | 0 | 1 | |
| 1 | 1 | 0 | |




For a '**T**' FF, whenever the **T** input is '**0**', the output will be in a '**Hold**' condition in which whatever $Q_p$ is will be held there until the next trigger or clock event.

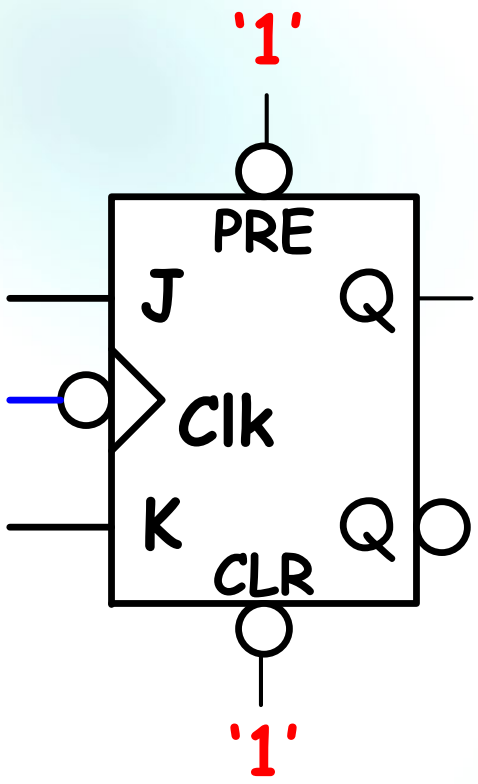
# The 'T' Flip-flop Transitions

| T | $Q_p$ | $Q_N$ | |
|---|---|---|---|
| 0 | 0 | 0 | } Hold |
| 0 | 1 | 1 | |
| 1 | 0 | 1 | } Toggle |
| 1 | 1 | 0 | |




Whenever the T input is '1'. the output will be in a "Toggle' condition where if $Q_p$ is '0' the output will toggle to a '1' and vice versa.

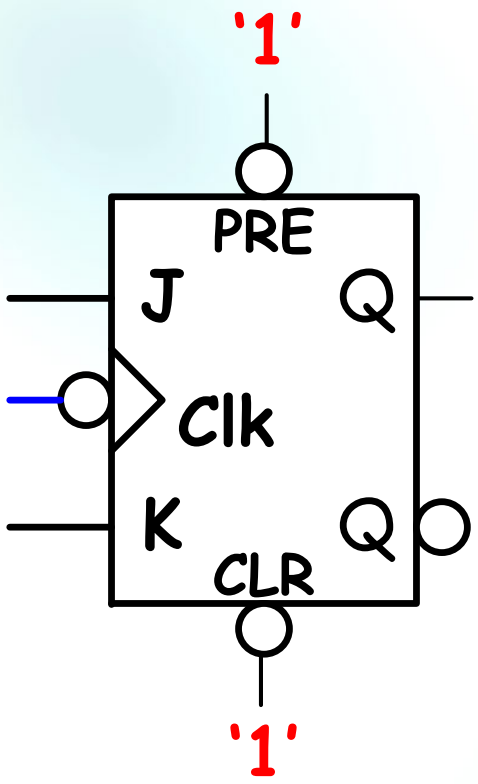
# The 'JK' Flip-flop Transitions




| Row | J | K | $Q_p$ | $Q_N$ | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | } Hold |
| 1 | 0 | 0 | 1 | 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 | | |

For a JK-ff, when **both the J and K inputs are at a logic '0'** the output will be in a **HOLD** condition.

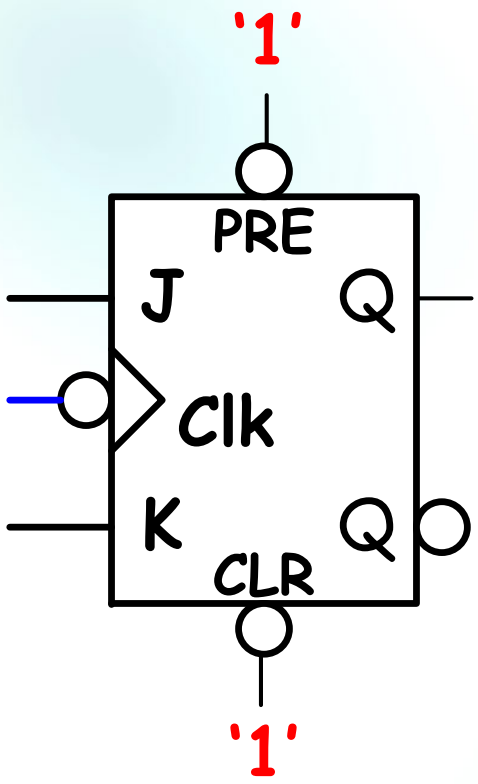
# The 'JK' Flip-flop Transitions




| Row | J | K | $Q_p$ | $Q_N$ | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | Hold |
| 1 | 0 | 0 | 1 | 1 | Hold |
| 2 | 0 | 1 | 0 | 0 | Reset |
| 3 | 0 | 1 | 1 | 0 | Reset |
| 4 | 1 | 0 | 0 | | |
| 5 | 1 | 0 | 1 | | |
| 6 | 1 | 1 | 0 | | |
| 7 | 1 | 1 | 1 | | |

When **J = 0 and K = 1** the output will be in a **RESET** condition no matter what present state is.

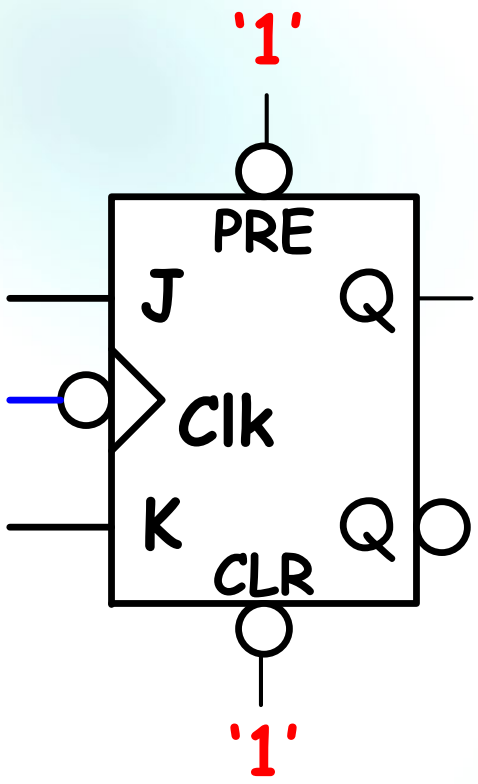
# The 'JK' Flip-flop Transitions




| Row | J | K | $Q_p$ | $Q_N$ | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | Hold |
| 1 | 0 | 0 | 1 | 1 | |
| 2 | 0 | 1 | 0 | 0 | Reset |
| 3 | 0 | 1 | 1 | 0 | |
| 4 | 1 | 0 | 0 | 1 | Set |
| 5 | 1 | 0 | 1 | 1 | |
| 6 | 1 | 1 | 0 | | |
| 7 | 1 | 1 | 1 | | |

When **J = 1 and K = 0** the output will be in a **SET** condition no matter what present state is.

# The 'JK' Flip-flop Transitions



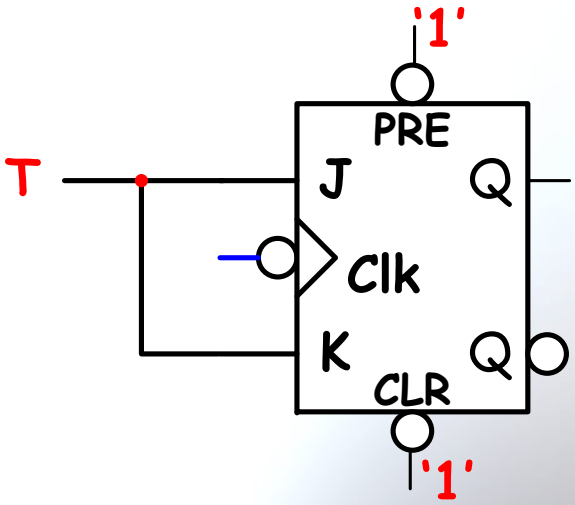

| Row | J | K | $Q_p$ | $Q_N$ | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | Hold |
| 1 | 0 | 0 | 1 | 1 | |
| 2 | 0 | 1 | 0 | 0 | Reset |
| 3 | 0 | 1 | 1 | 0 | |
| 4 | 1 | 0 | 0 | 1 | Set |
| 5 | 1 | 0 | 1 | 1 | |
| 6 | 1 | 1 | 0 | 1 | Toggle |
| 7 | 1 | 1 | 1 | 0 | |

When **J = 1 and K = 1** the output will be in a **TOGGLE** condition no matter what present state is.

# The JK used as a T-FF

Before simplifying the JK truth table we can note that the table looks like it is made up of both the T and the D flip-flops.

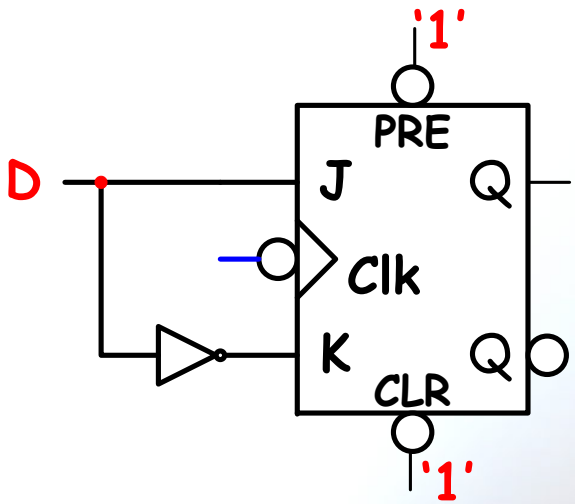
- We can convert the **JK** into a **T-FF** by simply **shorting the two inputs together** and using the resulting input as a T input.




| Row | J | K | $Q_p$ | $Q_N$ | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | } Hold |
| 1 | 0 | 0 | 1 | 1 | |
| 2 | | | | | |
| 3 | | | | | |
| 4 | | | | | |
| 5 | | | | | |
| 6 | 1 | 1 | 0 | 1 | } Toggle |
| 7 | 1 | 1 | 1 | 0 | |

# The JK used as a D-FF

All that is necessary to convert a JK into a D-FF is to place an inverter between the J and the K inputs so that they will always be opposite values.




| Row | J | K | $Q_p$ | $Q_N$ | |
|---|---|---|---|---|---|
| 0 | | | | | |
| 1 | | | | | |
| 2 | 0 | 1 | 0 | 0 | Reset |
| 3 | 0 | 1 | 1 | 0 | Reset |
| 4 | 1 | 0 | 0 | 1 | Set |
| 5 | 1 | 0 | 1 | 1 | Set |
| 6 | | | | | |
| 7 | | | | | |

# Simplifying the JK truth table

We can simplify the JK table and make it more useful at the same time by introducing the concept of the "Don't Care".

- An input is in a "Don't Care" state when it really doesn't matter what value is placed on the input. There will be no change on the output due to any value on the input.

# The State Transition Table

- **We start out by switching the order of the columns so that the state transitions are first. The table now becomes a "Transition Table".**

| $Q_P$ | $\Rightarrow$ | $Q_N$ | J | K |
|---|---|---|---|---|
| 0 | $\Rightarrow$ | 0 | | |
| 0 | $\Rightarrow$ | 1 | | |
| 1 | $\Rightarrow$ | 0 | | |
| 1 | $\Rightarrow$ | 1 | | |

# The '0' to '0' transition

Next we look for the two rows that have transitions from state '**0**' to state '**0**'. Note that it doesn't matter what value K assumes because the transition remains the same. So we place a '**0**' in the **J** column and an **X** for "don't care' in the **K** column.

| Row | J | K | $Q_P$ | $Q_N$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 2 | 0 | 1 | 0 | 0 |
| 3 | 0 | 1 | 1 | 0 |
| 4 | 1 | 0 | 0 | 1 |
| 5 | 1 | 0 | 1 | 1 |
| 6 | 1 | 1 | 0 | 1 |
| 7 | 1 | 1 | 1 | 0 |

| $Q_P$ | ⇒ | $Q_N$ | J | K |
|---|---|---|---|---|
| 0 | ⇒ | 0 | 0 | X |
| 0 | ⇒ | 1 | | |
| 1 | ⇒ | 0 | | |
| 1 | ⇒ | 1 | | |

# The '0' to '1' transition

Next we look for the two rows that have transitions from state '**0**' to state '**1**'. Note that it doesn't matter what value **K** assumes because the transition remains the same. So we place a '**1**' in the **J** column and an **X** for "don't care' in the **K** column.

| Row | J | K | $Q_P$ | $Q_N$ |
|---|---|---|---|---|
| 0 | | | | |
| 1 | 0 | 0 | 1 | 1 |
| 2 | | | | |
| 3 | 0 | 1 | 1 | 0 |
| 4 | 1 | 0 | 0 | 1 |
| 5 | 1 | 0 | 1 | 1 |
| 6 | 1 | 1 | 0 | 1 |
| 7 | 1 | 1 | 1 | 0 |

| $Q_P$ | $\Rightarrow$ | $Q_N$ | J | K |
|---|---|---|---|---|
| 0 | $\Rightarrow$ | 0 | 0 | X |
| 0 | $\Rightarrow$ | 1 | 1 | X |
| 1 | $\Rightarrow$ | 0 | | |
| 1 | $\Rightarrow$ | 1 | | |

# The '1' to '0' transition

Next we look for the two rows that have transitions from state '**1**' to state '**0**'. Note that it doesn't matter what value **J** assumes because the transition remains the same. So we place a '**1**' in the **K** column and an **X** for "don't care' in the **J** column.

| Row | J | K | $Q_P$ | | $Q_N$ |
|---|---|---|---|---|---|
| 0 | | | | | |
| 1 | 0 | 0 | 1 | | 1 |
| 2 | | | | | |
| 3 | 0 | 1 | 1 | → | 0 |
| 4 | | | | | |
| 5 | 1 | 0 | 1 | | 1 |
| 6 | | | | | |
| 7 | 1 | 1 | 1 | → | 0 |

| $Q_P$ | ⇒ | $Q_N$ | J | K |
|---|---|---|---|---|
| 0 | ⇒ | 0 | 0 | X |
| 0 | ⇒ | 1 | 1 | X |
| 1 | ⇒ | 0 | X | 1 |
| 1 | ⇒ | 1 | | |

# The '1' to '1' transition

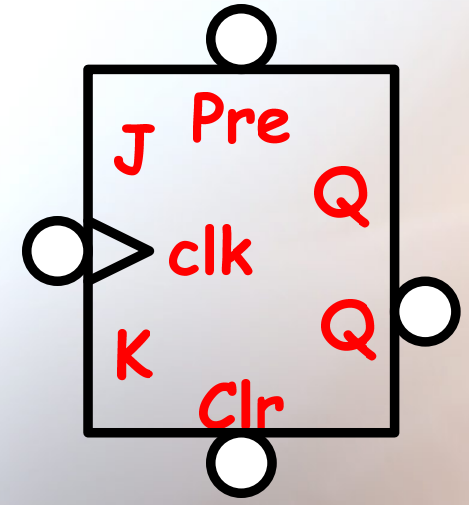
Next we look for the two rows that have transitions from state '**1**' to state '**1**'. Note that it doesn't matter what value **J** assumes because the transition remains the same. So we place a '**0**' in the **K** column and an **X** for "don't care' in the **J** column.

| Row | J | K | $Q_P$ | | $Q_N$ |
|---|---|---|---|---|---|
| 0 | | | | | |
| 1 | 0 | 0 | 1 | → | 1 |
| 2 | | | | | |
| 3 | | | | | |
| 4 | | | | | |
| 5 | 1 | 0 | 1 | → | 1 |
| 6 | | | | | |
| 7 | | | | | |

| $Q_P$ | ⇒ | $Q_N$ | J | K |
|---|---|---|---|---|
| 0 | ⇒ | 0 | 0 | X |
| 0 | ⇒ | 1 | 1 | X |
| 1 | ⇒ | 0 | X | 1 |
| 1 | ⇒ | 1 | X | 0 |

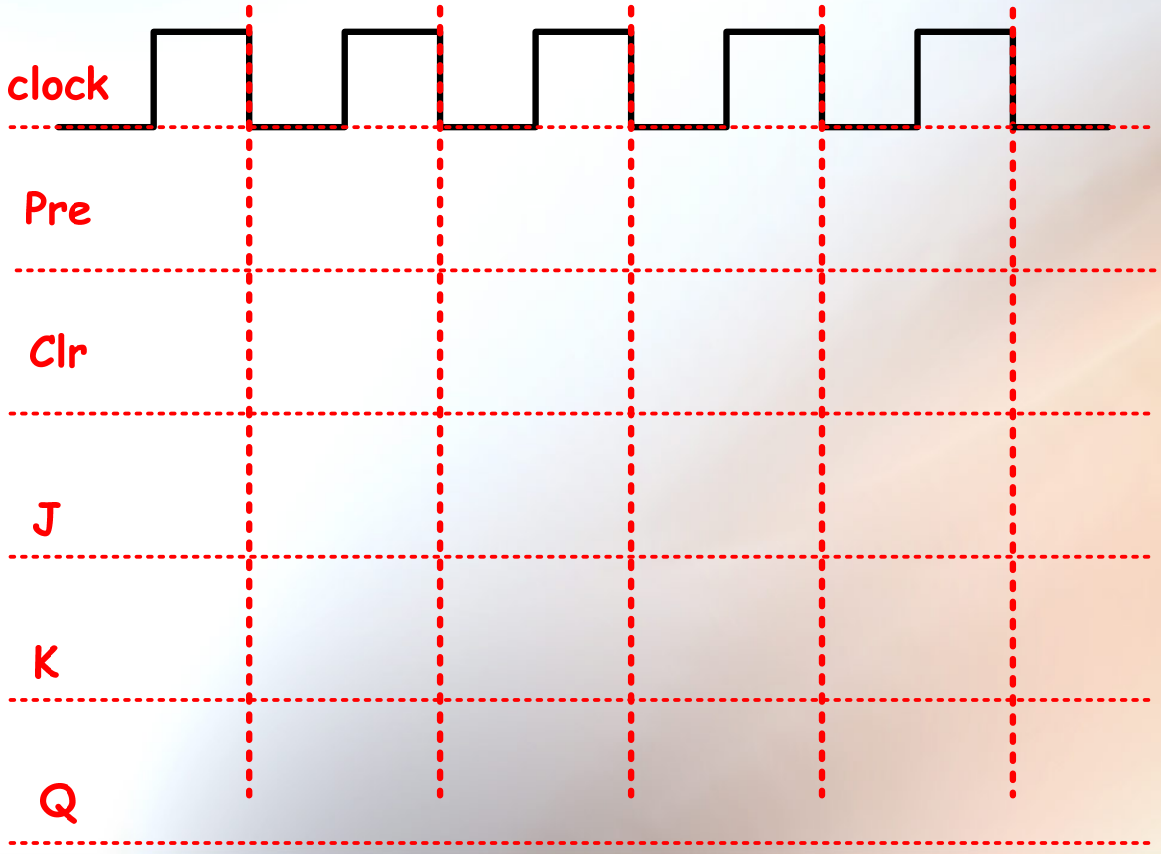
# Timing Diagrams

- Quite often it is necessary to examine how a circuit acts given a set of input signals. The tool used to accomplish this is the Timing Diagram.
- Use a set of signals to examine how the JK-FF below reacts. Note that the PRE and CLR inputs are not set at any logic level.



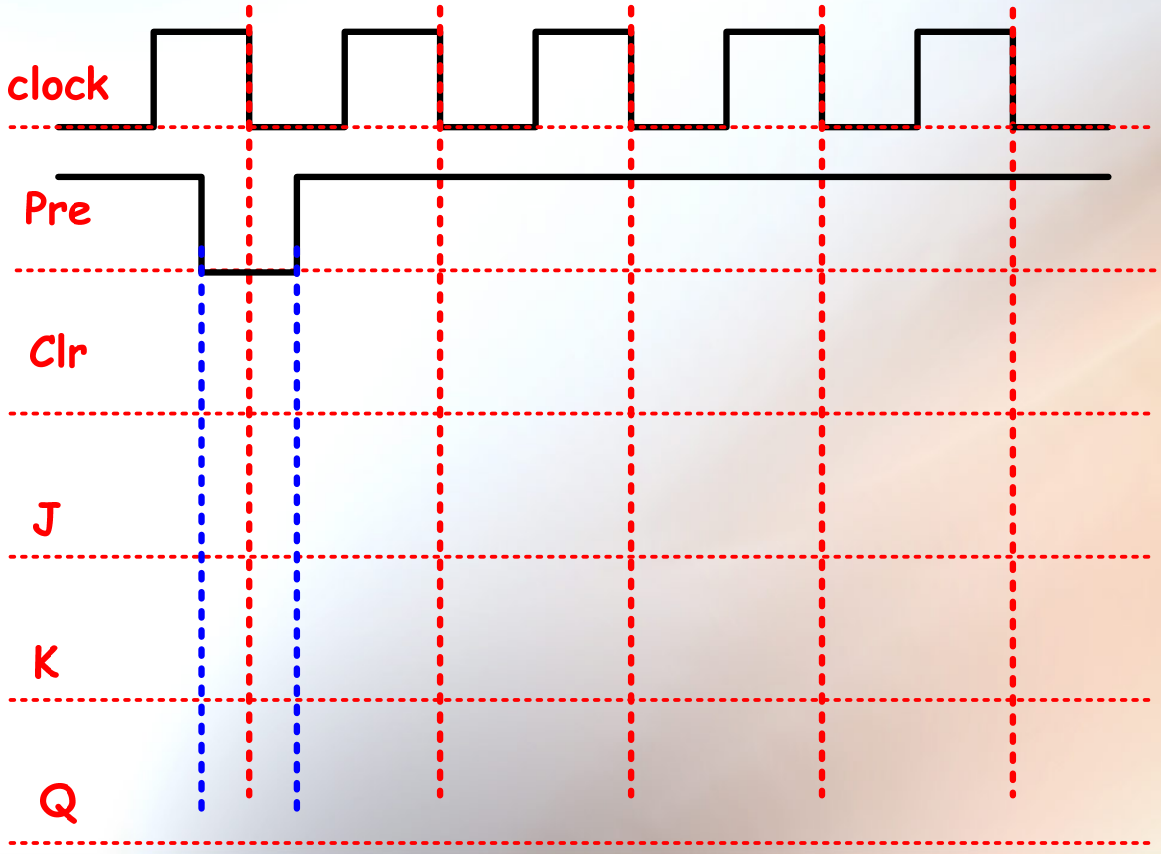
# Timing Diagrams

- The figure below is the framework for a "Timing" diagram. Since the JK-FF had a trailing edge trigger, vertical lines have been added to remind the user about trigger events.



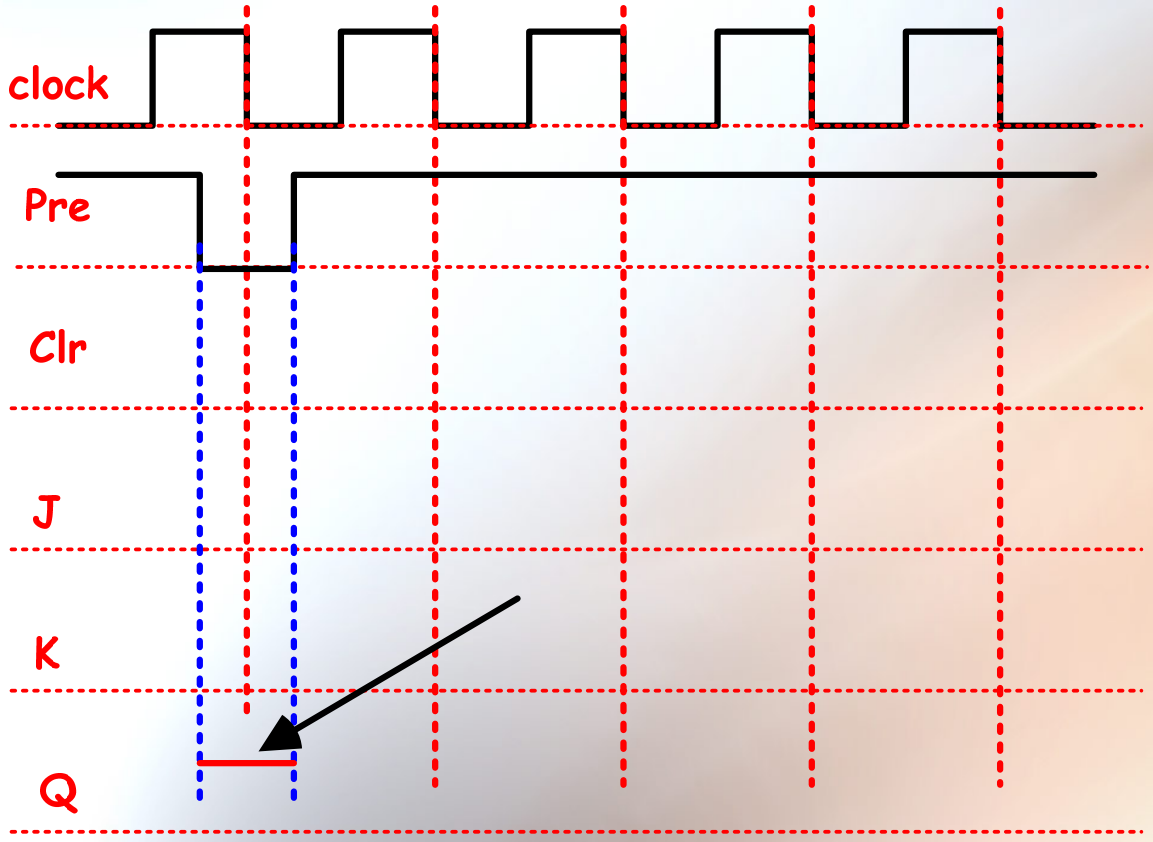
# Timing Diagrams

- A signals has been added for the PRE input. Since it is an asynchronous input, a vertical line has been to mark the edges where the input was active.



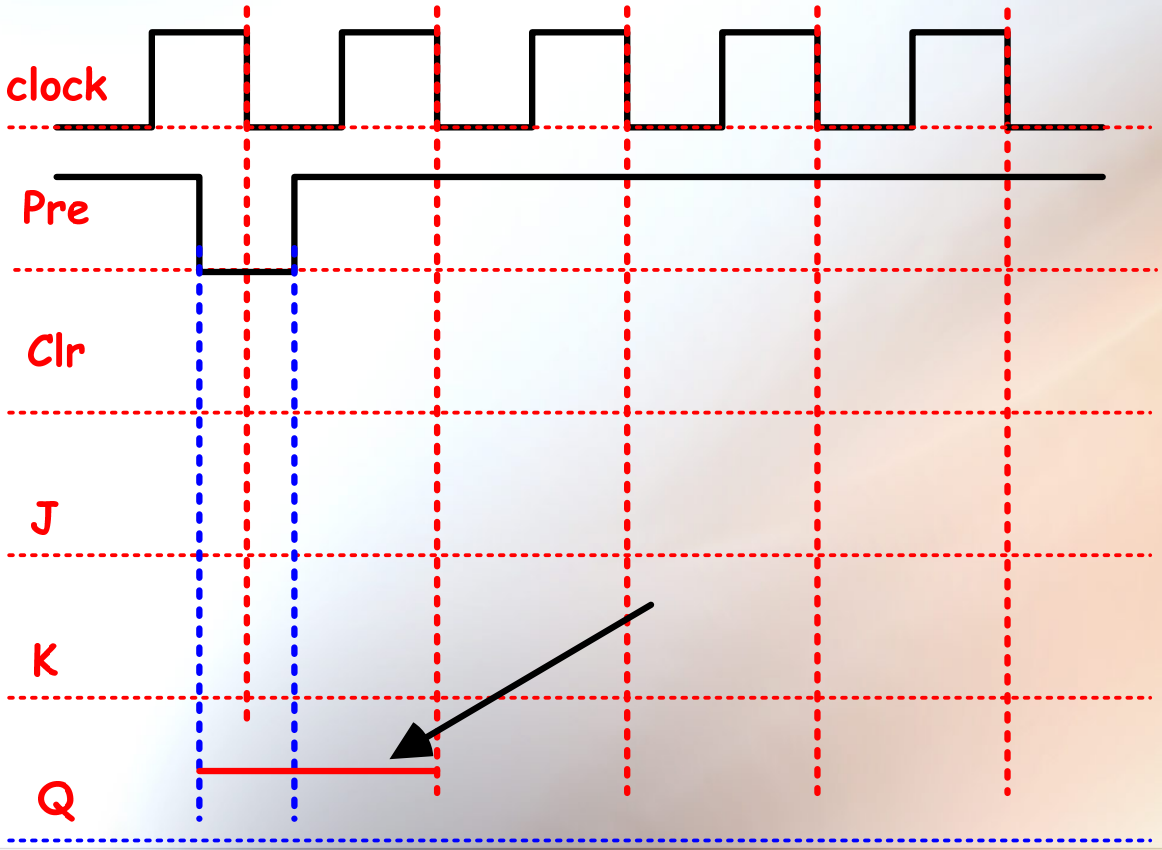
# Timing Diagrams

- The Q output is marked at a HIGH level during the time that PRE was LOW. Again, it makes no difference what the clock is doing here.



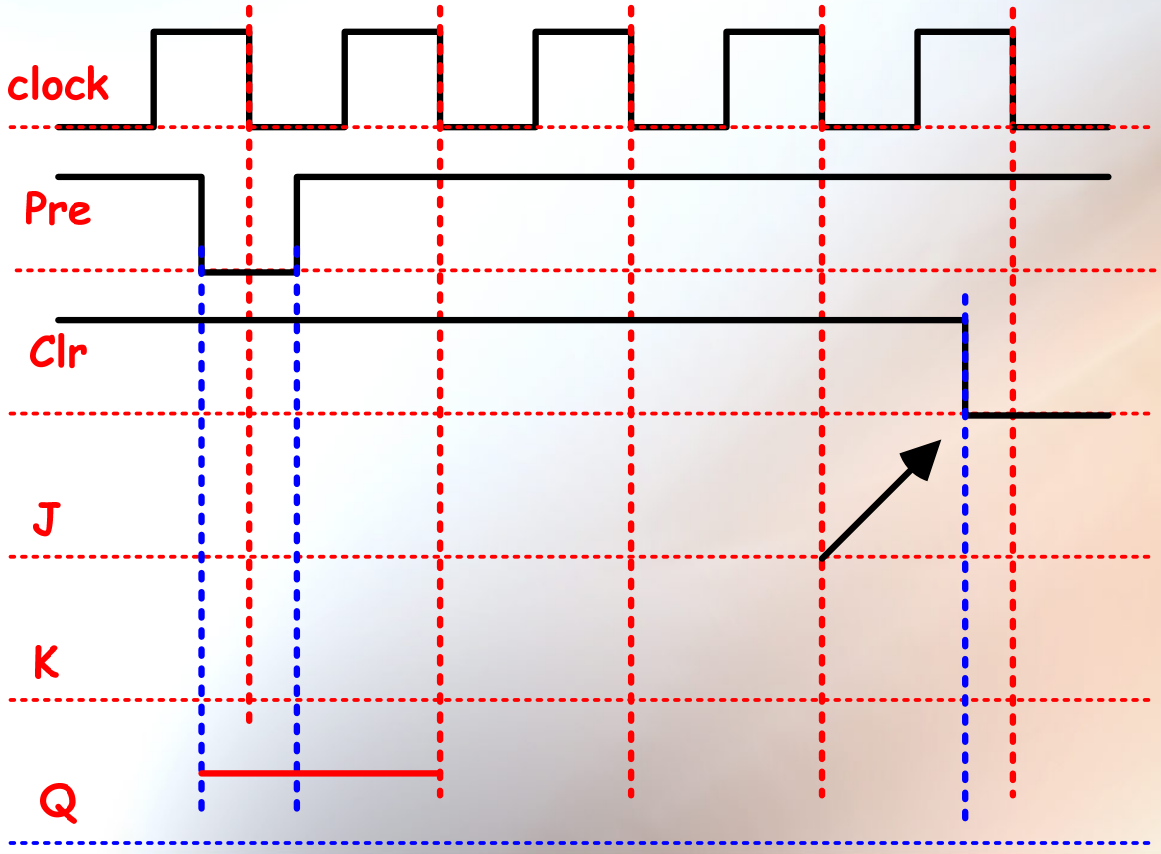
# Timing Diagrams

- Since there is no way that Q can change until the next trailing edge, the HIGH level extended.



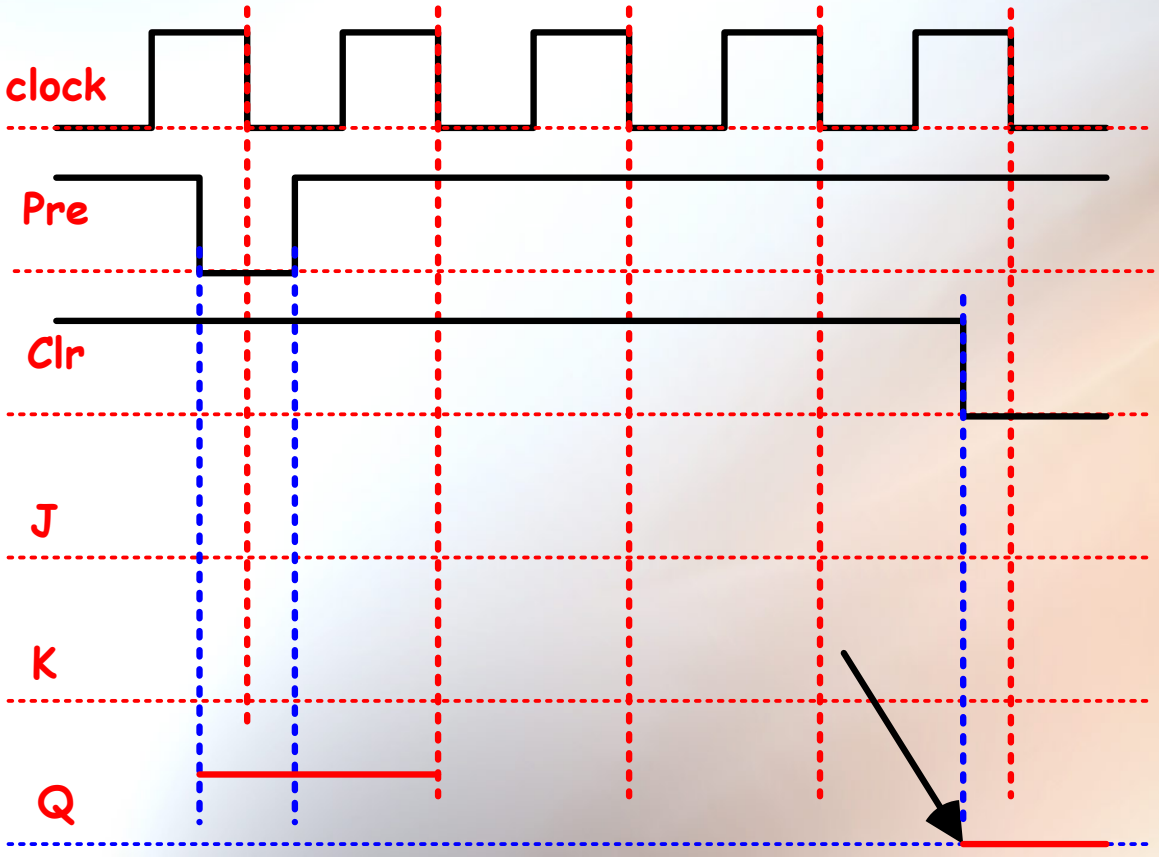
# Timing Diagrams

- Now a CLR signal is added with its vertical markers showing the period when the signal is low.



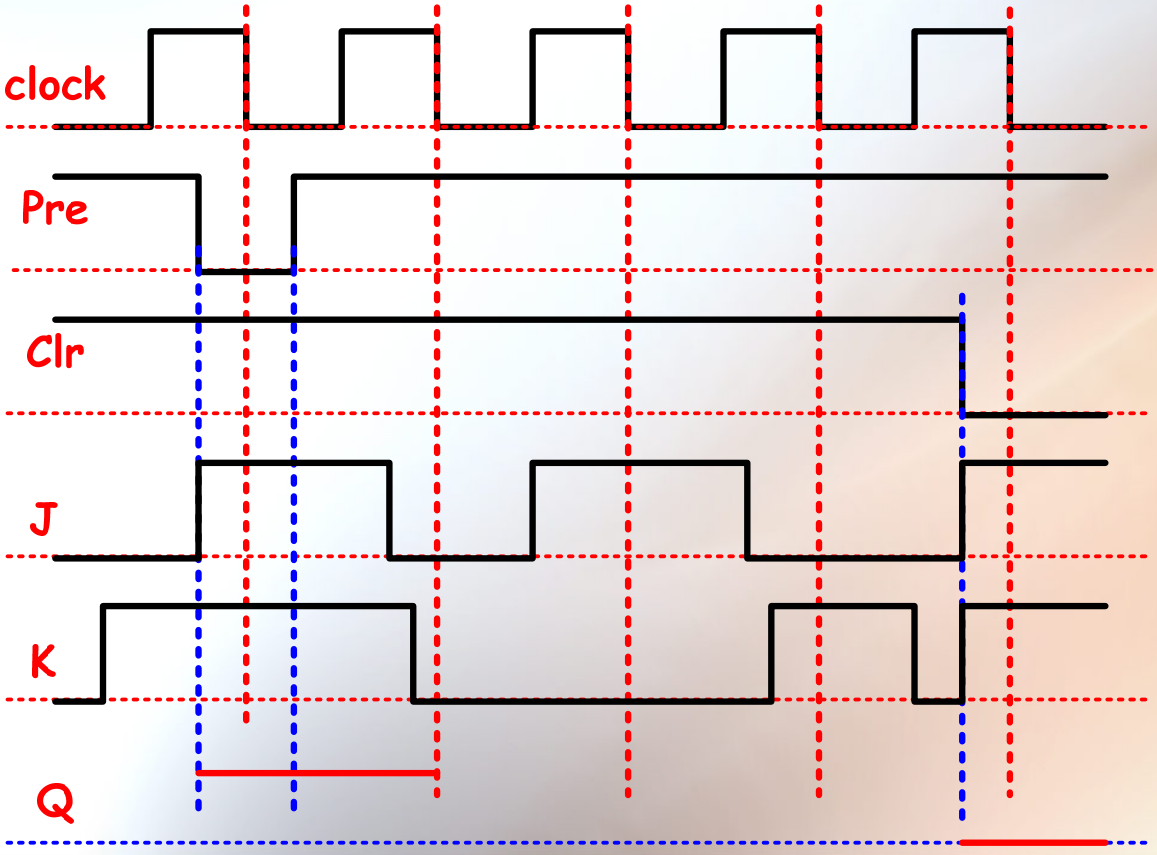
# Timing Diagrams

- **The LOW on the CLR line causes Q to go LOW asynchronously.**



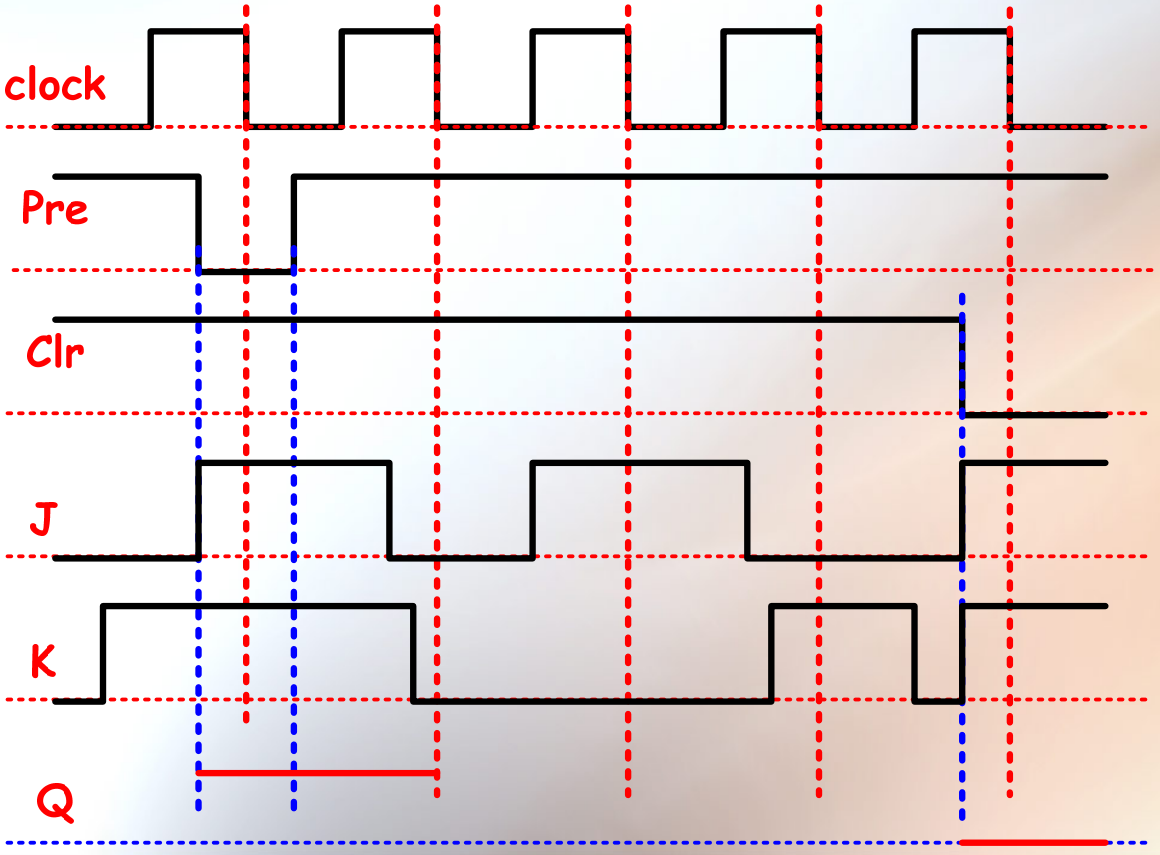
# Timing Diagrams

- **Next, J and K signals are added.**



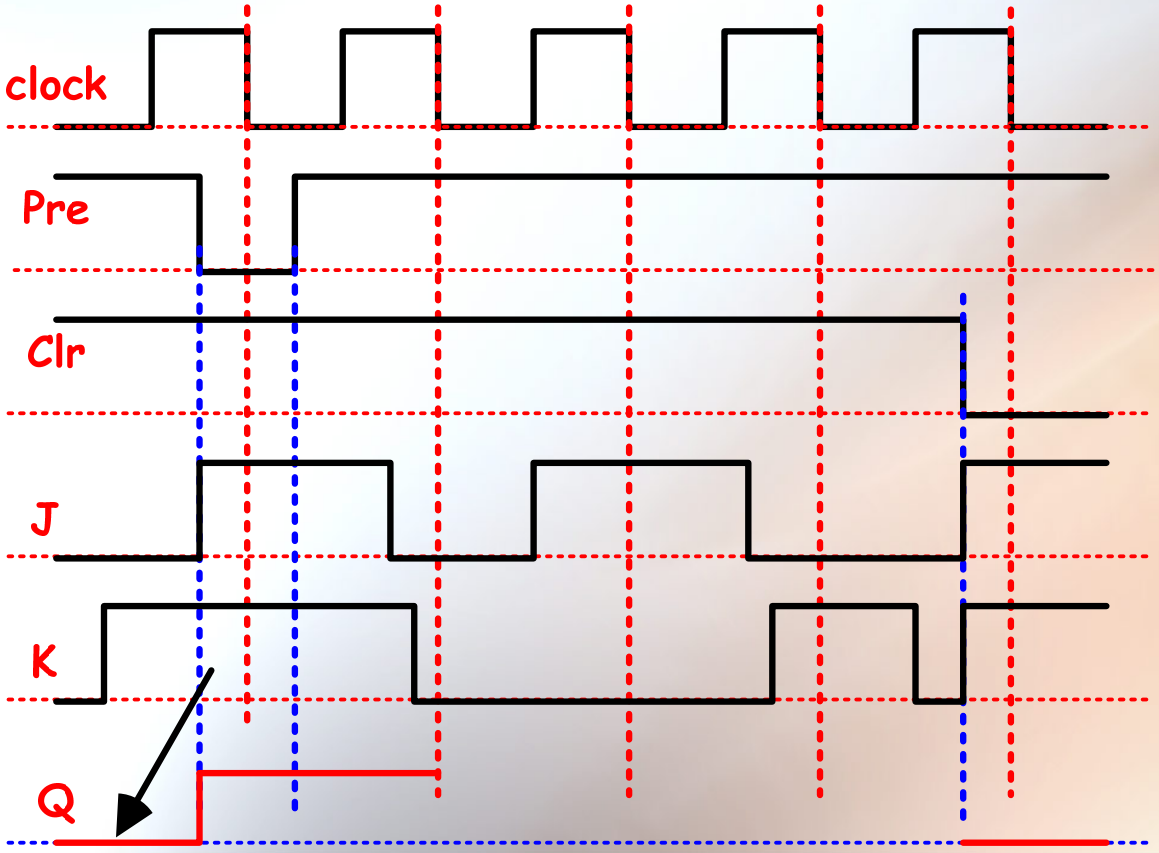
# Timing Diagrams

- At this point the problem would need to state what the initial state is. In this case, the initial state is $y_0 = 0$.



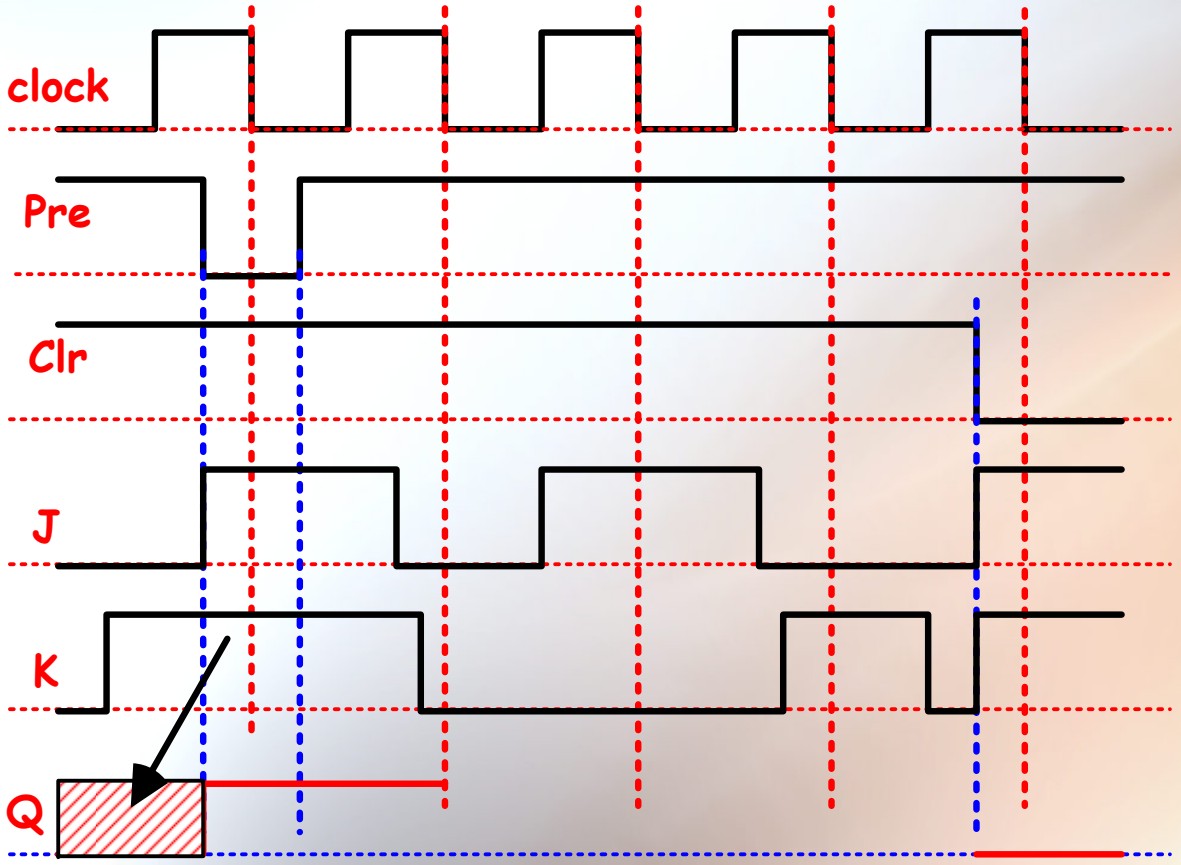
# Timing Diagrams

- The Initial state of $y_0 = 0$ is added to the Q output. It should go up to the 1st trailing edge but this time it goes up to where PRE forces the output HIGH.



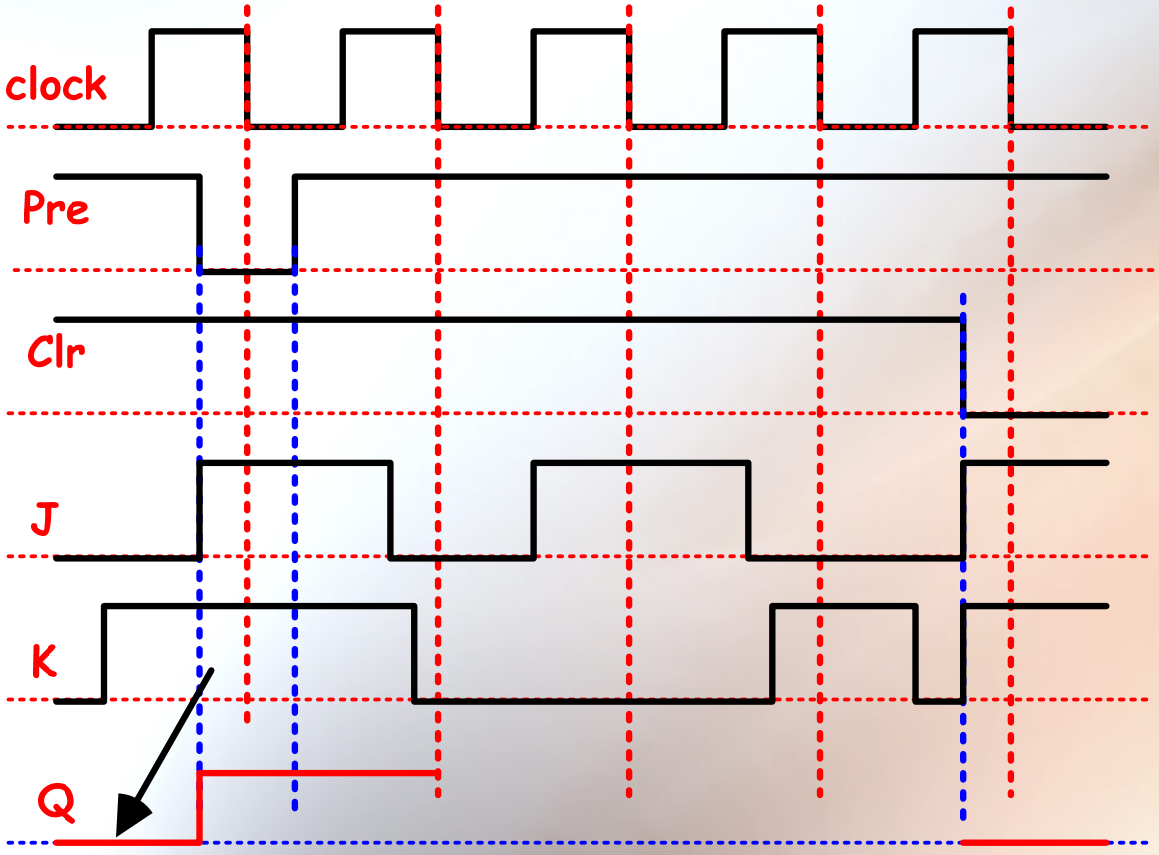
# Timing Diagrams

- Note that if the initial state isn't given, the state becomes indeterminate until the first trigger event.



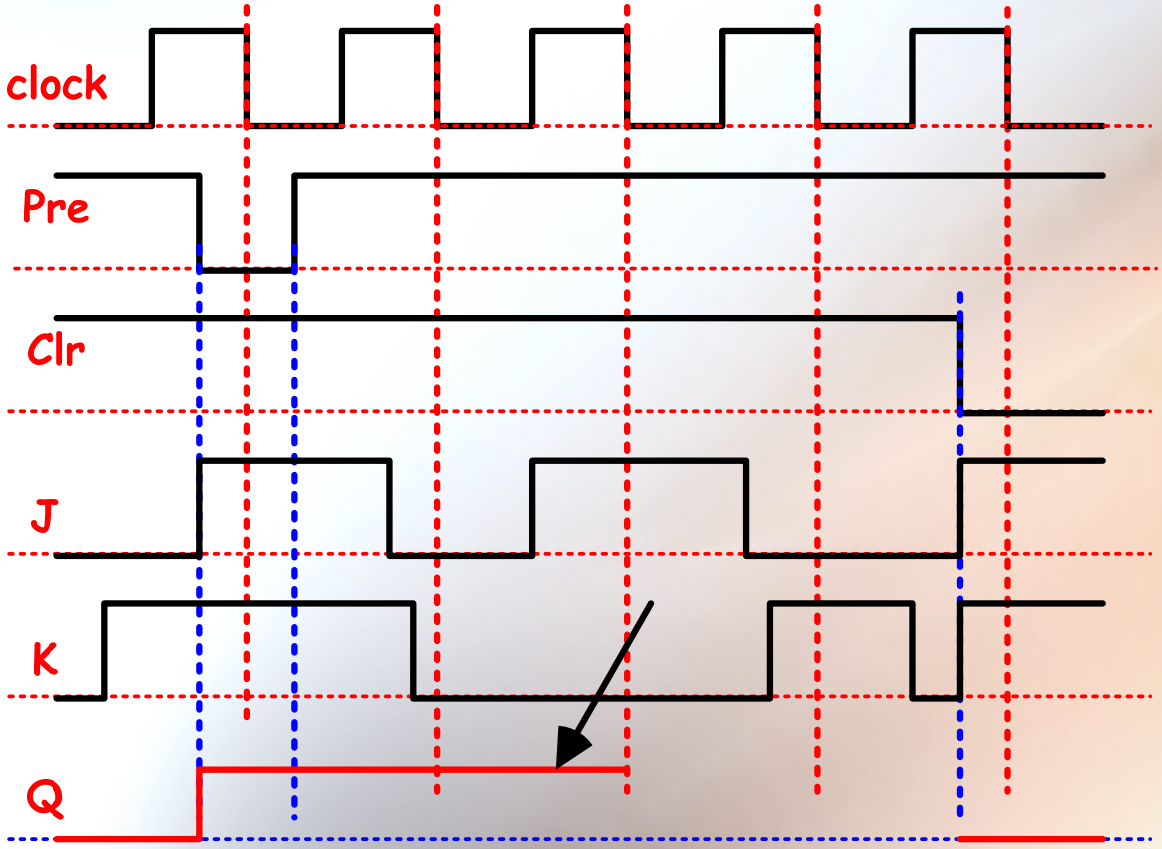
# Timing Diagrams

- OK, back to the original initial state. Note that PRE holds the output HIGH till the second trailing edge so J and K don't matter till then.



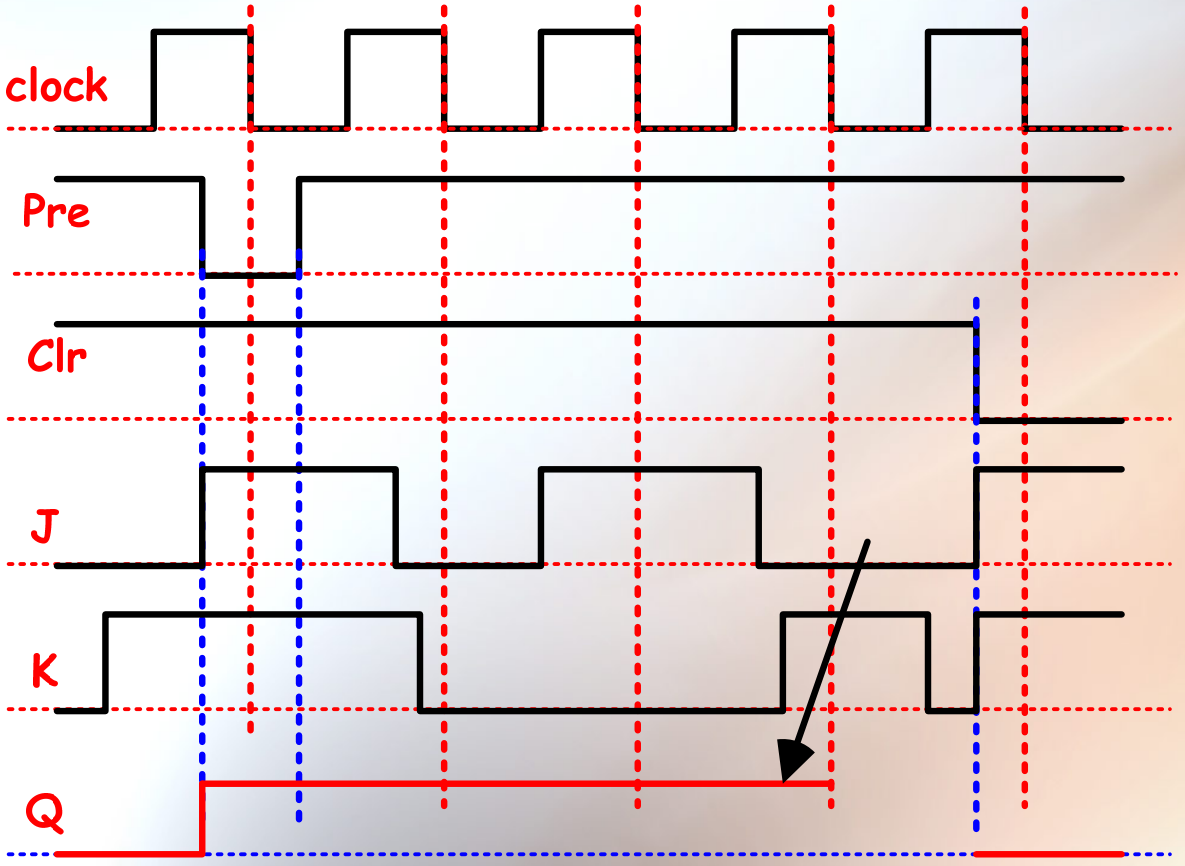
# Timing Diagrams

- At the 2nd trigger, J = 0 and K = 0, so the FF is in a HOLD condition. Q stays HIGH till trigger 3.



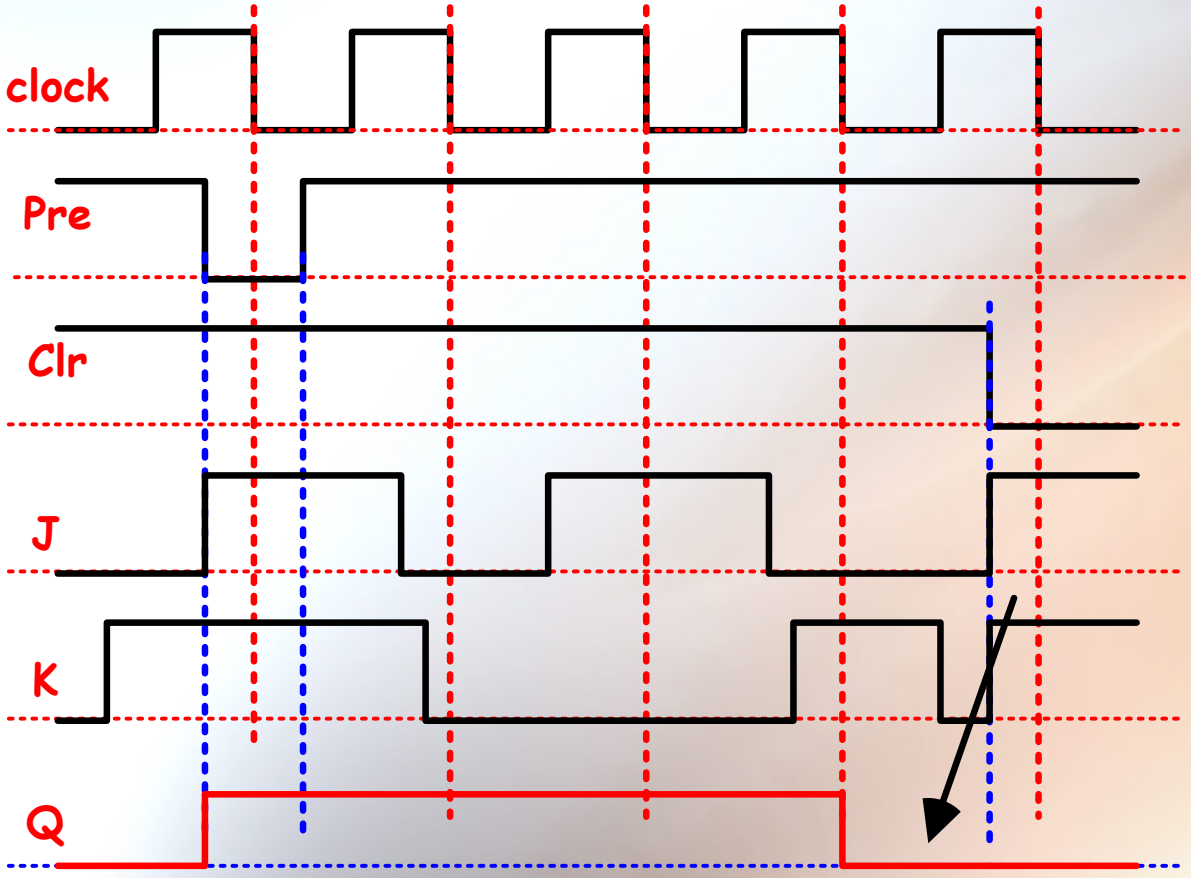
# Timing Diagrams

- At the 3rd trigger, J = 1 and K = 0, so the FF is in a SET condition. Q stays HIGH till trigger 4 since it was already set.



# Timing Diagrams

- At the 4th trigger, J = 0 and K = 1, so the FF is in a RESET condition. Q goes low until the next trigger (or in this case until it runs into the CLR'd output).



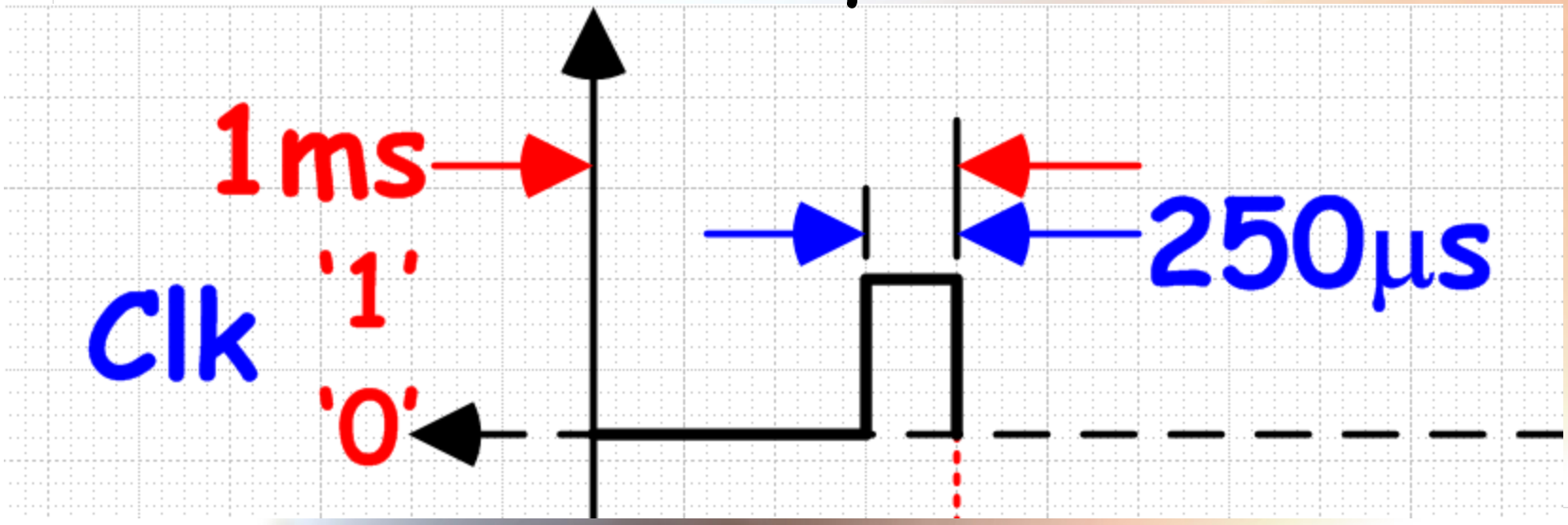
# Frequency Division

- **A T- FF is also known as a frequency divider.**
- **Obviously, if it divides frequency then it multiplies time (or periods).**

# Frequency Division

- Before we go any further, lets review the concept of duty cycle.
- Duty cycle is the ratio of Time High to the Period expressed as a percentage.

$$D = \frac{T_H}{T} \bullet 100$$

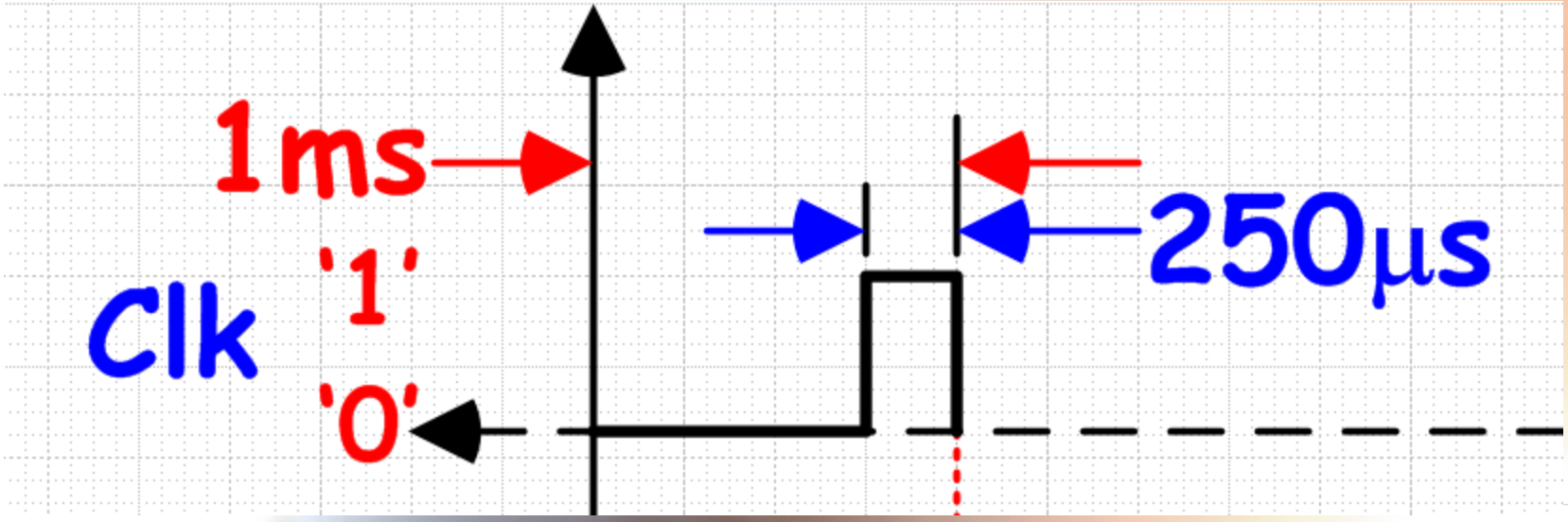


# Frequency Division

- Duty cycle is the ratio of Time High to the Period expressed as a percentage.

$$D = \frac{T_H}{T} \bullet 100$$

- So, if you look at the waveform below, you see the measurements for time high and period.



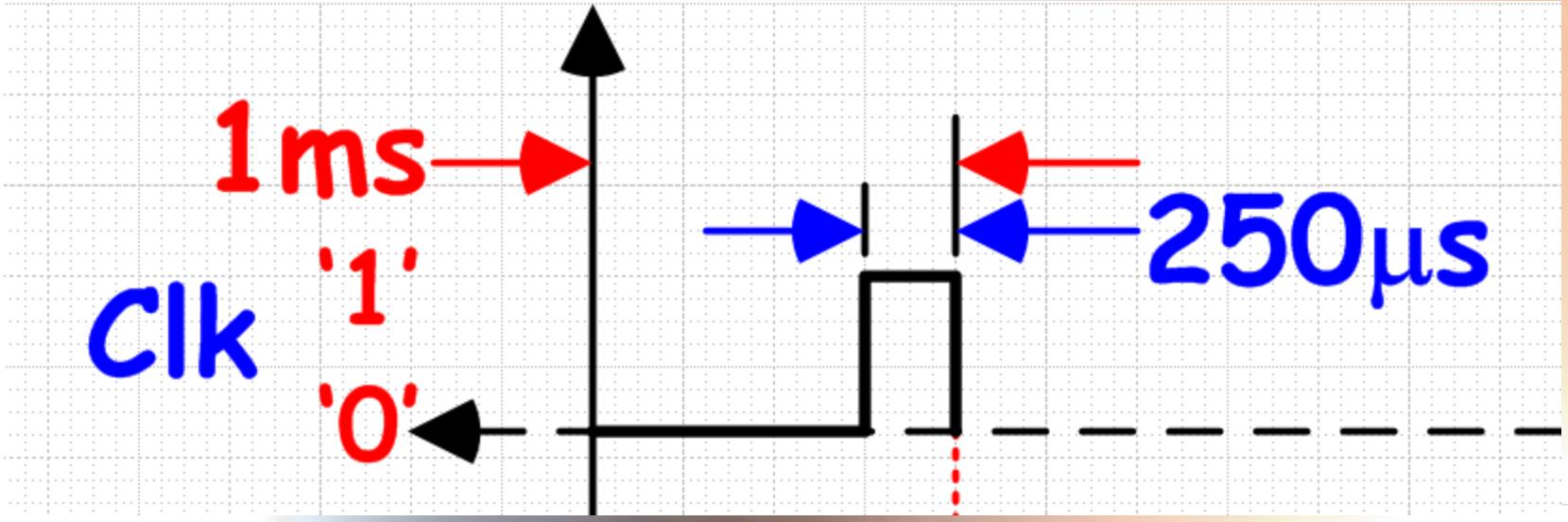
# Frequency Division

- **The duty cycle for this waveform is:**

$$D = \frac{T_H}{T} \bullet 100$$

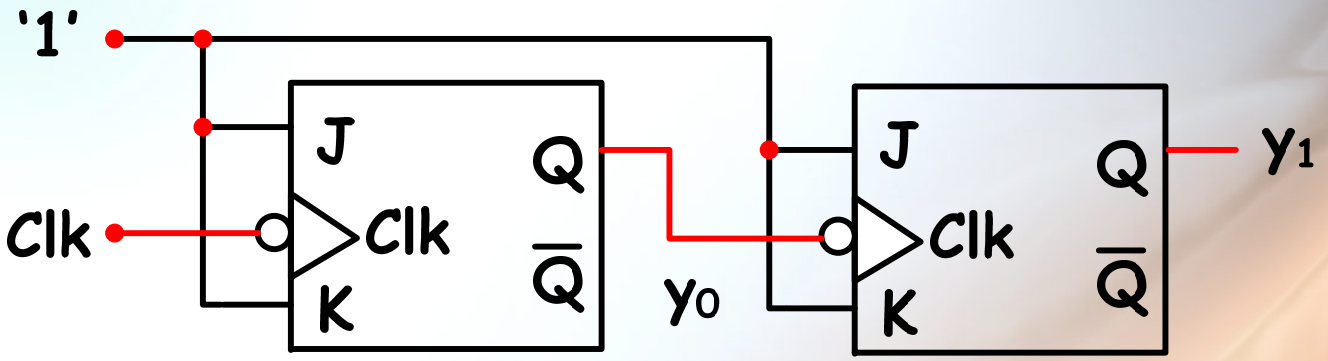
$$= \frac{250\mu s}{1ms} \bullet 100$$

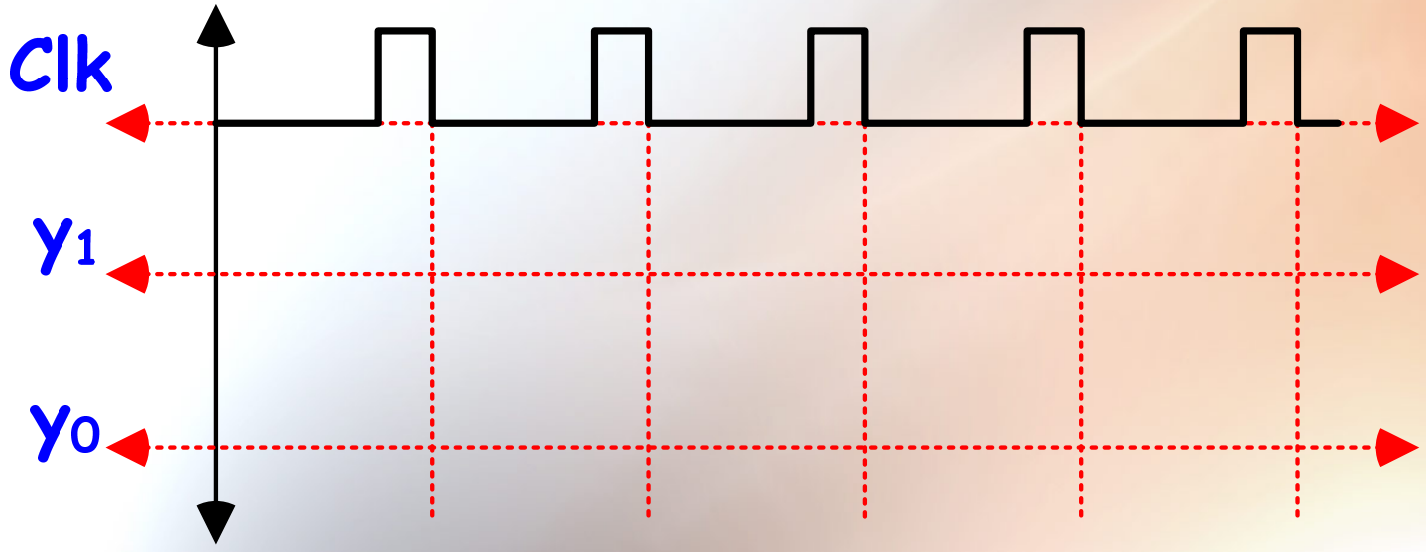
$$= 25\%$$



# Frequency Division

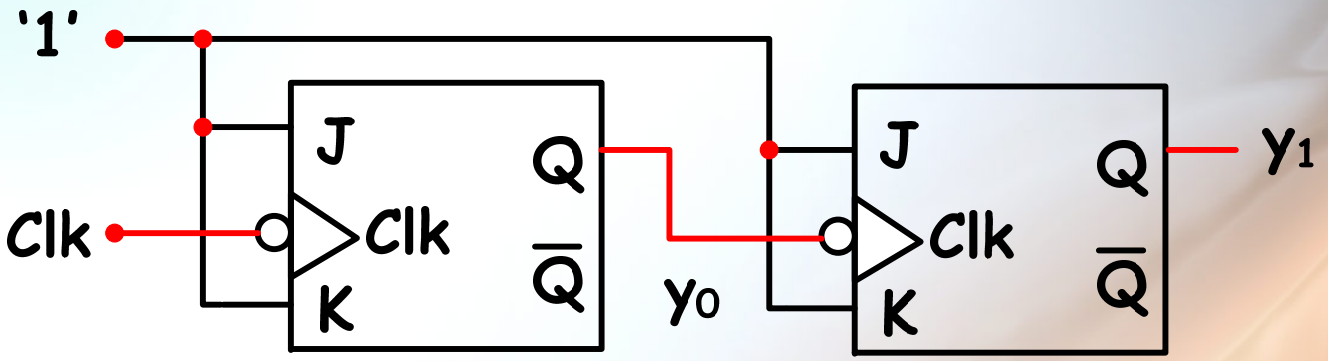
- Now, lets supply a 25% duty cycle clock to a two T-FF ripple counter.

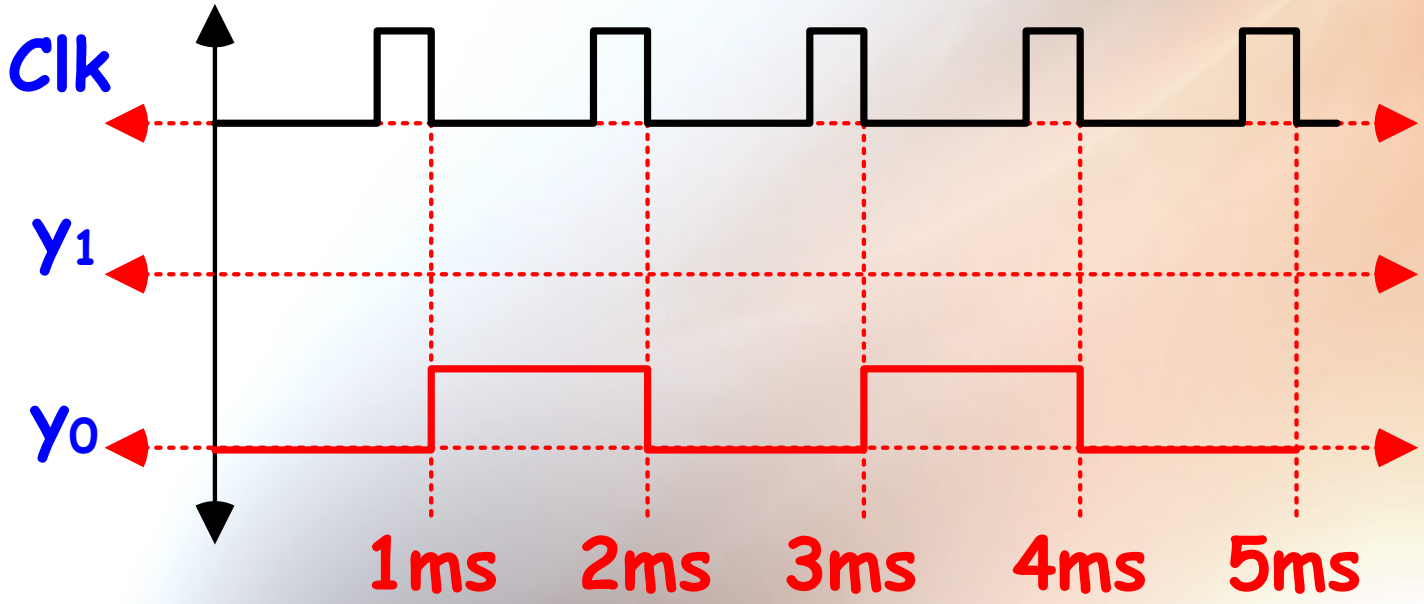


# Frequency Division

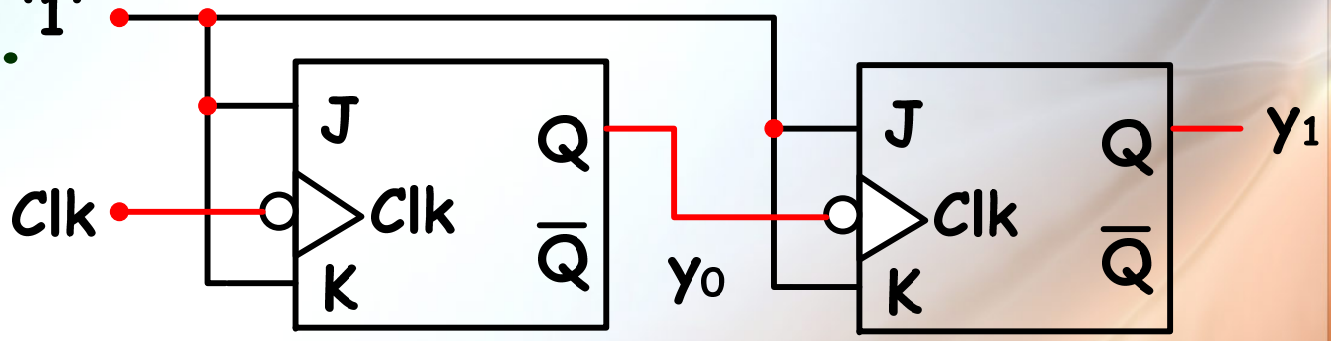
- The 1st FF output, $y_0$, will toggle with each clock trailing edge.

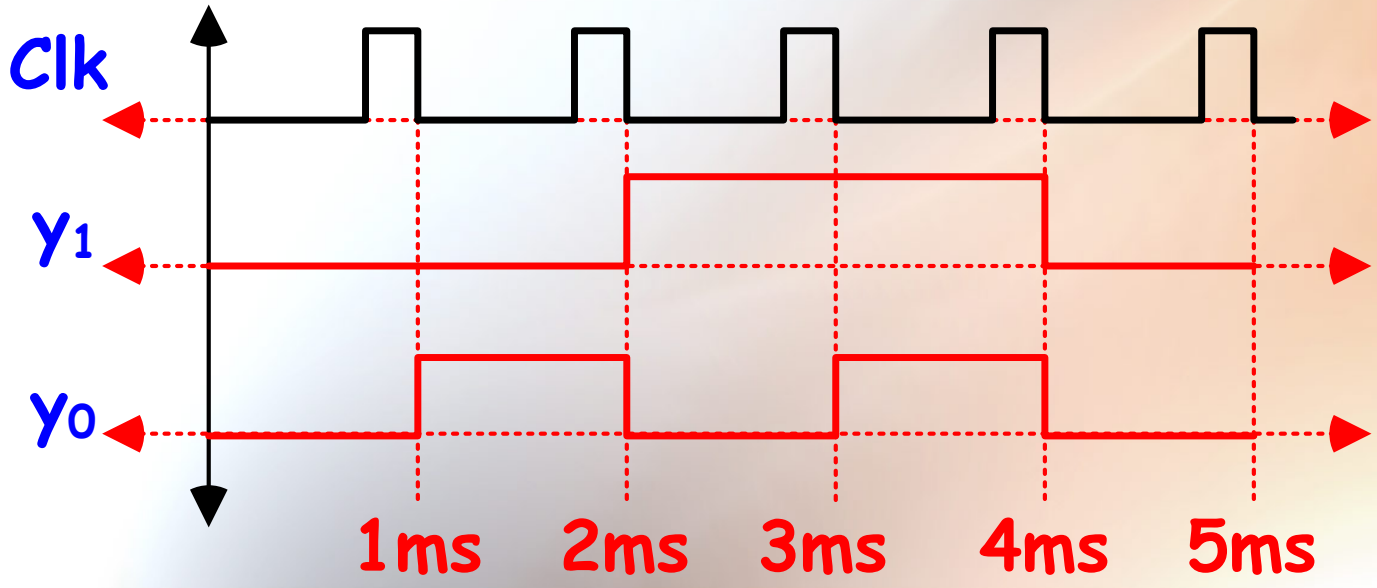


# Frequency Division

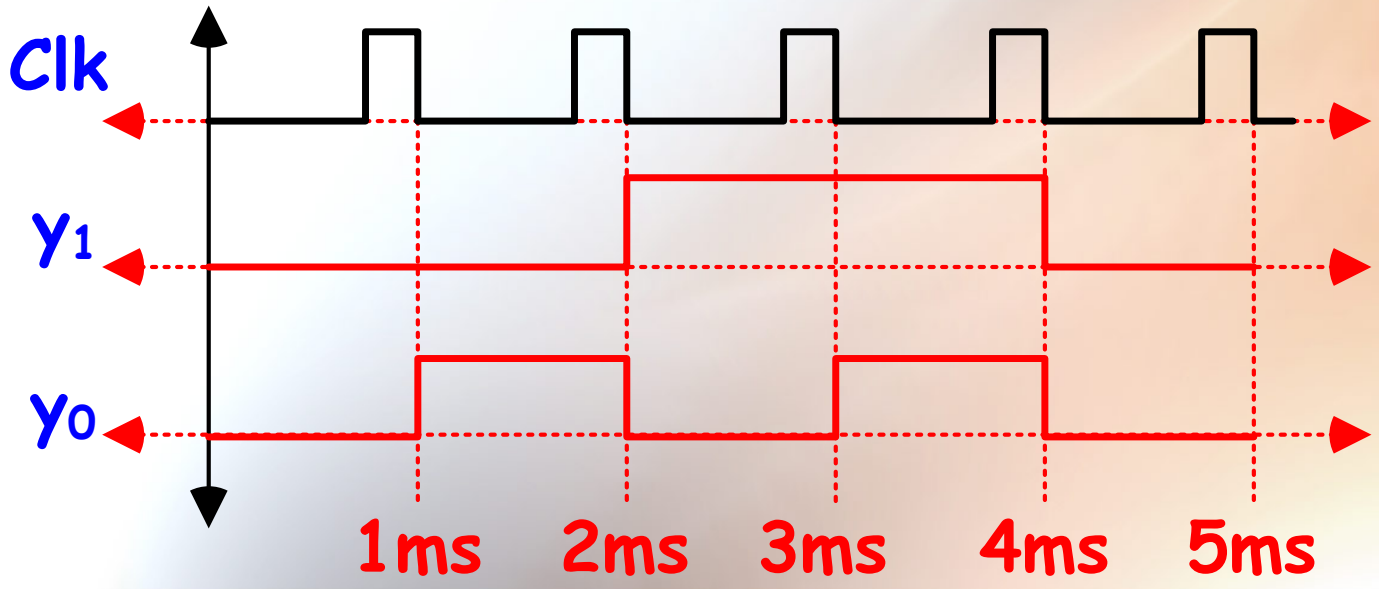
- The 2nd FF output, $y_1$, will toggle with each of the 1st FF's trailing edges.



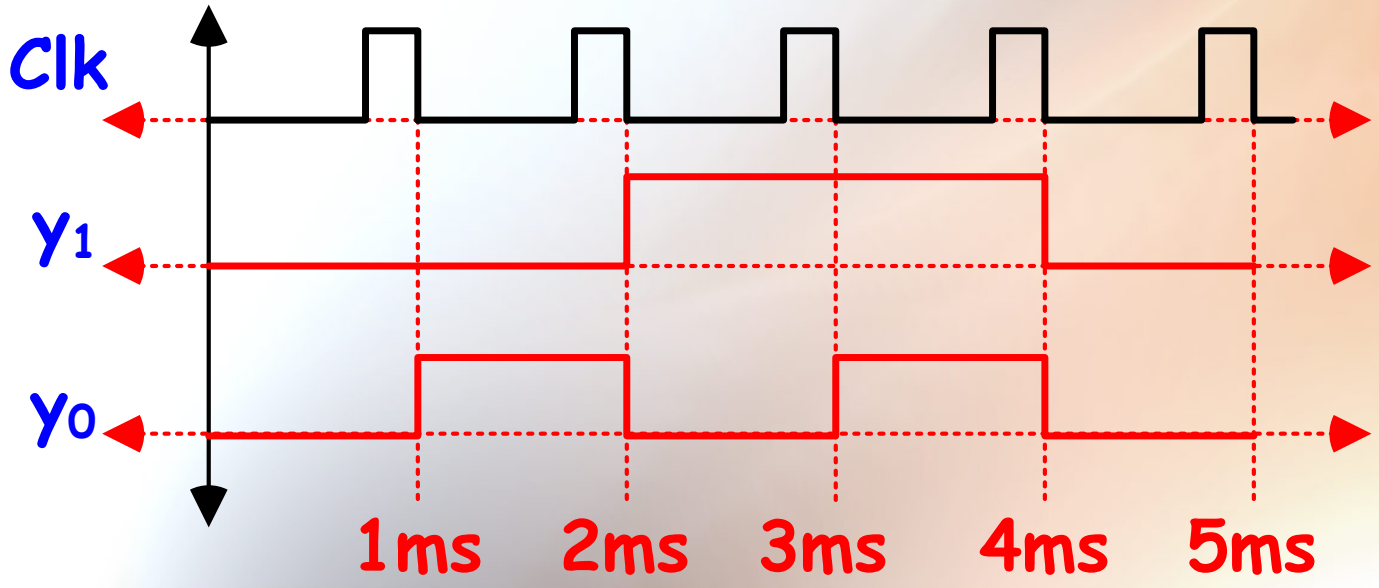
# Frequency Division

- Now, lets examine the waveform for $y_0$.
- Note that the duty cycle is now 50%. This is one of the effects of a T-FF.



# Frequency Division

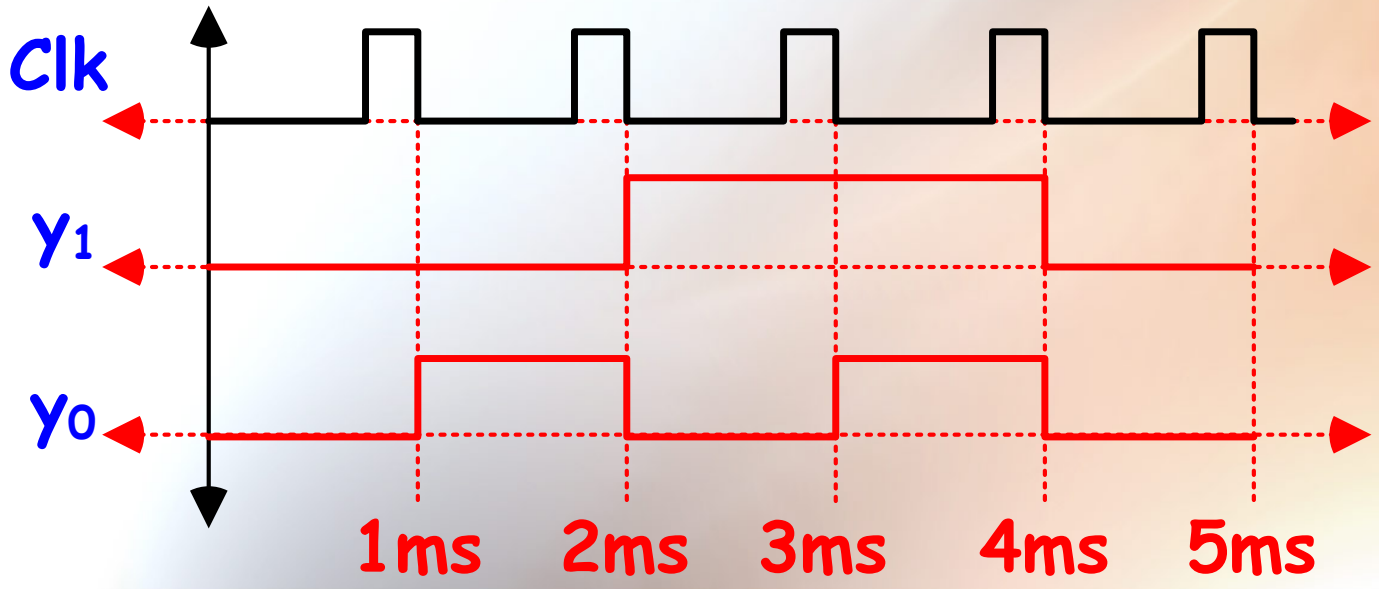
- The period (T) of $y_0$ is 2ms which is twice the period of the clock. The period has been MULTIPLIED by 2!



# Frequency Division

- The frequency of the clock was:

$$f = \frac{1}{T} = \frac{1}{1ms} = 1khz$$

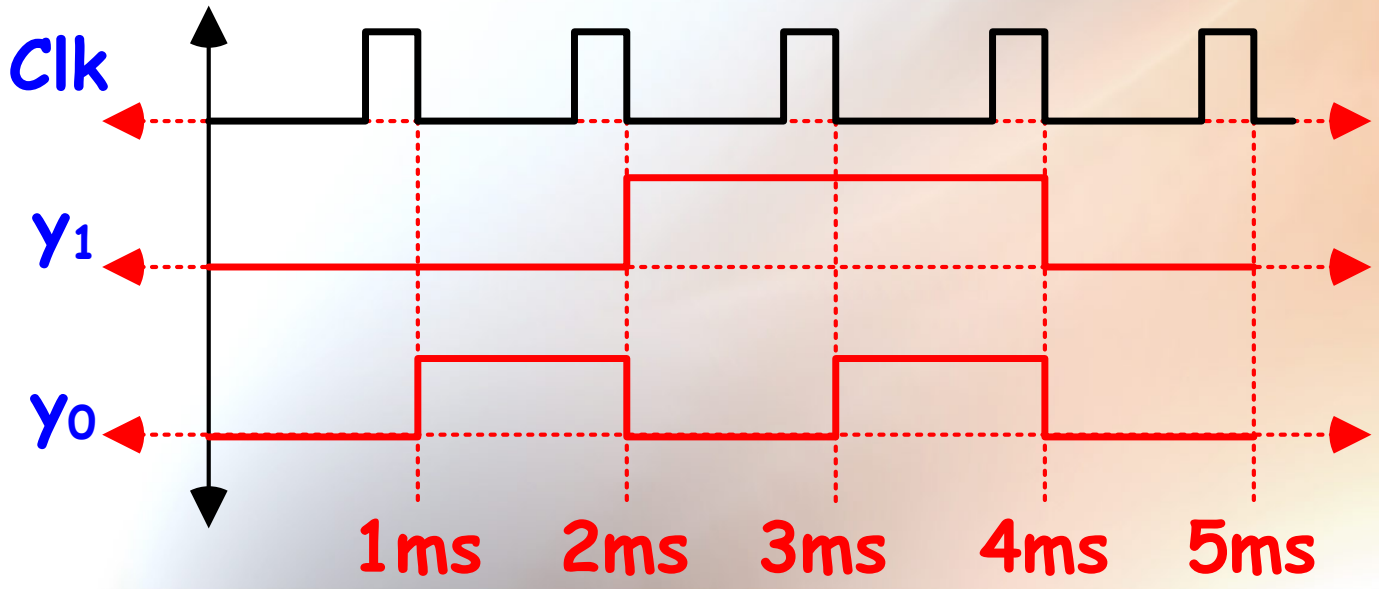


# Frequency Division

**The Frequency of $y_0$ is:**

$$f = \frac{1}{2ms} = 500hz$$

**The Frequency has been divided by 2.**

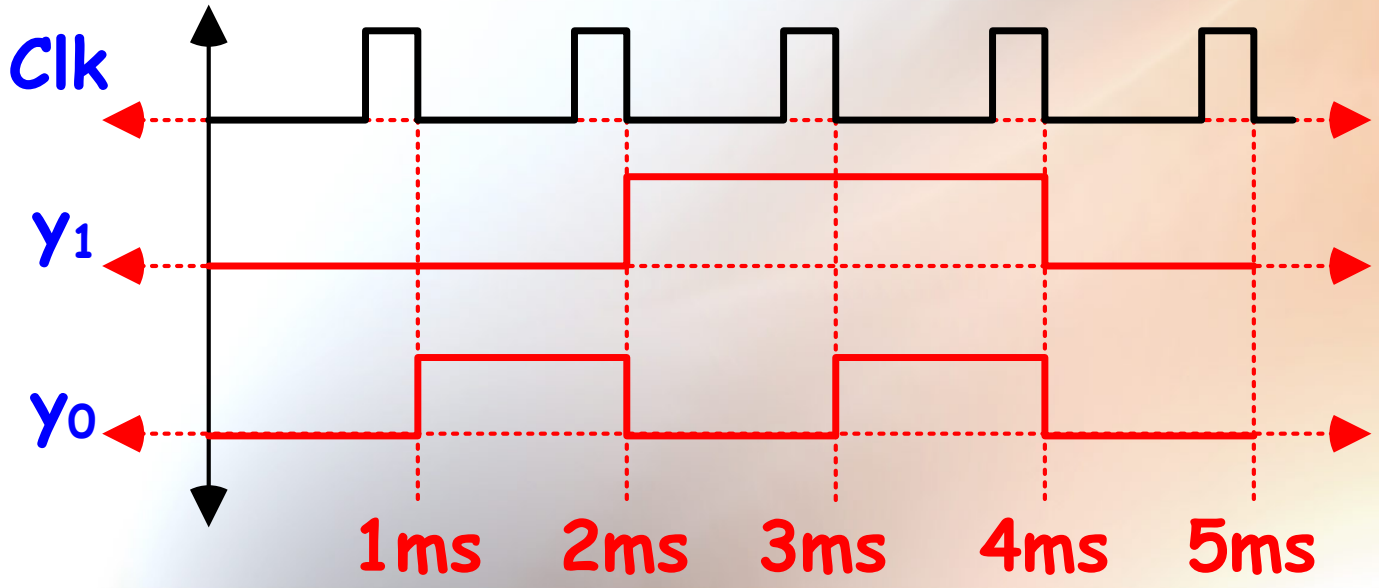


# Frequency Division

The Frequency of $y_1$ is:

$$f = \frac{1}{4ms} = 250hz$$

- This is one half of $y_1$ and one fourth of the clocks period!



# Clock Slew Error

- Also known as timing skew, Clock skew is a phenomenon in synchronous circuits in which the clock signal reaches different components at different times.

# Clock Slew Error

- Also known as timing skew, Clock skew is a phenomenon in synchronous circuits in which the clock signal reaches different components at different times.
- The causes vary but the problem can cause serious glitches in some types of circuits.

# Master/Slave JK-FF's

- The Master/Slave JK was designed to compensate for the clock skew problem.

# Master/Slave JK-FF's

- The Master/Slave JK was designed to compensate for the clock skew problem.
- This JK contains two latches connected serially. The first latch is designed to accept the data from the JK inputs on the leading clock edge.

# Master/Slave JK-FF's

- The Master/Slave JK was designed to compensate for the clock skew problem.
- This JK contains two latches connected serially. The first latch is designed to accept the data from the JK inputs on the leading clock edge.
- After the leading edge the contents of the JK inputs are held in the first latch. The inputs could continue to change but will not be accepted. (of course, in reality there exists about a 20ns setup time after the edge as well).

# Master/Slave JK-FF's

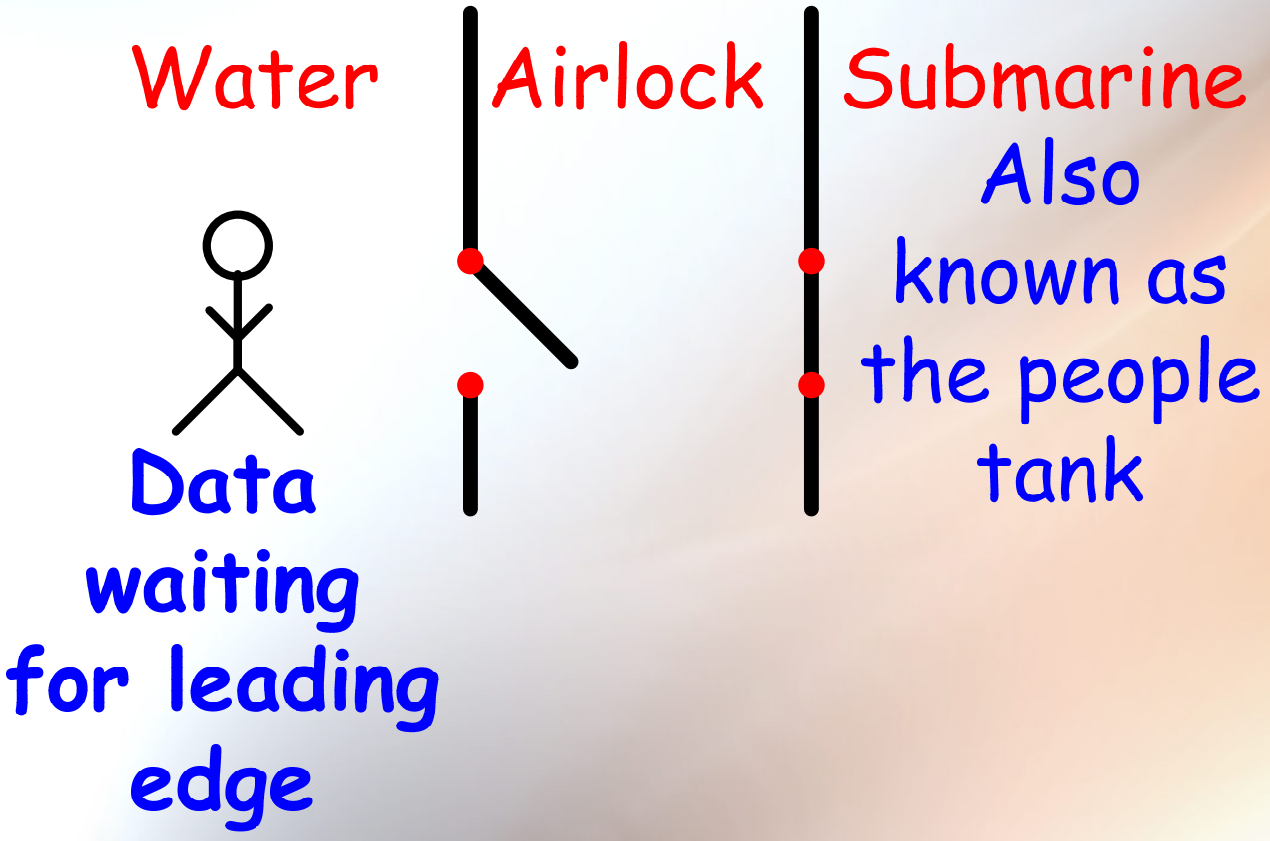
- The 2nd latch doesn't act on the JK input instructions till the Trailing Edge of the clock.

# Master/Slave JK-FF's

- The 2nd latch doesn't act on the JK input instructions till the Trailing Edge of the clock.
- This concept is known as "Data Lockout".

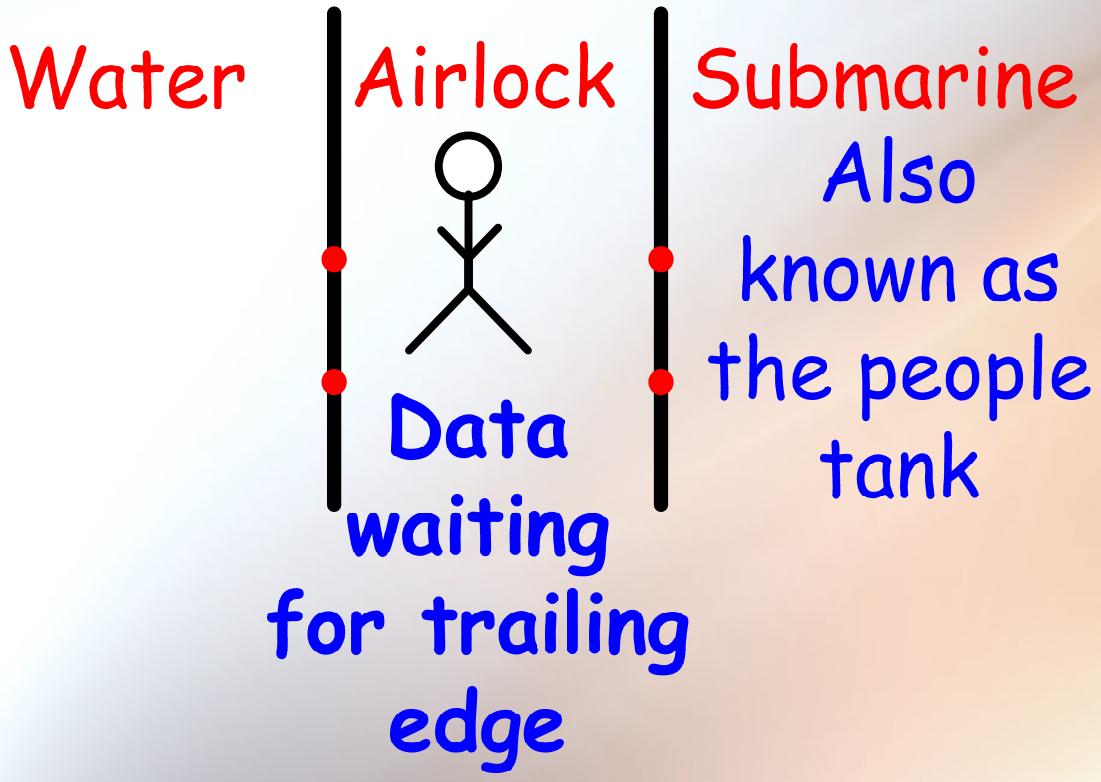
# Master/Slave JK-FF's

- **Data Lockout is analogous to a diver entering a submarine thru an airlock.**



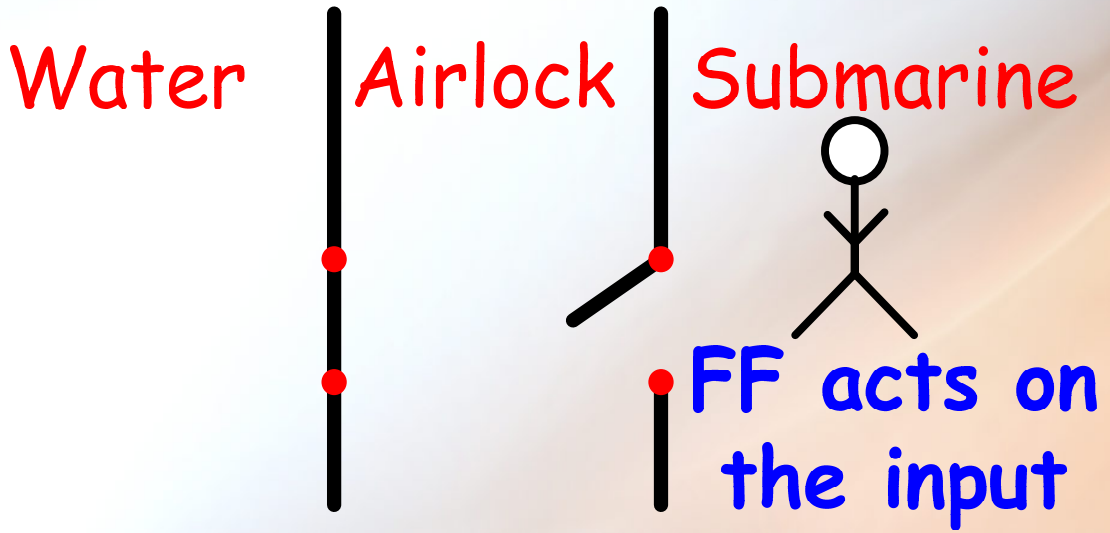
# Master/Slave JK-FF's

- **Data Lockout is analogous to a diver entering a submarine thru an airlock.**



# Master/Slave JK-FF's

- Data Lockout is analogous to a diver entering a submarine thru an airlock.



# The State Diagram

- **A very useful design tool is the State Diagram.**

# The State Diagram

- A very useful design tool is the State Diagram.
- A State Diagram allows you visualize the state progression of a state machine (Ch. 8).

# The State Diagram

- A very useful design tool is the State Diagram.
- A State Diagram allows you visualize the state progression of a state machine (Ch. 8).
- The State Diagram can precede the state table or follow it.

# The State Diagram

- A very useful design tool is the State Diagram.
- A State Diagram allows you visualize the state progression of a state machine (Ch. 8).
- The State Diagram can precede the state table or follow it.
- There are some categories of state machine which depend totally on the State Diagram (Sequence Detector, Ch. 8).

# A Two-State Diagram

- **Lets start out with a two-state diagram.**

# A Two-State Diagram

- Lets start out with a two-state diagram.
- A state is represented by a bubble with the state ID in the center.

# A Two-State Diagram

- Lets start out with a two-state diagram.
- A state is represented by a bubble with the state ID in the center.
- This ID could be a state variable, or it could be the actual state.

# A Two-State Diagram

- Lets start out with a two-state diagram.
- A state is represented by a bubble with the state ID in the center.
- This ID could be a state variable, or it could be the actual state.
- Or, it could have both as shown here.

# A Two-State Diagram

- All state diagrams MUST have a legend.

# A Two-State Diagram

- All state diagrams MUST have a legend.
- This legend is normally a rectangle placed somewhere on the periphery of the diagram

# A Two-State Diagram

- All state diagrams MUST have a legend.
- This legend is normally a rectangle placed somewhere on the periphery of the diagram
- The contents vary with the problem but normally consists of an input followed by a / and then the output.

# A Two-State Diagram

- All state diagrams MUST have a legend.
- This legend is normally a rectangle placed somewhere on the periphery of the diagram
- The contents vary with the problem but normally consists of an input (X) followed by a / and then the output (Z).

# A Two-State Diagram

- Note that the Q output of a FF (or a counter chip) is not really an output (from a state machine design perspective)
- Instead, it is a STATE.
- Outputs are the result of additional circuitry attached to the circuit.

# A Two-State Diagram

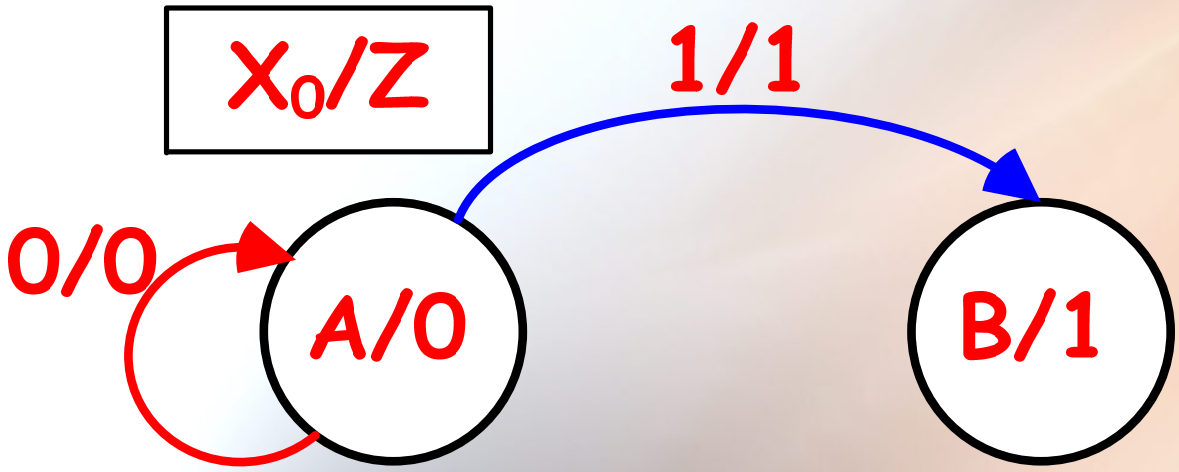
- For this example, if the input (X) is a 0 and the Present State is A, then the Next state will be A. The circuit output will be a 0.

# A Two-State Diagram

- For this example, if the input (X) is a 0 and the Present State is A, then the Next state will be A. The circuit output will be a 0.
- Note that if the legend was not present there would be no way to determine what the contents of the arrow meant!

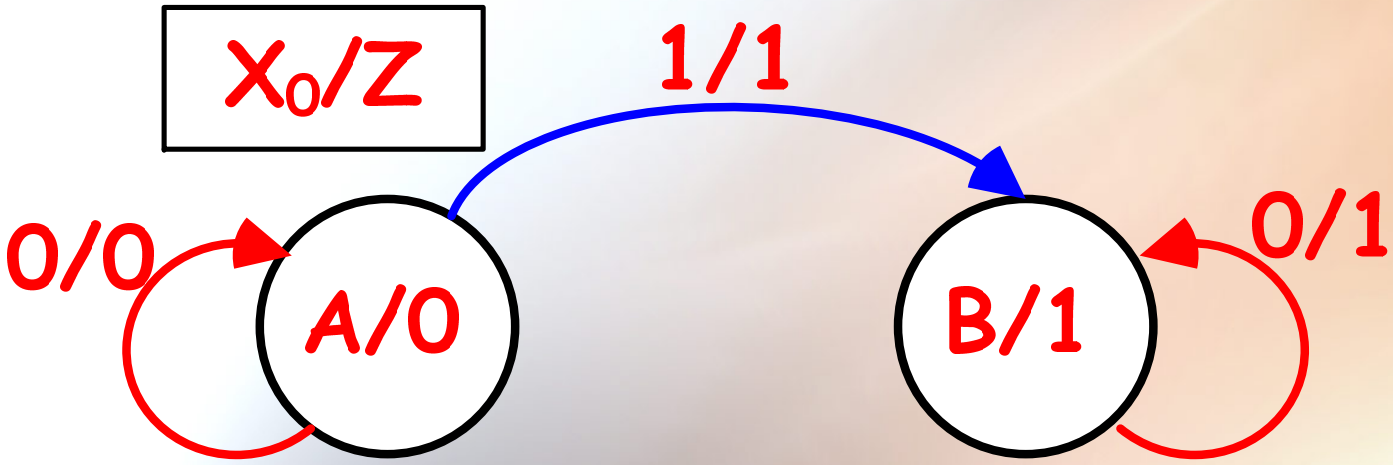
# A Two-State Diagram

- Next, if the input (X) is a 1 and the Present State is A, then the Next state will be B and the circuit output will be a 1.



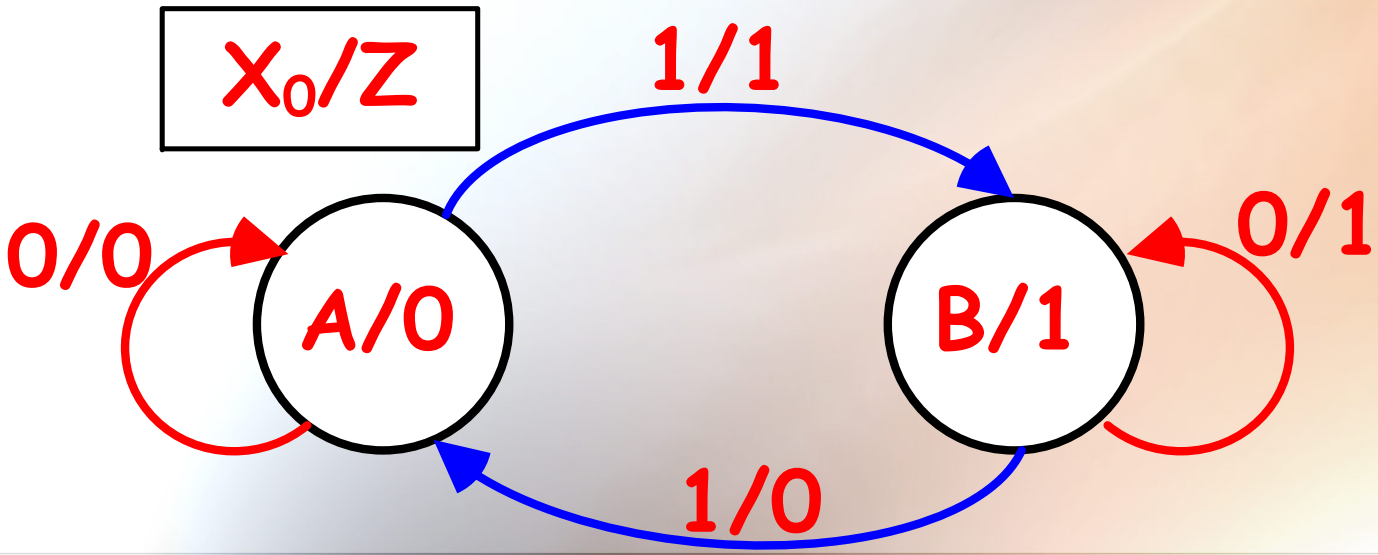
# A Two-State Diagram

- Next, if the input (X) is a 0 and the Present State is B, then the Next state will be B and the circuit output will be a 1.



# A Two-State Diagram

- And finally, if the input (X) is a 1 and the Present State is B, then the Next state will be A and the circuit output will be a 0.



# A JK FF State Diagram

- Lets examine what the state diagram for a JK-FF would look like. Note that a JK-FF, just like any FF, does not have an output, just a state out. For this reason, the legend will be J/K.

J/k

# A JK FF State Diagram

- Any single FF will result in a two-state diagram. We will call state 0, A and state 1, B.

# A JK FF State Diagram

- It might help at this point to remind ourselves of the JK Transition Table

| $Q_P$ | ⇒ | $Q_N$ | J | K |
|---|---|---|---|---|
| 0 | ⇒ | 0 | 0 | X |
| 0 | ⇒ | 1 | 1 | X |
| 1 | ⇒ | 0 | X | 1 |
| 1 | ⇒ | 1 | X | 0 |

# A JK FF State Diagram

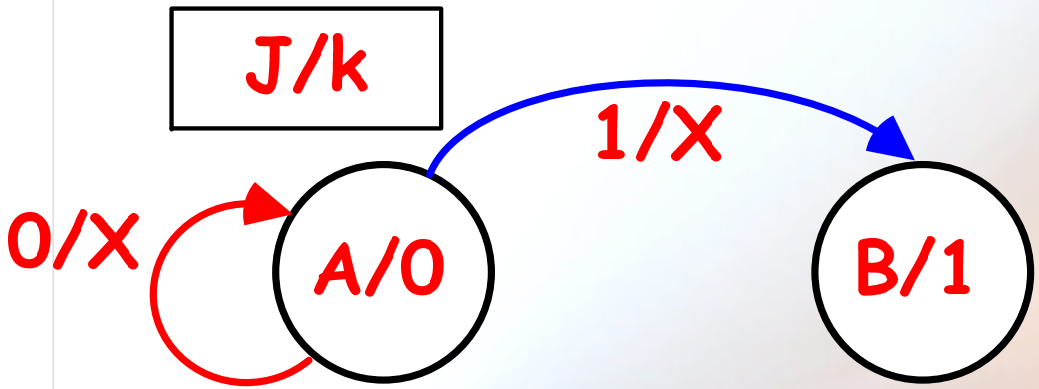
- According to the table, if J is 0 and $Q_p$ is 0, then $Q_N$ will be 0. Note that K is a don't care.

| $Q_P$ | ⇒ | $Q_N$ | J | K |
|---|---|---|---|---|
| 0 | ⇒ | 0 | 0 | X |
| 0 | ⇒ | 1 | 1 | X |
| 1 | ⇒ | 0 | X | 1 |
| 1 | ⇒ | 1 | X | 0 |

# A JK FF State Diagram

- According to the table, if J is 1 and $Q_p$ is 0, then $Q_N$ will be 1.

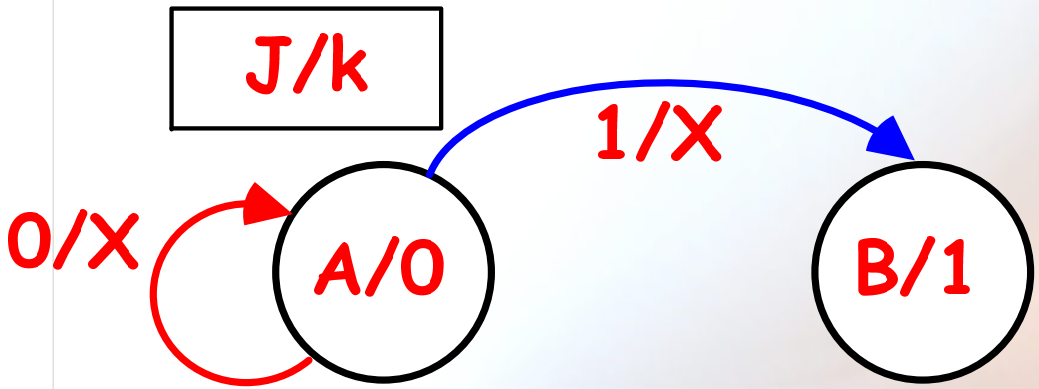
| $Q_P$ | ⇒ | $Q_N$ | J | K |
|---|---|---|---|---|
| 0 | ⇒ | 0 | 0 | X |
| 0 | ⇒ | 1 | 1 | X |
| 1 | ⇒ | 0 | X | 1 |
| 1 | ⇒ | 1 | X | 0 |



# A JK FF State Diagram

- According to the table, if J is 1 and $Q_p$ is 0, then $Q_N$ will be 1.

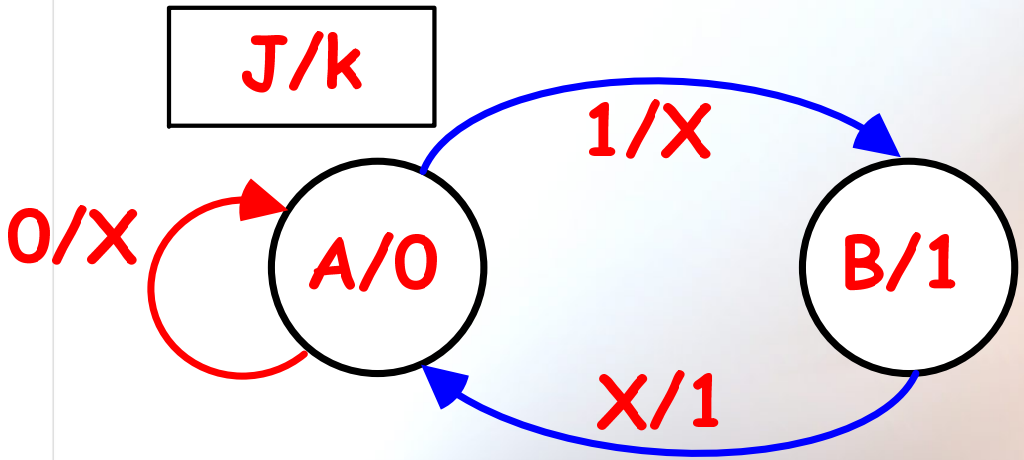
| $Q_P$ | ⇒ | $Q_N$ | J | K |
|---|---|---|---|---|
| 0 | ⇒ | 0 | 0 | X |
| 0 | ⇒ | 1 | 1 | X |
| 1 | ⇒ | 0 | X | 1 |
| 1 | ⇒ | 1 | X | 0 |



# A JK FF State Diagram

- According to the table, if K is 1 and $Q_p$ is 1, then $Q_N$ will be 0. Note that J is now a don't care.

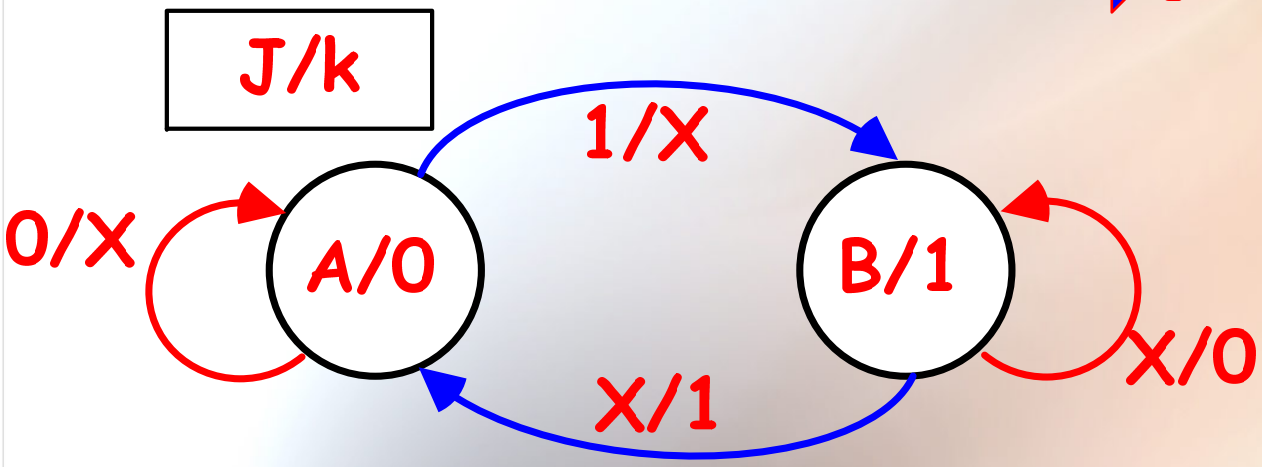
| $Q_P$ | ⇒ | $Q_N$ | J | K |
|---|---|---|---|---|
| 0 | ⇒ | 0 | 0 | X |
| 0 | ⇒ | 1 | 1 | X |
| 1 | ⇒ | 0 | X | 1 |
| 1 | ⇒ | 1 | X | 0 |



# A JK FF State Diagram

- And finally, if K is 0 and $Q_p$ is 1, then $Q_N$ will be 1.

| $Q_P$ | ⇒ | $Q_N$ | J | K |
|---|---|---|---|---|
| 0 | ⇒ | 0 | 0 | X |
| 0 | ⇒ | 1 | 1 | X |
| 1 | ⇒ | 0 | X | 1 |
| 1 | ⇒ | 1 | X | 0 |



# Problem Example

- Consider a sequential circuit having one input (x), 2 states ($y_2$ and $y_1$), and one output variable. (Note: The book got the $y_2$ and $y_1$ backwards.
- Make the indicated state assignments. The initial state of the machine is $y^0$ = A.

| State | $y_2y_1$ |
|---|---|
| A | 00 |
| B | 01 |
| C | 10 |
| D | 11 |

# Problem Example

- If the circuit has the following input (x) sequence applied to it: X=0110101100, the following behavior will be observed:

| Trigger | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| x | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | |
| PS | A | D | B | A | D | B | B | A | C | C | C |
| NS | D | B | A | D | B | B | A | C | C | C | |
| z | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | |

| State | $y_2y_1$ |
|---|---|
| A | 00 |
| B | 01 |
| C | 10 |
| D | 11 |

# Problem Example

- Using the provided answer, create a state table. The left side of the table starts the same as always. The state assignment table was used to assign the state variables.

| State | $y_2y_1$ |
|---|---|
| A | 00 |
| B | 01 |
| C | 10 |
| D | 11 |

| x | $y_2$ | | $y_1$ | $Y_2$ | $Y_1$ | z |
|---|---|---|---|---|---|---|
| 0 | 0 | A | 0 | | | |
| 0 | 0 | B | 1 | | | |
| 0 | 1 | C | 0 | | | |
| 0 | 1 | D | 1 | | | |
| 1 | 0 | A | 0 | | | |
| 1 | 0 | B | 1 | | | |
| 1 | 1 | C | 0 | | | |
| 1 | 1 | D | 1 | | | |

# Problem Example

- **Use the behavior table to complete the next state and output columns:**

| x | $y_2$ | | $y_1$ | $Y_2$ | $Y_1$ | z |
|---|---|---|---|---|---|---|
| 0 | 0 | A | 0 | D | | 0 |
| 0 | 0 | B | 1 | | | |
| 0 | 1 | C | 0 | | | |
| 0 | 1 | D | 1 | | | |
| 1 | 0 | A | 0 | | | |
| 1 | 0 | B | 1 | | | |
| 1 | 1 | C | 0 | | | |
| 1 | 1 | D | 1 | | | |

| Trig | 0 | 1 | ... |
|---|---|---|---|
| x | 0 | 1 | ... |
| PS | A | D | ... |
| NS | D | B | ... |
| z | 0 | 1 | ... |

# Problem Example

- **Use the behavior table to complete the next state and output columns:**

| x | $y_2$ | | $y_1$ | $Y_2$ | $Y_1$ | z |
|---|---|---|---|---|---|---|
| 0 | 0 | A | 0 | D | | 0 |
| 0 | 0 | B | 1 | | | |
| 0 | 1 | C | 0 | | | |
| 0 | 1 | D | 1 | | | |
| 1 | 0 | A | 0 | | | |
| 1 | 0 | B | 1 | | | |
| 1 | 1 | C | 0 | | | |
| 1 | 1 | D | 1 | B | | 1 |

| Trig | 0 | 1 | ... |
|---|---|---|---|
| x | 0 | 1 | ... |
| PS | A | D | ... |
| NS | D | B | ... |
| z | 0 | 1 | ... |

# Problem Example

- **Use the behavior table to complete the next state and output columns:**

| x | $y_2$ | | $y_1$ | $Y_2$ | $Y_1$ | z |
|---|---|---|---|---|---|---|
| 0 | 0 | A | 0 | D | | 0 |
| 0 | 0 | B | 1 | | | |
| 0 | 1 | C | 0 | | | |
| 0 | 1 | D | 1 | | | |
| 1 | 0 | A | 0 | | | |
| 1 | 0 | B | 1 | A | | 0 |
| 1 | 1 | C | 0 | | | |
| 1 | 1 | D | 1 | B | | 1 |

| Trig | | 2 | 3 | 4 | |
|---|---|---|---|---|---|
| x | ... | 1 | 0 | 1 | ... |
| PS | ... | B | A | D | ... |
| NS | ... | A | D | B | ... |
| z | ... | 0 | 0 | 1 | ... |

# Problem Example

- **Use the behavior table to complete the next state and output columns:**

| x | $y_2$ | $y_1$ | $Y_2$ | $Y_1$ | z |
|---|---|---|---|---|---|
| 0 | 0 A | 0 | D | | 0 |
| 0 | 0 B | 1 | | | |
| 0 | 1 C | 0 | | | |
| 0 | 1 D | 1 | | | |
| 1 | 0 A | 0 | | | |
| 1 | 0 B | 1 | A | | 0 |
| 1 | 1 C | 0 | | | |
| 1 | 1 D | 1 | B | | 1 |

| Trig | | 2 | 3 | 4 | |
|---|---|---|---|---|---|
| x | ... | 1 | 0 | 1 | ... |
| PS | ... | B | A | D | ... |
| NS | ... | A | D | B | ... |
| z | ... | 0 | 0 | 1 | ... |

Already finished columns 3 and 4 associated table rows.

# Problem Example

- **Use the behavior table to complete the next state and output columns:**

| x | $y_2$ | | $y_1$ | $Y_2$ | $Y_1$ | z |
|---|---|---|---|---|---|---|
| 0 | 0 | A | 0 | D | | 0 |
| 0 | 0 | B | 1 | B | | 1 |
| 0 | 1 | C | 0 | | | |
| 0 | 1 | D | 1 | | | |
| 1 | 0 | A | 0 | | | |
| 1 | 0 | B | 1 | A | | 0 |
| 1 | 1 | C | 0 | | | |
| 1 | 1 | D | 1 | B | | 1 |

| Trig | | 5 | 6 | 7 | |
|---|---|---|---|---|---|
| x | ... | 0 | 1 | 1 | ... |
| PS | ... | B | B | A | ... |
| NS | ... | B | A | C | ... |
| z | ... | 1 | 0 | 1 | ... |

# Problem Example

- **Use the behavior table to complete the next state and output columns:**

| x | $y_2$ | | $y_1$ | $Y_2$ | $Y_1$ | z |
|---|---|---|---|---|---|---|
| 0 | 0 | A | 0 | D | | 0 |
| 0 | 0 | B | 1 | B | | 1 |
| 0 | 1 | C | 0 | | | |
| 0 | 1 | D | 1 | | | |
| 1 | 0 | A | 0 | C | | 1 |
| 1 | 0 | B | 1 | A | | 0 |
| 1 | 1 | C | 0 | | | |
| 1 | 1 | D | 1 | B | | 1 |

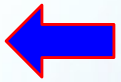
| Trig | | 5 | 6 | 7 | |
|---|---|---|---|---|---|
| x | ... | 0 | 1 | 1 | ... |
| PS | ... | B | B | A | ... |
| NS | ... | B | A | C | ... |
| z | ... | 1 | 0 | 1 | ... |

Column 6 wasn't needed.

# Problem Example

- **Use the behavior table to complete the next state and output columns:**

| x | $y_2$ | | $y_1$ | $Y_2$ | $Y_1$ | z |
|---|---|---|---|---|---|---|
| 0 | 0 | A | 0 | D | | 0 |
| 0 | 0 | B | 1 | B | | 1 |
| 0 | 1 | C | 0 | C | | 1 |
| 0 | 1 | D | 1 | | | |
| 1 | 0 | A | 0 | C | | 1 |
| 1 | 0 | B | 1 | A | | 0 |
| 1 | 1 | C | 0 | | | |
| 1 | 1 | D | 1 | B | | 1 |



| Trigger | | 8 | 9 | 10 |
|---|---|---|---|---|
| x | ... | 0 | 0 | |
| PS | ... | C | C | C |
| NS | ... | C | C | |
| z | ... | 1 | 1 | |

# Problem Example

- **Use the behavior table to complete the next state and output columns:**

| x | $y_2$ | | $y_1$ | $Y_2$ | $Y_1$ | z |
|---|---|---|---|---|---|---|
| 0 | 0 | A | 0 | D | | 0 |
| 0 | 0 | B | 1 | B | | 1 |
| 0 | 1 | C | 0 | C | | 1 |
| 0 | 1 | D | 1 | | | |
| 1 | 0 | A | 0 | C | | 1 |
| 1 | 0 | B | 1 | A | | 0 |
| 1 | 1 | C | 0 | | | |
| 1 | 1 | D | 1 | B | | 1 |

Rows 3 and 6 are not covered by the problem so are considered to be undefined or "illegal" states.

# Problem Example

- Use the behavior table to complete the next state and output columns:

| x | $y_2$ | | $y_1$ | $Y_2$ | $Y_1$ | z | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | A | 0 | D | | 0 | |
| 0 | 0 | B | 1 | B | | 1 | |
| 0 | 1 | C | 0 | C | | 1 | |
| 0 | 1 | D | 1 | A | | 0 | ← |
| 1 | 0 | A | 0 | C | | 1 | |
| 1 | 0 | B | 1 | A | | 0 | |
| 1 | 1 | C | 0 | D | | 0 | ← |
| 1 | 1 | D | 1 | B | | 1 | |

So, the designer is free to assign them in such a way as to lead to "legal" states.

# Problem Example

- **Use the state assignment table to complete the Next state columns ID's.**

| x | $y_2$ | | $y_1$ | $Y_2$ | | $Y_1$ | z |
|---|---|---|---|---|---|---|---|
| 0 | 0 | A | 0 | | D | | 0 |
| 0 | 0 | B | 1 | | B | | 1 |
| 0 | 1 | C | 0 | | C | | 1 |
| 0 | 1 | D | 1 | 0 | A | 0 | 0 |
| 1 | 0 | A | 0 | | C | | 1 |
| 1 | 0 | B | 1 | 0 | A | 0 | 0 |
| 1 | 1 | C | 0 | | D | | 0 |
| 1 | 1 | D | 1 | | B | | 1 |

| State | $y_2y_1$ |
|---|---|
| A | 00 |
| B | 01 |
| C | 10 |
| D | 11 |

# Problem Example

- Use the state assignment table to complete the Next state columns ID's.

| x | $y_2$ | | $y_1$ | $Y_2$ | | $Y_1$ | z |
|---|---|---|---|---|---|---|---|
| 0 | 0 | A | 0 | | D | | 0 |
| 0 | 0 | B | 1 | 0 | B | 1 | 1 |
| 0 | 1 | C | 0 | | C | | 1 |
| 0 | 1 | D | 1 | 0 | A | 0 | 0 |
| 1 | 0 | A | 0 | | C | | 1 |
| 1 | 0 | B | 1 | 0 | A | 0 | 0 |
| 1 | 1 | C | 0 | | D | | 0 |
| 1 | 1 | D | 1 | 0 | B | 1 | 1 |

| State | $y_2y_1$ |
|---|---|
| A | 00 |
| B | 01 |
| C | 10 |
| D | 11 |

# Problem Example

- Use the state assignment table to complete the Next state columns ID's.

| x | $y_2$ | | $y_1$ | $Y_2$ | | $Y_1$ | z |
|---|---|---|---|---|---|---|---|
| 0 | 0 | A | 0 | | D | | 0 |
| 0 | 0 | B | 1 | 0 | B | 1 | 1 |
| 0 | 1 | C | 0 | 1 | C | 0 | 1 |
| 0 | 1 | D | 1 | 0 | A | 0 | 0 |
| 1 | 0 | A | 0 | 1 | C | 0 | 1 |
| 1 | 0 | B | 1 | 0 | A | 0 | 0 |
| 1 | 1 | C | 0 | | D | | 0 |
| 1 | 1 | D | 1 | 0 | B | 1 | 1 |

| State | $y_2y_1$ |
|---|---|
| A | 00 |
| B | 01 |
| C | 10 |
| D | 11 |

# Problem Example

- Use the state assignment table to complete the Next state columns ID's.

| x | $y_2$ | | $y_1$ | $Y_2$ | | $Y_1$ | z |
|---|---|---|---|---|---|---|---|
| 0 | 0 | A | 0 | 1 | D | 1 | 0 |
| 0 | 0 | B | 1 | 0 | B | 1 | 1 |
| 0 | 1 | C | 0 | 1 | C | 0 | 1 |
| 0 | 1 | D | 1 | 0 | A | 0 | 0 |
| 1 | 0 | A | 0 | 1 | C | 0 | 1 |
| 1 | 0 | B | 1 | 0 | A | 0 | 0 |
| 1 | 1 | C | 0 | 1 | D | 1 | 0 |
| 1 | 1 | D | 1 | 0 | B | 1 | 1 |

| State | $y_2y_1$ |
|---|---|
| A | 00 |
| B | 01 |
| C | 10 |
| D | 11 |

# Problem Example

- **Before going, convert the table to a more concise form.**

| x | $y_2$ | | $y_1$ | $Y_2$ | | $Y_1$ | z |
|---|---|---|---|---|---|---|---|
| 0 | 0 | A | 0 | 1 | D | 1 | 0 |
| 0 | 0 | B | 1 | 0 | B | 1 | 1 |
| 0 | 1 | C | 0 | 1 | C | 0 | 1 |
| 0 | 1 | D | 1 | 0 | A | 0 | 0 |
| 1 | 0 | A | 0 | 1 | C | 0 | 1 |
| 1 | 0 | B | 1 | 0 | A | 0 | 0 |
| 1 | 1 | C | 0 | 1 | D | 1 | 0 |
| 1 | 1 | D | 1 | 0 | B | 1 | 1 |

| PS | | | NS<br>X = 0<br>$Y_2Y_1$ / z | NS<br>X = 1<br>$Y_2Y_1$ / z |
|---|---|---|---|---|
| $y_2$ | | $y_1$ | | |
| 0 | A | 0 | D / 0 | |
| 0 | B | 1 | | |
| 1 | C | 0 | | |
| 1 | D | 1 | | |

# Problem Example

- **Before going, convert the table to a more concise form.**

| x | $y_2$ | | $y_1$ | $Y_2$ | | $Y_1$ | z |
|---|---|---|---|---|---|---|---|
| 0 | 0 | B | 1 | 0 | B | 1 | 1 |
| 0 | 1 | C | 0 | 1 | C | 0 | 1 |
| 0 | 1 | D | 1 | 0 | A | 0 | 0 |
| 1 | 0 | A | 0 | 1 | C | 0 | 1 |
| 1 | 0 | B | 1 | 0 | A | 0 | 0 |
| 1 | 1 | C | 0 | 1 | D | 1 | 0 |
| 1 | 1 | D | 1 | 0 | B | 1 | 1 |

| PS | | | NS X = 0 | NS X = 1 |
|---|---|---|---|---|
| $y_2$ | | $y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | | |
| 1 | C | 0 | | |
| 1 | D | 1 | | |

# Problem Example

- **Before going, convert the table to a more concise form.**

| x | $y_2$ | | $y_1$ | $Y_2$ | | $Y_1$ | z |
|---|---|---|---|---|---|---|---|
| 0 | 0 | B | 1 | 0 | B | 1 | 1 |
| 0 | 1 | C | 0 | 1 | C | 0 | 1 |
| 0 | 1 | D | 1 | 0 | A | 0 | 0 |
| 1 | 0 | B | 1 | 0 | A | 0 | 0 |
| 1 | 1 | C | 0 | 1 | D | 1 | 0 |
| 1 | 1 | D | 1 | 0 | B | 1 | 1 |

| PS | | | NS<br>X = 0 | NS<br>X = 1 |
|---|---|---|---|---|
| $y_2$ | | $y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | |
| 1 | C | 0 | | |
| 1 | D | 1 | | |

# Problem Example

- **Before going, convert the table to a more concise form.**

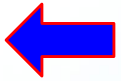
| x | $y_2$ | | $y_1$ | $Y_2$ | | $Y_1$ | z |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| 0 | 1 | C | 0 | 1 | C | 0 | 1 |
| 0 | 1 | D | 1 | 0 | A | 0 | 0 |
| | | | | | | | |
| 1 | 0 | B | 1 | 0 | A | 0 | 0 |
| 1 | 1 | C | 0 | 1 | D | 1 | 0 |
| 1 | 1 | D | 1 | 0 | B | 1 | 1 |

| PS | | | NS<br>X = 0 | NS<br>X = 1 |
|---|---|---|---|---|
| $y_2$ | | $y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | | |
| 1 | D | 1 | | |

# Problem Example

- **Before going, convert the table to a more concise form.**

| x | $y_2$ | $y_1$ | $Y_2$ | $Y_1$ | z |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | | | | |
| 0 | 1 | C | 0 | 1 | C | 0 | 1 |
| 0 | 1 | D | 1 | 0 | A | 0 | 0 |
| | | | | | |
| | | | | | |
| | | | | | |
| 1 | 1 | C | 0 | 1 | D | 1 | 0 |
| 1 | 1 | D | 1 | 0 | B | 1 | 1 |



| PS | | | NS X = 0 | NS X = 1 |
|---|---|---|---|---|
| $y_2$ | | $y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | C / 1 | |
| 1 | D | 1 | | |

# Problem Example

- **Before going, convert the table to a more concise form.**

| x | $y_2$ | | $y_1$ | $Y_2$ | | $Y_1$ | z |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| 0 | 1 | D | 1 | 0 | A | 0 | 0 |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| 1 | 1 | C | 0 | 1 | D | 1 | 0 |
| 1 | 1 | D | 1 | 0 | B | 1 | 1 |

| PS | | | NS X = 0 | NS X = 1 |
|---|---|---|---|---|
| $y_2$ | | $y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | C / 1 | D / 0 |
| 1 | D | 1 | | |

# Problem Example

- **Before going, convert the table to a more concise form.**

| x | $y_2$ | | $y_1$ | $Y_2$ | | $Y_1$ | z |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| 0 | 1 | D | 1 | 0 | A | 0 | 0 |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| 1 | 1 | D | 1 | 0 | B | 1 | 1 |

| PS | | | NS $X = 0$ | NS $X = 1$ |
|---|---|---|---|---|
| $y_2$ | | $y_1$ | $Y_2Y_1 / z$ | $Y_2Y_1 / z$ |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | C / 1 | D / 0 |
| 1 | D | 1 | A / 0 | |

# Problem Example

- **Before going, convert the table to a more concise form.**

| x | $y_2$ | $y_1$ | $Y_2$ | $Y_1$ | z |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| 1 | 1 | D | 1 | 0 | B | 1 | 1 |

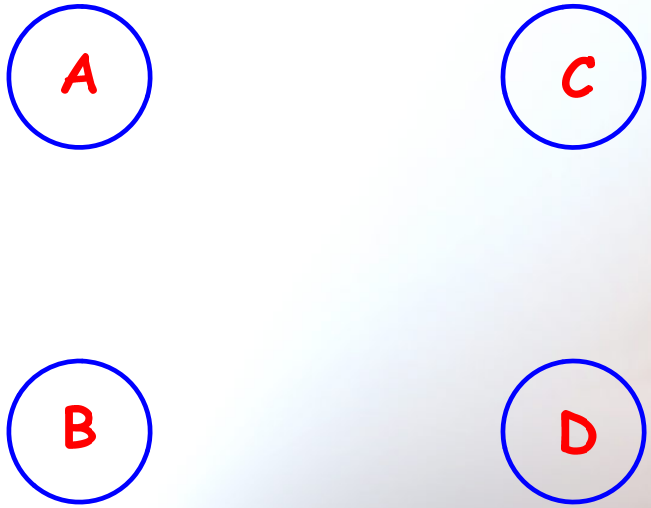
| PS | | | NS, X = 0 | NS, X = 1 |
|---|---|---|---|---|
| $y_2$ | | $y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | C / 1 | D / 0 |
| 1 | D | 1 | A / 0 | B / 1 |

# Problem Example

- Finally, create the state diagram.

| PS | | | NS X = 0 | NS X = 1 |
|---|---|---|---|---|
| $Y_2$ | | $Y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | C / 1 | D / 0 |
| 1 | D | 1 | A / 0 | B / 1 |

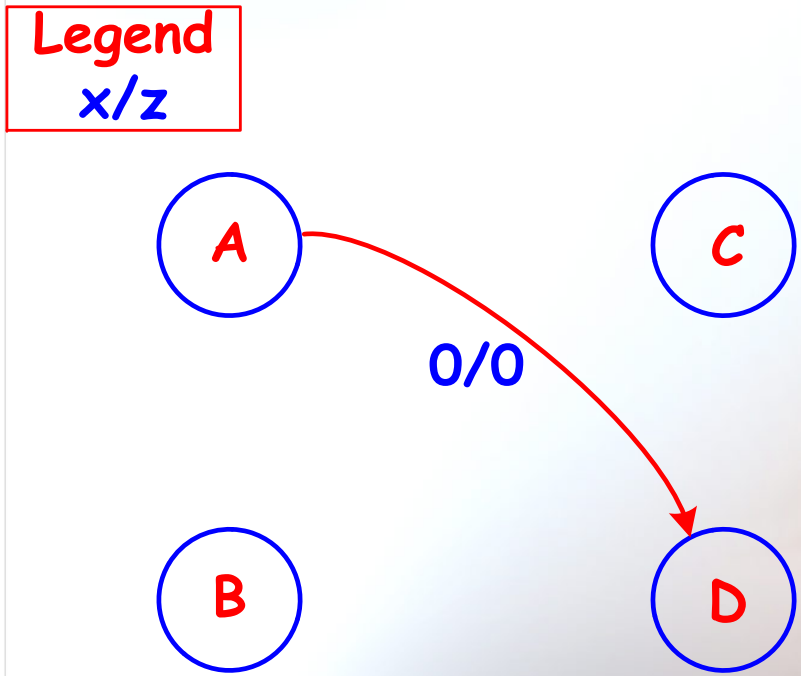
Legend
x/z



# Problem Example

- **Finally, create the state diagram.**

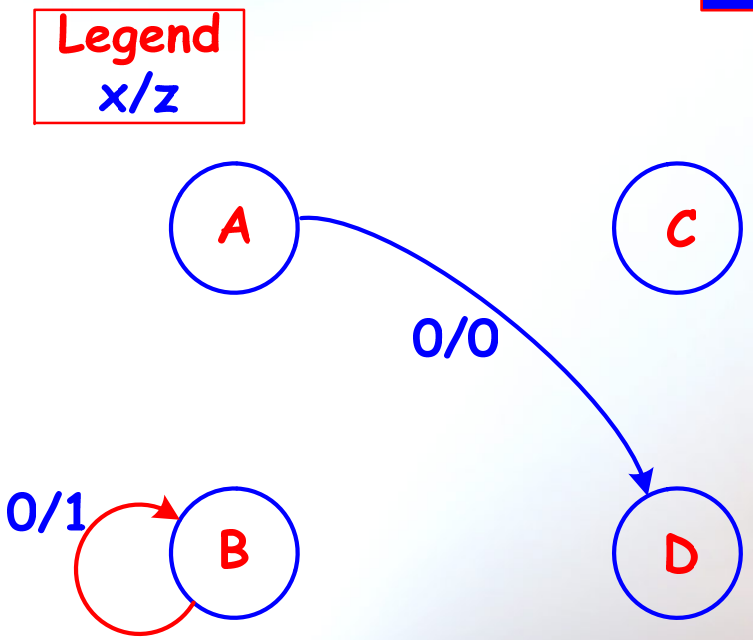
| PS | | | NS X = 0 | NS X = 1 |
|---|---|---|---|---|
| $y_2$ | | $y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | C / 1 | D / 0 |
| 1 | D | 1 | A / 0 | B / 1 |



# Problem Example

- **Finally, create the state diagram.**

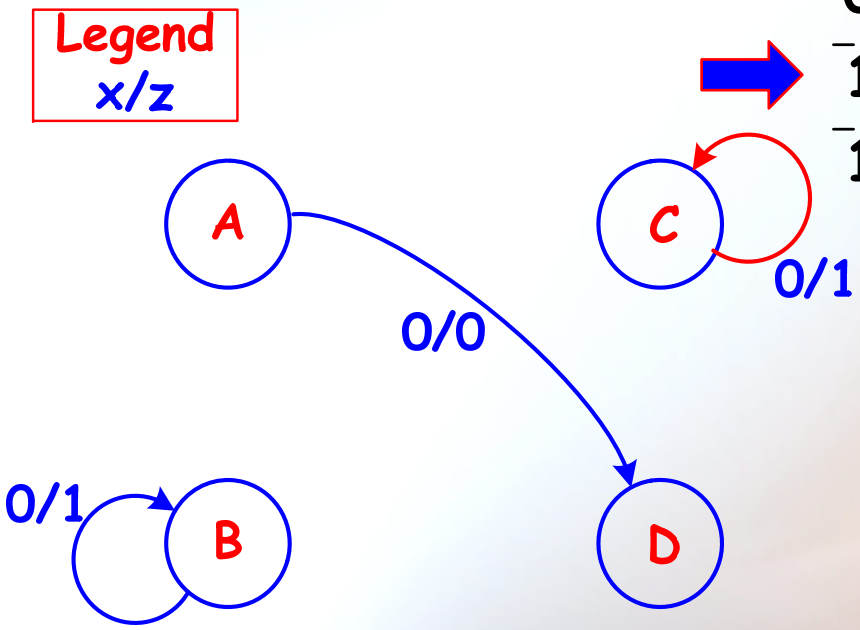
| PS | | | NS X = 0 | NS X = 1 |
|---|---|---|---|---|
| $y_2$ | | $y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | C / 1 | D / 0 |
| 1 | D | 1 | A / 0 | B / 1 |



# Problem Example

- **Finally, create the state diagram.**

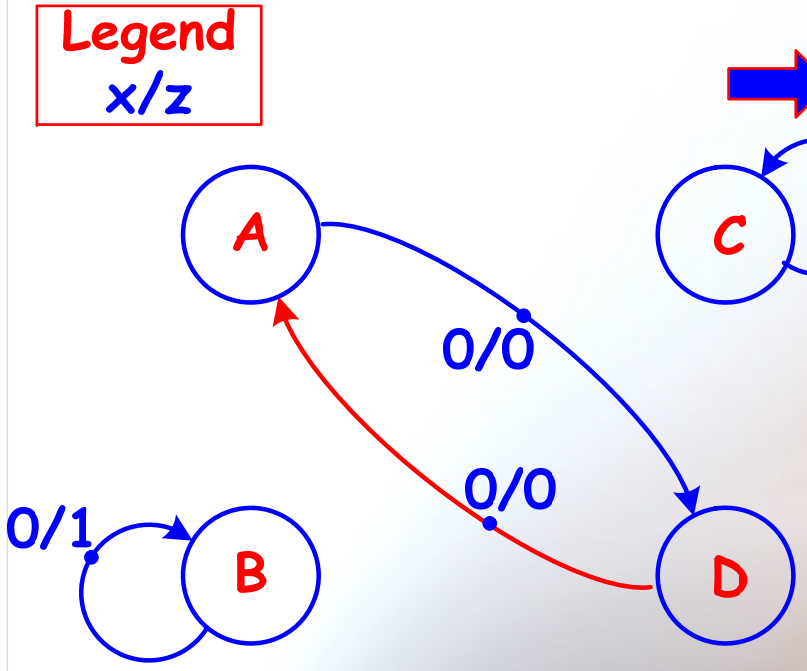
| PS | | | NS X = 0 | NS X = 1 |
|---|---|---|---|---|
| $Y_2$ | | $Y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | C / 1 | D / 0 |
| 1 | D | 1 | A / 0 | B / 1 |



# Problem Example

- Finally, create the state diagram.

| PS | | | NS X = 0 | NS X = 1 |
|---|---|---|---|---|
| $y_2$ | | $y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | C / 1 | D / 0 |
| 1 | D | 1 | A / 0 | B / 1 |

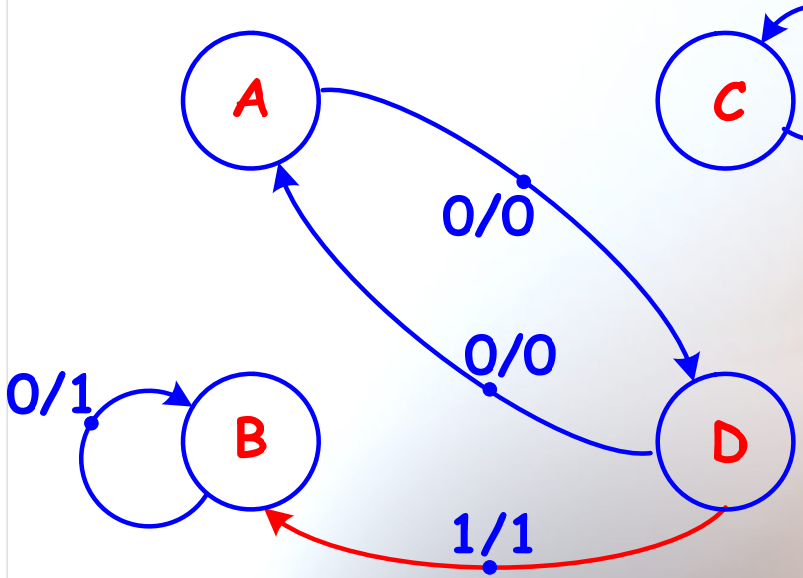


# Problem Example

- Finally, create the state diagram.

| PS | | | NS X = 0 | NS X = 1 |
|---|---|---|---|---|
| $y_2$ | | $y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | C / 1 | D / 0 |
| 1 | D | 1 | A / 0 | B / 1 |

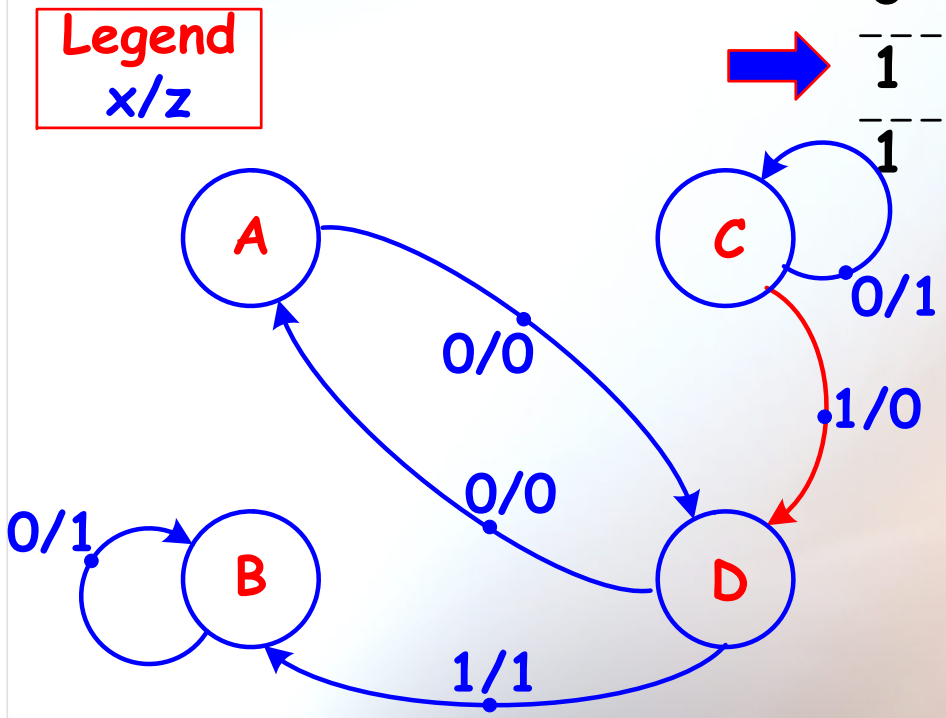
Legend
x/z



# Problem Example

- **Finally, create the state diagram.**

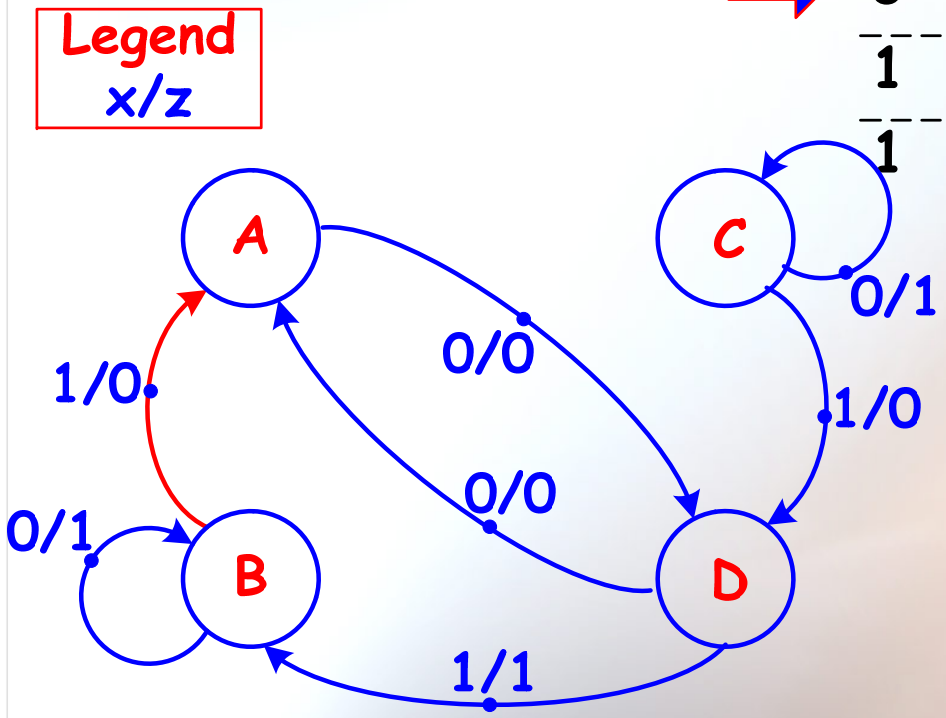
| PS | | | NS X = 0 | NS X = 1 |
|---|---|---|---|---|
| $Y_2$ | | $Y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | C / 1 | D / 0 |
| 1 | D | 1 | A / 0 | B / 1 |



# Problem Example

- **Finally, create the state diagram.**

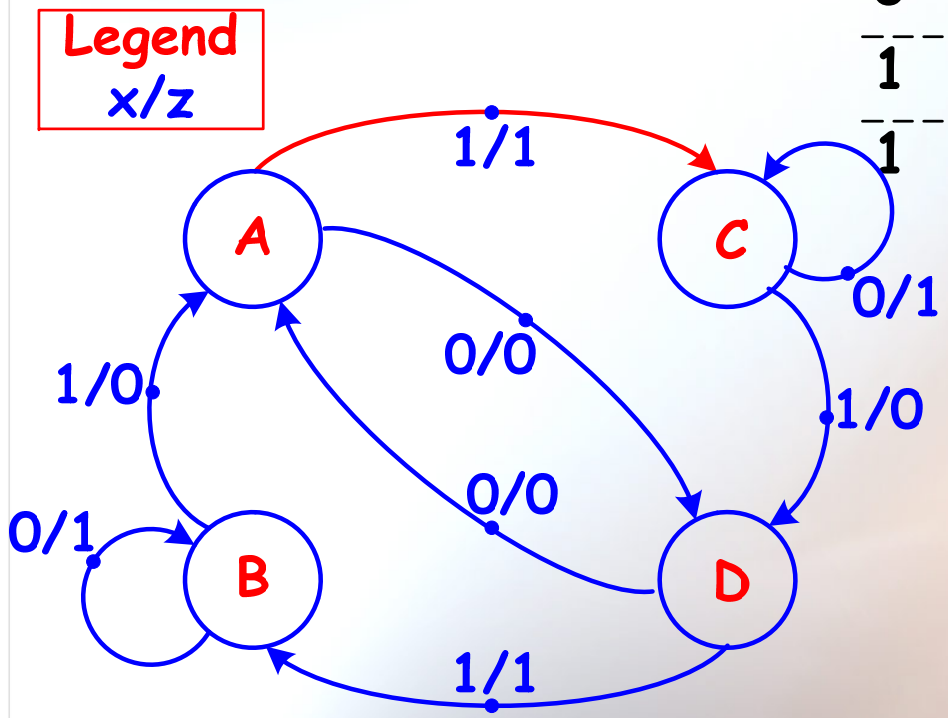
| PS | | | NS X = 0 | NS X = 1 |
|---|---|---|---|---|
| $Y_2$ | | $Y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | C / 1 | D / 0 |
| 1 | D | 1 | A / 0 | B / 1 |



# Problem Example

- **Finally, create the state diagram.**

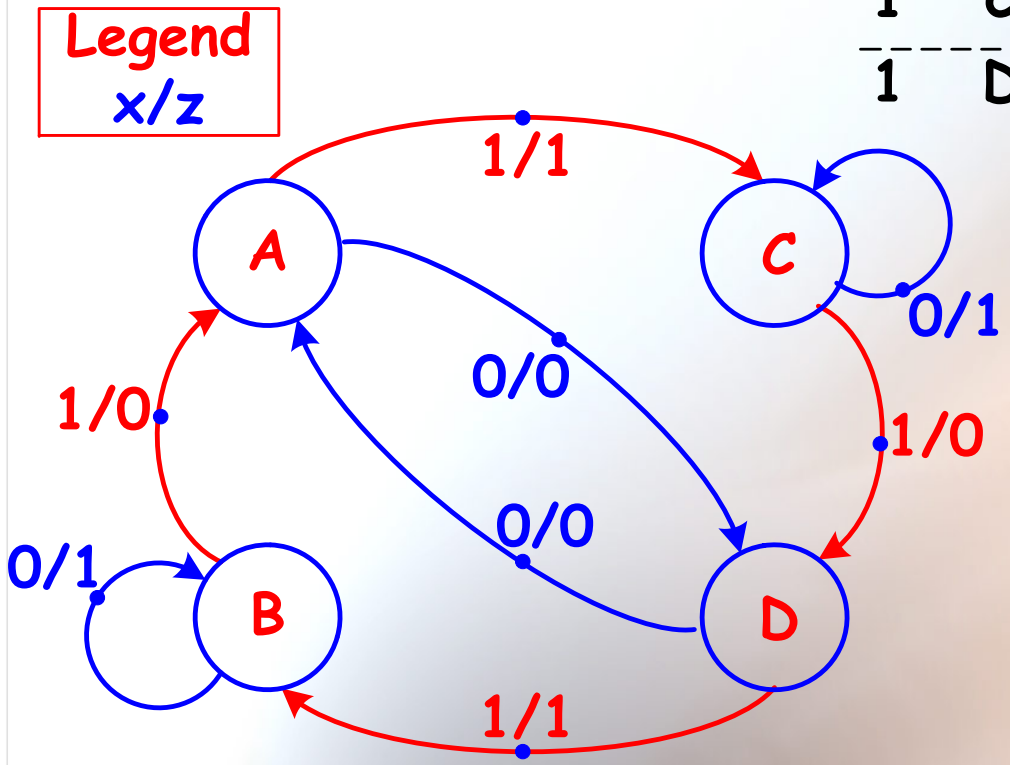
| PS | | | NS X = 0 | NS X = 1 |
|---|---|---|---|---|
| $Y_2$ | | $Y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | C / 1 | D / 0 |
| 1 | D | 1 | A / 0 | B / 1 |



# Problem Example

- **The completed solution.**

| PS | | | NS X = 0 | NS X = 1 |
|---|---|---|---|---|
| $y_2$ | | $y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | C / 1 | D / 0 |
| 1 | D | 1 | A / 0 | B / 1 |



# An alternate concise table

| PS | | | NS X = 0 | NS X = 1 |
|---|---|---|---|---|
| $y_2$ | | $y_1$ | $Y_2Y_1$ / z | $Y_2Y_1$ / z |
| 0 | A | 0 | D / 0 | C / 1 |
| 0 | B | 1 | B / 1 | A / 0 |
| 1 | C | 0 | C / 1 | D / 0 |
| 1 | D | 1 | A / 0 | B / 1 |

| PS | | | NS X = 0 | NS X = 1 | NS X = 0 | NS X = 1 |
|---|---|---|---|---|---|---|
| $y_2$ | | $y_1$ | $Y_2Y_1$ | $Y_2Y_1$ | z | z |
| 0 | A | 0 | D | C | 0 | 1 |
| 0 | B | 1 | B | A | 1 | 0 |
| 1 | C | 0 | C | D | 1 | 0 |
| 1 | D | 1 | A | B | 0 | 1 |