# FE REVIEW

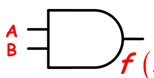
## LOGIC

1

8/24/2010

## The ‘AND’ gate

2




$f(A,B) = AB$

The ‘**AND**’ gates output will achieve its active state, “**ACTIVE HIGH**”, when **BOTH of its inputs** achieve their active state, “**ACTIVE HIGH**”.

| *A* | *B* | *AB* |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

$$f(A,B) = AB$$
$$= \sum m(3)$$

8/24/2010

## The ‘OR’ gate

3


$f(A,B) = A + B$

The ‘**OR**’ gates output will achieve its active state, “**ACTIVE HIGH**”, when **ONE OR MORE of its inputs** achieve their active state, “**ACTIVE HIGH**”.

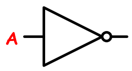
| A | B | A + B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

$$f(A,B) = A + B$$
$$= \sum m(1,2,3)$$

8/24/2010

## The 'NOT' gate (Inverter)

4



$f(A) = \overline{A}$

The '**NOT**' gates output will achieve its active state, "**ACTIVE LOW**", when **ITS SINGLE input** achieves its active state, "**ACTIVE HIGH**".

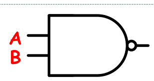
| $A$ | $f(A)$ |
|---|---|
| 0 | 1 |
| 1 | 0 |

$$f(A) = \overline{A} = \sum m(0)$$

8/24/2010

## The 'NAND' gate

5



$f(A,B) = \overline{AB}$

The '**NAND**' gates output will achieve its active state, "**ACTIVE LOW**", when **BOTH of its inputs** achieve their active state, "**ACTIVE HIGH**".

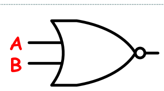
| $A$ | $B$ | $AB$ | $\overline{AB}$ |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

$$f(A,B) = \overline{AB} = \sum m(0,1,2)$$

8/24/2010

## The 'NOR' gate

6



$f(A,B) = \overline{A+B}$

The '**NOR**' gates output will achieve its active state, "**ACTIVE LOW**", when **ONE OR MORE of its inputs** achieve their active state, "**ACTIVE HIGH**".

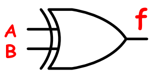
| $A$ | $B$ | $A+B$ | $\overline{A+B}$ |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 |

$$f(A,B) = \overline{A+B} = \sum m(0)$$

8/24/2010

## The 'XOR' gate

7




$$f(A,B) = A \oplus B$$
$$= A\bar{B} + \bar{A}B$$

The '**XOR**' gates output will achieve its active state, "**ACTIVE HIGH**", when **AN ODD # of its inputs** achieve their active state, "**ACTIVE HIGH**".

| $A$ | $B$ | $A \oplus B$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

$$f(A,B) = A \oplus B$$
$$= \sum m(1,2)$$

8/24/2010

## A Few Boolean Rules

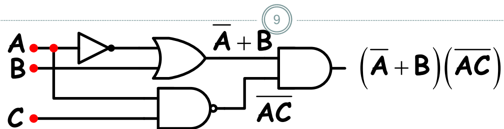
8

| | |
|---|---|
| $a + 0 = a$ | $a + \bar{a} = 1$ |
| $a + a = a$ | $a + 1 = 1$ |
| $\bar{\bar{a}} = a$ | $a + ab = a$ |
| $a + \bar{a}b = a + b$ | $ab + a\bar{b} = a$ |
| $ab + a\bar{b}c = ab + ac$ | $\overline{a + b} = \bar{a} \bullet \bar{b}$ |

8/24/2010

## Combinational Logic Example

9



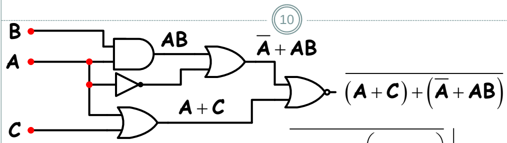

$$(\bar{A} + B)(\overline{AC})$$
$$(\bar{A} + B)(\bar{A} + \bar{C})$$
$$\bar{A}(\bar{A} + \bar{C}) + B(\bar{A} + \bar{C})$$
$$(\bar{A}\bar{A} + \bar{A}\bar{C}) + (\bar{A}B + B\bar{C})$$

$$\bar{A} + \bar{A}\bar{C} + \bar{A}B + B\bar{C}$$
$$\underbrace{\underbrace{\bar{A}}_{a} + \underbrace{\bar{A}}_{a}\underbrace{\bar{C}}_{b}}_{a+ab=a} + \bar{A}B + B\bar{C}$$
$$\bar{A} + \bar{A}B + B\bar{C}$$
$$\bar{A} + B\bar{C}$$

8/24/2010

## Another Boolean Example

10




$$\overline{(A+C)+\left(\underbrace{\underbrace{\overline{A}}_{a}+\underbrace{A}_{\overline{a}}\underbrace{B}_{b}}_{\overline{A}+B}\right)}$$

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

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

$$(\overline{A}\,\overline{C})(\overline{\overline{A}}\,\overline{B})$$

$$(\overline{A}\,\overline{C})(A\,\overline{B})$$

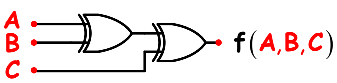
$$(\underbrace{A\overline{A}}_{0}\,\overline{B}\,\overline{C})$$

$0$

8/24/2010

## State Table Example

11




| A | B | C | f(A,B,C) |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

8/24/2010

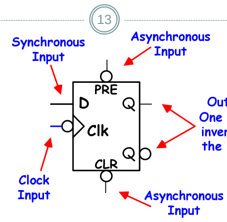
## Flip-Flops

12

8/24/2010

## FF's and some Definitions

13



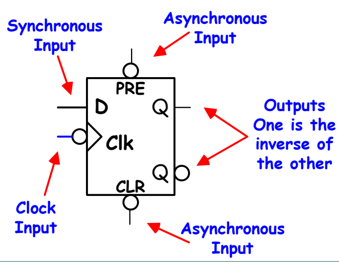

**Synchronous Input:** Controls the device based on the timing from a clock signal. When the clocking signal occurs, the synchronous input is allowed to affect the output. If the signal does not occur the synchronous input can not affect anything.



## Asynchronous Input

14

Affects the output without waiting for a clocking signal. It IGNORES the clock.

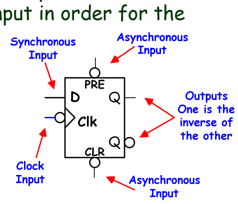





## ACTIVE HIGH OR LOW

15

**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.



## PRE and CLR

16

**'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).**

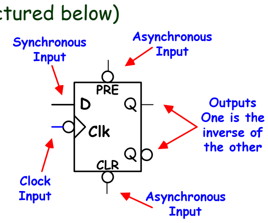


## The CLOCK

17

There are three types of clock inputs:

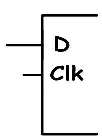
- Level Triggered (not used often)
- Leading Edge Triggered
- Trailing Edge Triggered (pictured below)



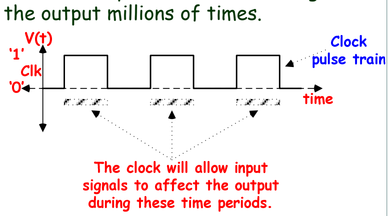



## Level Triggering

18

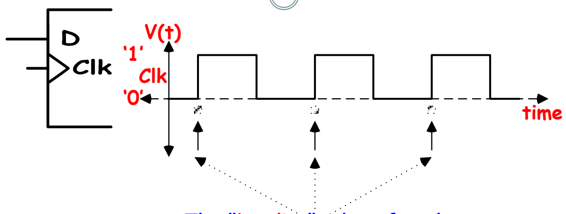



The problem with this type of triggering is that the input could change millions of times during each 'Level" period, thus affecting the output millions of times.



## Leading Edge Triggering

19

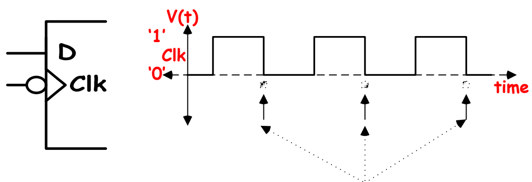



The "Leading" edge of each pulse will allow "one and only one" input event for each edge to affect the output.

8/24/2010

## Trailing Edge Triggering

20

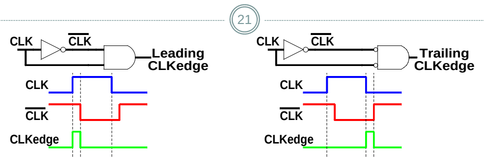



The "Trailing" edge of each pulse will allow "one and only one" input event for each edge to affect the output.

8/24/2010

## How the edge trigger is accomplished

21




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.

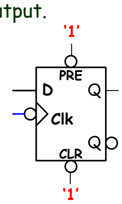
8/24/2010

## The 'D' Flip-flop

22

**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.

| D | $Q_P$ | $Q_N$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |




8/24/2010

## Present State and Next State

23

- $Q_P$ = Q present = Present state:
  The state of the output before the clock signal.
- $Q_N$ = Q next = 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!

8/24/2010

## D FF wrap up

24

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

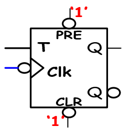
We note that whenever the **D input** is a '**0**', the output will be '**Reset**', while whenever the **D input** is a '**1**', the output will be '**Set**'.

The D FF is 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.

8/24/2010

## The 'T' Flip-flop

25



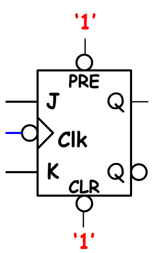

| T | $Q_p$ | $Q_N$ | |
|---|---|---|---|
| 0 | 0 | 0 | Hold |
| 0 | 1 | 1 | |
| 1 | 0 | 1 | Toggle |
| 1 | 1 | 0 | |

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.
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.

8/24/2010

## The 'JK' Flip-flop

26




| 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 | |

8/24/2010

## The JK used as a T-FF

27

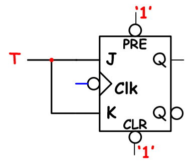
Before simplifying the JK truth table we can note that the table looks like it is made up of both the T and the D flip-flops.

8/24/2010

## The JK used as a T-FF

28

- 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.



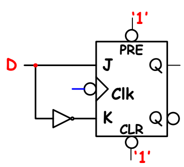

| Row | J | K | $Q_p$ | $Q_N$ | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | Hold |
| 1 | 0 | 0 | 1 | 1 | |
| 2 | | | | | |
| 3 | | | | | |
| 4 | | | | | |
| 5 | | | | | |
| 6 | 1 | 1 | 0 | 1 | Toggle |
| 7 | 1 | 1 | 1 | 0 | |

8/24/2010

## The JK used as a D-FF

29

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 | |
| 4 | 1 | 0 | 0 | 1 | Set |
| 5 | 1 | 0 | 1 | 1 | |
| 6 | | | | | |
| 7 | | | | | |

8/24/2010

## Simplifying the JK truth table

30

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.

8/24/2010

## The State Transition Table

31

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

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

8/24/2010

## The '0' to '0' transition

32

Next we look for the two rows that have transitions from state '0' to state '0'. It doesn't matter what K is 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 | | |

8/24/2010

## The '0' to '1' transition

33

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 |

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

8/24/2010

## The '1' to '0' transition

34

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 |

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



## The '1' to '1' transition

35

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 | | | | |

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



## Timing Diagrams

36

- The figure below is a "Timing" diagram for a JK-FF. Note that the 'Pre' and 'Clr' inputs as well as the leading edge trigger are demonstrated as well.