**XOR and XNOR Gates**

\[ F_{\text{xor}} = \overline{ab} + a\overline{b} \quad \Rightarrow \quad F_{\text{xor}} = \overline{ab} + ab = (\overline{a}b)(a\overline{b}) \]

**PMOS:**
\[ F_{\text{xor}} = \overline{a}b + a\overline{b} \]
\[ = (\overline{a} + b)(\overline{a} + b) \]

**NMOS:**
\[ F_{\text{xor}} = ab + a\overline{b} \]

\[ F_{\text{xnor}} = \overline{a}b + ab \]

**PMOS:**
\[ F_{\text{xnor}} = \overline{a}b + a\overline{b} \]
\[ = (\overline{a} + b)(\overline{a} + b) \]

**NMOS:**
\[ F_{\text{xnor}} = ab + a\overline{b} \]

---

**Multiplexer (Two Input)**

\[ F_{\text{mux}} = \overline{a}s + b\overline{s} \]

**PMOS:**
\[ F = \overline{a}s + b\overline{s} \]

**NMOS:**
\[ F = as + b\overline{s} \]

\[ = (a + \overline{s})(\overline{b} + s) \]
FLIP-FLOPS AND LATCHES

DEF:
- Combinational logic ext: no feedback, output = f(inputs)
- Sequential: new output = f(inputs, old outputs)
use positive feedback

Basic Bistable ext:
A bistable ext can be triggered to move from one state to another by moving past Vh for a duration of 2\(\tau_p\)
\(\tau_p = (\text{rise} + \text{fall}) / 2\)
- In the absence of a trigger, the bistable remains in that state

SR latch w/ NOR gates:

Delay from S\to\overline{Q} or R\to\overline{Q} is one NOR delay
" " S\to Q or R\to Q " two "

SR latch w/ NAND gates:
S\to Q or R\to \overline{Q} : one NAND delay
S\to \overline{Q} or R\to Q : two "
Ex. 5.4 NAND-based SR Latch, \( t_p = 400 \text{ ps}, \quad C_L = 100 \text{ fF} \)

0.13 µm CMOS, \( L = 0.1 \text{ µm} \) \( W \) sizes?

\[
t_{PHL} = t_{PLH} = 0.7 \sqrt{R \text{ eff} C_L} = 200 \text{ ps}
\]

\[
t_{PHL} = 0.7 \left( \frac{2 \times 1.25 \text{ kΩ}}{2} \right) \frac{L}{W}
\]

\[
\Rightarrow W_L = \frac{(0.7)(25)(100)}{200} = 8.75
\]

\[
\Rightarrow W_L = 0.875 \text{ µm}
\]

\[
t_{PLH} = (0.7) \left( \frac{30 \text{ kΩ}}{2} \right) \frac{L}{W_L}
\]

\[
\Rightarrow W/L = \frac{(0.7)(30)(100)}{200} = 10.5 \Rightarrow W = 1.05 \text{ µm}
\]

JK Flip-Flop. Two additional feedback lines remove ambiguity

\[
S \oplus R \Rightarrow J \oplus K
\]

<table>
<thead>
<tr>
<th>Jn Kn</th>
<th>Qt+1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0</td>
<td>( \overline{Q}_n ) (Reset)</td>
</tr>
<tr>
<td>0 1</td>
<td>0 (Set)</td>
</tr>
<tr>
<td>1 0</td>
<td>1 (Set)</td>
</tr>
<tr>
<td>1 1</td>
<td>( Q_n )</td>
</tr>
</tbody>
</table>

CMOS implementation
**JK Master-Slave Flip Flop**

Removes minimum width limit of clock.

![JK Master-Slave Flip Flop Diagram](image)

**Operation:**

1. **CK** rises, **\overline{CK}** goes down and disables the slave NAND gate.
2. **CK** enables the input NAND gates of the master.
3. The influence of **J** and **K** is entered into the master.
4. On the falling edge of **CK**, the master is disabled and the slave enabled.
5. The state of the master is transferred to the slave.

**One-catch problem:**

With **CK** high and the slave in the reset state, if the **J** input gate is enabled for a short while and returns to 0, the high value on **J** will propagate into the master and lock in place.

**JK Edge-Triggered Flip-Flop**

![JK Edge-Triggered Flip Flop Diagram](image)

**Operation:**

- **Set up time**: Time **JK** must be stable before **CK**.
- **Hold time**: Time **JK** must be stable after **CK**.

<table>
<thead>
<tr>
<th>Jn</th>
<th>Kn</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Qn</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Qn</td>
</tr>
</tbody>
</table>

1. With **CK** high, entry into the input NAND gates is controlled by **J** and **K**.
2. When **CK** goes low, the input NAND gates are disabled and the output of flip-flop changes due to the state of the **J** and/or **K** inputs prior to **CK** going low.
D Flip-Flops and Latches (D-Flops)

D-Latch -

```
D Q
CK Q
```

"Transparent" to the input.

Edge-triggered D-Flop -

```
D Q
CK Q
```

Comparison between a D-latch and D-flop -

Timing Issues -

\[ T_{\text{setup}} = \text{time meaningful data must be stable before the clock} \]

\[ T_{\text{hold}} = \text{"""""""""""""""""""""" after""""""""""""""

\[ T_{d-q} = \text{delay from the time the clock arrives to the point when } Q \text{ stabilizes}\]

\[ T_{d-q} = \text{time it takes the data to propagate through the D-latch} \]

Implementation of a D-latch -