Sequential Logic

Abanoub Emad Hanna

Important Definitions

Combinational vs Sequential

- **Combinational Logic:** Output depends only on the current input.
- **Sequential Logic:** Output depends on the current and previous inputs, thus it has memory.

Static vs Dynamic

- **Combinational Logic:**
  - Static: No clock input.
  - Dynamic: Clock input is required.
- **Sequential Logic:**
  - Static: Feedback used to retain output indefinitely.
  - Dynamic: Temporary charge on a capacitor.

Synchronous vs Asynchronous

- **Synchronous:** Clock-driven.
- **Asynchronous:** Event-driven.

Latches

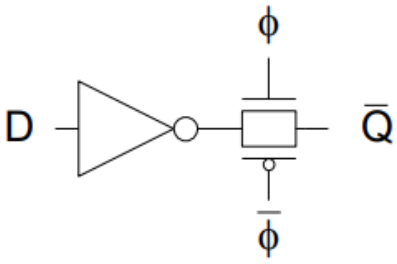
Latches are level-sensitive devices. Data passes only when transparent and is stored when opaque.

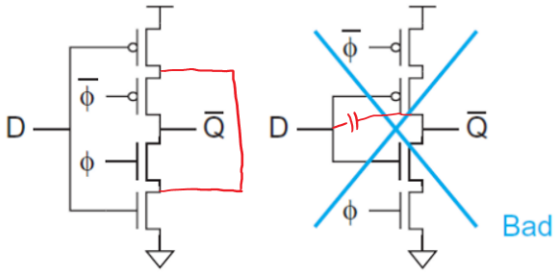
Types

- **Positive Latch:**
  - Transparent when CLK is high.
  - Setup & hold times are defined at the negative edge of CLK.
- **Negative Latch:**
  - Transparent when CLK is low.
  - Setup & hold times are defined at the positive edge of CLK.
- **Note:** Setup & hold times are defined when we are going to store the input, so in the case of a positive-edge triggered FF, setup & hold times are defined at the positive edge of CLK. In the case of a positive latch, we want to store when opaque, so setup & hold times are defined at the negative edge of CLK.

Design

Dynamic Latch

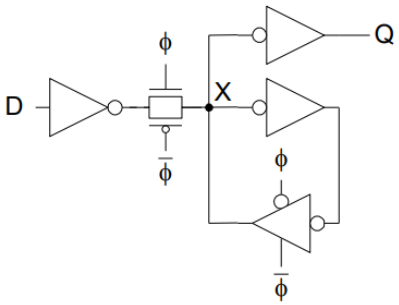




If the red wire is removed, this topology is known as C^2MOS latch, which is immune to clock skew. Second topology is bad due to the capactior between the output and the input.

Static Latch

- Buffered input and ouptut so no backdriving.
- large and slightly slow but very robust.
- High clock loading.
- Recommended for all but performance- or area-critical designs



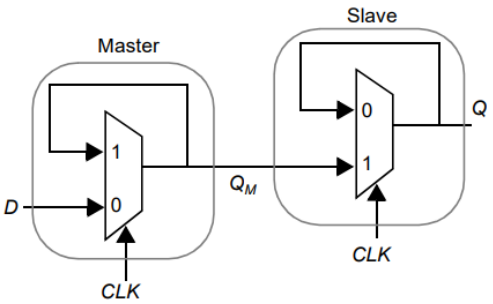
Flip-Flops

Flip-Flops are (+ve or -ve) edge-triggered devices. Data passes only at the edge of the clock signal and stored till the next cycle.

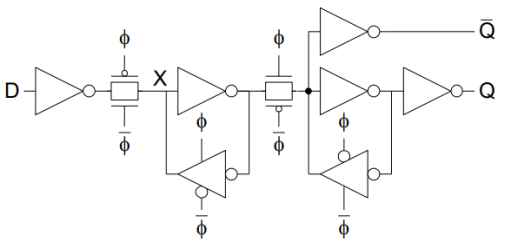
Design

Master-slave configuration is the most popular. For a positive-edge triggered FF, the master is negative latch and the slave is positive latch. When clock is low, master is transparent (captures the input) and slave is opaque (holds old value; ouptut = Q_n). When clock is high, slave latch is transparent (ouptut = D).
Note: Slave latch controls the ouptut, so it is same type as the FF.

MUX-Based



Standard FF



Types

D Flip-Flop

$$Q_{n+1} = D$$

JK Flip-Flop

J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q_n}$

SR Flip-Flop

S	R	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	X

T Flip-Flop

T	Q_{n+1}
0	Q_n
1	$\overline{Q_n}$

Type Conversion

Steps:

1. Construct truth table for the desired FF.
2. Add Q_n column if converting from D or T.
3. Fill the values of the current FF's truth table depending on the ouput.
4. Use K-Maps or intuition to simplify the equations.

SR to D

D	S	R	Q_{n+1}
0	0	1	0
1	1	0	1

$S = D$ & R=$\overline{D}$

D to JK

J	K	Q_n	D	Q_{n+1}
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	0	0

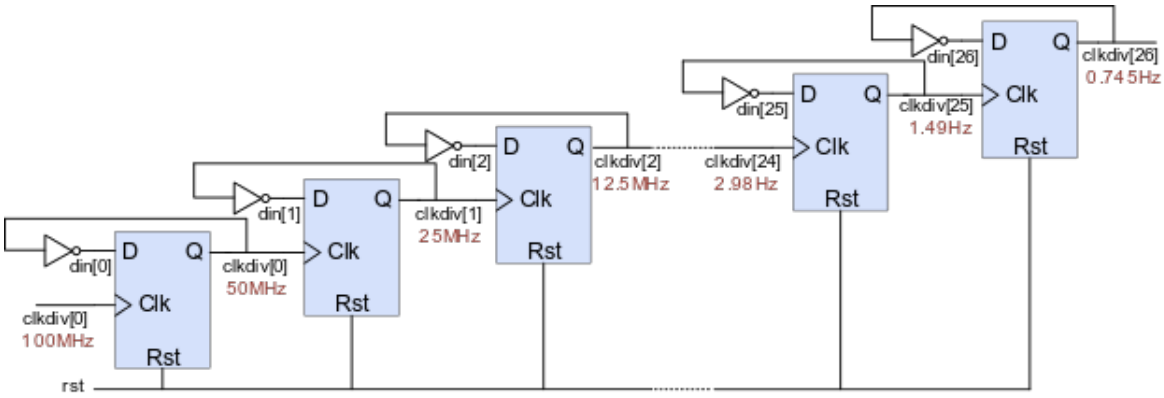
$D = J\overline{Q_n} + \overline{K}Q_n$

JK to T

T	J	K	Q_{n+1}
0	0	0	Q_n
1	1	1	$\overline{Q_n}$

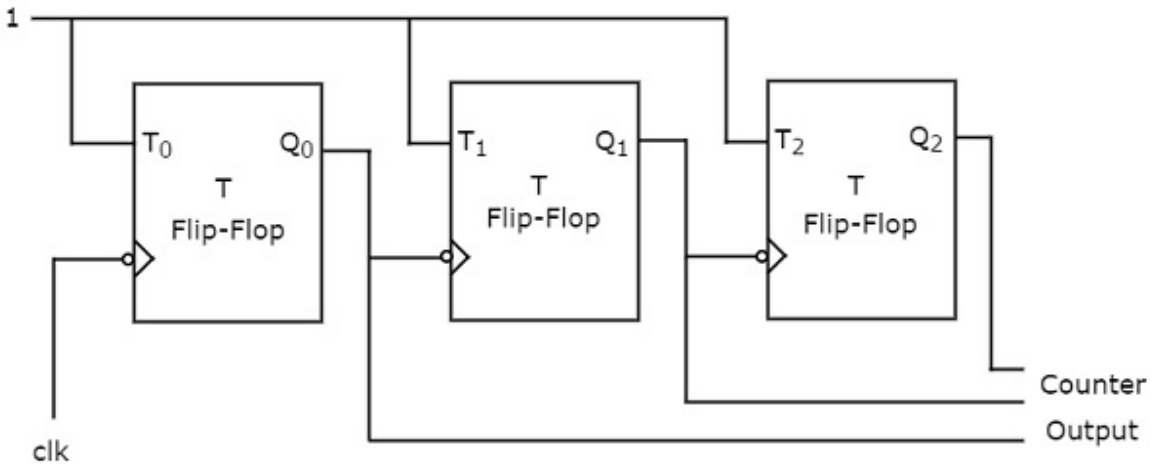
$J = T$ & R=$K = T$

Frequency Dividers



Frequency divider can also be done by making T = 1 for T FF and J = K = 1 for JK FF.

Counter



- We try to find any patterns in the truth table.
- Q0 toggles every clock cycle, so we set $T = 1$ for the first FF.
- Q1 toggles only at the falling edge of Q0, so we set $T = 1$ and invert the clock for the second FF.
- Q2 toggles only at the falling edge of Q1, so we set $T = 1$ and invert the clock for the third FF.

Q2	Q1	Q0
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1