Combinational Logic

- outputs depend only on present values of inputs

Sequential Circuit (simple example)

- outputs depend on both present and states of memory (past) elements
- clock determines when info stored in memory can change

RS latch

recall NOR gate

if \( A = 0 \), \( X = \overline{0+B} = \overline{B} \)

\[ X = \overline{A+B} \]

\[ X = \overline{B} \]
Latch has **asynchronous** response, i.e., output changes whenever the input changes.

Contrary to **flip-flop**, which requires a clock.
two stable states

\[ Q = \overline{Q} = 0 \]

\[ Q = 0 \quad Q = 1 \]

\[ \overline{Q} = 1 \quad \overline{Q} = 0 \]

stable stable

- similar to ball analogy, need a strong enough force to send the latch from one stable state into another

- there exists a min pulse length needed for switching (typically \( \mu \)s \( \text{ns} \))

Flip-flop (i.e. clocked or synchronized clock)

problems with latches: asynchronous circuits hard to combine/integrate

E.g. sequence of operations depends on exact values of prop. delays between gates

Synchronous circuits (= clocked) are usually preferred

Inputs/outputs change at well-defined times

Improvements

1) clocked RS flip-flop

State can change only when \( \text{CLK} = 1 \)

2) triggering: level-sensitive or transition-sensitive (better)

\( \text{CLK (EN)} = 1 \)

latch "transparent" when \( \text{CLK} = 0 \) or \( \overline{\text{S}} \)
Master-slave configuration

\[ S \quad Q \quad Y \quad R \quad Q \quad \overline{Q} \]

\[ \text{CLK} \quad \text{R} \quad \text{Q} \quad \text{S} \quad \overline{Q} \]

\[ \text{CLK} \quad \text{R} \quad \text{Q} \quad \text{S} \quad \overline{Q} \]

A) CLK 0→1 master enabled & slave disabled
B) CLK 1→0 master disabled & slave enabled

- Q ends up controlled by S, R when CLK \( \overline{E} \) only
- switch for multiple FF's can occur at the same time

Symbol

\[ S \quad Q \]
\[ \overline{Q} \]
\[ \text{CLK} \]

\[ +^\text{ edge} \]
\[ \text{triggered} \]

\[ -^\text{ edge} \]
\[ \text{triggered} \]

3) direct inputs: used to set FF state asynchronously (= force a state)

\[ \text{CLR (clear)} \]

\[ R \quad Q \quad \text{CLK} \quad S \quad \overline{Q} \]

PRE \( \rightarrow 1 \): Q \( \rightarrow 1 \) regardless
CLR \( \rightarrow 1 \): Q \( \rightarrow 0 \) of CLK, S, R

RRE (preset)