Sequential logic

**Asynchronous sequential logic** – state changes may occur whenever inputs change (elements may be simple wires or delay elements)

**Synchronous sequential logic** – state changes occur in lock step across all storage elements (using a clock signal – a periodic waveform)
Basis of sequential circuits: the R-S latch

Cross-coupled NOR gates

can force output to 0 (reset) or 1 (set)

fundamental component of ALL latches and flip-flops
Two stable states when $R=S=0$
S changes 0→1
R changes 0 → 1
S and R = 1

Inconsistent values
S and R change $1 \rightarrow 0$
Summary: the R-S latch

Timing waveform
Gated R-S Latch

CLK operates as R-S latch

R and S better not both be 1 here
Gated D Latch

\[ \begin{array}{c|c|c}
    \text{Clk} & D & Q(t+1) \\
    \hline
    0 & -- & Q(t) \\
    1 & 0 & 0 \\
    1 & 1 & 1 \\
\end{array} \]
Latches vs Flip-Flops

- Behavior is the same unless input changes while the clock is high.

Positive edge-triggered flip-flop

Transparent (level-sensitive) latch

Diagram illustrating the timing and behavior of latches and flip-flops with respect to clock inputs and data signals.
Master Slave Flip-Flops

Negative edge-triggered Flip-Flop
A Smaller Negative edge-triggered flip-flop

Sensitive to inputs only near edge of clock signal

4-5 gate delays

Characteristic equation: $Q(t+1) = D(t)$

Setup and hold times necessary to successfully latch the input
Analysis of negative edge-triggered flip-flop

When Clk=0 two stable states

newD
Analysis of negative edge-triggered flip-flop

Hold or setup time violation: 
D changes before the effects of the clock edge have propagated through flip-flop
Clocking Terminology

clock: periodic event, causes state of memory element to change can be rising edge or falling edge or high level or low level

setup time: minimum time before the clocking event by which the input must be stable ($T_{su}$)

hold time: minimum time after the clocking event for which the input must remain stable ($T_{h}$)

there is a timing "window" around the clocking event during which the input must remain stable and unchanged in order to be recognized
Typical timing specifications

Positive edge-triggered D flip-flop

- Setup and hold times
- Minimum clock width
- Propagation delays (low to high, high to low, max and typical)

All measurements are made from the clocking event in this case, the rising edge of the clock.
Cascaded Flip-Flops

shift register:
new value to first stage while second stage
obtains current value of first stage

\[ \text{IN} \rightarrow Q_0 \rightarrow D_1 \rightarrow Q_1 \]

\[ \text{CLK} \]

\[ \text{Clk} \]

\[ \text{IN} \]

\[ Q_0 \]

\[ Q_1 \]
Cascaded Flip-Flops (continued)

Setup/hold/propagation delays must be balanced
Works when: propagation delays far exceed hold times
clock period exceeds setup time
(guarantees following stage will latch current value
before it is replaced by new value)

assuming perfect clock distribution !!!
Cascaded Flip-Flops (continued)

A **hold time** violation can occur if the propagation delay plus logic/wire delay is less than the hold time.

A **setup time** violation can occur if the propagation delay plus logic/wire delay plus the setup time is more than the clock period.

assuming perfect clock distribution !!!
Timing problems

Setup time violations
must lengthen clock period or speedup signal, get faster logic

Hold time violations
slow down signal, slower logic

Clock skew
shifts relative time clock edge arrives at FFs
may lengthen setup and hold time requirements

Asynchronous signals
real world interfaces - real world isn't controlled by the same clock
targets to other systems with different clocks
Clock skew

Ideally – all storage elements clocked at the same time

Reality -- different wire delay to different points in the circuit causes skew between clock inputs

Effect of skew on cascaded flip-flops:
Clock skew

Can shorten time available for logic propagation

Time for logic to propagate
Strategies for minimizing clock skew

Distribute clock signals in general direction of data flow

Wires carrying clock between communicating components should be as short as possible

Make all wires from the clock source the same length

When skew is of same order as FF propagation delays, problems arise.

Worsens as systems get faster (wire delays don't improve as fast as circuit delays).
Metastability and asynchronous inputs

**Clocked synchronous circuits**
Inputs, state, and outputs sampled or changed in relation to a common reference signal (called the clock)

**Asynchronous circuits**
Inputs, state, and outputs sampled or changed independently of a common reference signal (glitches/hazards a major concern) (e.g., R-S latch)

**Asynchronous inputs to synchronous circuits**
Inputs can change at any time, will not meet setup/hold times
Dangerous, synchronous inputs are greatly preferred
Unavoidable (e.g., reset signal, memory wait, user input)
Handling asynchronous inputs

Never allow asynchronous inputs to be fanned out to more than one FF.
Different FFs could decide differently and the result could be an incorrect or illegal state.

adds delay to input into system
Synchronizer failure

When FF input changes near clock edge, the FF may enter a metastable state – neither a logic 0 nor 1 – it may stay in this state an indefinite amount of time, although this is not likely in real circuits.

small, but non-zero probability that FF output will get stuck in an in-between state

oscilloscope traces demonstrating synchronizer failure and eventual decay to steady state
Solutions to synchronizer failure

Slow down the system clock
to give synchronizer more time to decay into steady state

Use fastest possible logic in the synchronizer
this makes for a very sharp "peak" upon which to balance

Cascade two synchronizers

Probability of failure can never be made 0, but it can be substantially reduced