Sequential Circuit Design: Part 1

- Design of memory elements
  - Static latches
  - Pseudo-static latches
  - Dynamic latches
- Timing parameters
- Two-phase clocking
- Clocked inverters

Sequential Logic

\[ \Phi \]

FFs

LOGIC

\[ t_{p,\text{comb}} \]

2 storage mechanisms
- positive feedback
- charge-based
Sequencing

- **Combinational logic**
  - output depends on current inputs
- **Sequential logic**
  - output depends on current and previous inputs
  - Requires separating previous, current, future
  - Called *state* or *tokens*
  - Ex: FSM, pipeline

![Diagram of Finite State Machine and Pipeline](image)

Sequencing Cont.

- If tokens moved through pipeline at constant speed, no sequencing elements would be necessary
- Ex: fiber-optic cable
  - Light pulses (tokens) are sent down cable
  - Next pulse sent before first reaches end of cable
  - No need for hardware to separate pulses
  - But *dispersion* sets min time between pulses
- This is called *wave pipelining* in circuits
- In most circuits, dispersion is high
  - Delay fast tokens so they don’t catch slow ones.
Sequencing Overhead

• Use flip-flops to delay fast tokens so they move through exactly one stage each cycle.
• Inevitably adds some delay to the slow tokens
• Makes circuit slower than just the logic delay
  – Called sequencing overhead
• Some people call this clocking overhead
  – But it applies to asynchronous circuits too
  – Inevitable side effect of maintaining sequence

Sequencing Elements

• **Latch**: Level sensitive
  – a.k.a. transparent latch, D latch
• **Flip-flop**: edge triggered
  – A.k.a. master-slave flip-flop, D flip-flop, D register
• **Timing Diagrams**
  – Transparent
  – Opaque
  – Edge-trigger
Flip-Flop: Timing Definitions

Maximum Clock Frequency
Latch Design

- **Pass Transistor Latch**
  - Pros
    + Tiny
    + Low clock load
  - Cons
    - $V_t$ drop
    - nonrestoring
    - backdriving
    - output noise sensitivity
    - dynamic
    - diffusion input

Used in 1970’s

Latch Design

- **Transmission gate**
  + No $V_t$ drop
  - Requires inverted clock
Latch Design

- Inverting buffer
  - Restoring
  - No backdriving
  - Fixes either
    - Output noise sensitivity
    - Or diffusion input
    - Inverted output

Latch Design

- Buffered input
  - Fixes diffusion input
  - Noninverting
Latch Design

- Buffered output
  + No backdriving

- Widely used in standard cells
  + Very robust (most important)
  - Rather large
  - Rather slow (1.5 – 2 FO4 delays)
  - High clock loading

Latch Design

- Tristate feedback
  + Static
    - Backdriving risk

- Static latches are now essential
Latch Design

- Datapath latch
  - Smaller, faster
  - unbuffered input

Design of Memory Elements

Positive edge-triggered D flip-flop
Why use inverters on outputs?
Skew Problem: \( \Phi \) may be delayed with respect to \( \Phi \) (both may be 1 at the same time)

This is what happens-

Eliminating/Reducing skew:

Transmission gate acts a buffer, should have same delay as inverter
Latch design

Static D latch

“Jamb” latch

Weak inverter

Latch Design

Variant of D latch
Flip-Flop Design

- Flip-flop is built as pair of back-to-back latches

```
D -> X -> Q
```

Enable

- Enable: ignore clock when en = 0
  - Mux: increase latch D-Q delay
  - Clock Gating: increase en setup time, skew

```
Symbol     Multiplexer Design     Clock Gating Design

![](symbol1.png)  ![](mux1.png)  ![](gating1.png)

![](symbol2.png)  ![](mux2.png)  ![](gating2.png)
```
Reset

- Force output low when reset asserted
- Synchronous vs. asynchronous

Set / Reset

- Set forces output high when enabled
- Flip-flop with asynchronous set and reset
Dynamic Latches

- So far, all latches have been static-store state when clock is stopped but power is maintained
- Dynamic latches reduce transistor count
- Eliminate feedback inverter and transmission gate
- Latch value stored on the capacitance of the input (gate capacitance)

Dynamic Latch and Flip-Flop

Dynamic D latch

- Data stored as Charge on gate capacitance

Dynamic negative edge-triggered D flip-flop

- Difficult to ensure reliable operation
- Similar to DRAM
- Refresh cycles are required
**Charge-Based Storage**

Schematic diagram

**Pseudo-static Latch**

**Master-Slave Flip-Flop**

To reduce skew: generate complement of clock within the cell
Extra inverter per cell

**Overlapping Clocks Can Cause**
- Race Conditions
- Undefined Signals
Two-Phase Clocking

- Inverting a single clock can lead to skew problems
- Employ two non-overlapping clocks for master and slave sections of a flip-flop
- Also, use two phases for alternating pipeline stages
2-phase non-overlapping clocks

Important:
Non-overlap time t must be kept small

2-phase dynamic flip-flop

Input Sampled

Output Enable
**Use of “p” Leakers**

Flip-flop based on nMOS pass gates

Degraded voltage $V_{DD}-V_t$

Problem: Increased delay (extra inverter)

pMOS leaker transistors provide full-restored logic levels

No need to route $\Phi$ signals

---

**Clocked Inverters**

Similar to tristate buffer

$\Phi = 1$, acts as inverter

$\Phi = 0$, output = Z

D Latch

---
Flip-flop insensitive to clock overlap

C²MOS master-slave negative edge-triggered D flip-flop

C²MOS flip-flop

C²MOS avoids Race Conditions

(a) (1-1) overlap
(b) (0-0) overlap