Outline

• Sequencing
• Sequencing Element Design
• Max and Min-Delay
• Clock Skew
• Time Borrowing
• Two-Phase Clocking
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
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
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

- Tristate feedback
  + Static
    - Backdriving risk

- Static latches are now essential because of leakage
Latch Design

• Buffered input
  + Fixes diffusion input
  + Noninverting
  - Still potential for partial backdriving
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

• Datapath latch
  + smaller
  + faster
  - unbuffered input
Flip-Flop Design

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

• Enable: ignore clock when en = 0;
• Two approaches:
  – Mux: increase latch D-Q delay
  – Clock Gating: increase en setup time, skew
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
Sequencing Methods

- Flip-flops
- 2-Phase Latches
- Pulsed Latches
# Timing Diagrams

## Contamination and Propagation Delays

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_{pd}$</td>
<td>Logic Prop. Delay</td>
</tr>
<tr>
<td>$t_{cd}$</td>
<td>Logic Cont. Delay</td>
</tr>
<tr>
<td>$t_{pcq}$</td>
<td>Latch/Flop Clk-&gt;Q Prop. Delay</td>
</tr>
<tr>
<td>$t_{ccq}$</td>
<td>Latch/Flop Clk-&gt;Q Cont. Delay</td>
</tr>
<tr>
<td>$t_{pdq}$</td>
<td>Latch D-&gt;Q Prop. Delay</td>
</tr>
<tr>
<td>$t_{cdq}$</td>
<td>Latch D-&gt;Q Cont. Delay</td>
</tr>
<tr>
<td>$t_{setup}$</td>
<td>Latch/Flop Setup Time</td>
</tr>
<tr>
<td>$t_{hold}$</td>
<td>Latch/Flop Hold Time</td>
</tr>
</tbody>
</table>
Max-Delay: Flip-Flops

\[ t_{pd} \leq T_c - \left( t_{\text{setup}}^{\text{Q1}} + t_{\text{pd}}^{\text{D2}} \right) \]

sequencing overhead
Max Delay: 2-Phase Latches

\[ t_{pd} = t_{pd1} + t_{pd2} \leq T_c - \left( \frac{2t_{pd3}}{3} \right) \]

sequencing overhead
Max Delay: Pulsed Latches

\[ t_{pd} \leq T_c - \max \left( \frac{1}{4} \left( t_{pd} + t_{pw} + t_{setup} - t_{pdq} \right) \right) \]

sequencing overhead

\[(a) \ t_{pw} > t_{setup}\]
\[(b) \ t_{pw} < t_{setup}\]
Min-Delay: Flip-Flops

\[ t_{cd} \geq t_{hold} - t_{ccq} \]
Min-Delay: 2-Phase Latches

\[ t_{cd1}, t_{cd2} \geq t_{\text{hold}} - t_{ccq} - t_{\text{nonoverlap}} \]

Hold time reduced by nonoverlap

Paradox: hold applies twice each cycle, vs. only once for flops.

But a flop is made of two latches!
Min-Delay: Pulsed Latches

\[ t_{cd} \geq \]

Hold time increased by pulse width
Time Borrowing

• In a flop-based system:
  – Data launches on one rising edge
  – Must setup before next rising edge
  – If it arrives late, system fails
  – If it arrives early, time is wasted
  – Flops have hard edges

• In a latch-based system
  – Data can pass through latch while transparent
  – Long cycle of logic can borrow time into next
  – As long as each loop completes in one cycle
Time Borrowing Example

Loops may borrow time internally but must complete within the cycle.
2-Phase Latches
\[ t_{\text{borrow}} \leq \frac{T_c}{2} - (t_{\text{setup}} + t_{\text{nonoverlap}}) \]

Pulsed Latches
\[ t_{\text{borrow}} \leq t_{\text{pw}} - t_{\text{setup}} \]
Clock Skew

• We have assumed zero clock skew
• Clocks really have uncertainty in arrival time
  – Decreases maximum propagation delay
  – Increases minimum contamination delay
  – Decreases time borrowing
Skew: Flip-Flops

\[
t_{pd} \leq T_c - \left( \frac{t_{pdq}}{4} + t_{setup} + t_{skew} \right)
\]

sequencing overhead

\[
t_{cd} \geq t_{hold} - t_{ccq} + t_{skew}
\]
Skew: Latches

2-Phase Latches

\[ t_{pd} \leq T_c - \left( \frac{2t_{pd}}{3} \right) \]

sequencing overhead

\[ t_{cd1}, t_{cd2} \geq t_{hold} - t_{ccq} - t_{nonoverlap} + t_{skew} \]

\[ t_{borrow} \leq \frac{T_c}{2} - \left( t_{setup} + t_{nonoverlap} + t_{skew} \right) \]

Pulsed Latches

\[ t_{pd} \leq T_c - \max \left( \frac{t_{pd}}{4}, \frac{t_{pcq}}{4}, \frac{t_{setup}}{4}, \frac{t_{pw}}{4}, \frac{t_{skew}}{4} \right) \]

sequencing overhead

\[ t_{cd} \geq t_{hold} + t_{pw} - t_{ccq} + t_{skew} \]

\[ t_{borrow} \leq t_{pw} - \left( t_{setup} + t_{skew} \right) \]
Two-Phase Clocking

• If setup times are violated, reduce clock speed
• If hold times are violated, chip fails at any speed
• In this class, working chips are most important
  – No tools to analyze clock skew
• An easy way to guarantee hold times is to use 2-phase latches with big nonoverlap times
• Call these clocks $\phi_1, \phi_2$ (ph1, ph2)
Safe Flip-Flop

• Past years used flip-flop with nonoverlapping clocks
  – Slow – nonoverlap adds to setup time
  – But no hold times

• In industry, use a better timing analyzer
  – Add buffers to fast signals if hold time is at risk
Adaptive Sequencing

- Designers include timing margin
  - Voltage
  - Temperature
  - Process variation
  - Data dependency
  - Tool inaccuracies
- Alternative: run faster and check for near failures
  - Idea introduced as “Razor”
    - Increase frequency until at the verge of error
    - Can reduce cycle time by ~30%
Summary

• **Flip-Flops:**
  – Very easy to use, supported by all tools
  – With external 2-phase clocks guaranteed* functionality

• **2-Phase Transparent Latches:**
  – Lots of skew tolerance and time borrowing

• **Pulsed Latches:**
  – Fast, some skew tol & borrow, hold time risk

<table>
<thead>
<tr>
<th></th>
<th>Sequencing overhead $(T_c - t_{pd})$</th>
<th>Minimum logic delay $t_{cd}$</th>
<th>Time borrowing $t_{borrow}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flip-Flops</td>
<td>$t_{pcq} + t_{setup} + t_{skew}$</td>
<td>$t_{hold} - t_{eq} + t_{skew}$</td>
<td>0</td>
</tr>
<tr>
<td>Two-Phase Transparent Latches</td>
<td>$2t_{pdq}$</td>
<td>$t_{hold} - t_{eq} - t_{nonoverlap} + t_{skew}$ in each half-cycle</td>
<td>$t - \frac{T_c}{2} - (t_{setup} + t_{nonoverlap} + t_{skew})$</td>
</tr>
<tr>
<td>Pulsed Latches</td>
<td>$\max(t_{pdq}, t_{pcq} + t_{setup} - t_{pw} + t_{skew})$</td>
<td>$t_{hold} - t_{eq} + t_{pw} + t_{skew}$</td>
<td>$t_{pw} - (t_{setup} + t_{skew})$</td>
</tr>
</tbody>
</table>

* For sufficient period and non-overlap