11. Sequential Elements

- Last module:
  - Wire resistance and capacitance
  - RC delay
  - Wire engineering
- This module
  - Floorplanning (basic physical design – determines wires)
  - Sequential circuit design
  - Clock skew

Floorplan

- How do you estimate block areas?
  - Begin with block diagram
  - Each block has
    - Inputs
    - Outputs
    - Function (draw schematic)
    - Type: array, datapath, random logic
  - Estimation depends on type of logic

MIPS Floorplan

Area Estimation

- Arrays:
  - Layout basic cell
  - Calculate core area from # of cells
  - Allow area for decoders, column circuitry
- Datapaths
  - Sketch slice plan
  - Count area of cells from cell library
  - Ensure wiring is possible
- Random logic
  - Compare complexity to a design you have done

MIPS Slice Plan

Typical Layout Densities

- Typical numbers of high-quality layout
- Derate by 2 for class projects to allow routing and some sloppy layout.
- Allocate space for big wiring channels

<table>
<thead>
<tr>
<th>Element</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random logic (2 metal layers)</td>
<td>1000-1500 $\lambda^2$ / transistor</td>
</tr>
<tr>
<td>Datapath</td>
<td>250 – 750 $\lambda^2$ / transistor</td>
</tr>
<tr>
<td></td>
<td>Or 6 WL + 360 $\lambda^2$ / transistor</td>
</tr>
<tr>
<td>SRAM</td>
<td>1000 $\lambda^2$ / bit</td>
</tr>
<tr>
<td>DRAM</td>
<td>100 $\lambda^2$ / bit</td>
</tr>
<tr>
<td>ROM</td>
<td>100 $\lambda^2$ / bit</td>
</tr>
</tbody>
</table>
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 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

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.
Latch Design

- Transmission gate
  + No $V_t$ drop
  - Requires inverted clock

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

- Tristate feedback
  + Static
  - Backdriving risk

- Static latches are now essential

- 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

- 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 $en = 0$
  - 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

Master-Slave Flip-Flop
Illustration of delays

Max-Delay: Flip-Flops
$ t_{pd} \leq T_s + t_{setup} + t_{prop} $
Max Delay: 2-Phase Latches

\[ t_{pd} = t_{pd1} + t_{pd2} \leq T_c - (2 t_{pdq}) \]

__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: 2-Phase Latches

\[ t_{pd} \leq T_c = \text{max}(t_{pth1}, t_{pth2} + t_{hold} + t_{pw}) \]

Min-Delay: Flip-Flops

\[ t_{pd} \geq t_{hold} - t_{cqs} \]

Hold time increased by pulse width

Min-Delay: Pulsed Latches

\[ t_{pd} \geq t_{hold} - t_{cqs} + t_{pw} \]

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

How Much Borrowing?

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

Skew: Latches

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

- In class, use flip-flop with nonoverlapping clocks
  - Very slow – nonoverlap adds to setup time
  - But no hold times
- In industry, use a better timing analyzer
  - Add buffers to slow signals if hold time is at risk

Summary

- Flip-Flops:
  - Very easy to use, supported by all tools
- 2-Phase Transparent Latches:
  - Lots of skew tolerance and time borrowing
- Pulsed Latches:
  - Fast, some skew tol. & borrow, hold time risk