Lecture 6

Flip-Flop and Clock Design

R. Saleh
Dept. of ECE
University of British Columbia
res@ece.ubc.ca

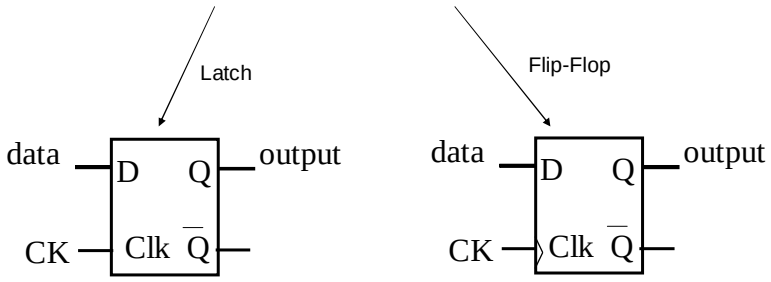
Design Considerations

- Basic role of clock is to perform synchronization operation in sequential logic circuits
- Clocks are used primary to drive the flip-flops in a logic chip
- Usually thousands of flops exist on the chip
- Design of the clock and the flops are related to each other so they should be studied together

- Design Issues:
  - flip-flop setup and hold times
  - clock power
  - clock latency, skew, jitter
  - impact of IR drop on clock
  - clock layout and routing
  - clock synchronization: PLL and DLL

Clocked D Flip-flop

- Very useful FF
- Widely used in IC design for temporary storage of data
- May be *level-sensitive or edge-triggered*



Latch vs. Flip-flop

Latch (level-sensitive, transparent)

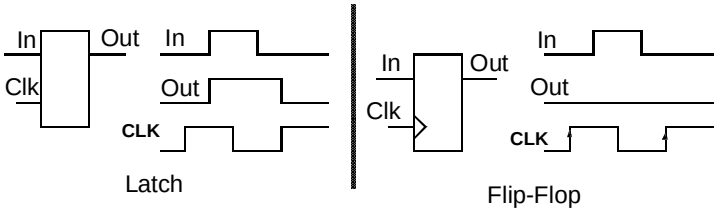
When the clock is high it passes **In** value to **Out**

When the clock is low, it holds value that **In** had when the clock fell

Flip-Flop (edge-triggered, non transparent)

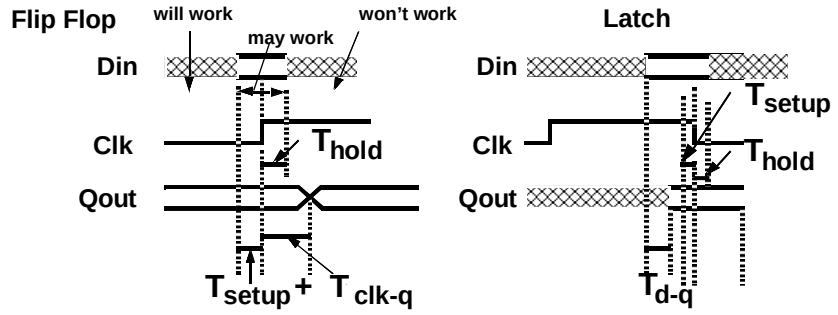
On the *rising* edge of clock (pos-edge trig), it transfers the value of **In** to **Out**

It holds the value at all other times.



Clocking Overhead

FF and Latches have setup and hold times that must be satisfied:



If Din arrives before setup time and is stable after the hold time, FF will work; if Din arrives after hold time, it will fail; in between, it may or may not work; FF delays the slowest signal by the setup + clk-q delay in the worst case

Latch has small setup and hold times; but it delays the late arriving signals by T_{d-q}

Clock Skew

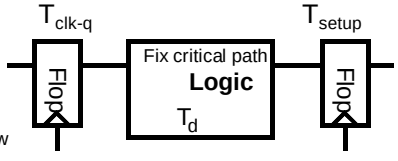
- Not all clocks arrive at the same time, i.e., they may be skewed.
  - SKEW = mismatch in the delays between arrival times of clock edges at FF's

SKEW causes two problems:

- The cycle time gets longer by the skew

$$T_{cycle} = T_d + T_{setup} + T_{clk-q} + T_{skew}$$

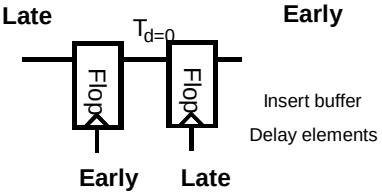
Shows up as a SETUP time violation



- The part can get the wrong answer

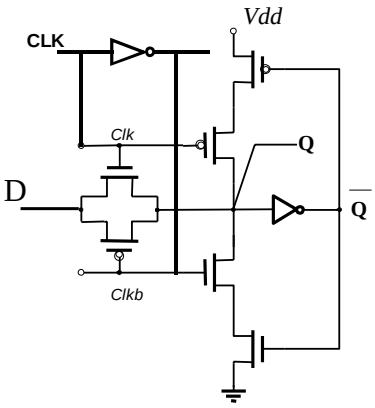
when $T_{skew} + T_{hold} > T_{clk-q}$

Shows up as a HOLD time violation



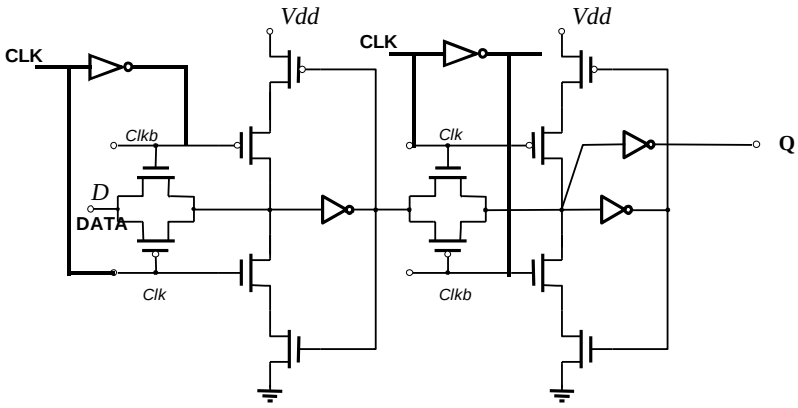
Transfer Gate D-Latch

- D-latch operation
  - When D arrives, if CLK is low then TG is off, and the previous output is held
  - When CLK goes high, D enters FF through TG and establishes Q and $\overline{Q}$
    - If data is 1, pull up network is enabled
    - If data is 0, pull down network is enabled
    - When clock goes low, the data is latched by one of the two networks
  - Setup time: time needed to charge Q
  - Hold time: time needed to shut off CLK and turn off TG

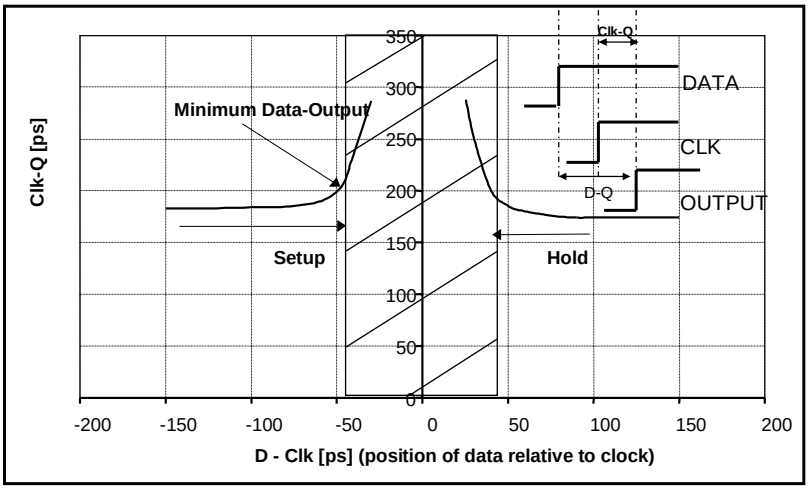


T-G Master-Slave D-FF

- Edge-Triggered Flip-flop

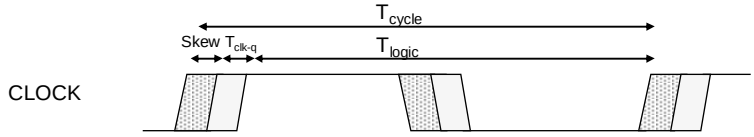


Delay vs. Setup/Hold Times



Overhead for a Clock

- CMOS FO4 delay is roughly 425ps/um x L_{eff}
- For 0.13um, FO4 delay ≈ 50ps
  - For a 1GHz clock, this allows < 20 FO4 gate delays/cycle
- Clock overhead (including margins for setup/hold)
  - 2 FF/Latches cost about 2 x1.2FO4 delays=2-3 FO4 delays
  - skew costs approximately 2-3 FO4 delays
- Overhead of clock is roughly 4-6 FO4 delays
- 14-16 FO4 delays left to work with for logic
- Need to reduce skew and FF cost.



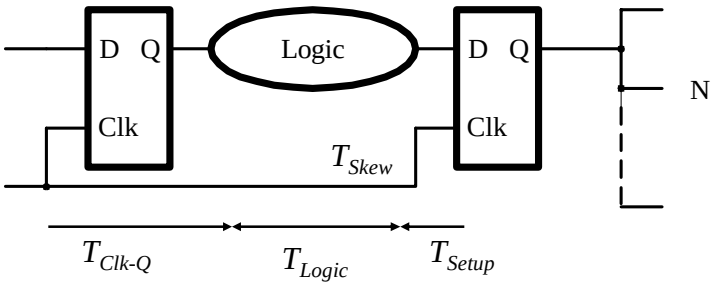
Requirements in Flip-Flop Design

- Minimize FF overhead: small clk-q delay, t_{setup}, t_{hold} times
- Minimize power
  - expensive packages and cooling systems
  - flops up to 20% of total power of high-performance systems
- High driving capability
  - Typical flip-flop load in a 0.18µm CMOS ranges from 50fF to over 200fF, with typical values of 100-150fF in critical paths
- Multiplexed or scan enabled
- Crosstalk insensitivity
  - dynamic/high impedance nodes are problematic
- Small load on clock to improve performance of clock and reduce power of clock
  - clocks can consume 40% of total chip power

Clock Design Issues

- Clock cycle depends on a number of factors:

$$T_{cycle} = T_{Clk\text{-}Q} + T_{Logic} + T_{setup} + T_{skew}$$



Sources of Clock Skew

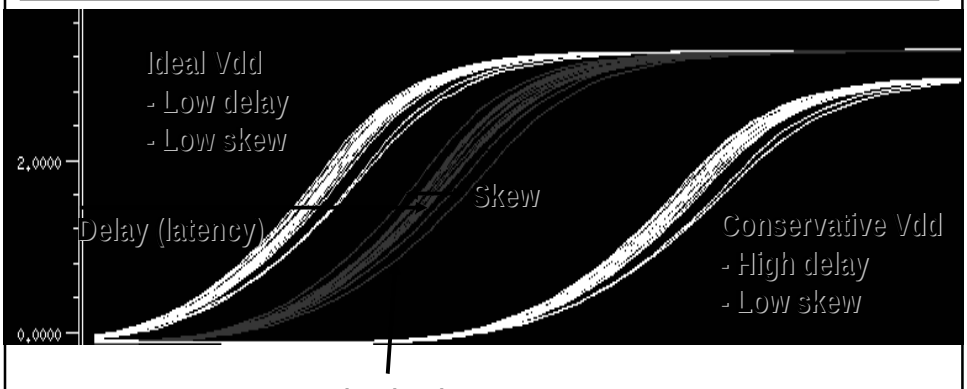
Main sources:

1. Imbalance between different paths from clock source to FF's
   - interconnect length determines RC delays
   - capacitive coupling effects cause delay variations
   - buffer sizing
   - number of loads driven
2. Process variations across die
   - interconnect and devices have different statistical variations

Secondary Sources:

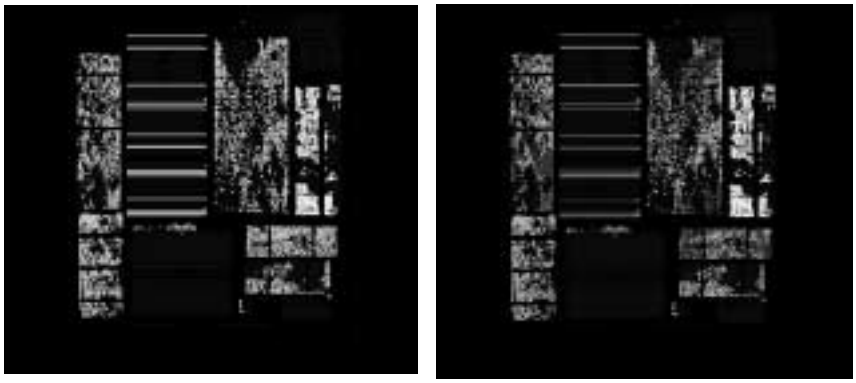
1. IR drop in power supply
2. Ldi/dt drop in supply

IR Drop Impacts on Clock Skew



Actual IR drop impact
- delay about 5-15% larger
- skew about 25-30% larger

Effects of IR-Drop on Clock Skew

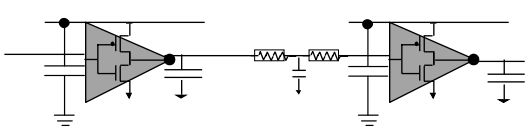


Without IR-drop **With IR-drop**

Plots courtesy of Simplex Solutions, Inc.

Reducing the Effects of IR drop and Ldi/dt

- Stagger the firing of buffers (bad idea: increases skew)
- Use different power grid tap points for clock buffers (but it makes routing more complicated for automated tools)
- Use smaller buffers (but it degrades edge rates/increases delay)
- Make power busses wider (requires area but should do it)
- Use more Vdd/Vss pins; adjust locations of Vdd/Vss pins
- Put in power straps where needed to deliver current
- Place decoupling capacitors wherever there is free space
- Integrate decoupling capacitors into buffer cells



Power dissipation in Clocks

- Significant power dissipation can occur in clocks in high-performance designs:
  - clock switches on every cycle so $P = CV^2f$ (i.e., $\alpha=1$)
  - clock capacitance can be ~nF range, say 1nF = 1000pF
  - assuming a power supply of 1.8V, CV = 1800pC of charge
  - if clock switches every 2ns (500MHz), that's 0.9A
  - for V_{DD} = 1.8V, P=IV=0.9(1.8)=1.6W in the clock circuit alone
- Much of the power (and the skew) occurs in the final drivers due to the sizing up of buffers to drive the flip-flops
- Key to reducing the power is to examine equation CV^2f and reduce the terms wherever possible
  - V_{DD} is usually given to us; would not want to reduce swing due to coupling noise, etc.
  - Look more closely at C and f

Reducing Power in Clocking

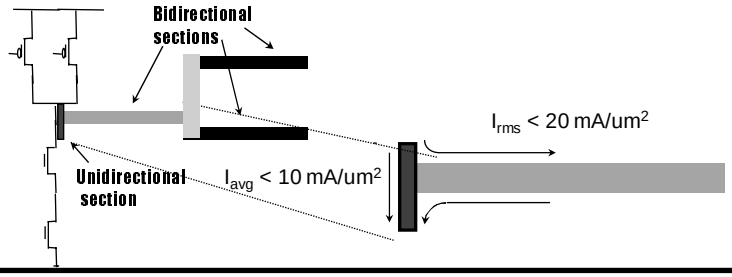
- Gated Clocks:
  - can gate clock signals through AND gate before applying to flip-flop; this is more of a total chip power savings
  - all clock trees should have the same type of gating whether they are used or not, and at the same level - total balance
- Reduce overall capacitance (again, shielding vs. spacing)

(a) higher total cap./less area (b) lower cap./ more area

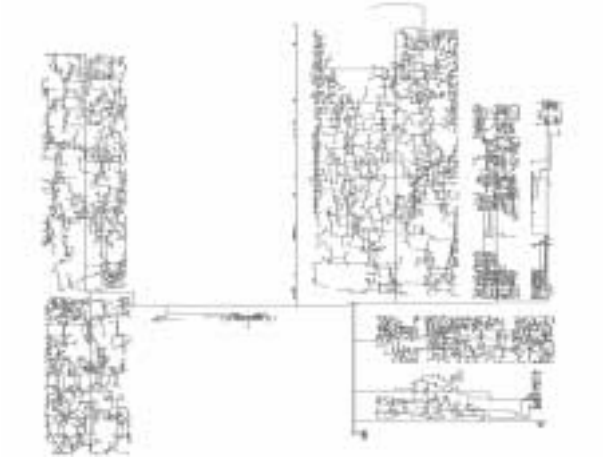
  - Tradeoff between the two approaches due to coupling noise
  - approach (a) is better for inductive noise; (b) is better for capacitive noise

Signal Electromigration

- Electromigration can occur on certain signal lines
- Clocks are prone to EM failures due to large current demand on every cycle
- Since current is bidirectional, we look at RMS current which lead to Joule heating effects (thermal)
- Based on signal activity (frequency of switching)



Clock Circuit of Multimedia Chip



Plots courtesy of Simplex Solutions, Inc.

Signal EM Example

Clock Design Objectives

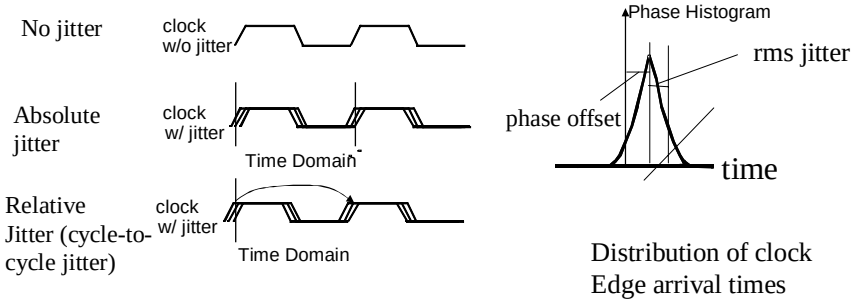
- Now that we understand the role of the clock and some of the key issues, how do we design it?
  - Minimize the clock skew (in presence of IR drop)
  - Minimize the clock delay (latency)
  - Minimize the clock power (and area)
  - Maximize noise immunity (due to coupling effects)
  - Maximize the clock reliability (signal EM)
- Problems that we will have to deal with
  - Routing the clock to all flip-flops on the chip
  - Driving unbalanced loading, which will not be known until the chip is nearly completed
  - On-chip process/temperature variations

Clock Design and Verification

- Many design styles
  - Low-speed designs: regular signals, symmetric tree
  - Medium-speed designs: balanced H-tree
  - High-speed designs
    - Balanced buffered H-tree
    - Grid
- Clock verification is more complex in DSM
  - RC Interconnect delays
  - Signal integrity (capacitive coupling, inductance)
  - IR drop
  - Signal Electromigration
  - Clock Jitter

Clock Jitter

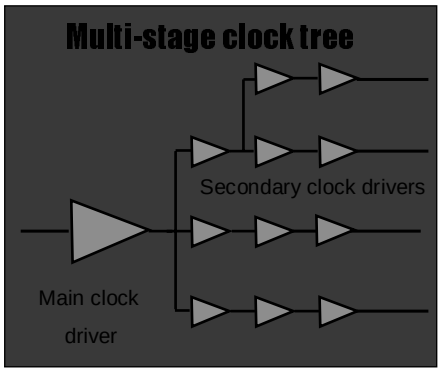
- Jitter is a term that applies to the shifting of a clock edge relative to its expected position due to noise (e.g., from power supply, random noise, temperature variation)
- Can be viewed as an uncertainty in the clock edge



Clock Design

Tree

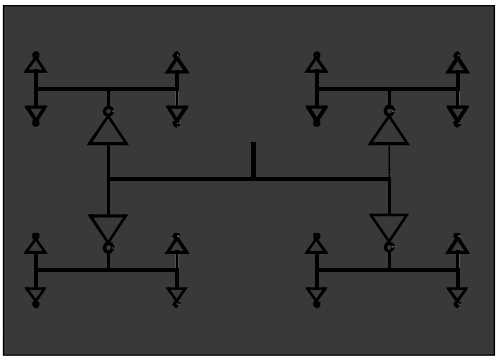
- Minimal area cost
- Requires clock-tree management
- Use a large superbuffer to drive downstream buffers
- Balancing may be an issue



Clock Configurations

H-Tree

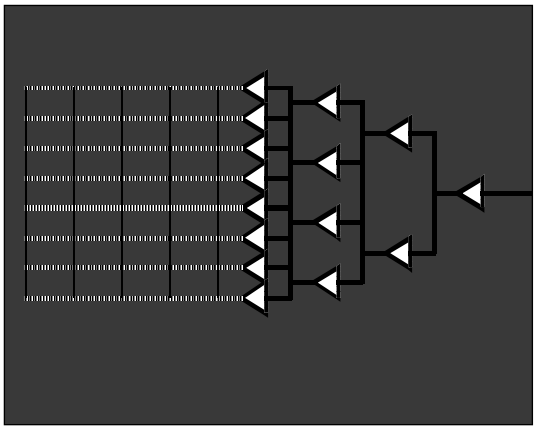
- Place clock root at center of chip and distribute as an H structure to all areas of the chip
- Clock is delayed by an equal amount to every section of the chip
- Local skew inside blocks is kept within tolerable limits



Clock Configurations

Grid

- Greater area cost
- Easier skew control
- Increased power consumption
- Electromigration risk increased at drivers
- Severely restricts floorplan and routing



Clock Design Today

Old methodology

- Route clock
- Route rest of nets
- Extract clock parasitics
- Perform timing verification
- Balance clock by “snaking” route in reserved areas

Advanced Clock Verification

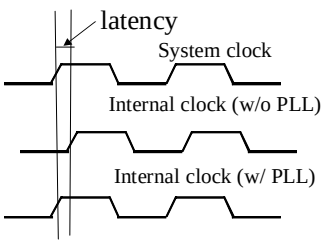
- IR Drop and Ldi/dt effects
- Coupling capacitance
- Electromigration checks
- Full-chip skew/slew analysis
- Jitter analysis
- Inductance Effects
- Process variations

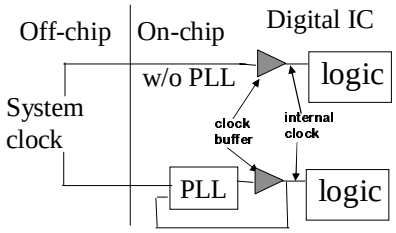
Good Practices in Clock Design

- Try to achieve the lowest Latency (Super Buffer/H-tree)
- Control transition times (keep edge rates sharp)
- Use 1 type of clock buffer for good matching (except perhaps in the last leg where you need to have adjustable buffers)
- Have min/max line lengths for good matching
- Determine whether spacing or shielding provides better tradeoff
- Use integral decoupling in buffers to reduce IR and Ldi/dt

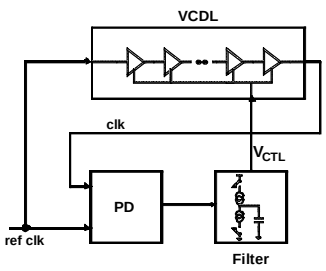
PLLs/DLLs

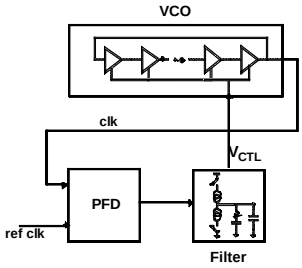
- So far in this course we have talked about clock design but not about the circuits that generate the clock and synchronize data around the clock
- These circuits are generally referred to as phase-locked loops (PLL) and delay-locked loops (DLLs)
- Applications of these circuits include: system synchronization, skew reduction, clock synthesis, clock and data synchronization





PLL/DLL Architecture





First order loop:
- easily stabilized
- frequency synthesis a problem
- ref clk jitter passes through

Second/Third order loop:
- stability is an issue
- frequency synthesis easy
- filtering of ref clk jitter

PLL Vs DLL

- PLL:
  - Second/Third order loop (stability is an issue)
  - Frequency synthesis possible (uses a VCO)
  - Input jitter is filtered
  - Phase error accumulates (takes longer to acquire lock)
  - Limited frequency capture range, unlimited phase capture range.
- DLL:
  - First order loop (always stable)
  - No self-generated jitter
  - Phase error does not accumulate
  - Not able to adjust its frequency (uses VCDL)
  - Limited phase capture range
  - Very attractive alternative when no frequency synthesis required.