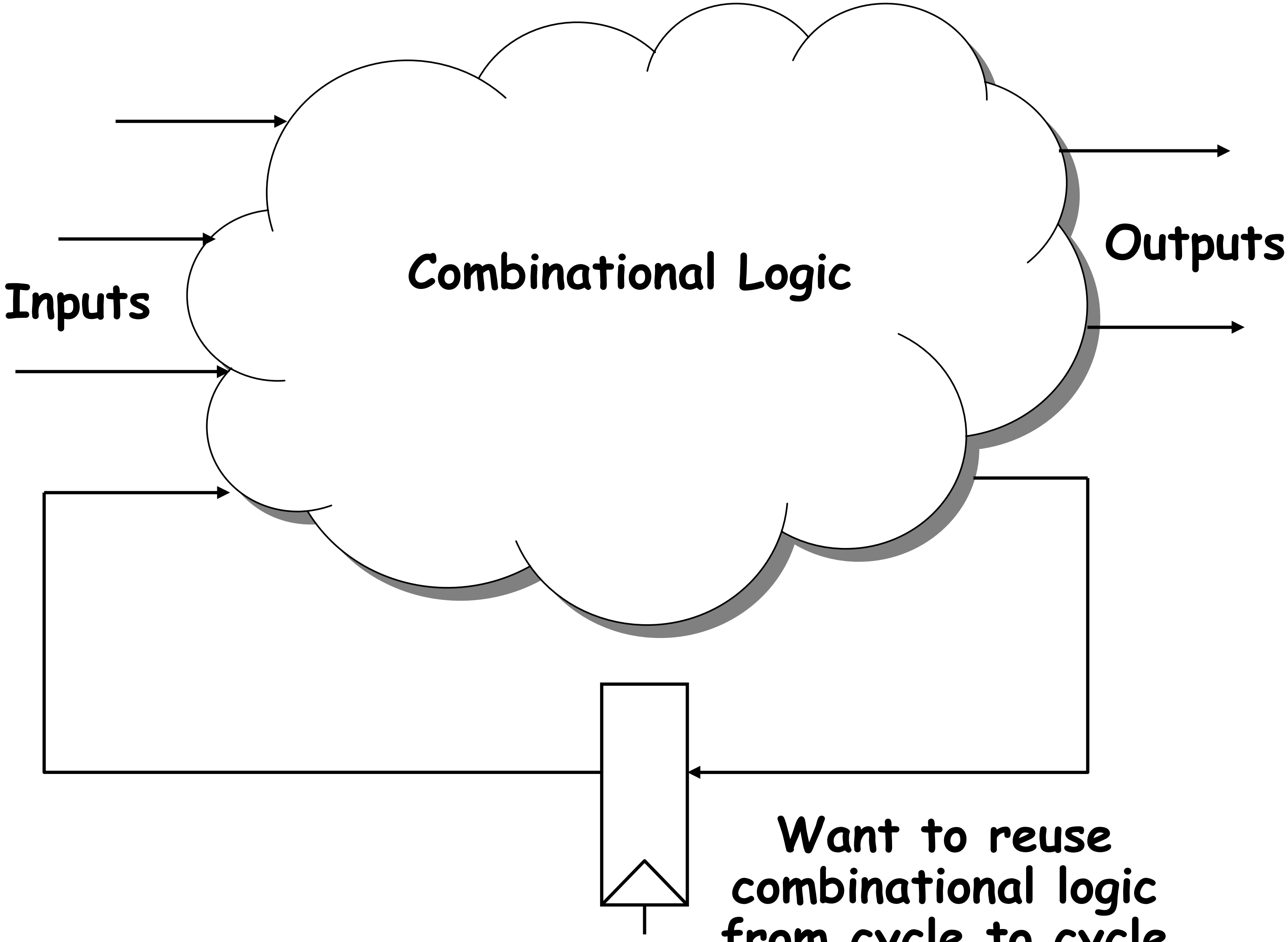
Clocking

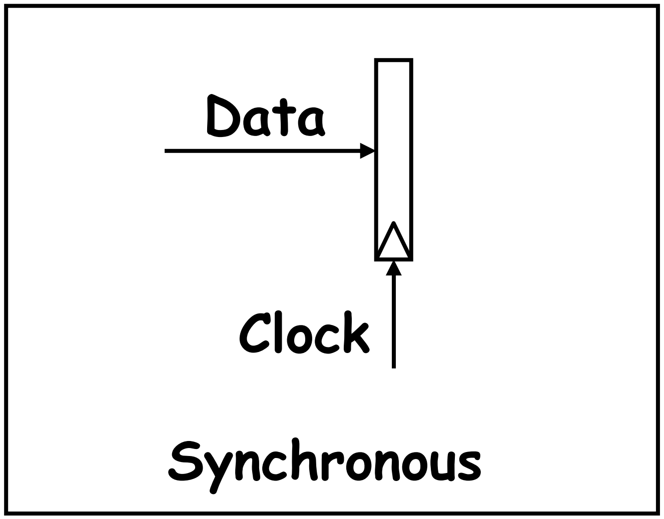
Why Clocks and Storage Elements?

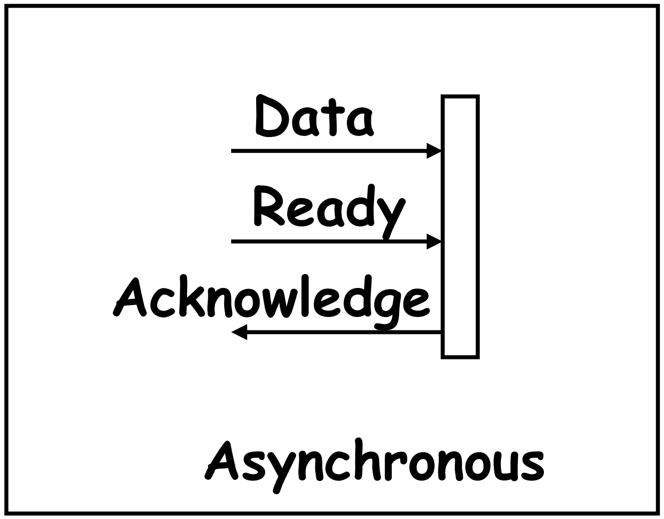


Want to reuse combinational logic from cycle to cycle

Digital Systems Timing Conventions

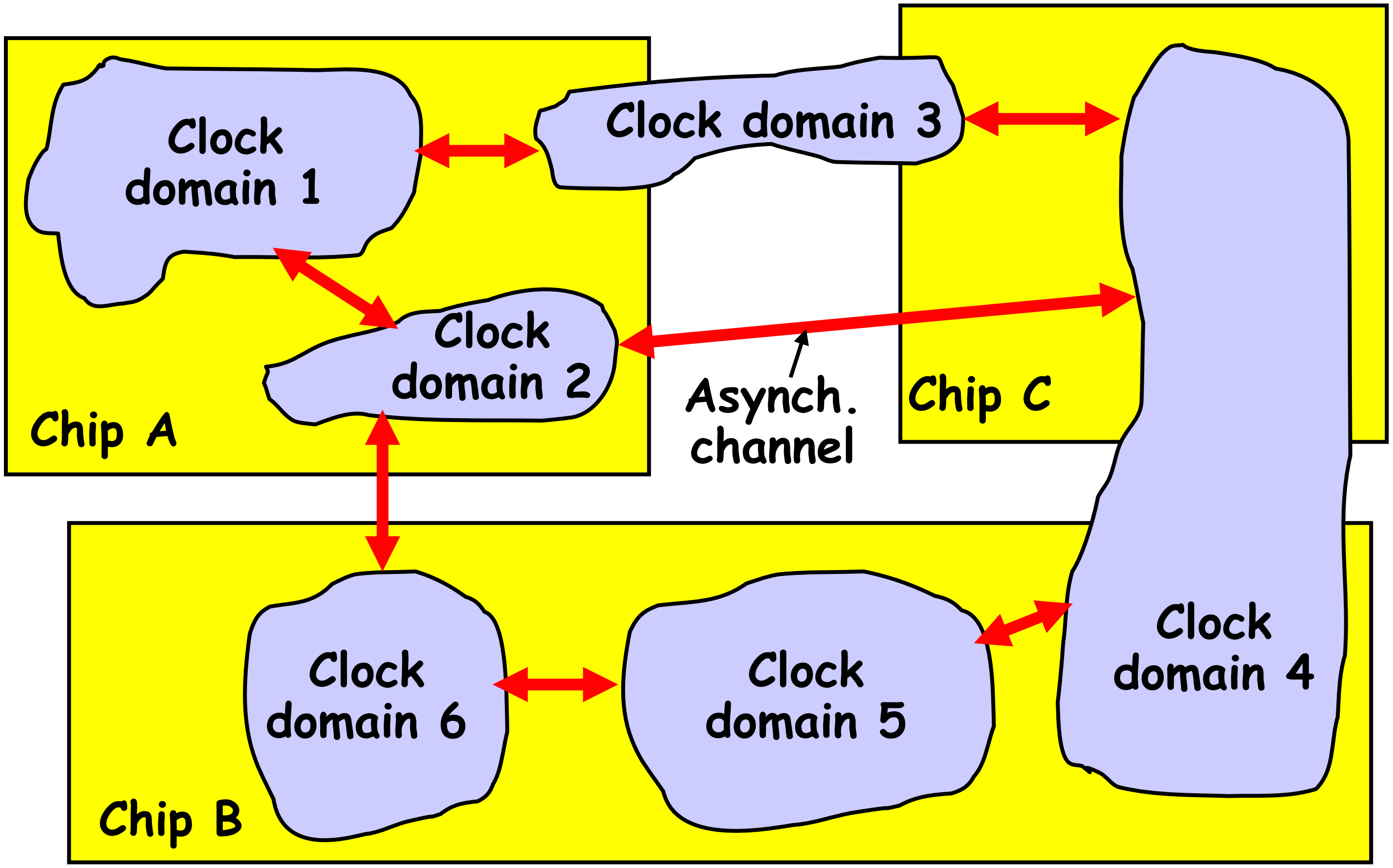
- All digital systems need a convention about when a receiver can sample an incoming data value
  - synchronous systems use a common clock
  - asynchronous systems encode "data ready" signals alongside, or encoded within, data signals
- Also need convention for when it's safe to send another value
  - synchronous systems, on next clock edge (after hold time)
  - asynchronous systems, acknowledge signal from receiver





Large Systems

- Most large scale ASICs, and systems built with these ASICs, have several synchronous clock domains connected by asynchronous communication channels

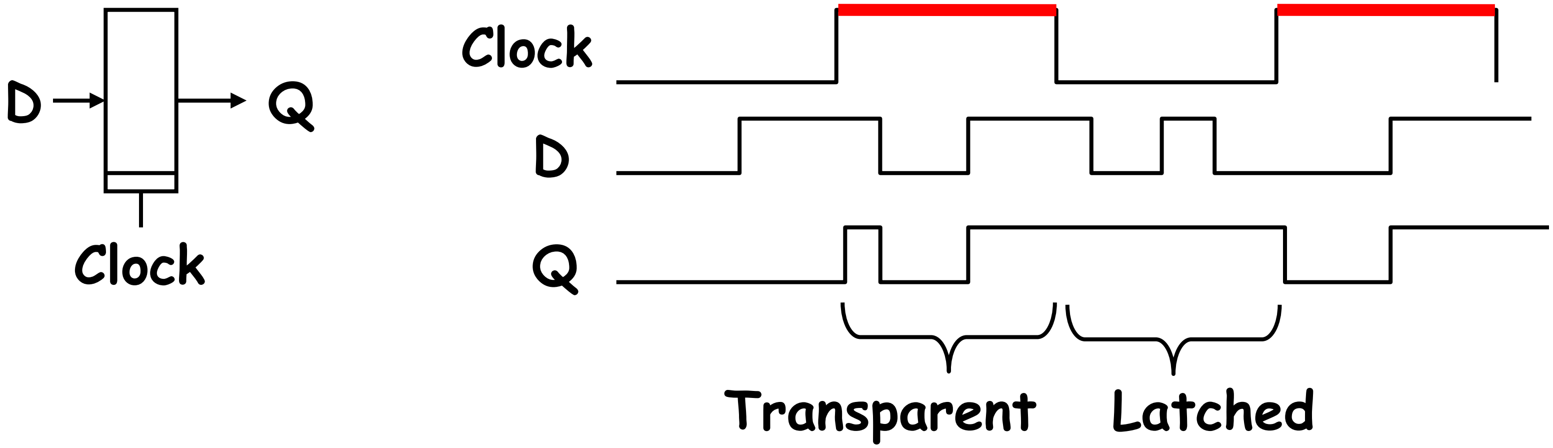


- We'll focus on a single synchronous clock domain today

Clocked Storage Elements

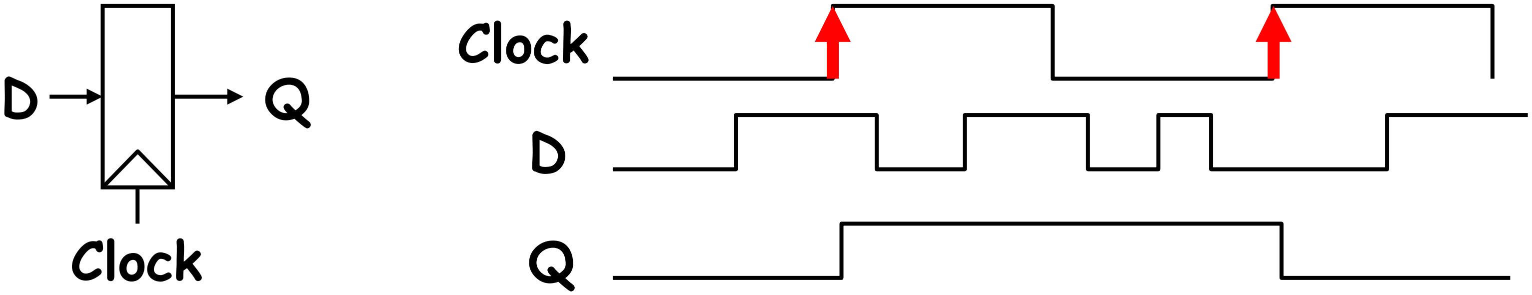
Transparent Latch, Level Sensitive

- data passes through when clock high, latched when clock low



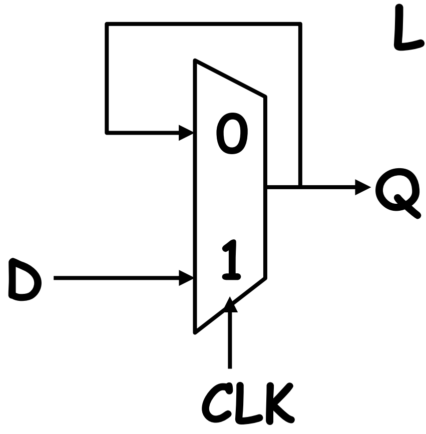
D-Type Register or Flip-Flop, Edge-Triggered

- data captured on rising edge of clock, held for rest of cycle



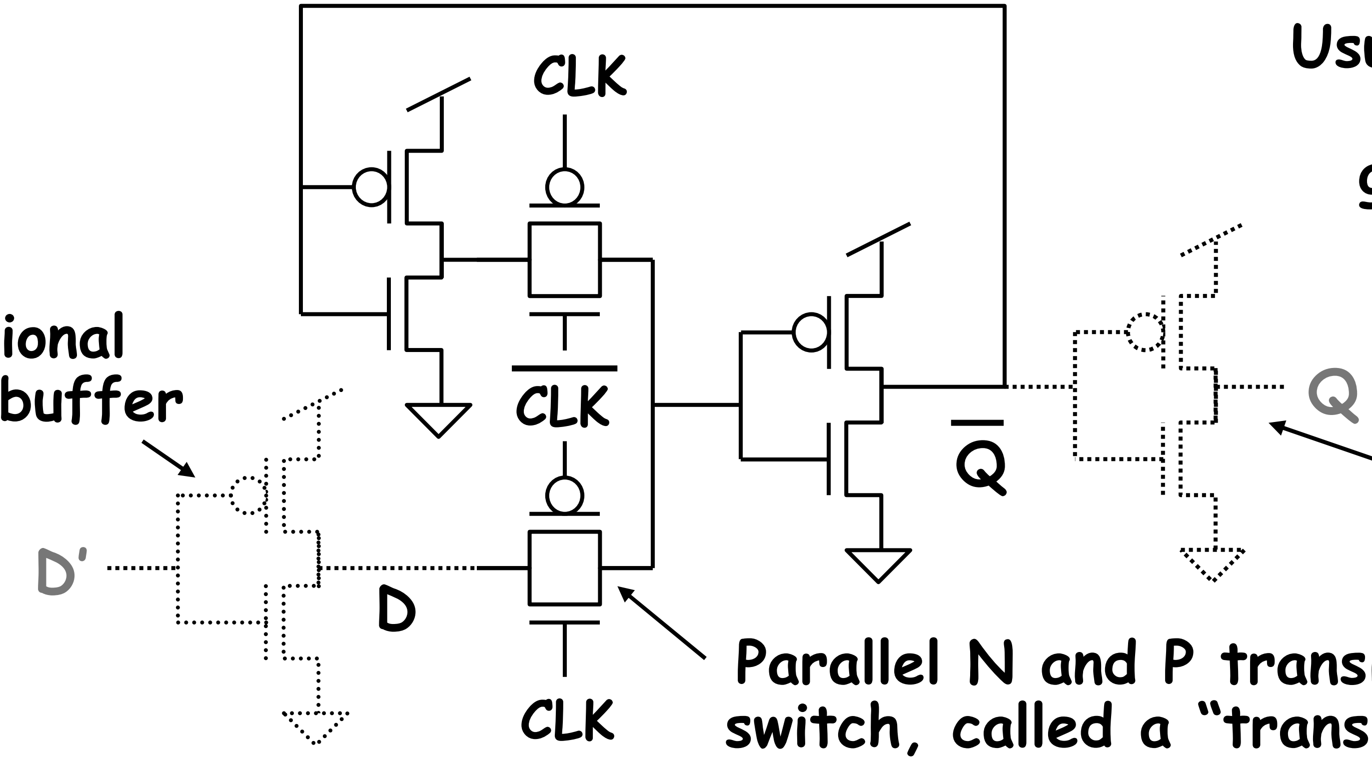
(Can also have latch transparent on clock
low, or negative-edge triggered flip-flop)

Building a Latch



Latches are a mux, clock selects either data or output value

CMOS Transmission Gate Latch



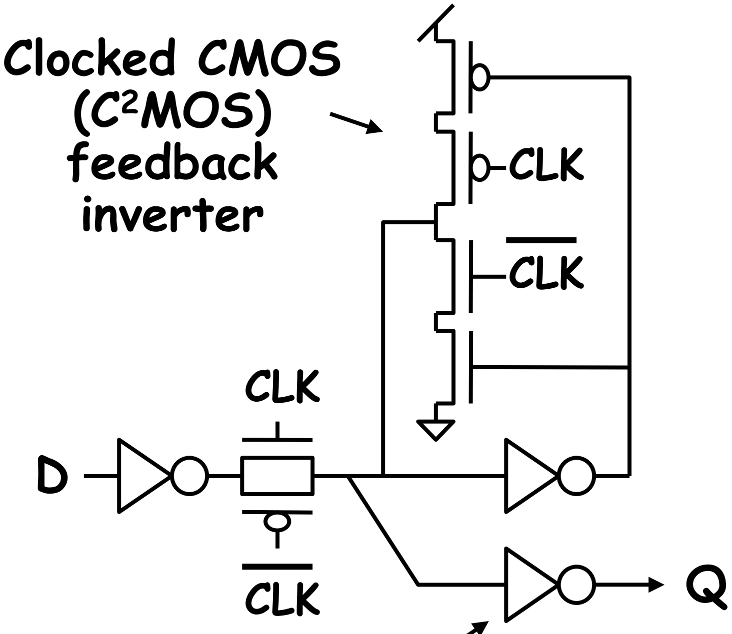
Usually have local inverter to generate $\overline{\text{CLK}}$

Optional input buffer

Optional output buffer

Parallel N and P transistors act as switch, called a "transmission gate"

Static CMOS Latch Variants

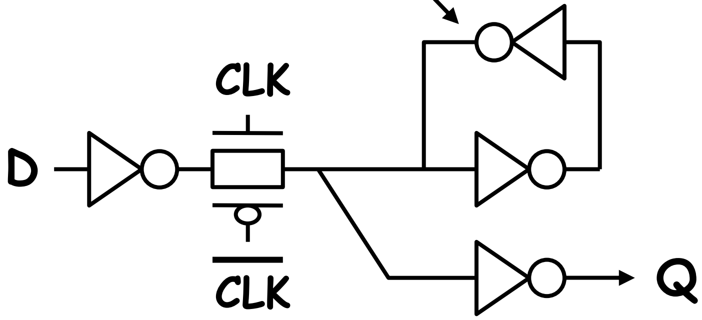


Clocked CMOS (C^2MOS) feedback inverter

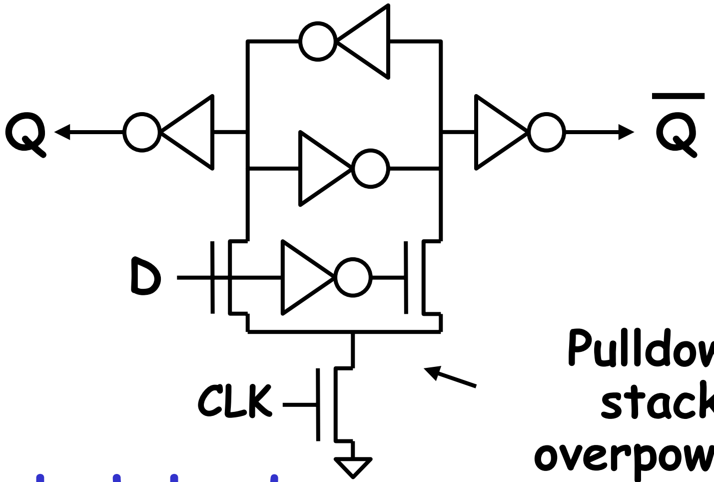
Output buffer shields storage node from downstream logic

Generally the best, fast and energy efficient

Weak feedback inverter so input can overpower it



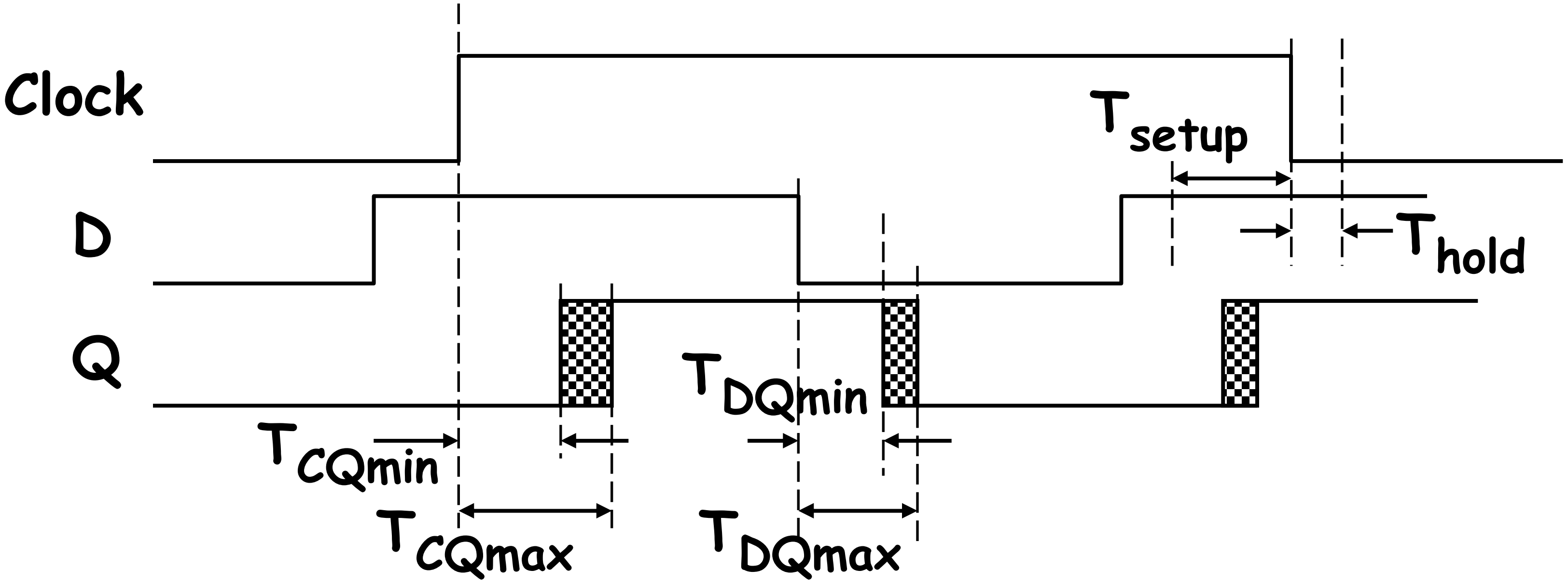
Can be small, lower clock load, but sizing problematic



Pulldown stack overpowers cross-coupled inverters

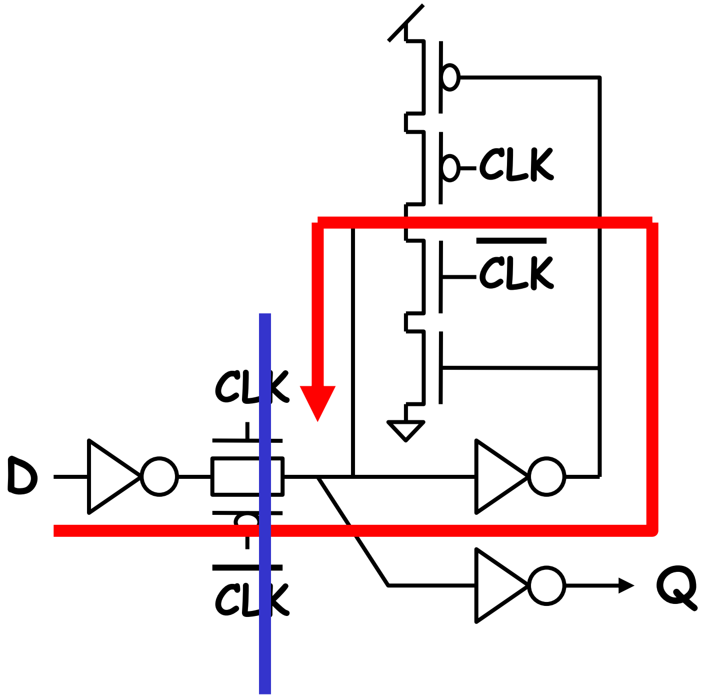
Has lowest clock load

Latch Timing Parameters



- T_{CQmin}/T_{CQmax}
  - propagation in→out when clock opens latch
- T_{DQmin}/T_{DQmax}
  - propagation in→out while transparent
  - usually the most important timing parameter for a latch
- T_{setup}/T_{hold}
  - define window around closing clock edge during which data must be steady to be sampled correctly

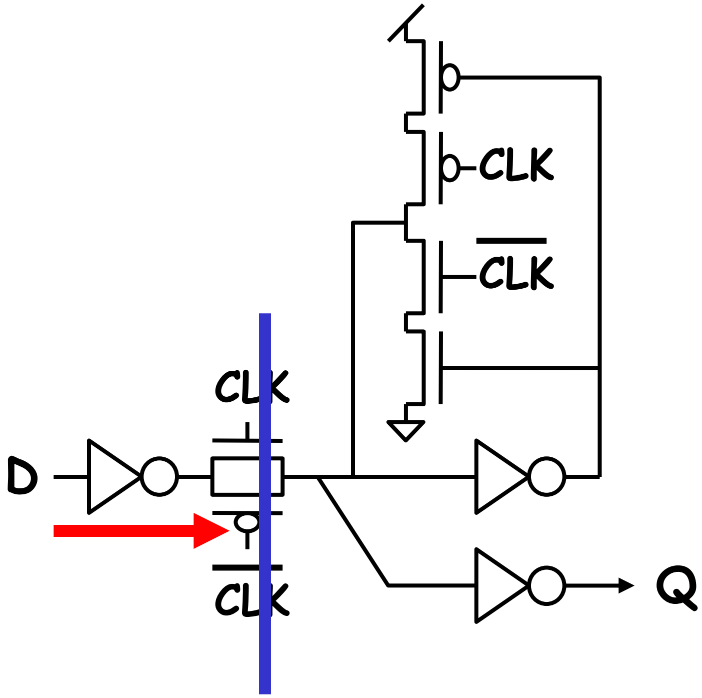
The Setup Time Race



Setup represents the race for new data to propagate around the feedback loop before clock closes the input gate.

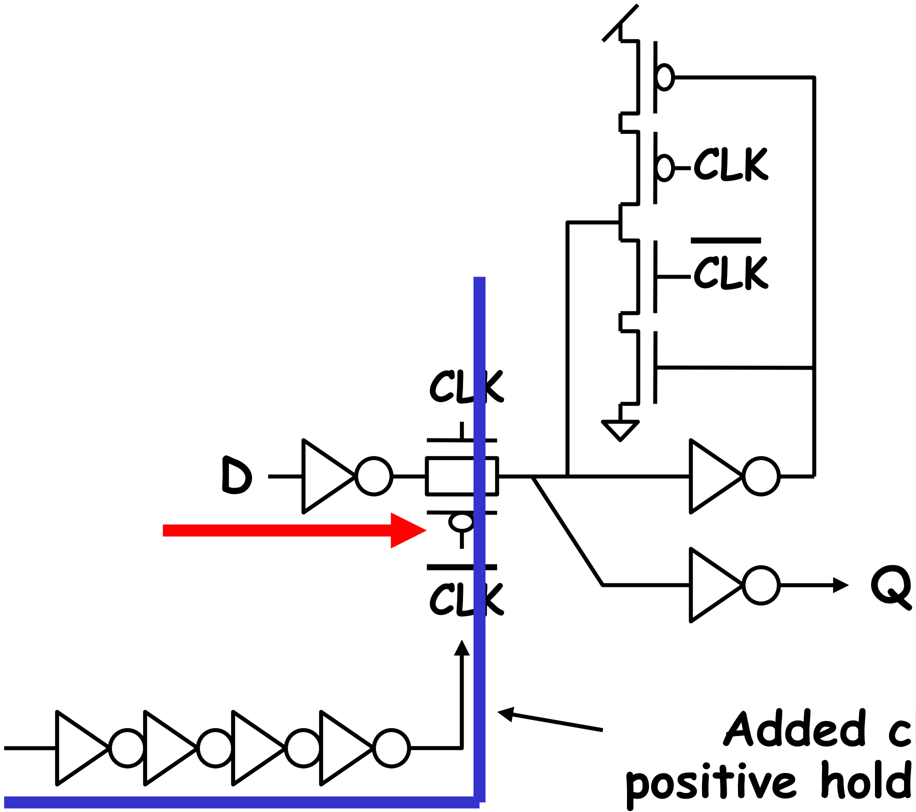
(Here, we're rooting for the data signal)

Failing Setup



If data arrives too close to clock edge, it won't set up the feedback loop before clock closes the input transmission gate.

The Hold Time Race

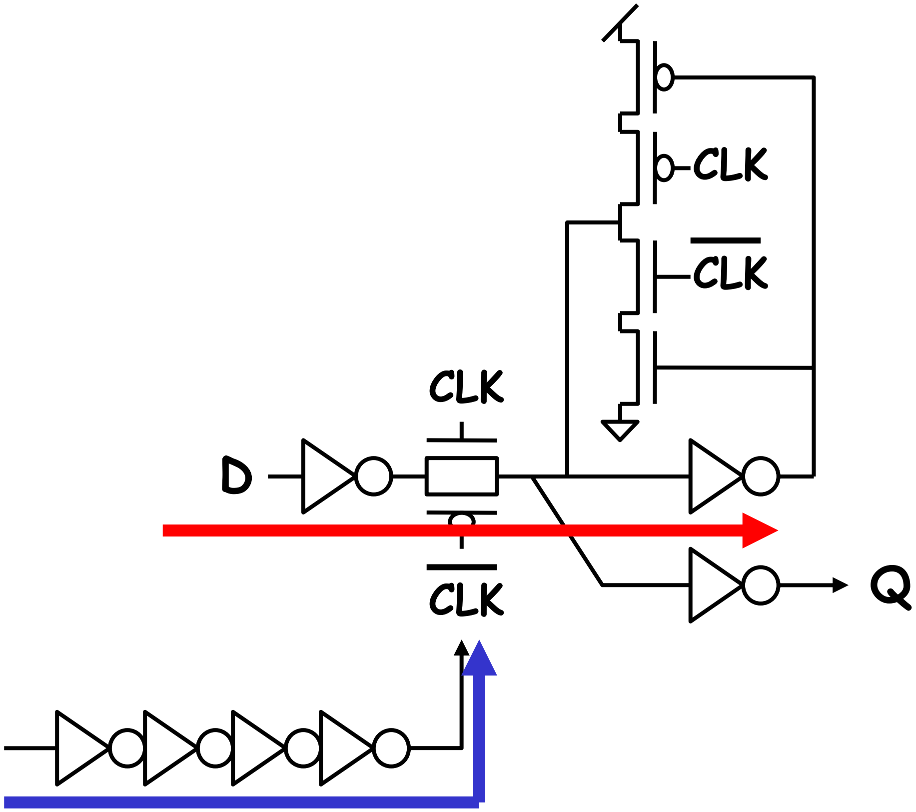


Added clock buffers to demonstrate positive hold time on this latch – other latch designs naturally have positive hold time

Hold time represents the race for clock to close the input gate before next cycle's data disturbs the stored value.

(Here we're rooting for the clock signal)

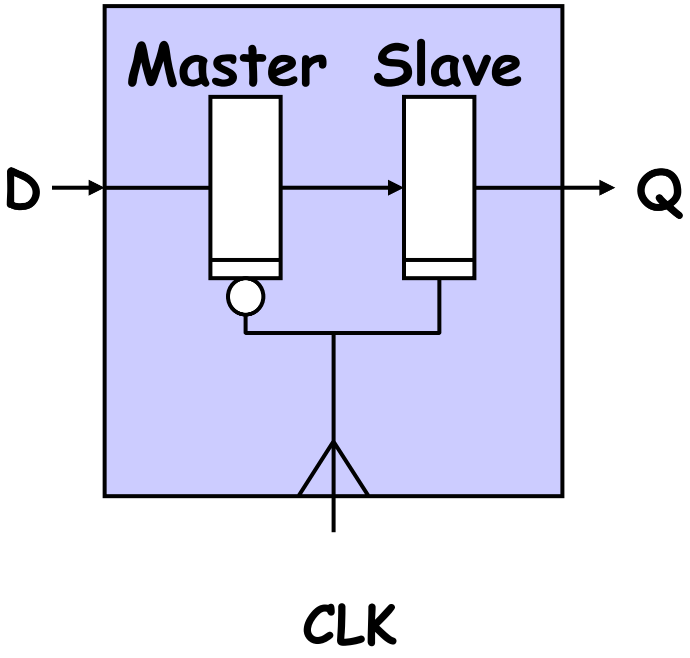
Failing Hold Time

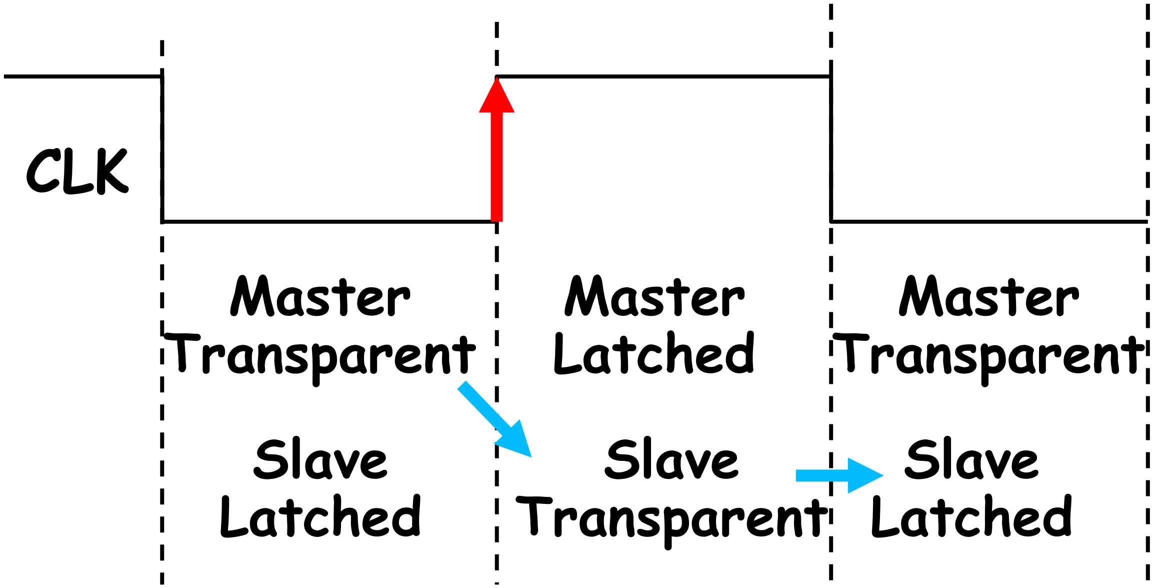


If data changes too soon after clock edge, clock might not have had time to shut off input gate and new data will corrupt feedback loop.

Flip-Flops

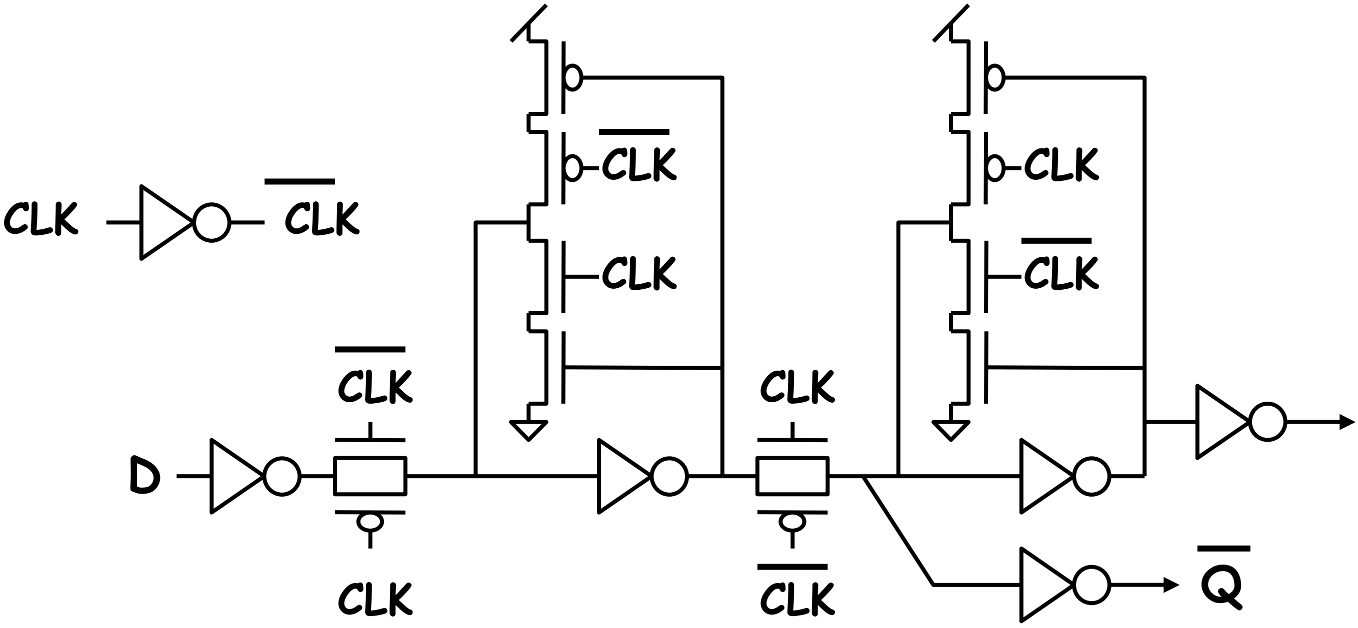
- Can build a flip-flop using two latches back to back





- On positive edge, master latches input D, slave becomes transparent to pass new D to output Q
- On negative edge, slave latches current Q, master goes transparent to sample input D again

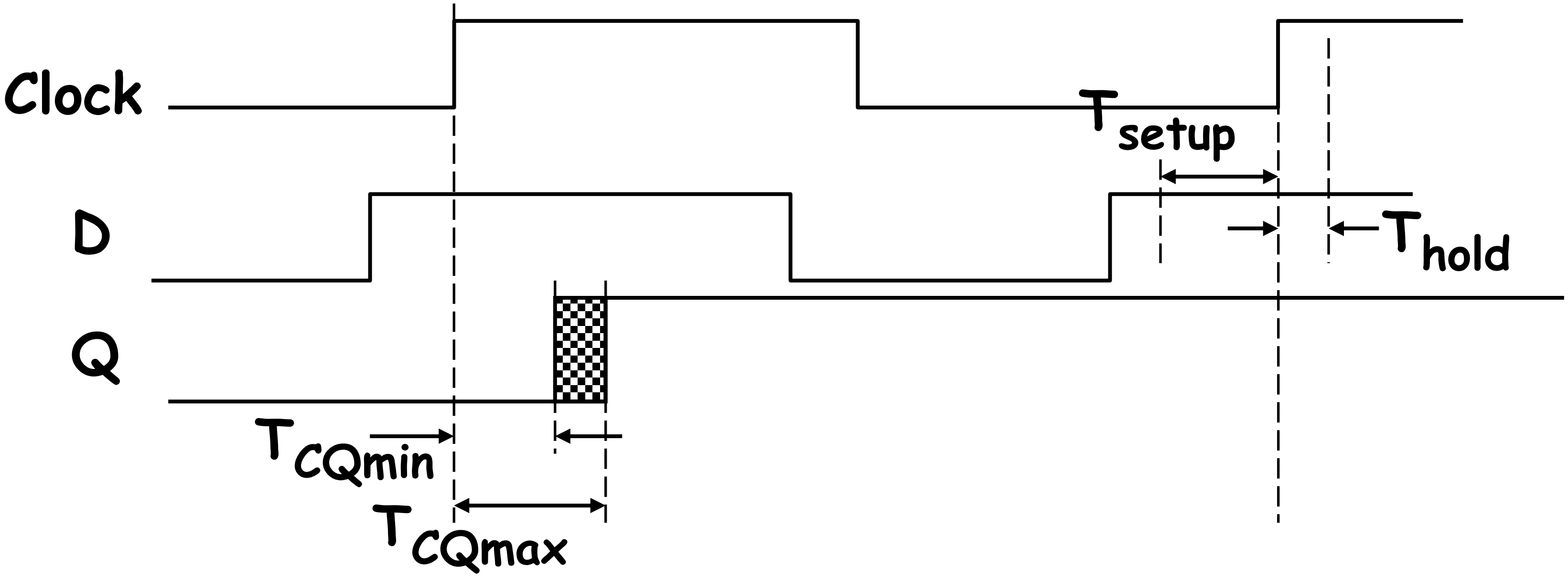
Flip-Flop Designs



Can have true or complementary output or both

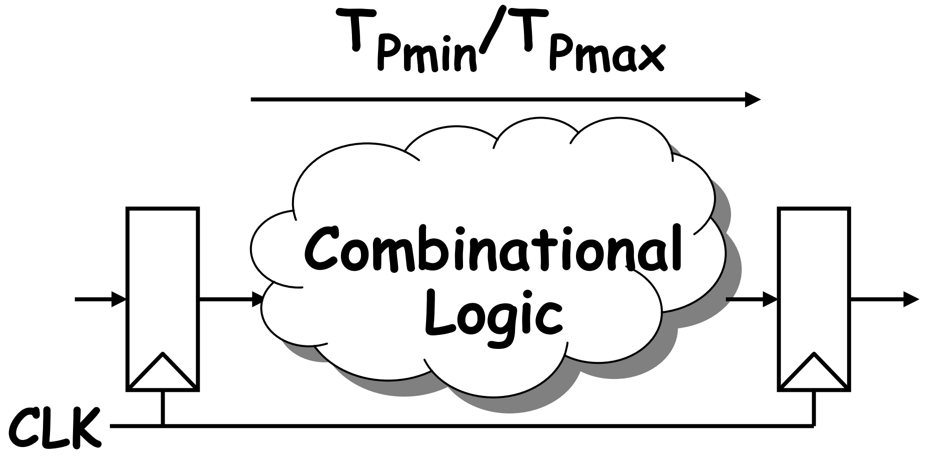
- Transmission-gate master-slave latches most popular in ASICs
  - robust, convenient timing parameters, energy-efficient
- Many other ways to build a flip-flop other than transmission gate master-slave latches
  - usually trickier timing parameters
  - only found in high performance custom devices

Flip-Flop Timing Parameters



- T_{CQmin}/T_{CQmax}
  - propagation in→out at clock edge
- T_{setup}/T_{hold}
  - define window around rising clock edge during which data must be steady to be sampled correctly
  - either setup or hold time can be negative

Single Clock Edge-Triggered Design



Single clock with edge-triggered registers most common design style in ASICs

- Slow path timing constraint

$$T_{cycle} \geq T_{CQmax} + T_{Pmax} + T_{setup}$$

  - can always work around slow path by using slower clock

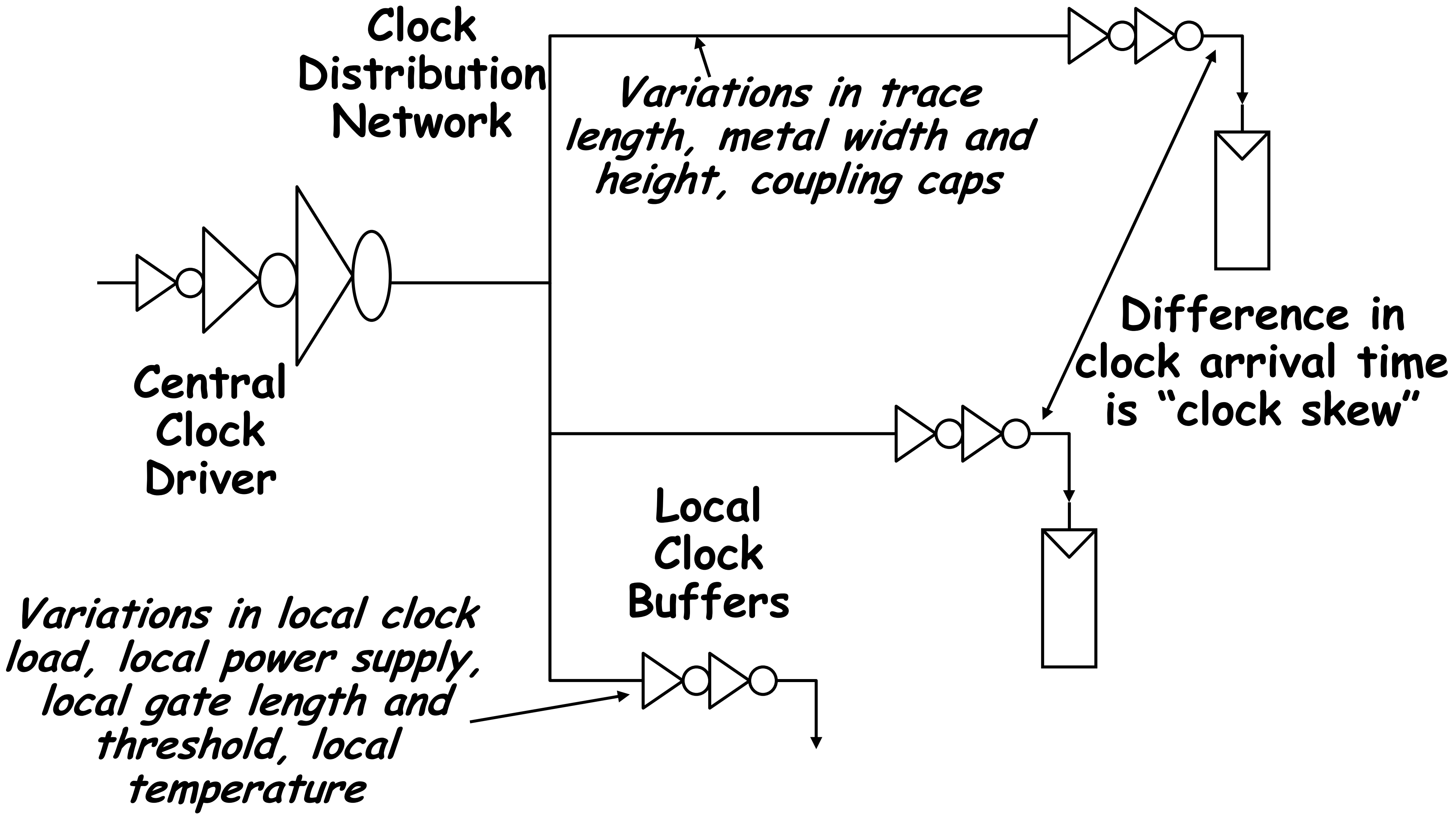
- Fast path timing constraint

$$T_{CQmin} + T_{Pmin} \geq T_{hold}$$

  - bad fast path cannot be fixed without redesign!
  - might have to add delay into paths to satisfy hold time

Clock Distribution

- **Can't really distribute clock at same instant to all flip-flops on chip**



Clock Distribution Network

Variations in trace length, metal width and height, coupling caps

Central Clock Driver

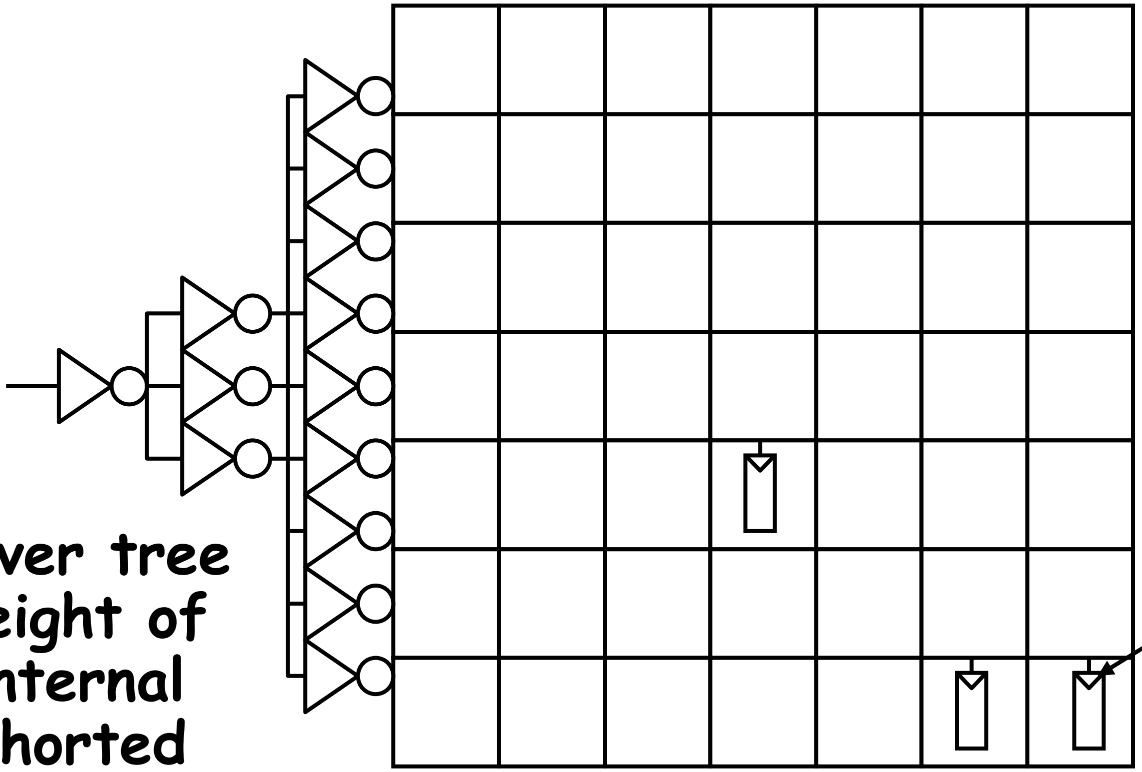
Difference in clock arrival time is "clock skew"

Local Clock Buffers

Variations in local clock load, local power supply, local gate length and threshold, local temperature

Clock Grids

- One approach for low skew is to use a single metal clock grid across whole chip (Alpha 21064)
- Low skew but very high power, no clock gating

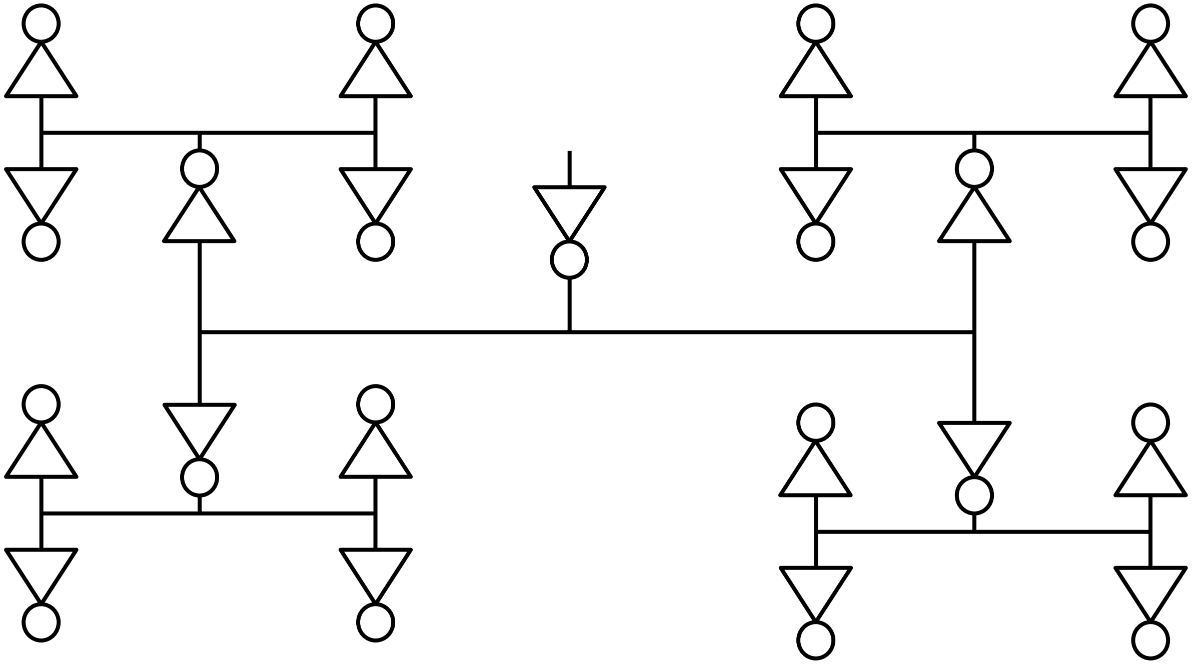


Clock driver tree spans height of chip. Internal levels shorted together.

Grid feeds flops directly, no local buffers

H-Trees

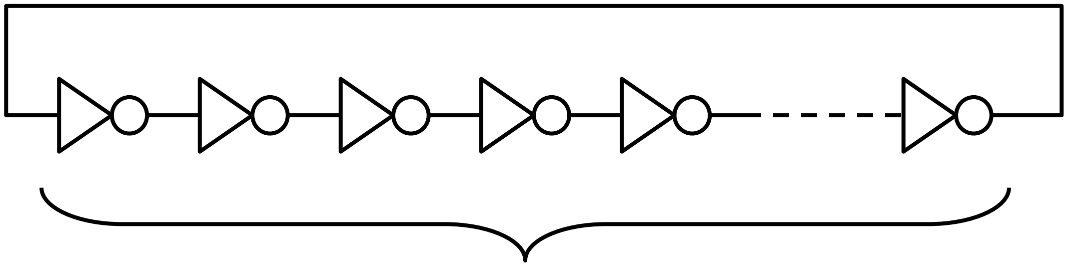
- Recursive pattern to distribute signals uniformly with equal delay over area



- Uses much less power than grid, but has more skew
- In practice, an approximate H-tree is used at the top level (has to route around functional blocks), with local clock buffers driving regions

Clock Oscillators

- Where does the clock signal come from?
- Simple approach: ring oscillator



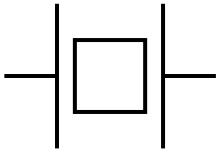
Odd number of inverter stages connected in a loop

Problem:

- What frequency does the ring run at?
  - Depends on voltage, temperature, fabrication run, …
- Where are the clock edges relative to an external observer?
  - Free running, no synchronization with external channel

Clock Crystals

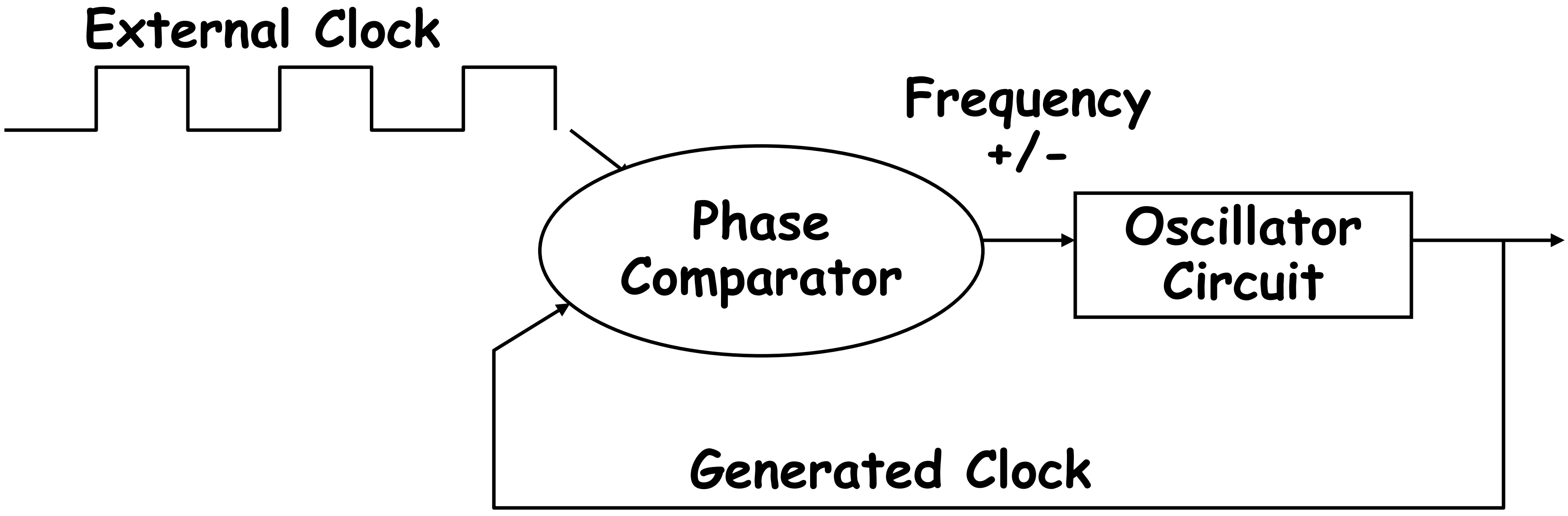
- Fix the clock frequency by using a crystal oscillator
- Exploit peizo-electric effect in quartz to create highly resonant peak in feedback loop of oscillator
- Easy to obtain frequency accuracy of ~50 parts per million



- Expensive to increase frequency to more than a few 100MHz

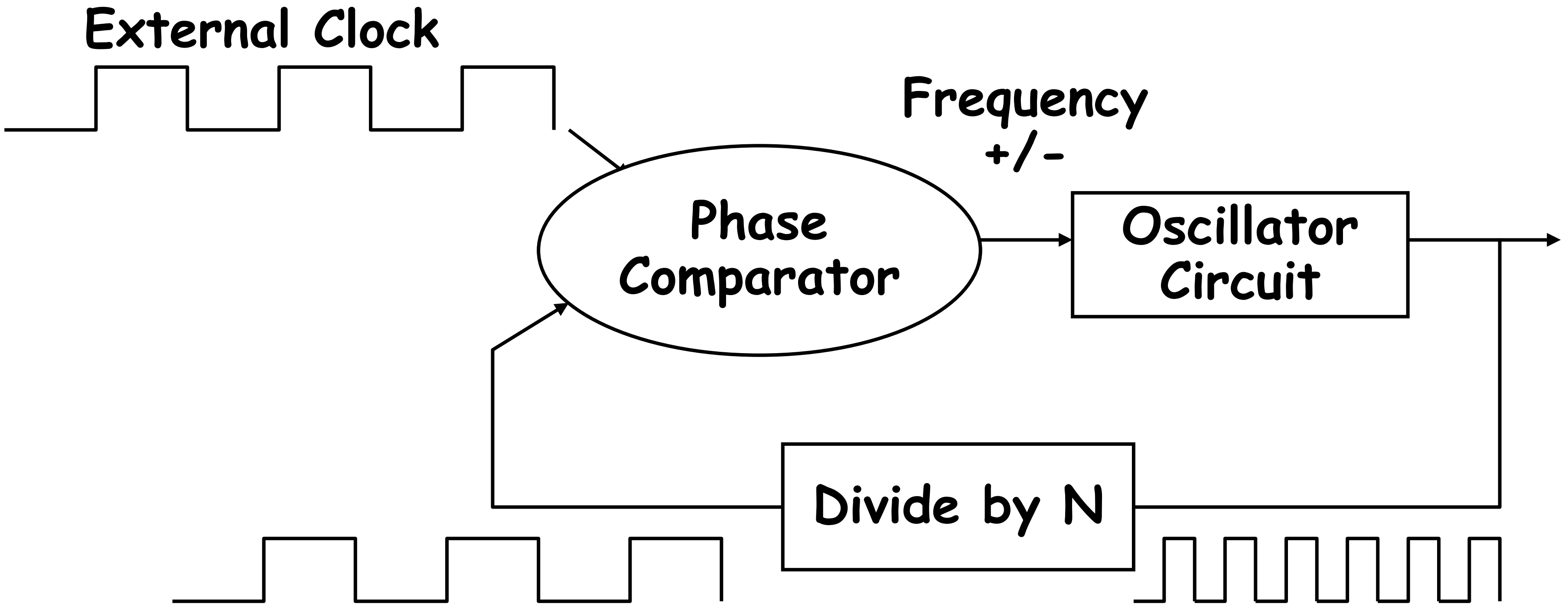
Phase Locked Loops (PLLs)

- Use a feedback control loop to force an oscillator to align frequency and phase with an external clock source.



Multiplying Frequency with a PLL

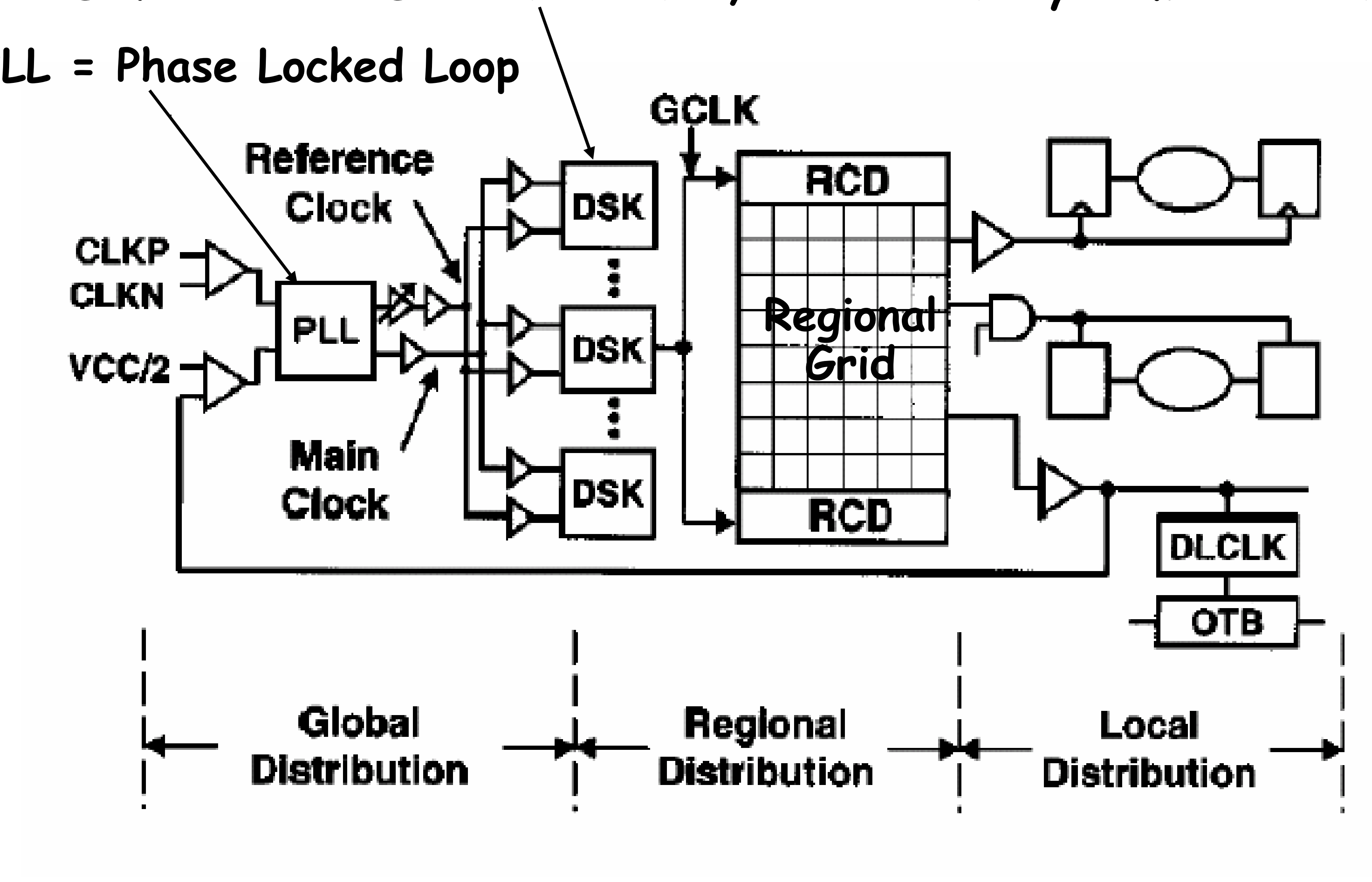
- By using a clock divider (a simple synchronous circuit) in the feedback loop, can force on-chip oscillator to run at rational multiple of external clock



Intel Itanium Clock Distribution

DSK = Active Deskew Circuits, cancels out systematic skew

PLL = Phase Locked Loop



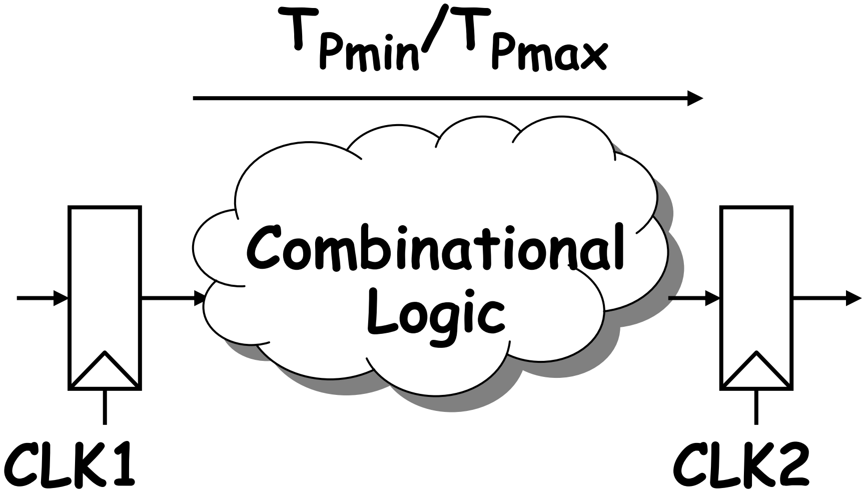
Skew Sources and Cures

- Systematic skew due to manufacturing variation can be mostly trimmed out with adaptive deskewing circuitry
  - cross chip skews of <10ps reported
- Main sources of remaining skew are temperature changes (low-frequency) and power supply noise (high frequency)
- Power supply noise affects clock buffer delay and also frequency of PLL
  - often power for PLL is provided through separate pins
  - clock buffers given large amounts of local on-chip decoupling capacitance

Skew versus Jitter

- Skew is spatial variation in clock arrival times
  - variation in when the *same* clock edge is seen by two *different* flip-flops
- Jitter is temporal variation in clock arrival times
  - variation in when two *successive* clock edges are seen by the *same* flip-flop
- Power supply noise is main source of jitter
- From now on, use "skew" as shorthand for untrimmable timing uncertainty

Timing Revisited



Skew eats into timing budget

- Slow path timing constraint

$$T_{cyc} \geq T_{CQmax} + T_{Pmax} + T_{setup} + T_{skew}$$

  - worst case is when CLK2 is earlier/later than CLK1

- Fast path timing constraint

$$T_{CQmin} + T_{Pmin} \geq T_{hold} + T_{skew}$$

  - worst case is when CLK2 is earlier/later than CLK1