11. Sequential Elements

J. A. Abraham

Department of Electrical and Computer Engineering
The University of Texas at Austin

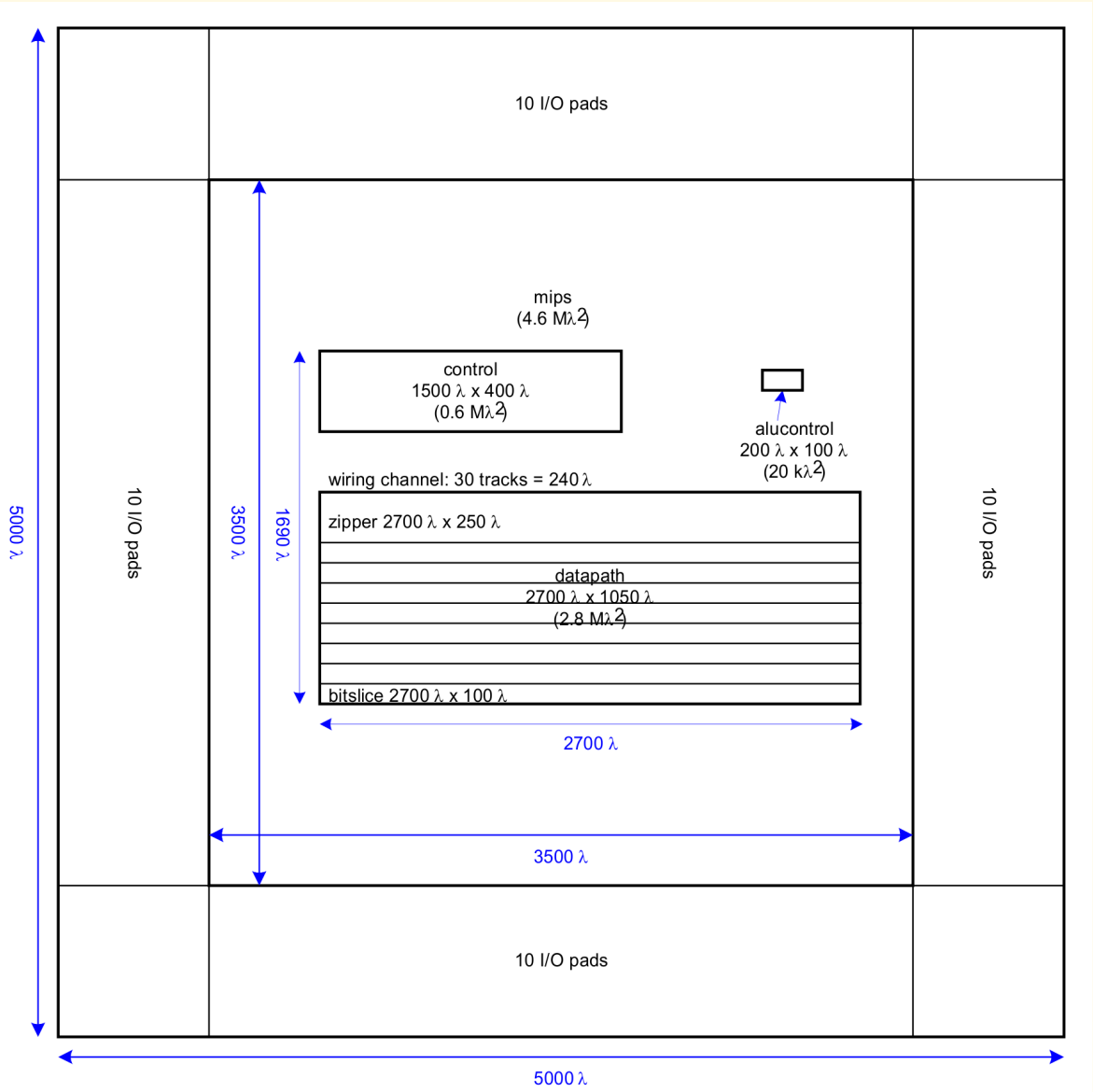
EE 382M, VLSI I
Fall 2010

October 4, 2010

Floor Plan

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

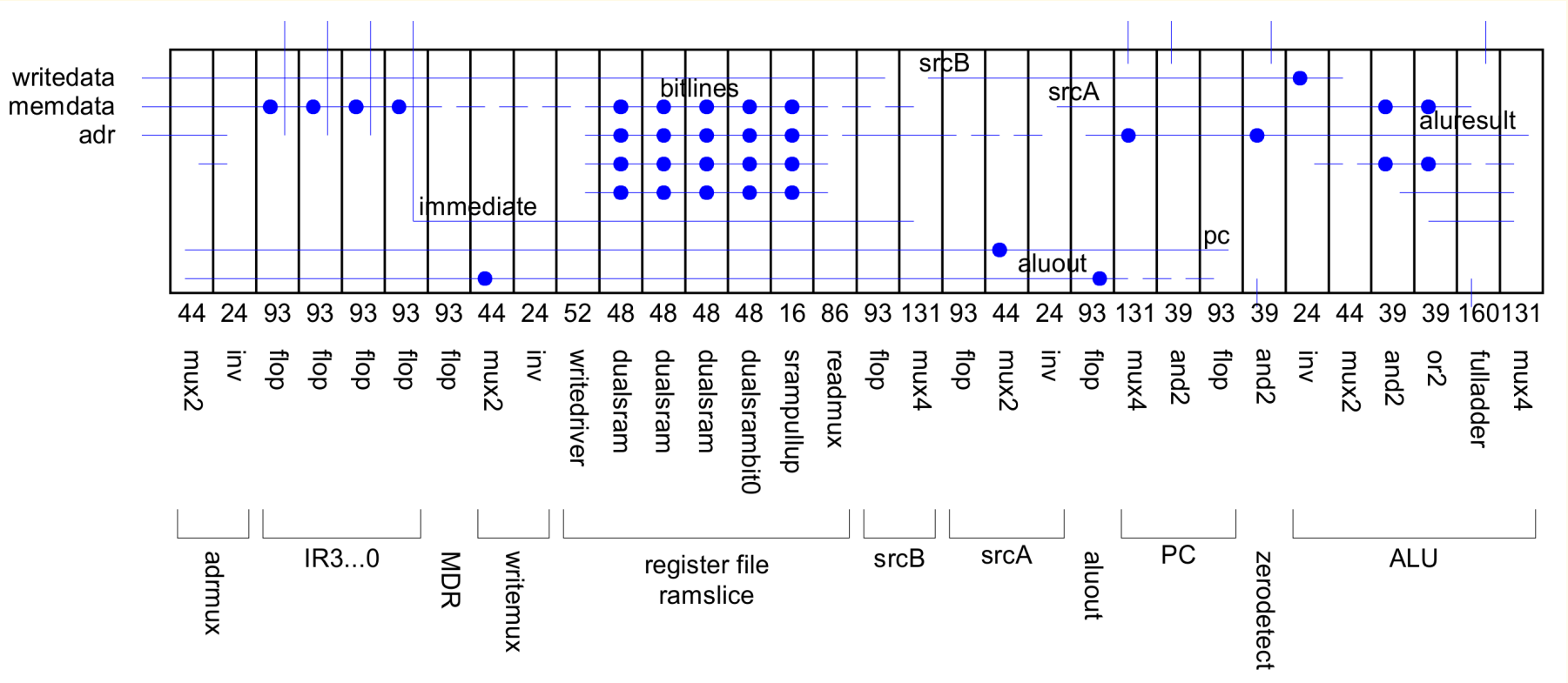
MIPS Floorplan



Area Estimation

- Arrays
  - Layout basic cell
  - Calculate core area from number 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 do a design you have done
  - For design in a new technology, estimate from scaling design in the old technology

MIPS Slice Plan



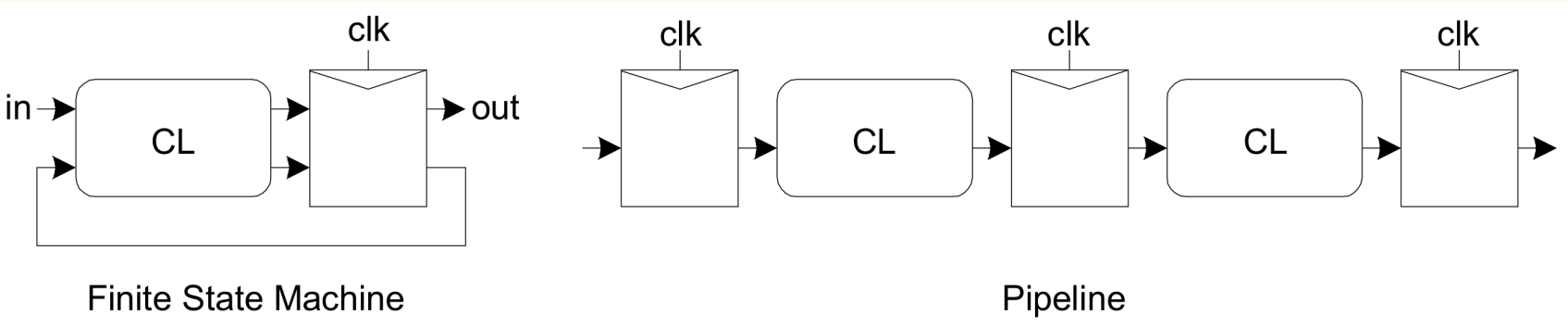
Typical Layout Densities

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

Element	**Area**
Random logic (2 metal layers)	1000 – 1500 λ^2/transistor
Datapath	250 – 750 λ^2/transistor or, $6WL + 360\lambda^2$/transistor
SRAM	1000 λ^2/bit
DRAM	100 λ^2/bit
ROM	100 λ^2/bit

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
  - Example, Finite-State Machine (FSM), pipeline



Sequencing, Cont'd

- If tokens moved through pipeline at constant speed, no sequencing elements would be necessary
- Example, 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

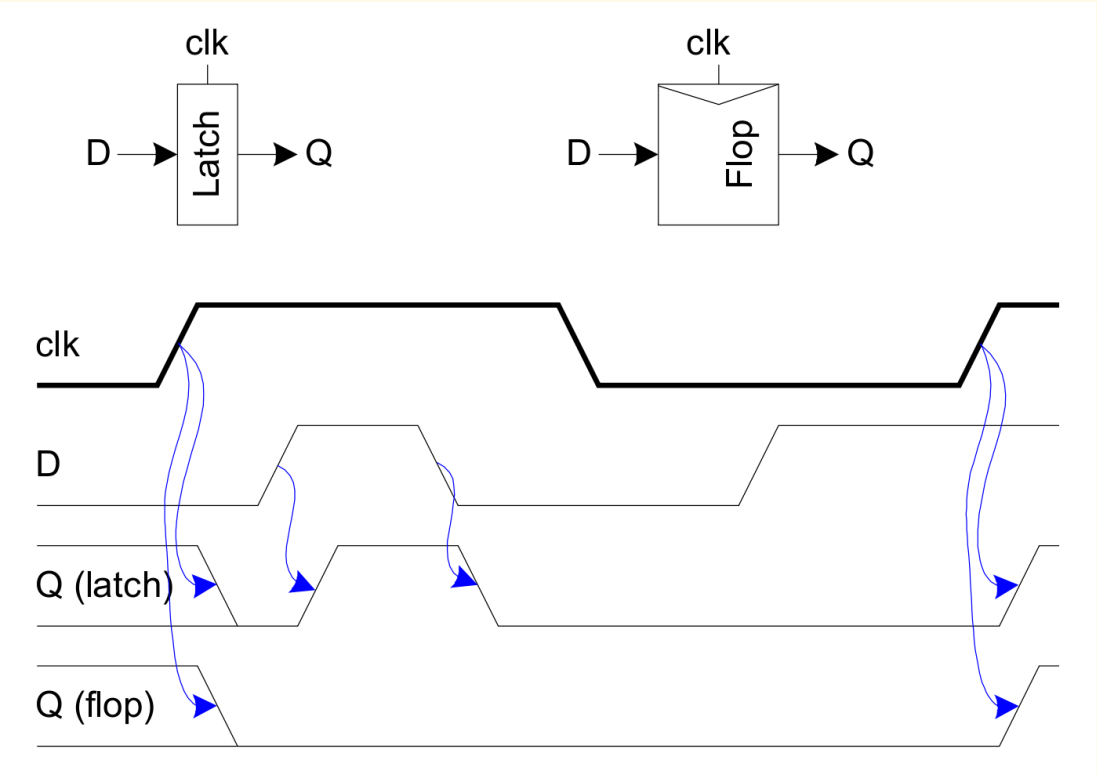
Sequencing Overhead

- 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
  - Also called transparent latch, D latch
- **Flip-Flop**: Edge triggered
  - Also called master-slave flip-flop, D flip-flop, D register, D Flop

- Timing Diagrams
  - Transparent
  - Opaque
  - Edge-trigger

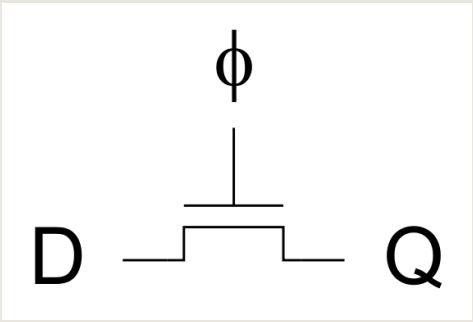


Latch Designs

Pass Transistor Latch

+ Tiny
+ Low clock load
– V_t drop
– Non-restoring
– Back driving
– Output noise sensitivity
– Dynamic
– Diffusion input

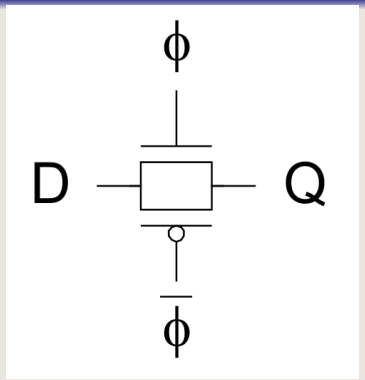
Used in the 1970s



Latch Designs, Cont'd

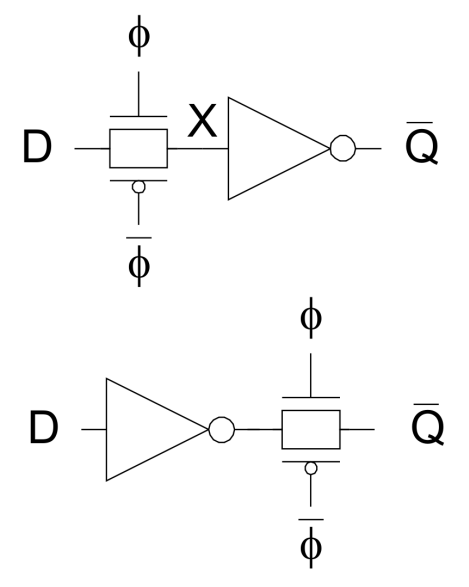
Transmission Gate Latch

+ No V_t drop
– Requires inverted clock



Inverting Buffer Latch

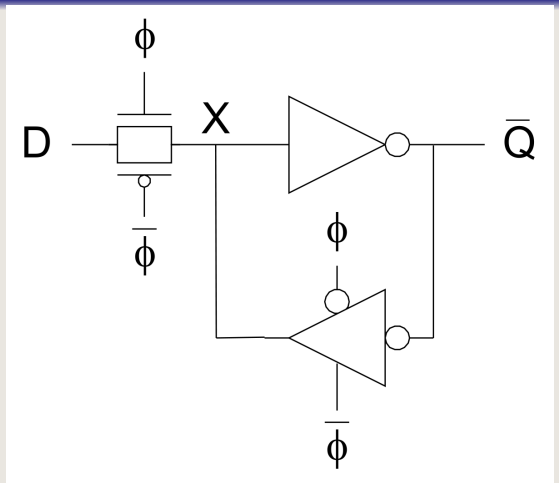
+ Restoring
+ No Backdriving
+ Fixes either
  - Output noise sensitivity
  - Or diffusion input
– Inverted output



Latch Designs, Cont'd

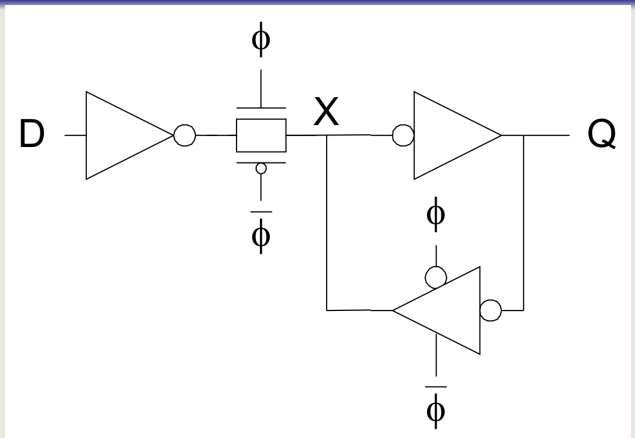
Latch with Tristate Feedback

+ Static
– Backdriving risk
- Static latches are now essential



Latch with Buffered Input

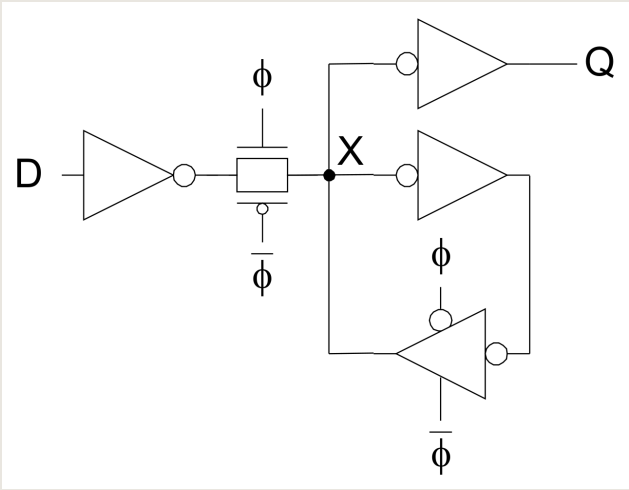
+ Fixes diffusion input
+ Non-inverting



Latch Designs, Cont'd

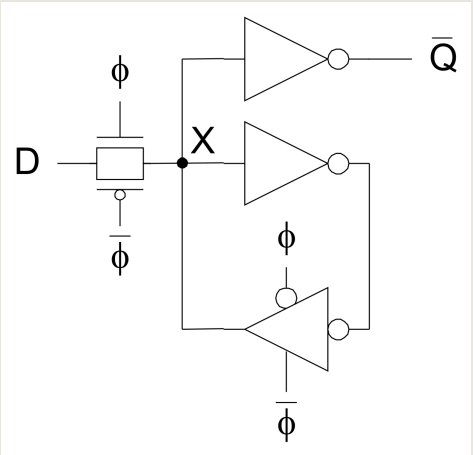
Latch with Buffered Output

- Widely used in standard cells
- \+ No backdriving
- \+ Very robust (most important)
- – Rather large
- – Rather slow (1.5 – 2 FO4 delays)
- – High clock loading



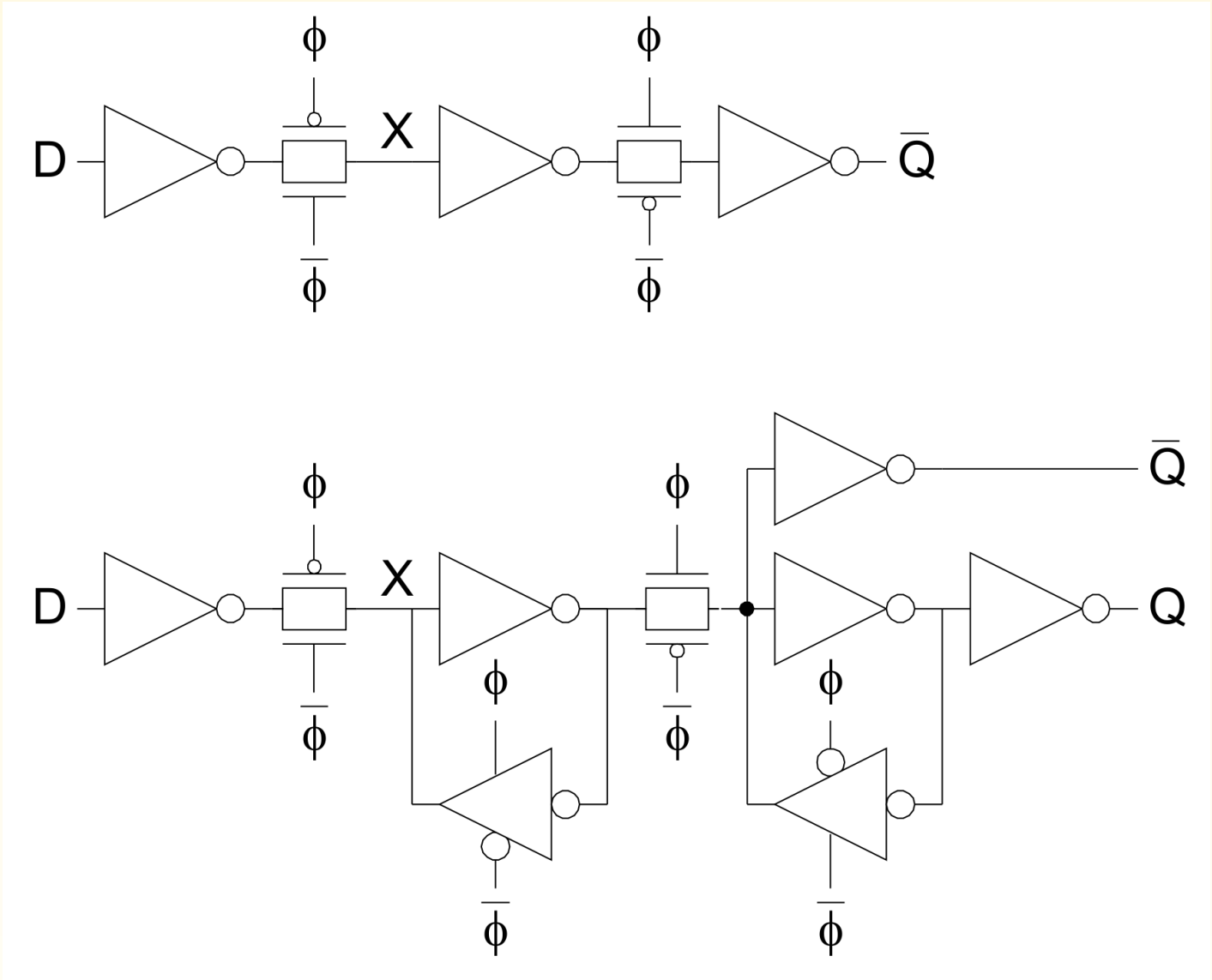
Datapath Latch

- \+ Smaller, faster
- \- Unbuffered input



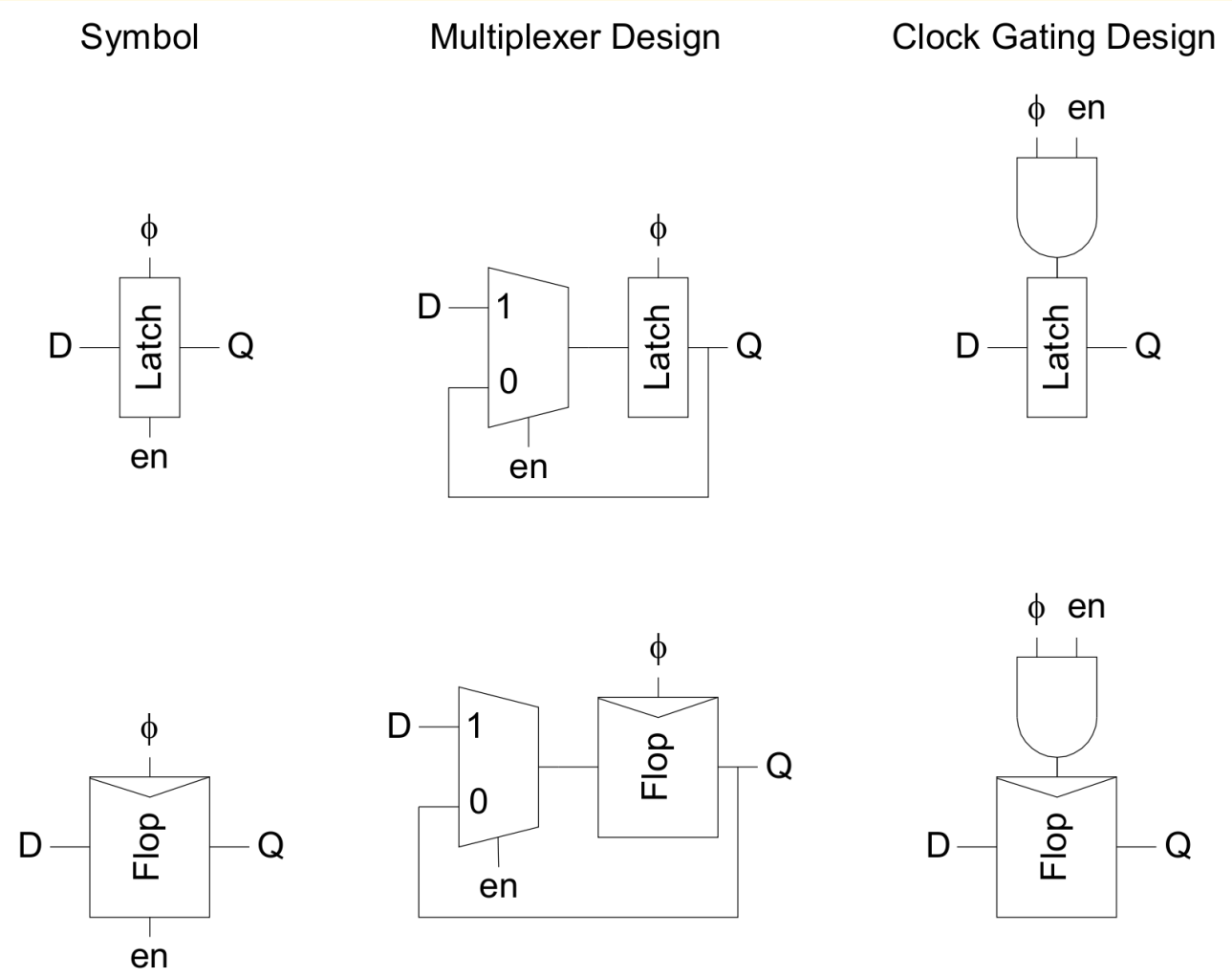
Flip-Flop Design

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



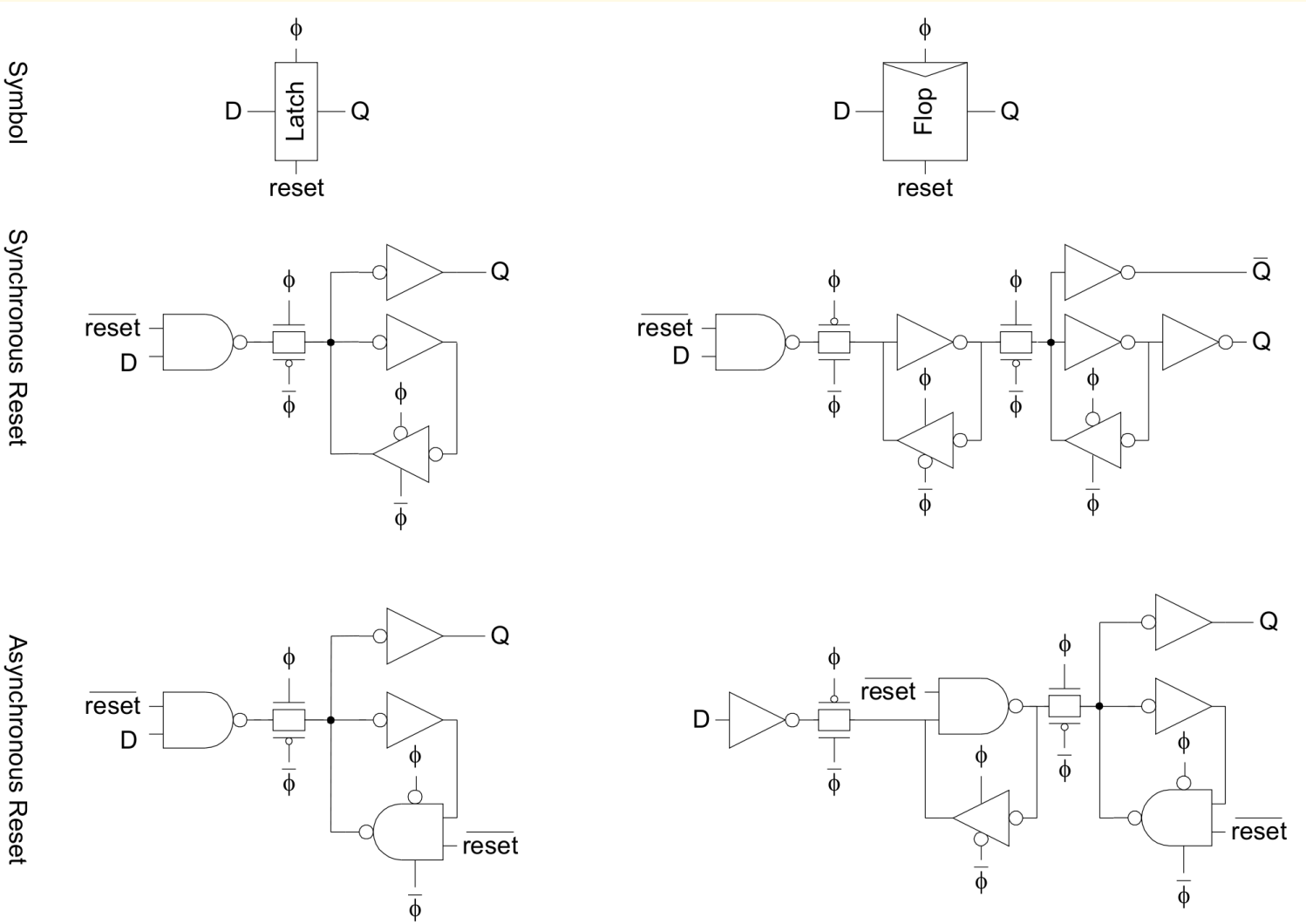
Enable

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



Reset

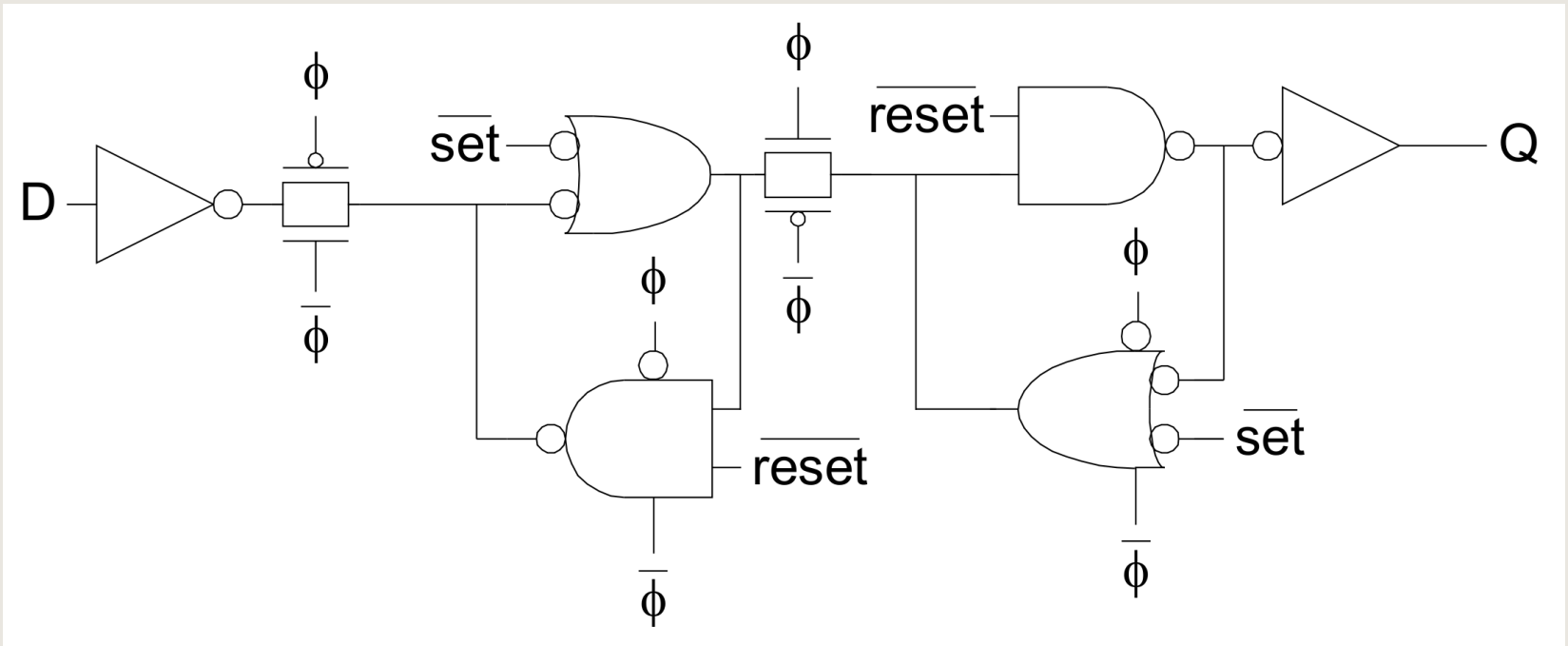
- Force output low when reset asserted
- Synchronous vs. asynchronous



Set/Reset

- Set forces output high when enabled

Flip-flop with asynchronous set and reset

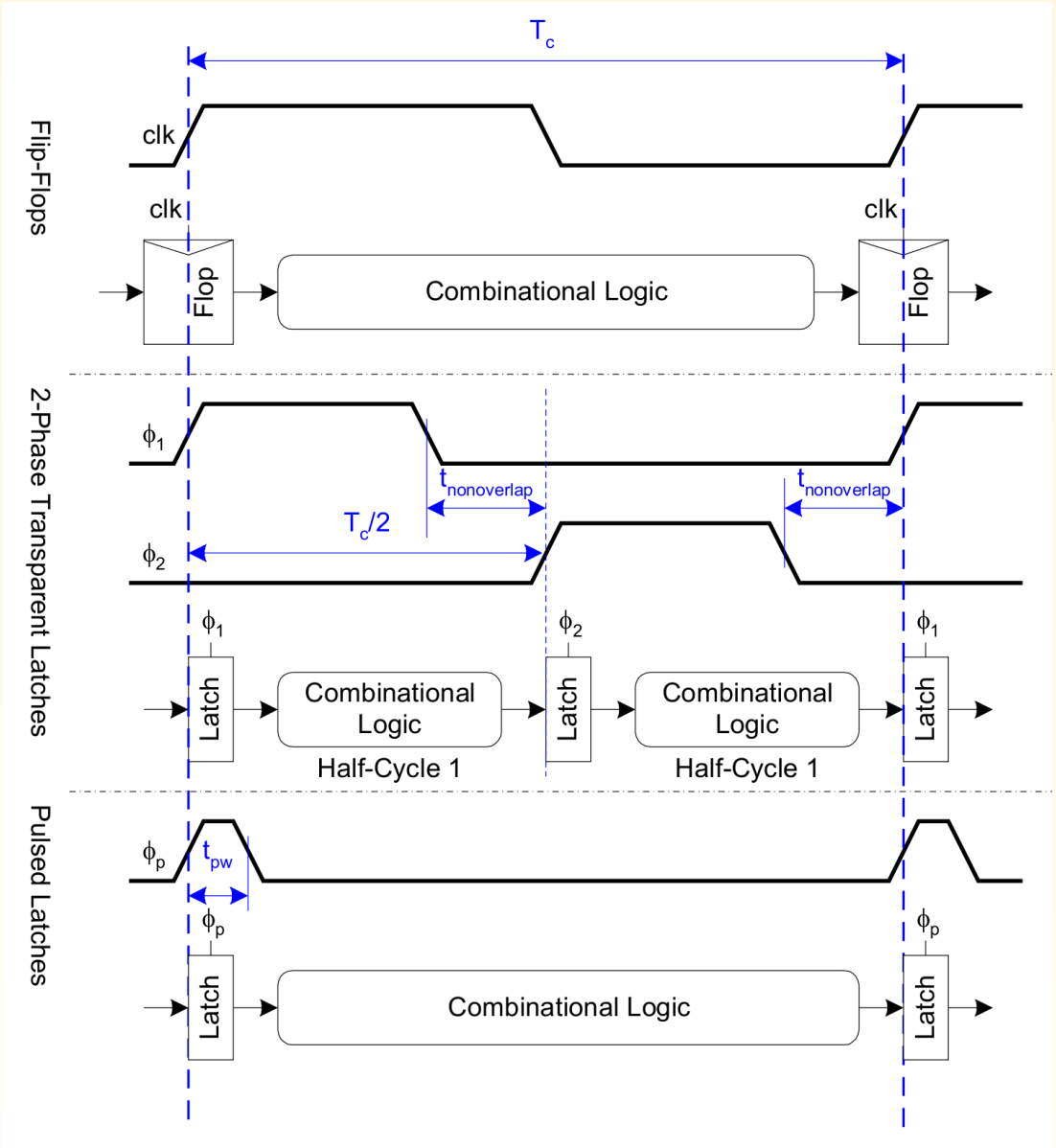


Sequencing Methods

Flip-Flops

2-Phase Transparent Latches

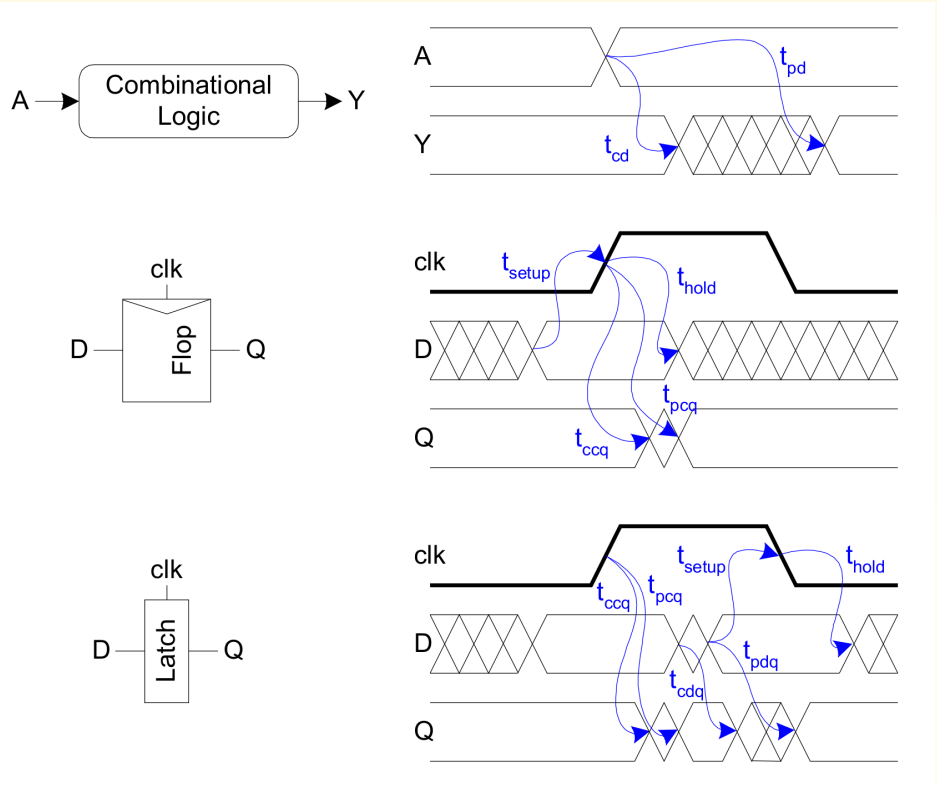
Pulsed Latches



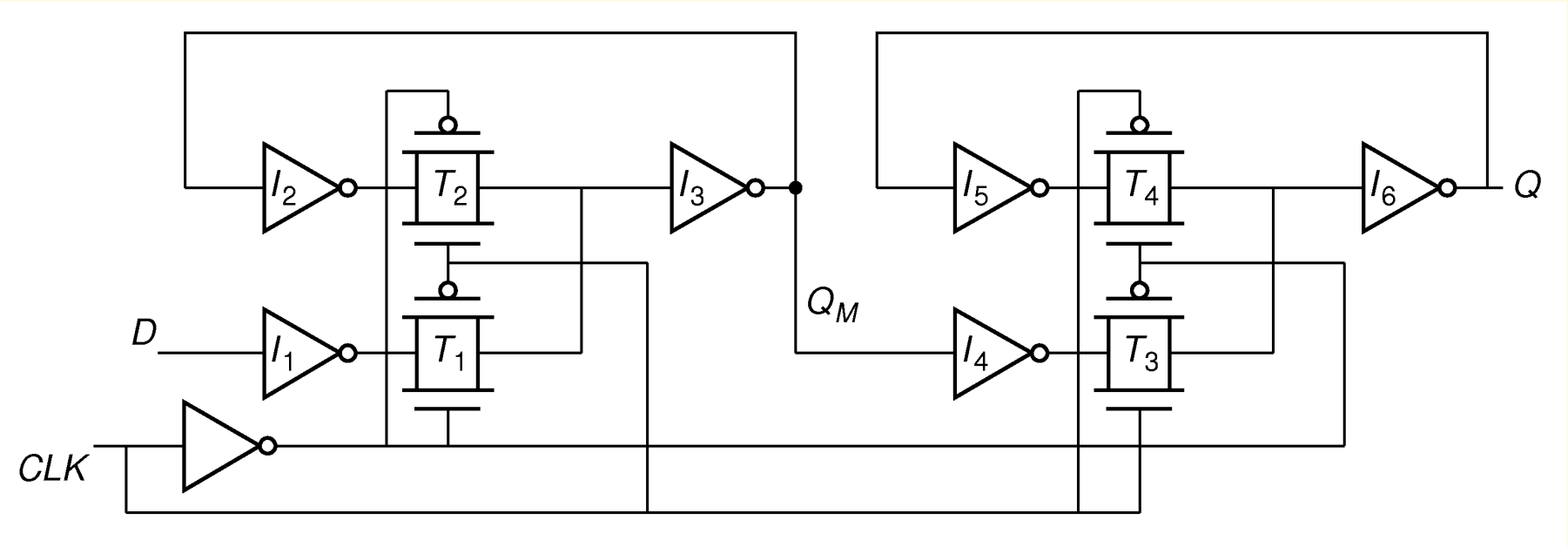
Timing Diagrams

Contamination and Propagation Delays

t_{pd}	Logic Propagation Delay
t_{cd}	Logic Contamination Delay
t_{pcq}	Latch/Flop Clk-Q Prop. Delay
t_{ccq}	Latch/Flop Clk-Q Cont. Delay
t_{pdq}	Latch D-Q Prop. Delay
t_{cdq}	Latch D-Q Cont. Delay
t_{setup}	Latch/Flop Setup Time
t_{hold}	Latch/Flop Hold Time



Example: Master-Slave Flip Flop



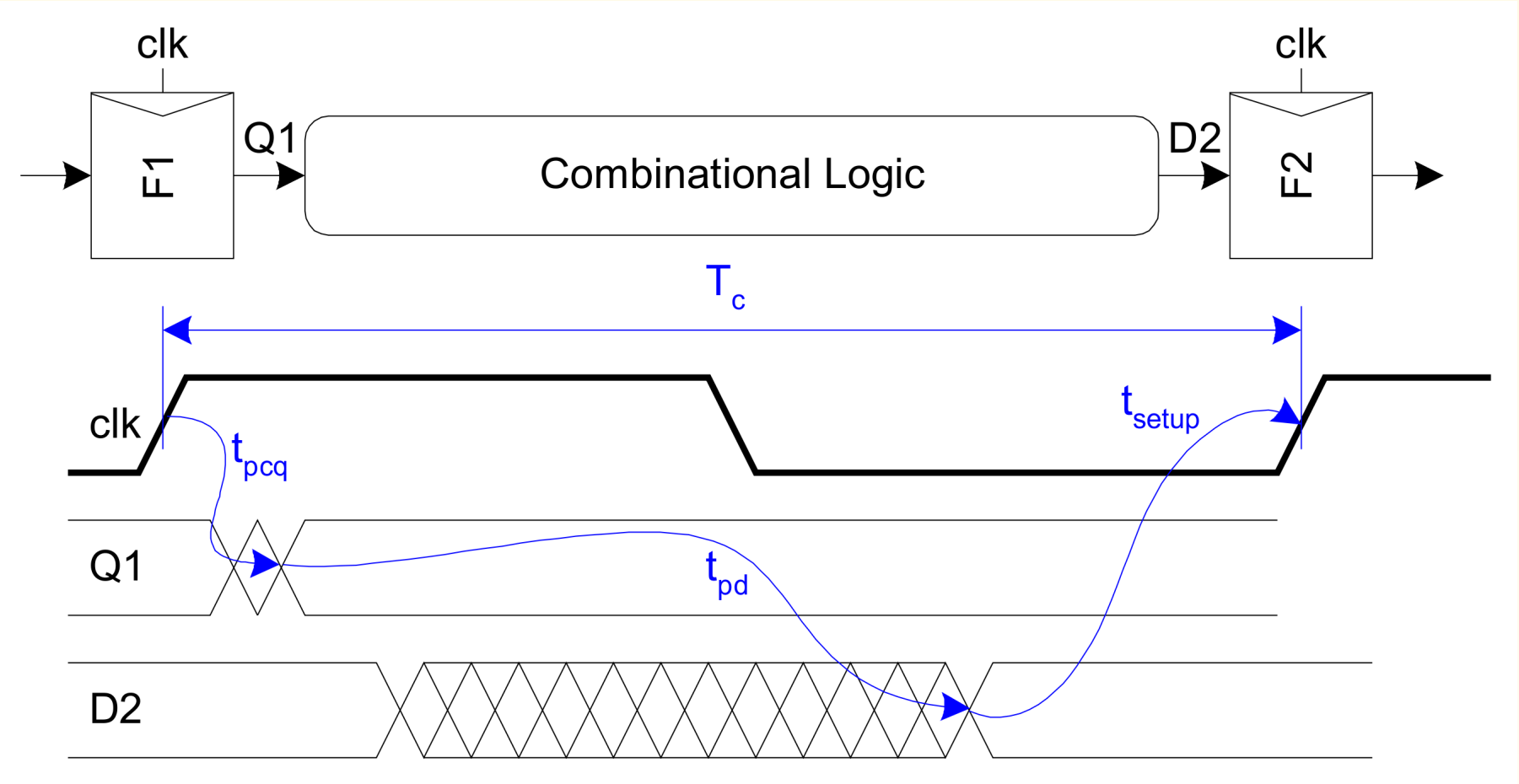
$$t_{setup} =$$

$$t_{pcq} =$$

$$t_{hold} =$$

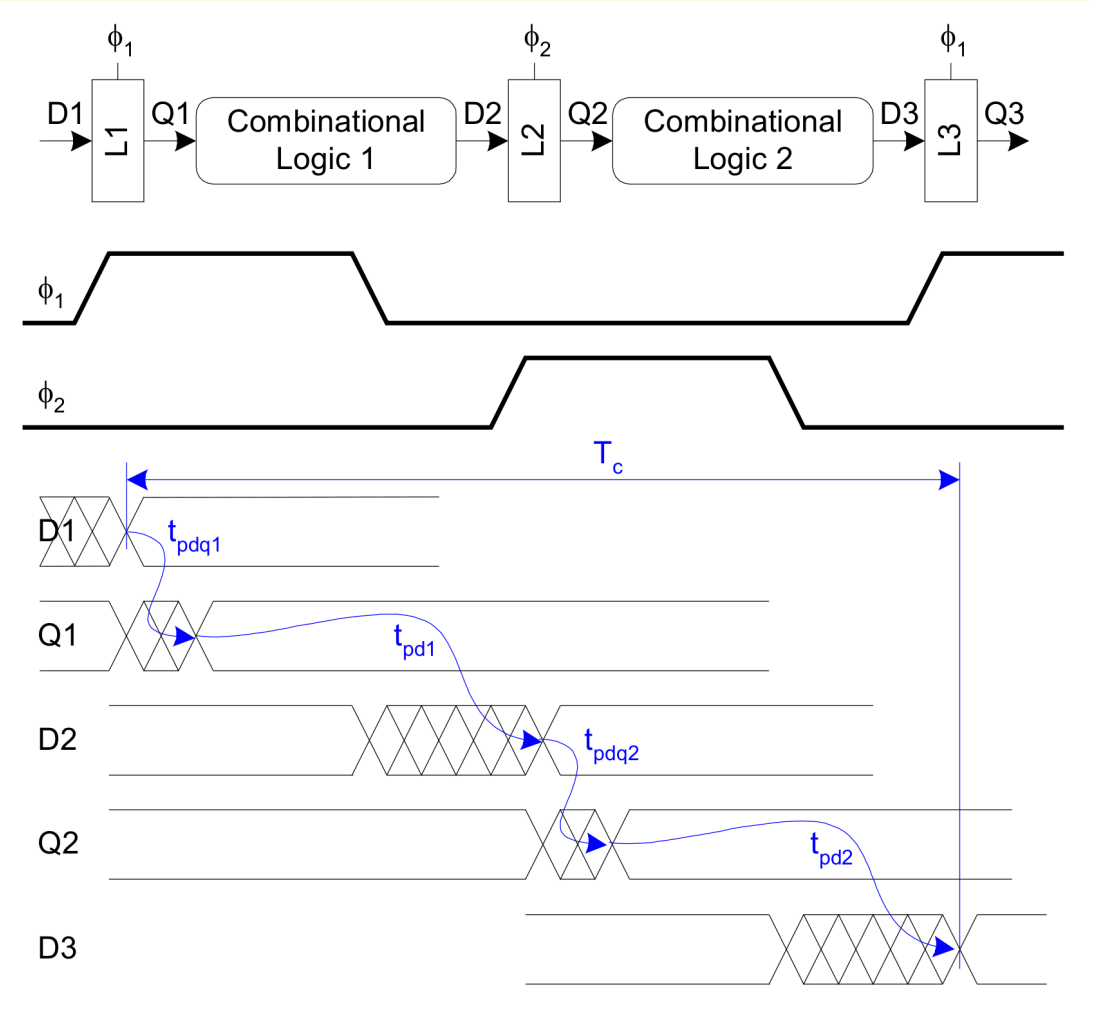
Maximum Delay: Flip-Flops

$$t_{pd} \leq T_c - \underbrace{(t_{setup} + t_{pcq})}_{\text{sequencing overhead}}$$



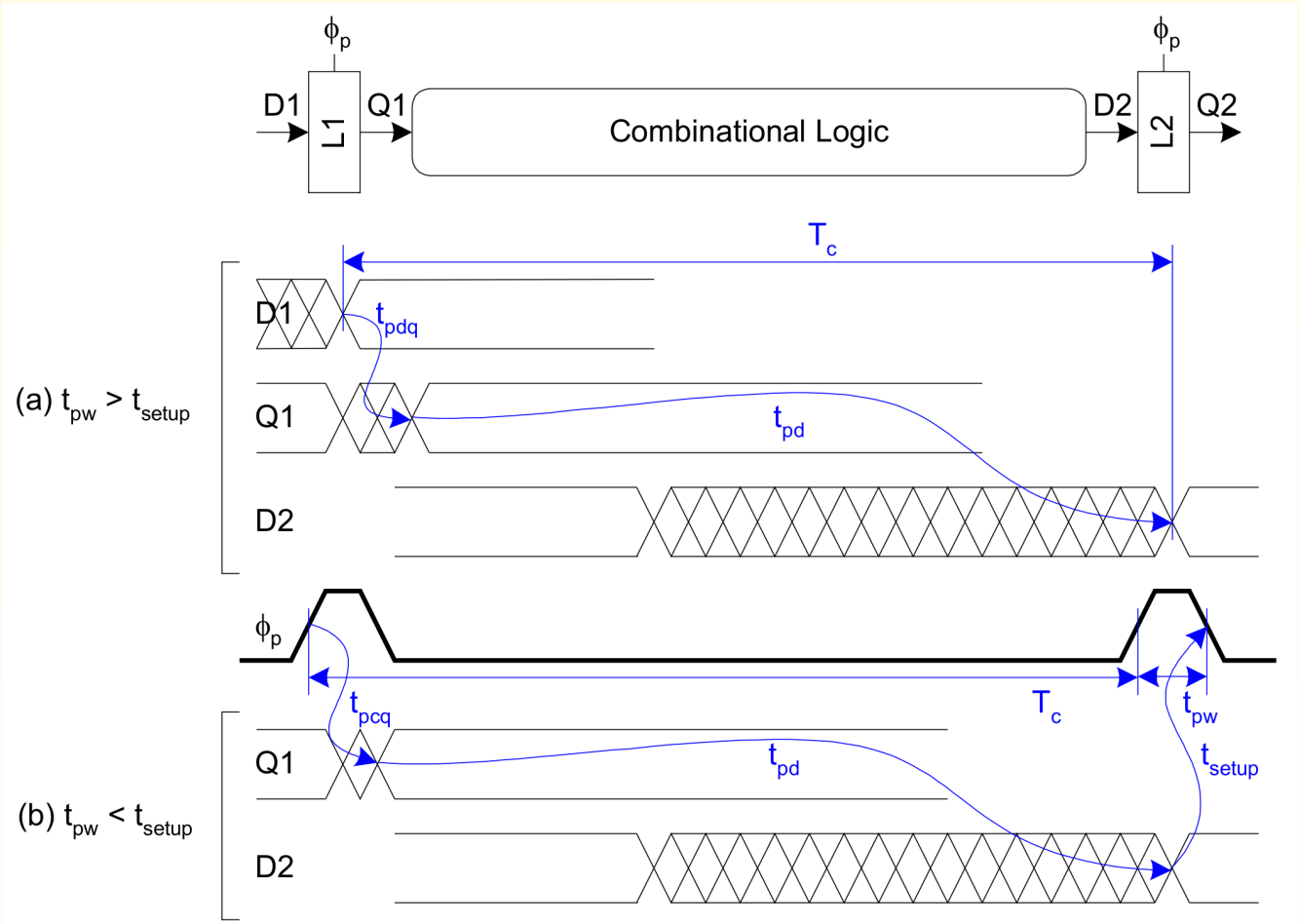
Maximum Delay: 2-Phase Latches

$$t_{pd} = t_{pd1} + t_{pd2} \leq T_c - \underbrace{(2t_{pdq})}_{\text{sequencing overhead}}$$



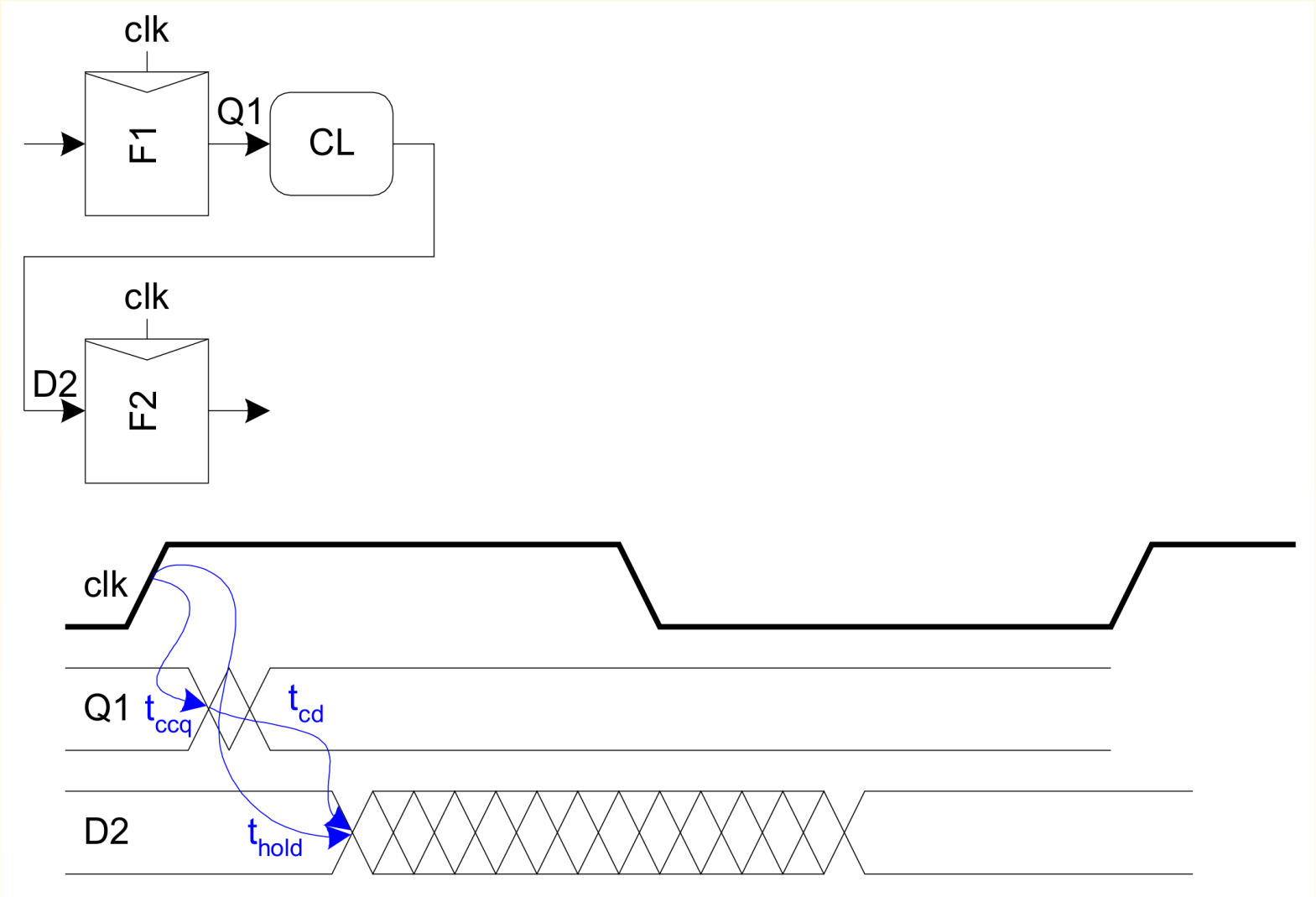
Maximum Delay: Pulsed Latches

$$t_{pd} \leq T_c - \underbrace{\max(t_{pdq}, t_{pcq} + t_{setup} - t_{pw})}_{\text{sequencing overhead}}$$



Minimum Delay: Flip-Flops

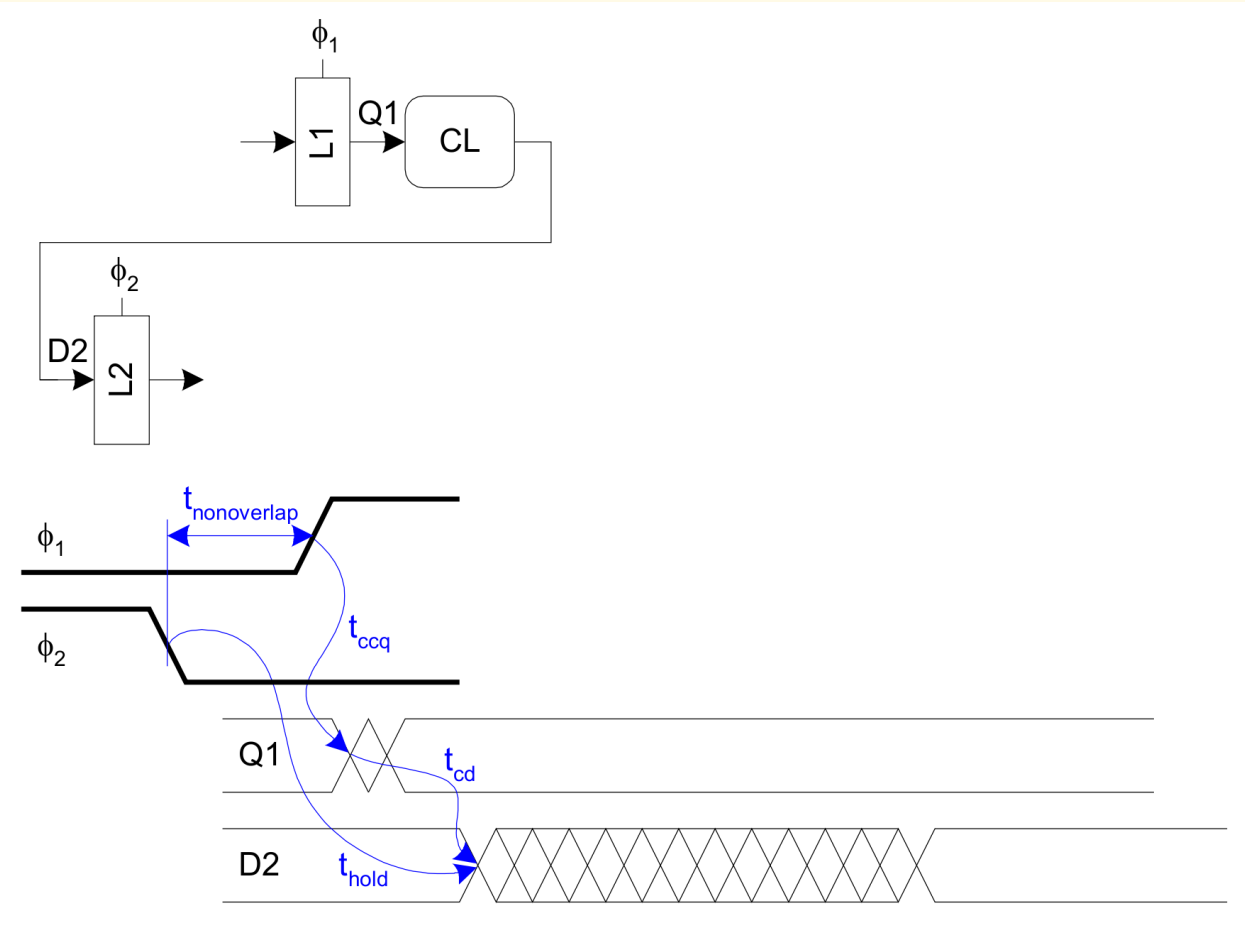
$$t_{cd} \geq t_{hold} - t_{ccq}$$



Minimum Delay: 2-Phase Latches

$$t_{cd1}, t_{cd2} \geq t_{hold} - t_{ccq} - t_{nonoverlap}$$

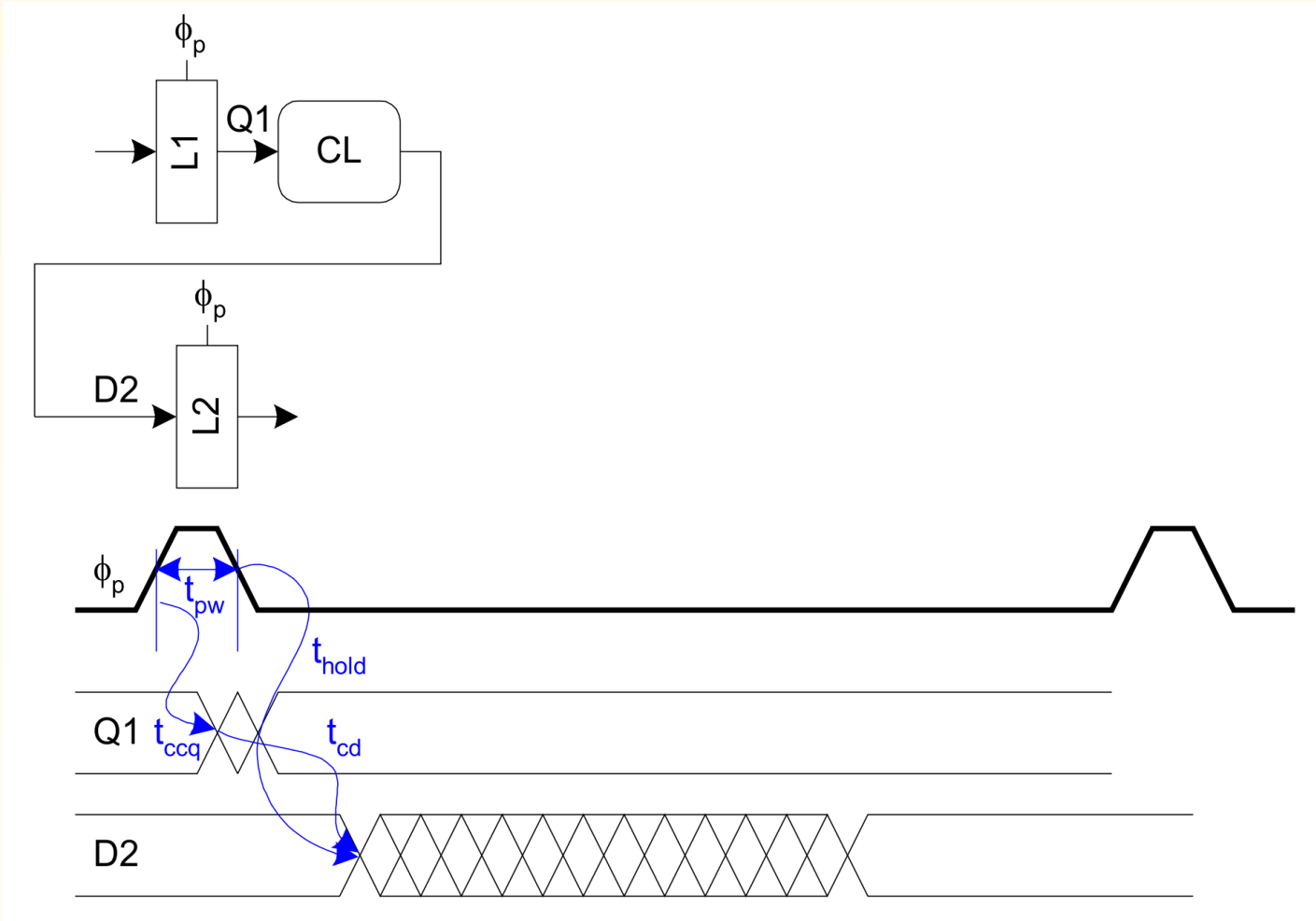
- Hole time reduced by nonoverlap
- Paradox: hold applies twice each cycle, versus only once for flops
  - But a flop is made of two latches!



Minimum Delay: Pulsed Latches

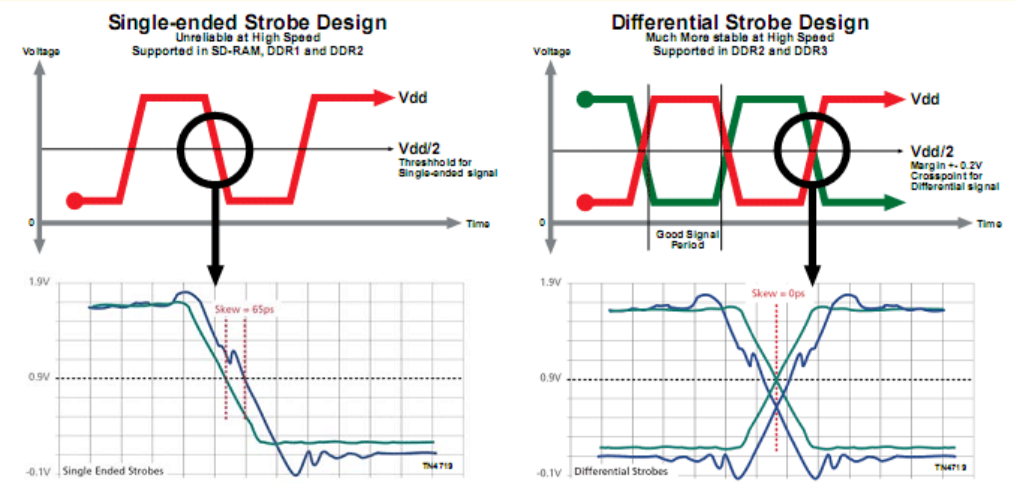
$$t_{cd} \geq t_{hold} - t_{ccq} + t_{pw}$$

Hold time increased by pulse width



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

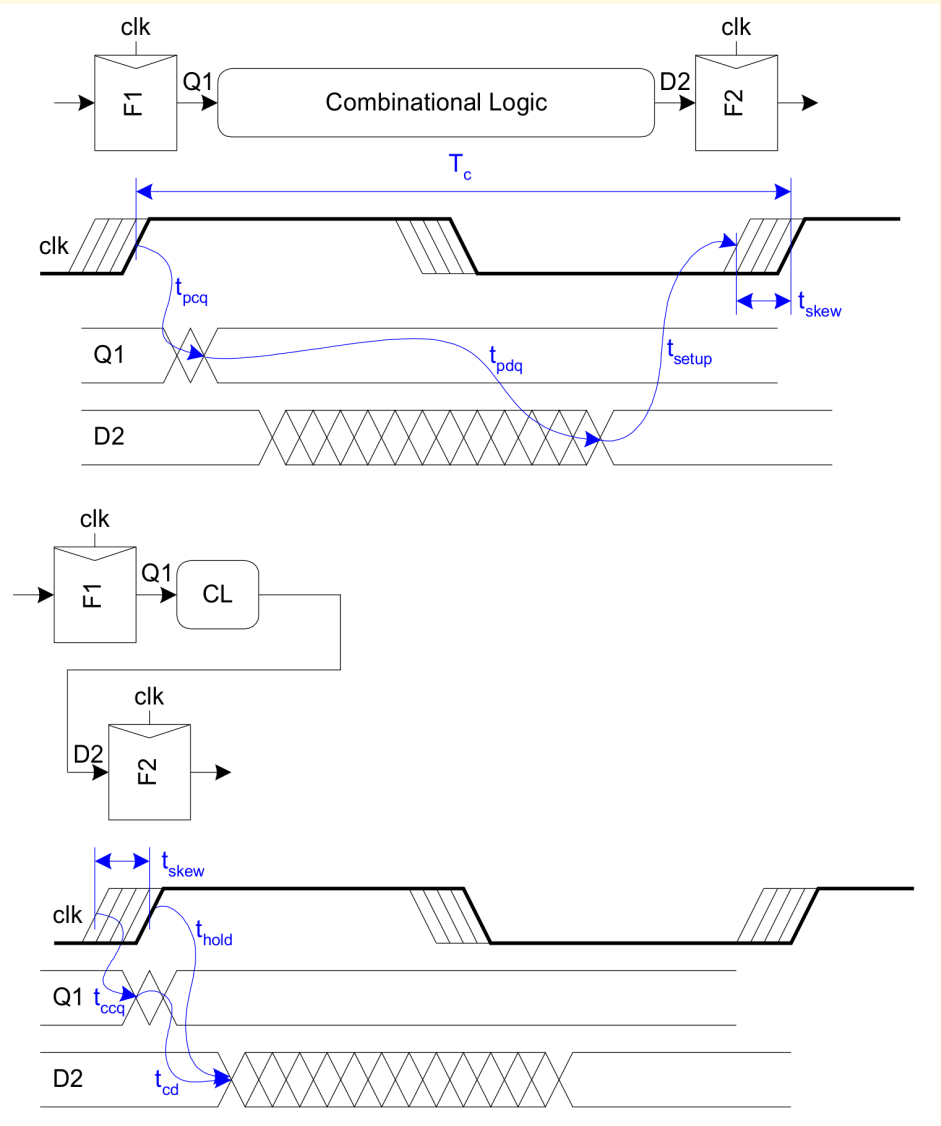


▲ Comparison of Single-ended and Differential Strobes
Strobes Chart Source: Micron Technology.
Illustration: Ryan J. Leng

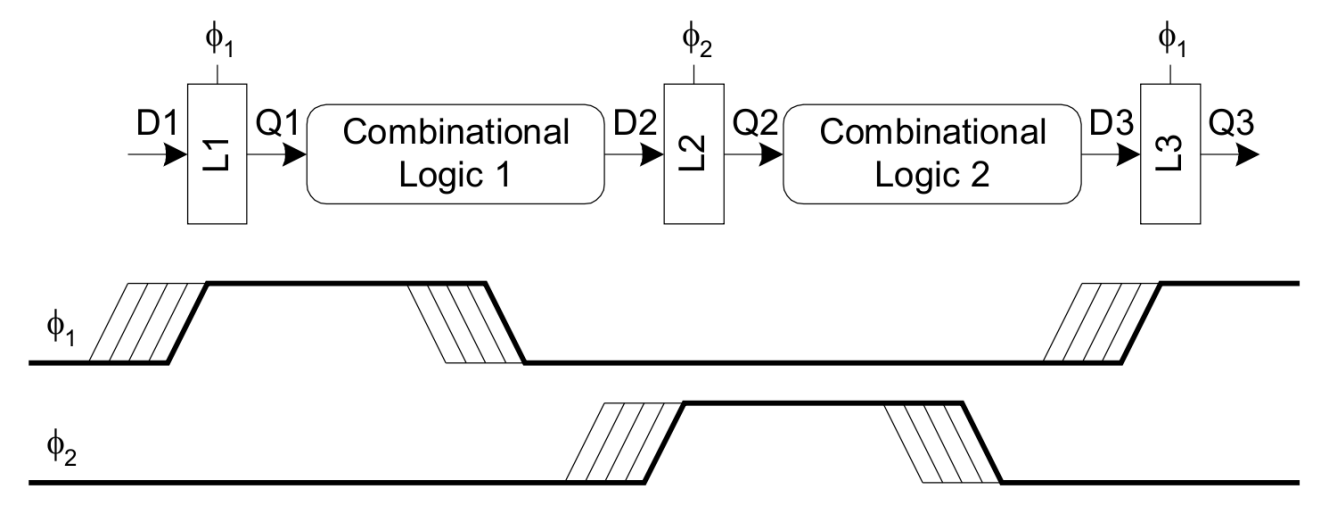
Skew: Flip-Flops

$$t_{pd} \leq T_c - \underbrace{(t_{pcq} + t_{setup} + t_{skew})}_{\text{sequencing overhead}}$$

$$t_{cd} \geq t_{hold} - t_{ccq} + t_{skew}$$



Skew: Latches



- 2-Phase Latches

$$t_{pd} \leq T_c - \underbrace{(2t_{pdq})}_{\text{sequencing overhead}}$$

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

$$t_{borrow} \leq T_c/2 - (t_{setup} + t_{nonoverlap} + t_{skew})$$

- Pulsed Latches

$$t_{pd} \leq T_c - \underbrace{\max(t_{pdq}, t_{pcq} + t_{setup} - t_{pw} + t_{skew})}_{\text{sequencing overhead}}$$

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

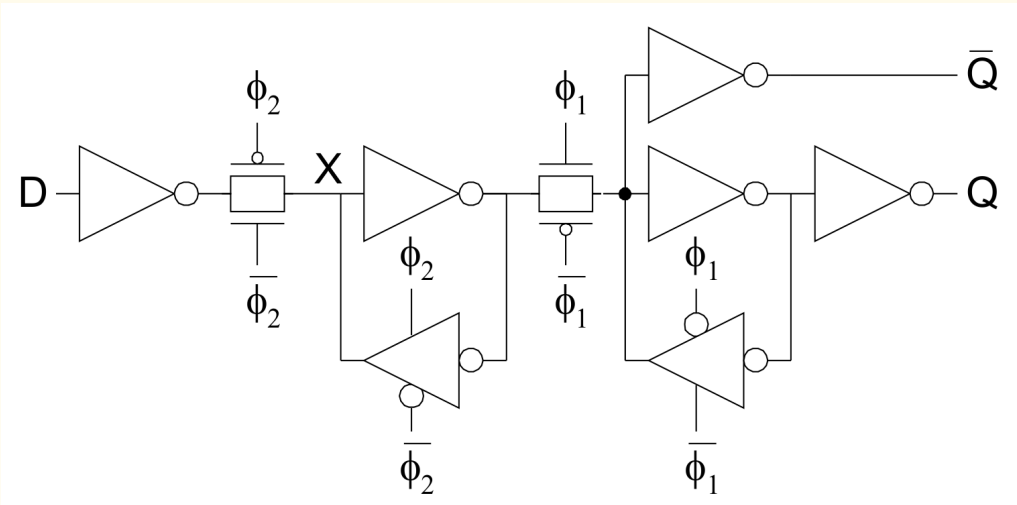
$$t_{borrow} \leq t_{pw} - (t_{setup} + t_{skew})$$

Two-Phase Clocking

- If setup times are violated, reduce clock speed
- If hold times are violated, chip fails at any speed
- Use tools to analyze clock skew
- Easy way to guarantee hold times: use 2-phase latches with big non-overlap times (used in academic designs)
- Call these clocks ϕ_1, ϕ_2 (ph1, ph2)

Safe Flip-Flop

- Flip-Flop with non-overlapping clocks
- Very slow – nonoverlap adds to setup time, but no hold time problem
- Use timing analysis and add buffers to slow signals if hold time is at risk



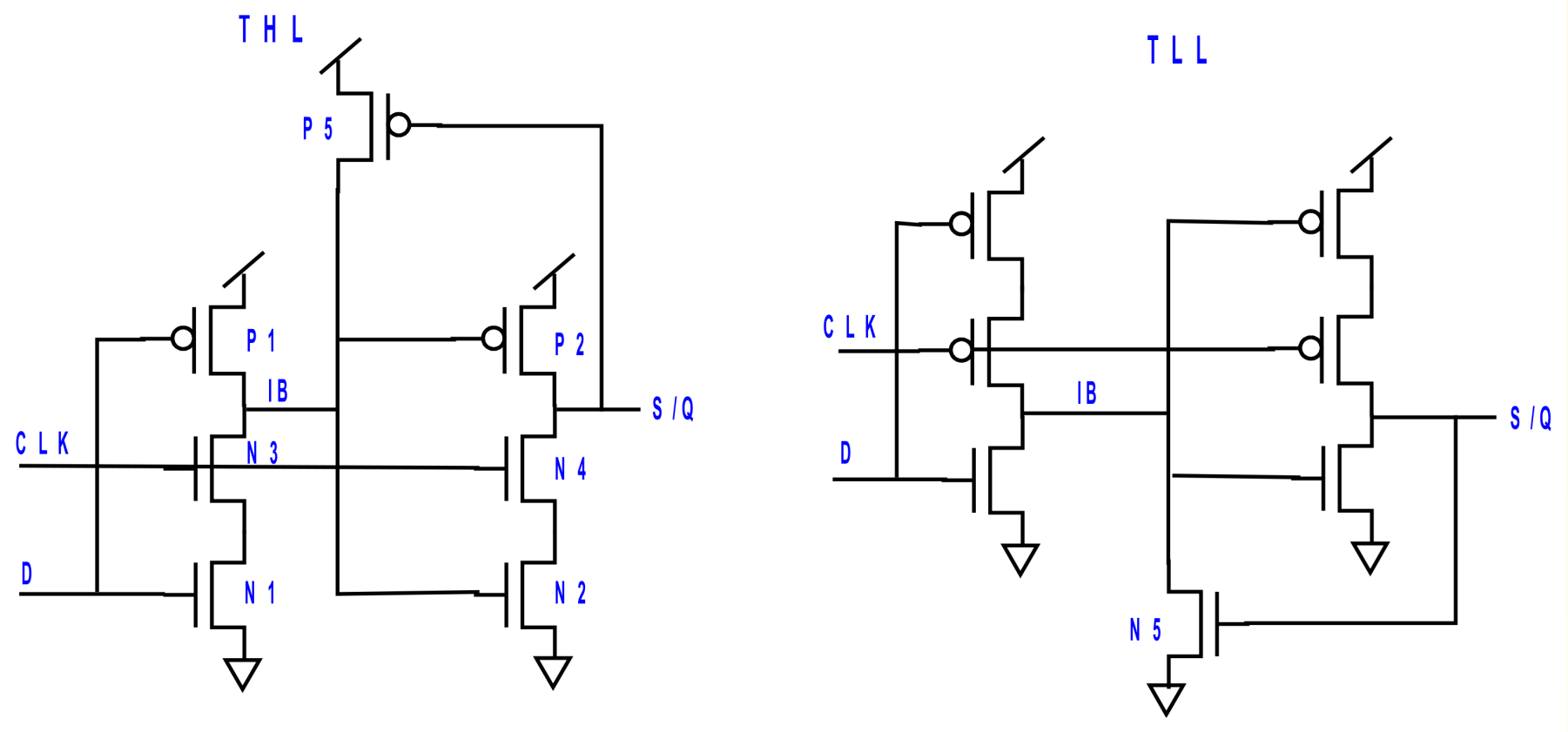
Summary

- Flip-Flops
  - Very easy to use, supported by all tools
- 2-Phase Transparent Latches
  - Lost of skew tolerance and time borrowing
- Pulsed Latches
  - Fast, some skew tolerance and borrowing, hold time risk

	Sequencing overhead ($T_c - t_{pd}$)	Minimum logic delay (t_{cd})	Time borrowing (t_{borrow})
Flip-flops	$t_{pcq} + t_{setup} + t_{skew}$	$t_{hold} - t_{ccq} + t_{skew}$	0
Two-phase transparent latches	$2t_{pdq}$	$t_{hold} - t_{ccq} - t_{nonoverlap} + t_{skew}$ in each half-cycle	$\frac{T_c}{2} - (t_{setup} + t_{nonoverlap} + t_{skew})$
Pulsed latches	$\max(t_{pdq}, t_{pcq} + t_{setup} - t_{pw} + t_{skew})$	$t_{hold} - t_{ccq} + t_{pw} + t_{skew}$	$t_{pw} - (t_{setup} + t_{skew})$

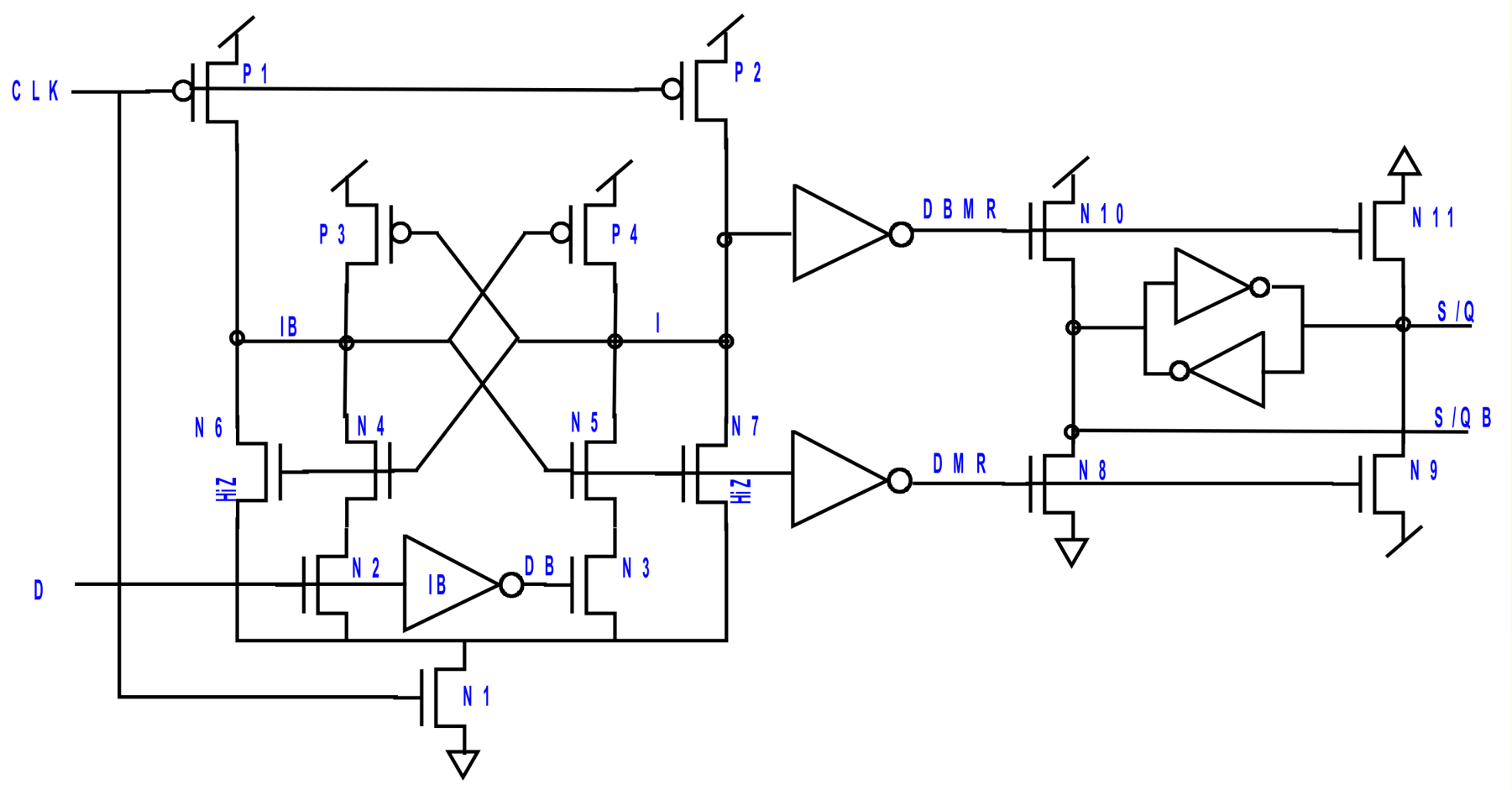
High Performance Flops

The modified Svensson latch, DEC Alpha 21064



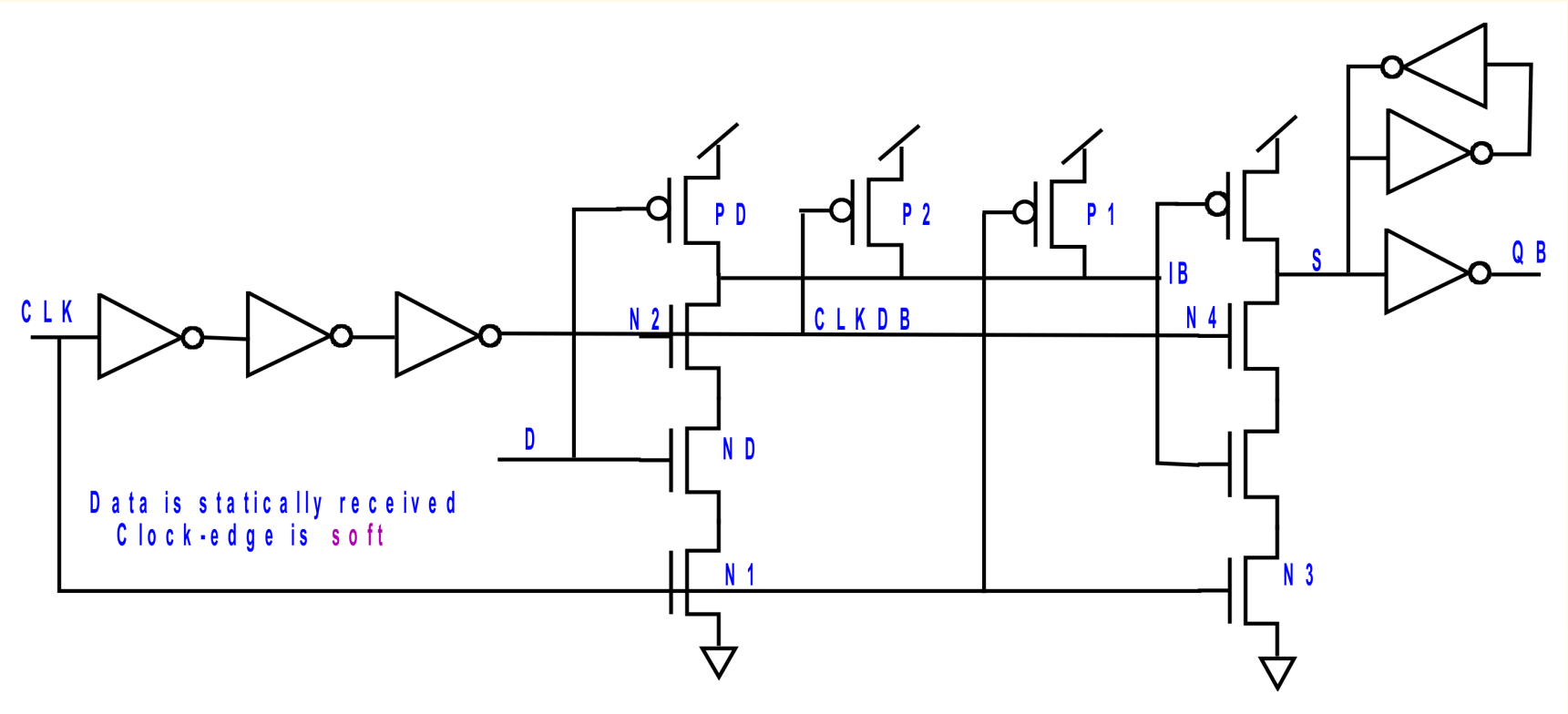
High Performance Flops, Cont'd

The amplifier-based flip-flop



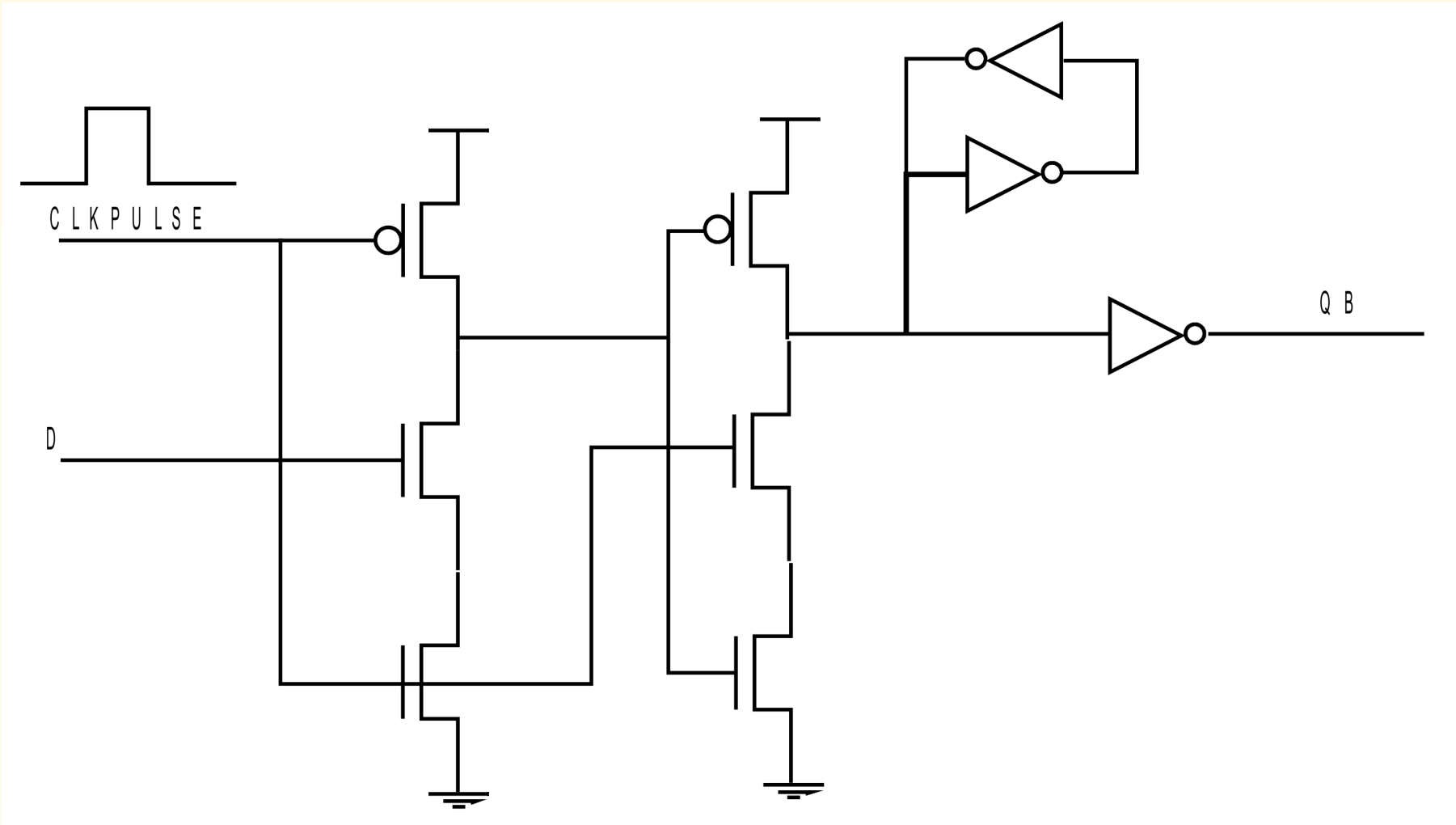
High Performance Flops, Cont'd

The hybrid latch flip-flop of AMD K6



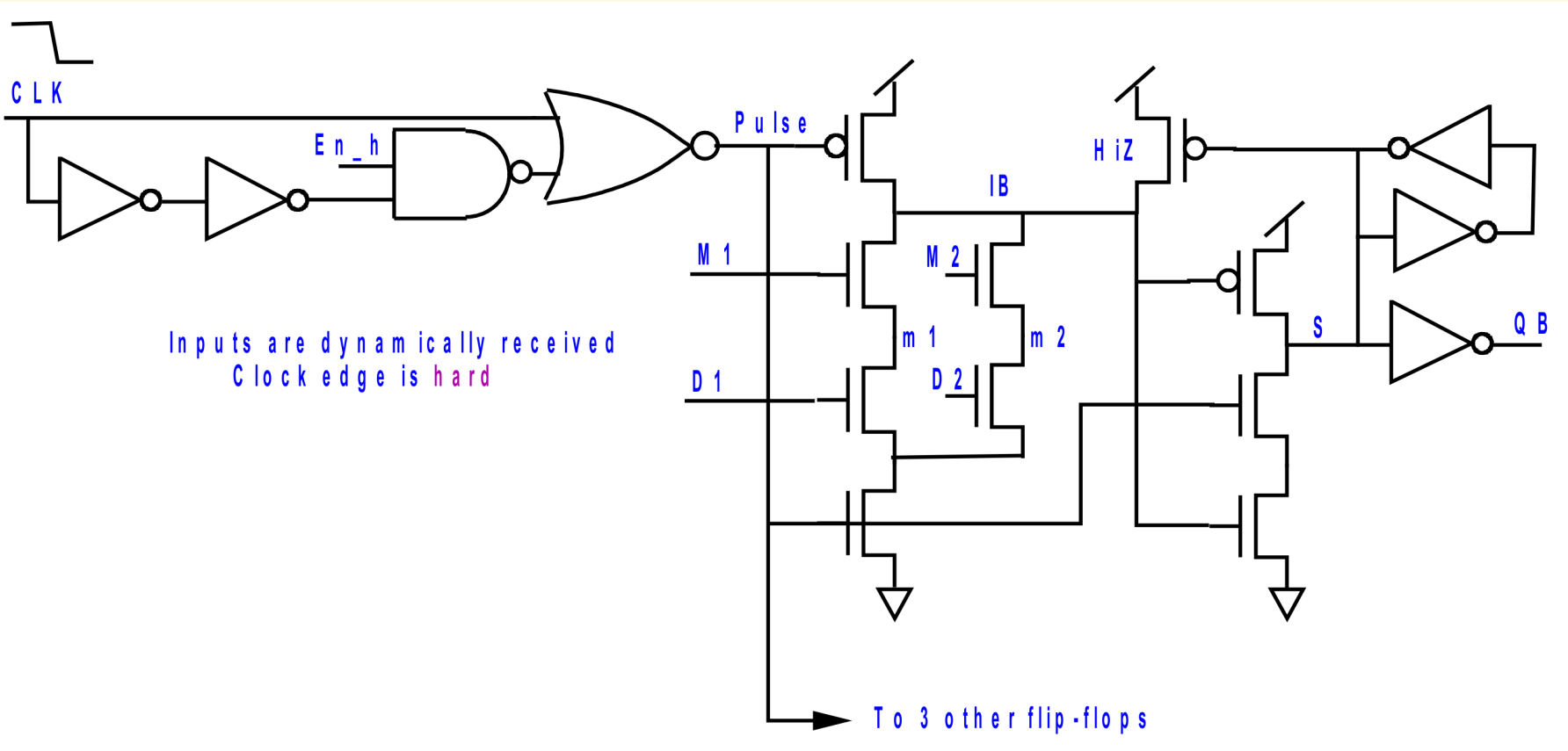
Basic Flop in AMD Athlon Processor

Clock pulse is generated using monoshots at the rising edge

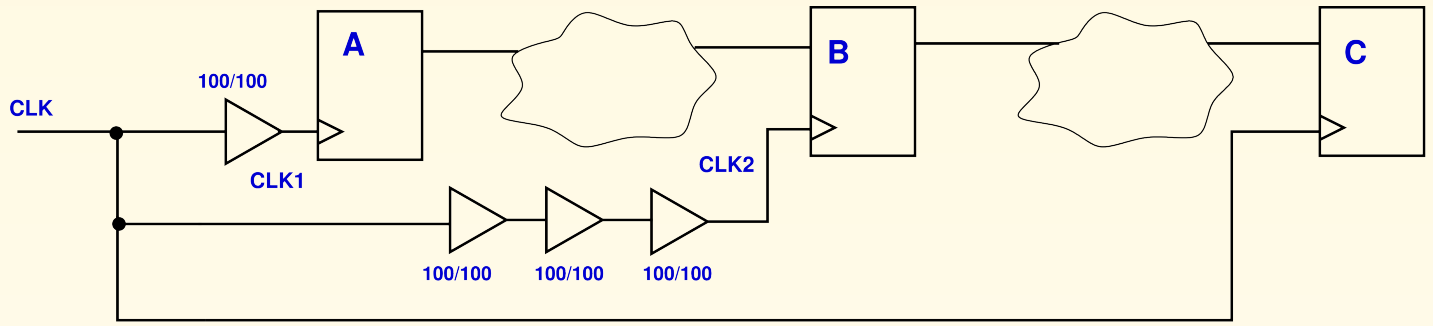


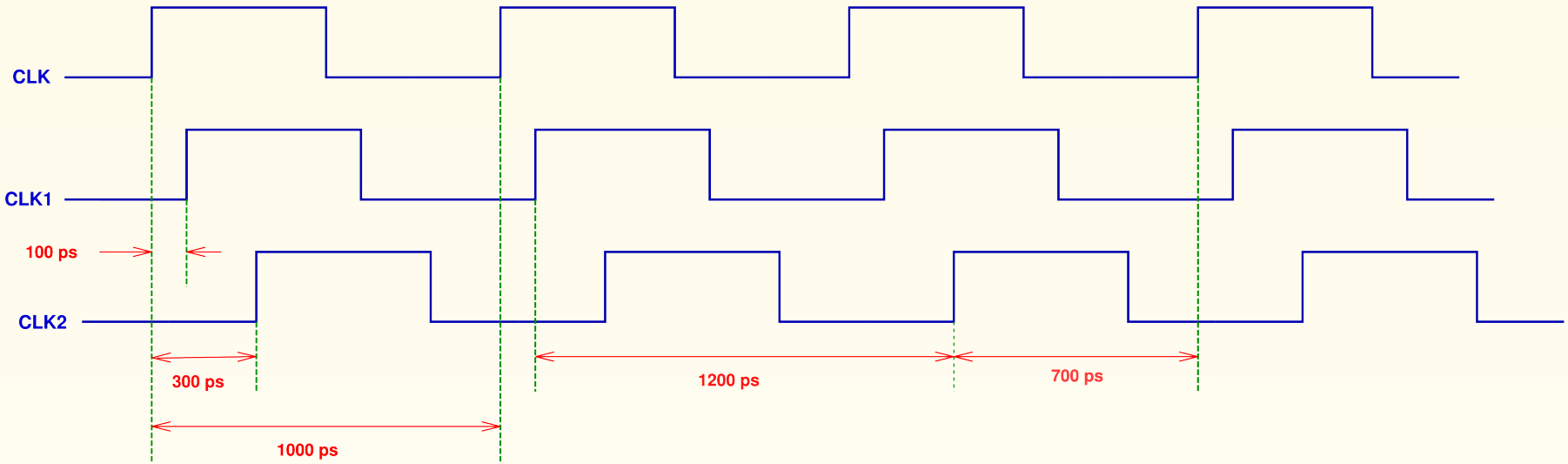
High Performance Flops, Cont'd

The enabled two-way MUX pulsed flip-flop of K7



Cycle Stretching





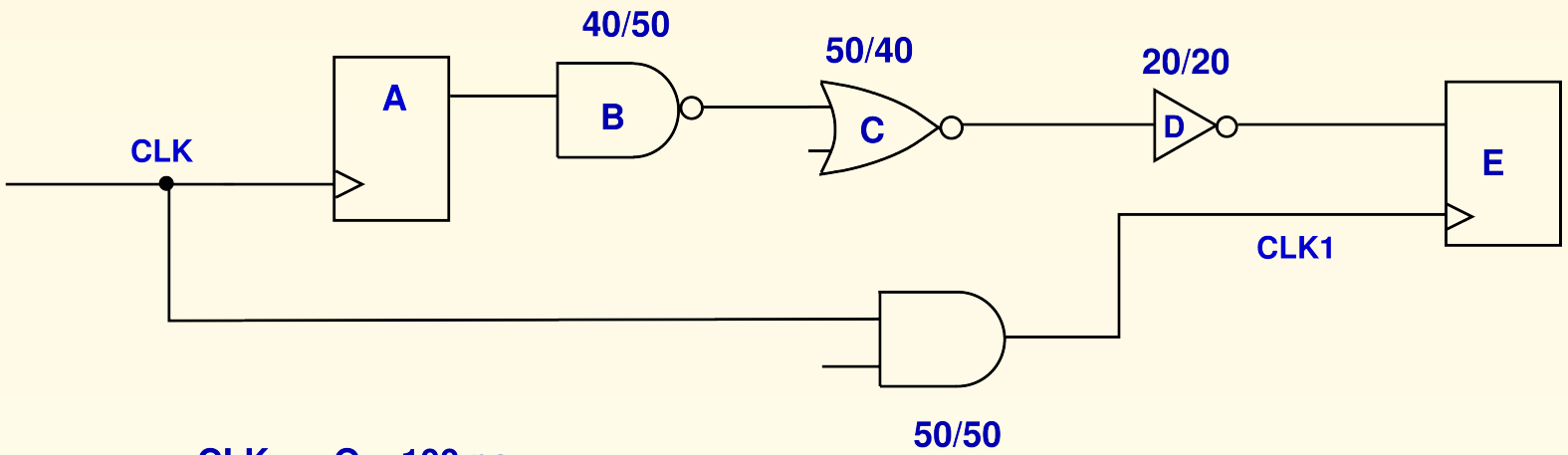
Cycle time between A & B = 1200 ps

Cycle time between B & C = 700 ps

Fixing Hold-Time Violations

- Measure all hold times with respect to the main clock
- Adjust the hold time if the flop is receiving a delayed clock
- Compute the shortest path delay from the rising edge of the clock
- Check to see if there are any hold time failures

Example: Fixing Hold-Time Violations



CLK --> Q = 100 ps

Setup = 10 ps

Hold = 200 ps (A large hold time is used to illustrate the problem)

Shortest path delay from A $\rightarrow$ E $= 100 + 40 + 40 + 20 = 200$ ps

Delay between CLK1 and CLK $= 50$ ps

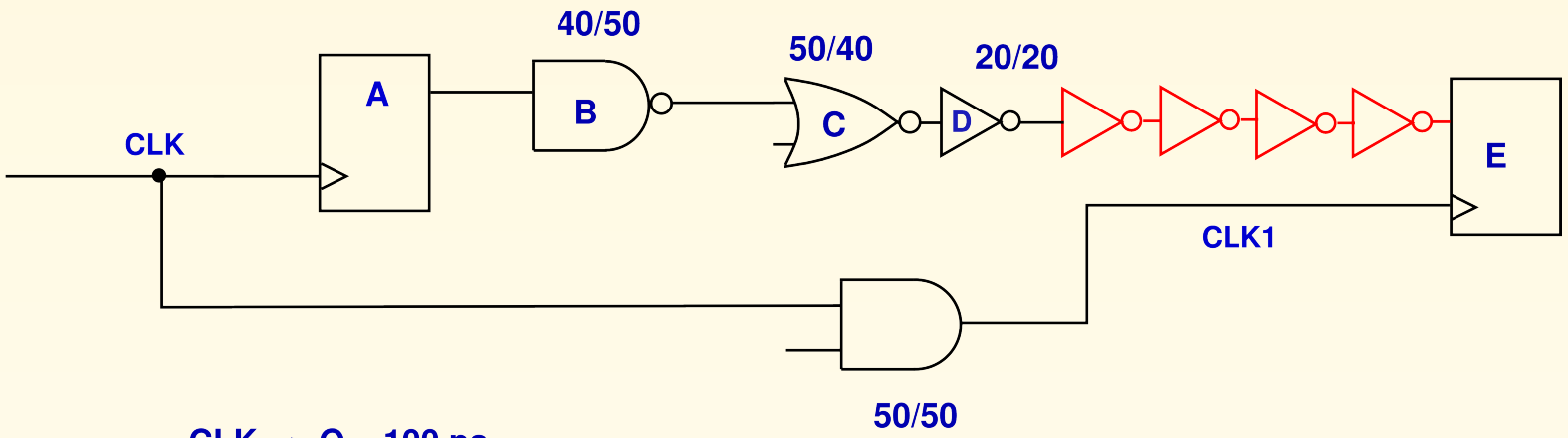
Adjusted hold time $= 200 + 50 = 250$ ps

Hold Slack $=$ (Path Delay) $-$ (Adjusted Hold Time) $= 200 - 250$

$= -50$ ps

$\Longrightarrow$ **FAIL (Hold slack should be $\geq$ 0)**

Example: Fixing Hold-Time Violations, Cont'd



Insert 4 inverters after D, with each adding a 20 ps (or can insert one AND gate)

Long path (4 invs.) $= 100 + 50 + 50 + 20 + 80 + 10 = 310$ ps

Now the minimum cycle time at which the path can operate = (Path Delay) $-$ (CLK $\rightarrow$ CLK1 Delay) $= 310 - 50 = 260$ ps

If possible, add the additional delay to fix hold time violations in the short path (without affecting the long paths)