EEC 116 Lecture #6: Sequential Logic

Rajeevan Amirtharajah
University of California, Davis
Jeff Parkhurst
Intel Corporation

Announcements

- **Lab 3 extended, same due date as Lab 4**
- **HW4 issued today**

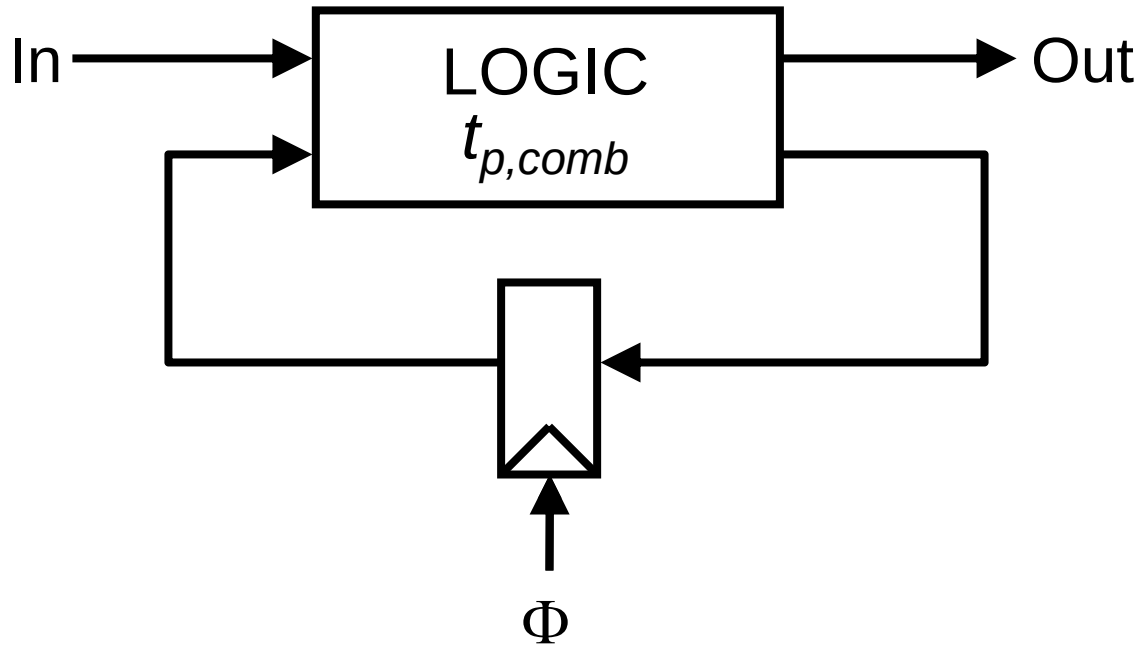
Outline

- **Review: Arithmetic**
- **Finish Transmission Gate Circuits and Multipliers**
- **Sequential MOS Logic Circuits: Rabaey, 7.1-7.3 (Kang & Leblebici, 8.1-8.5)**
- **Next Topic: Latchup and Layout Guidelines**

Sequential Logic Basic Definition

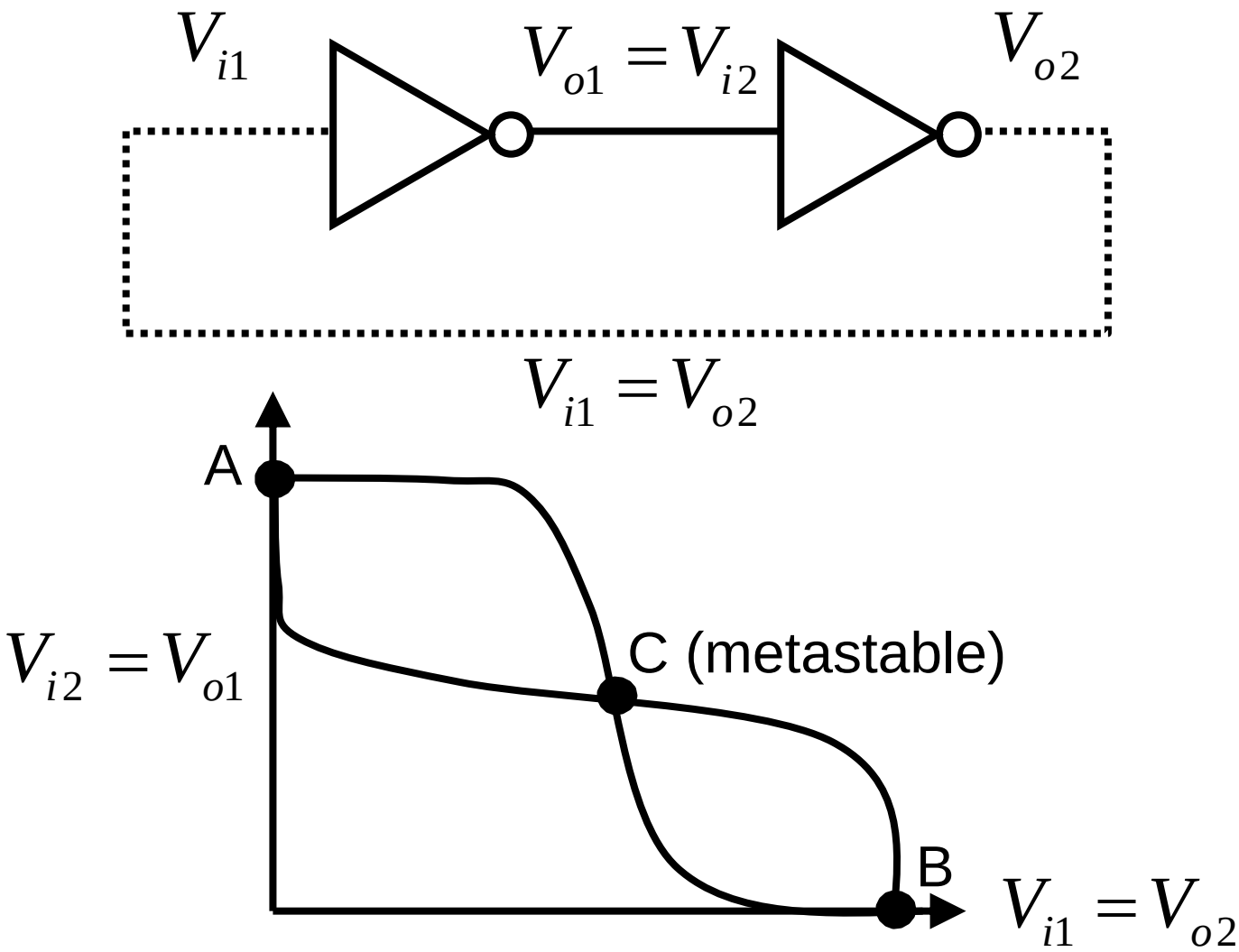
- **Combinational circuits' output is a function of the circuit inputs and a delay time**
  - Examples: NAND, NOR, XOR, adder, multiplier
- **Sequential circuits' output is a function of the circuit inputs, previous circuit state, and a delay time**
  - Examples: Latches, flip-flops, FSMs, pipelined adders and multipliers, microprocessors
  - Sequential elements are critical to implementing techniques such as feedback or blocks such as memory

Sequential Logic Example: Mealy FSM

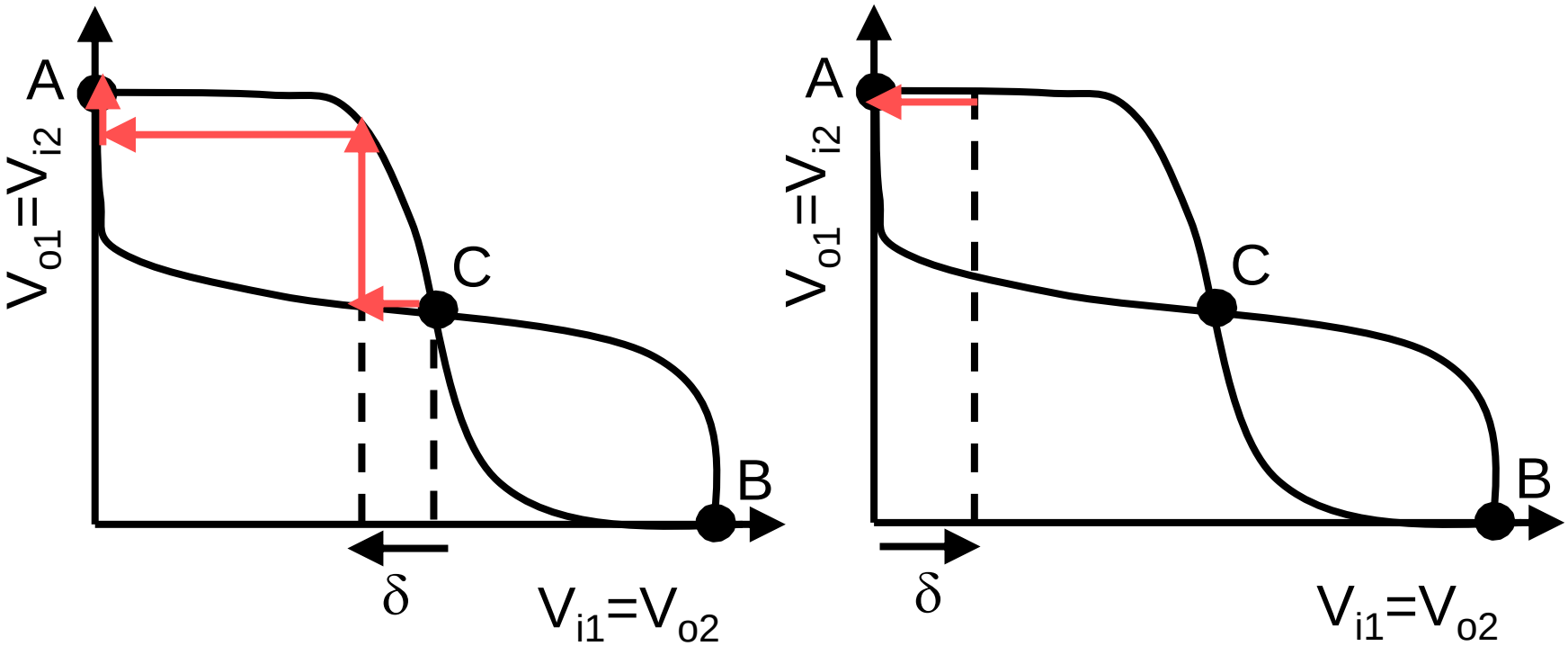


- **Two information storage mechanisms**
  - Positive feedback-based (static) circuits
  - Charge storage-based (dynamic) circuits
- **Clock signal Φ controls timing of state (memory) updates**

Positive Feedback: Bistability



Metastability



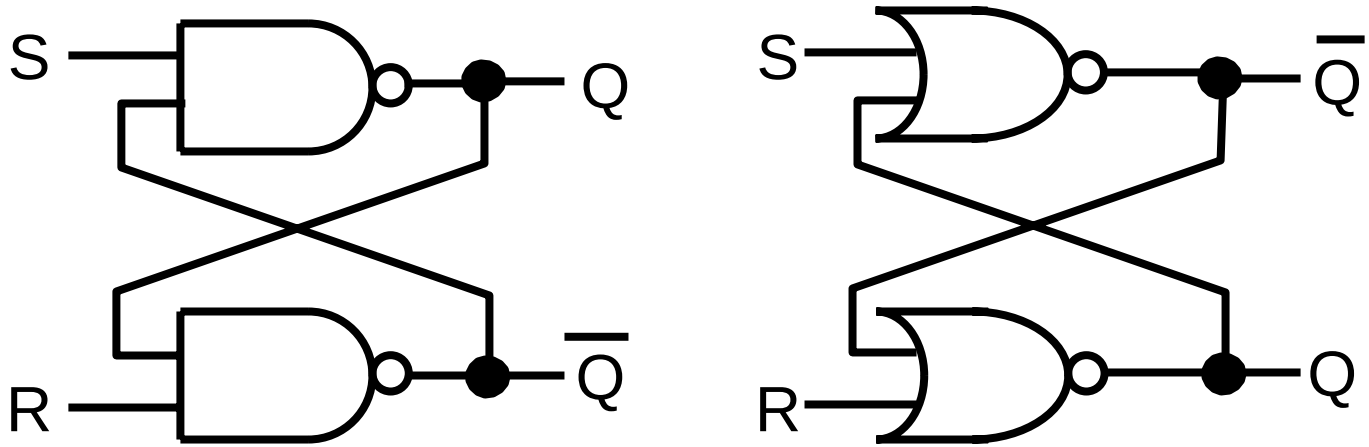
Gain should be larger than 1 in the transition region

Bistable Elements

- **Bistable elements have two stable states or operation modes**
- **Cross-coupled inverters are the most basic bistable element**
  - Circuit forms the basis of latches and SRAM memory
  - Stable points on the VTC are those with the lowest energy
  - Points with high energy are unstable, perturbations are amplified

Set-Reset (SR) Latch

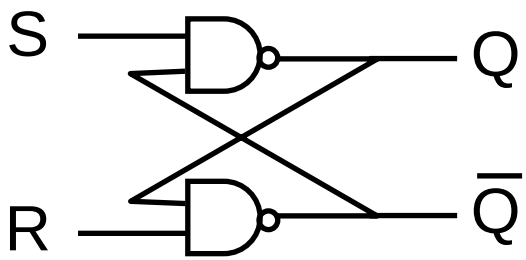
- **Change inverters to NAND or NOR gates, with second inputs = S(set) and R(reset)**



- **Allows control of the state of the bistable element**
- **One input state is not allowed**
- **Gating S and R with the clock prevents the latch from responding except during one phase of the clock cycle**

SR Latch

- **Sequential circuits: circuits which “store state”: circuits with memory elements**
- **Latches: store previous output value for certain input combinations**
- **SR latch (NAND-based):**



not allowed

memory

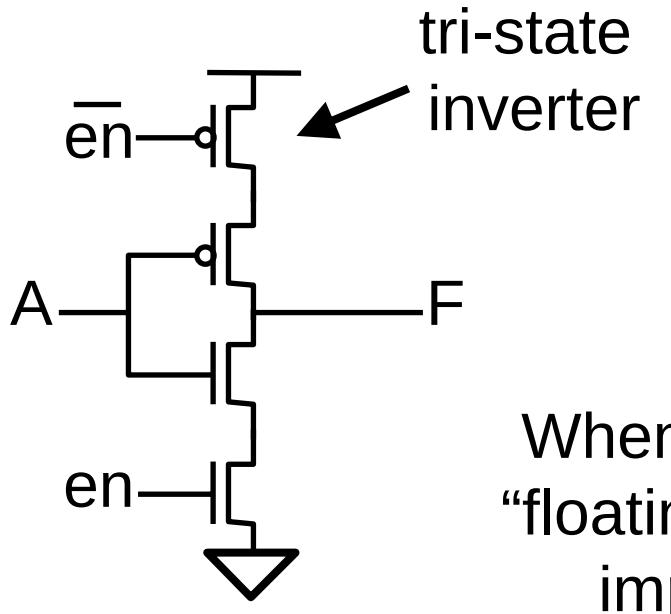
S	R	Q_{next}	$\overline{Q_{next}}$
0	0	1	1
0	1	1	0
1	0	0	1
1	1	Q	$\overline{Q}$

Other Latches

- **Clocked SR latch**
  - Adds clock input. Latch output can only be set/reset when clk=1 (or clk=0)
- **Other latch types:**
  - JK latch: Removes “not allowed” state – e.g., toggles when inputs are both 1
  - T latch: Toggles when T input = 1
  - D latch: Output = D input

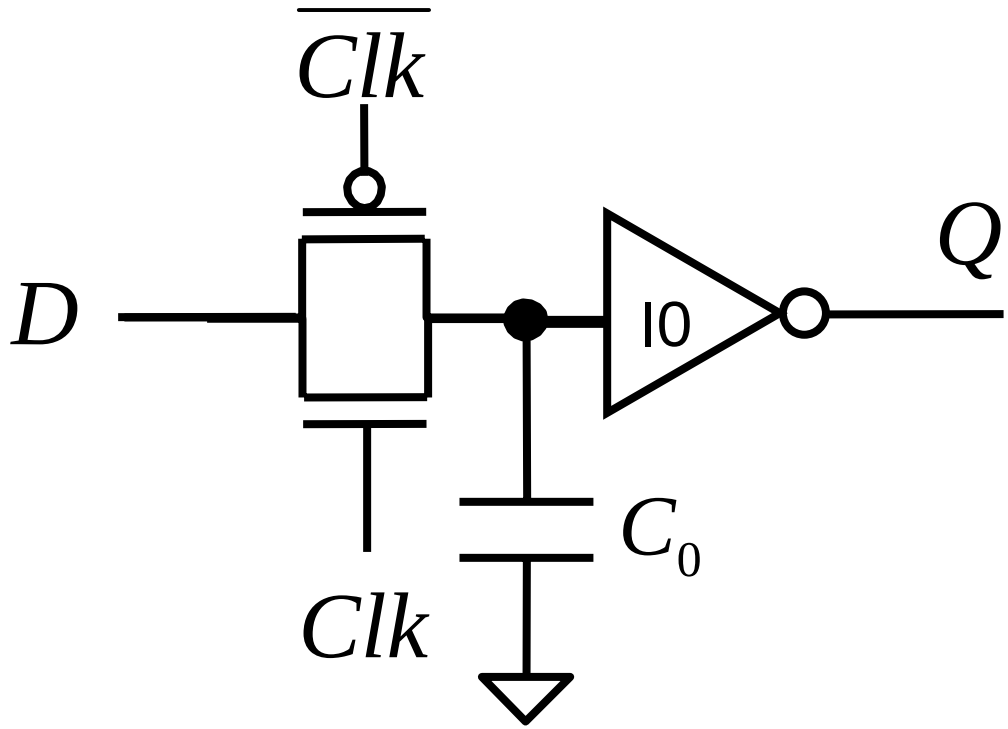
Latch Circuits

- **Many methods for implementing latches**
  - Standard CMOS gates (cross-coupled NAND, etc)
  - Transmission gates
  - Tri-state inverters



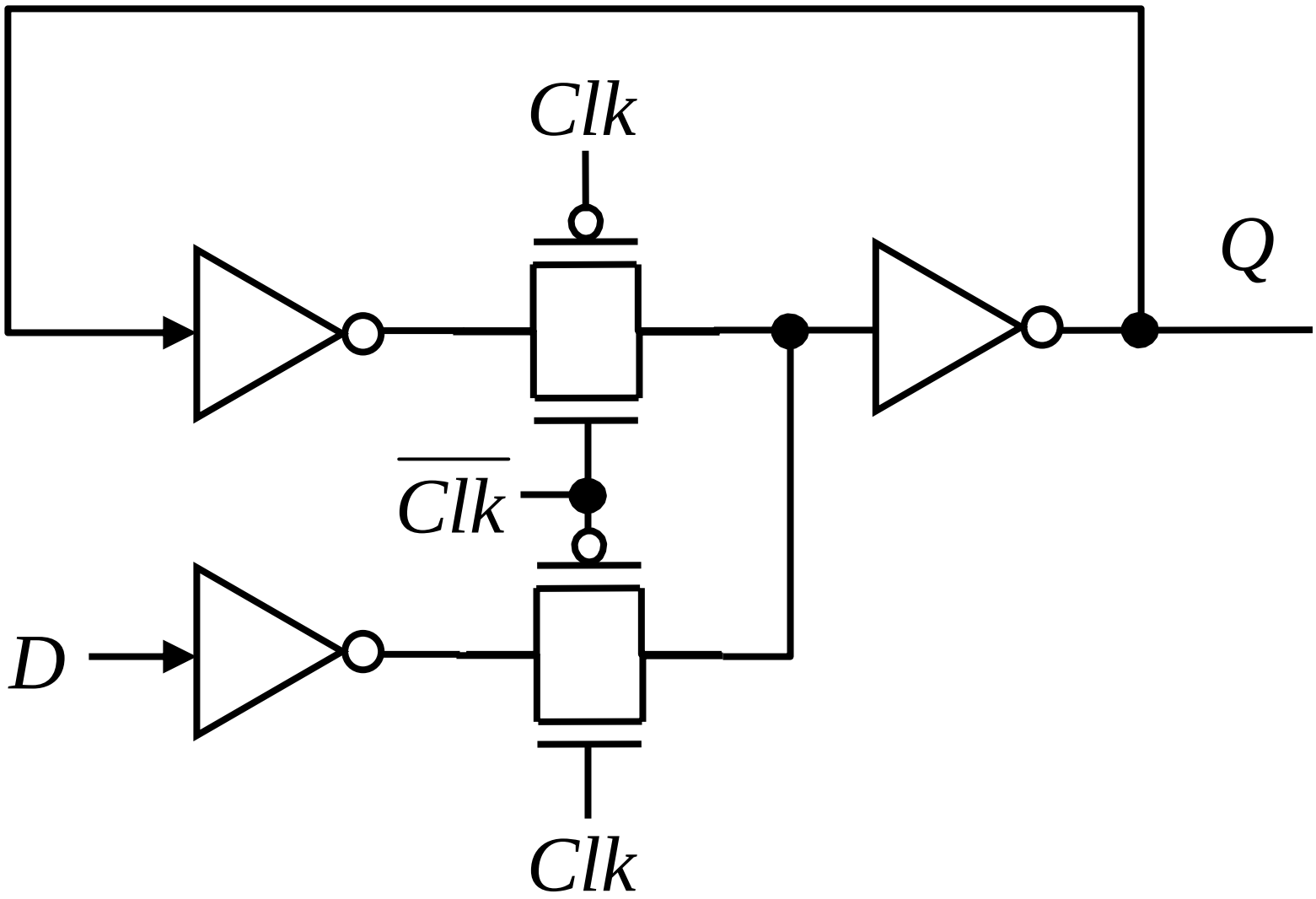
When en=0, F is "floating", i.e. high impedance

Positive Dynamic Transmission Gate Latch

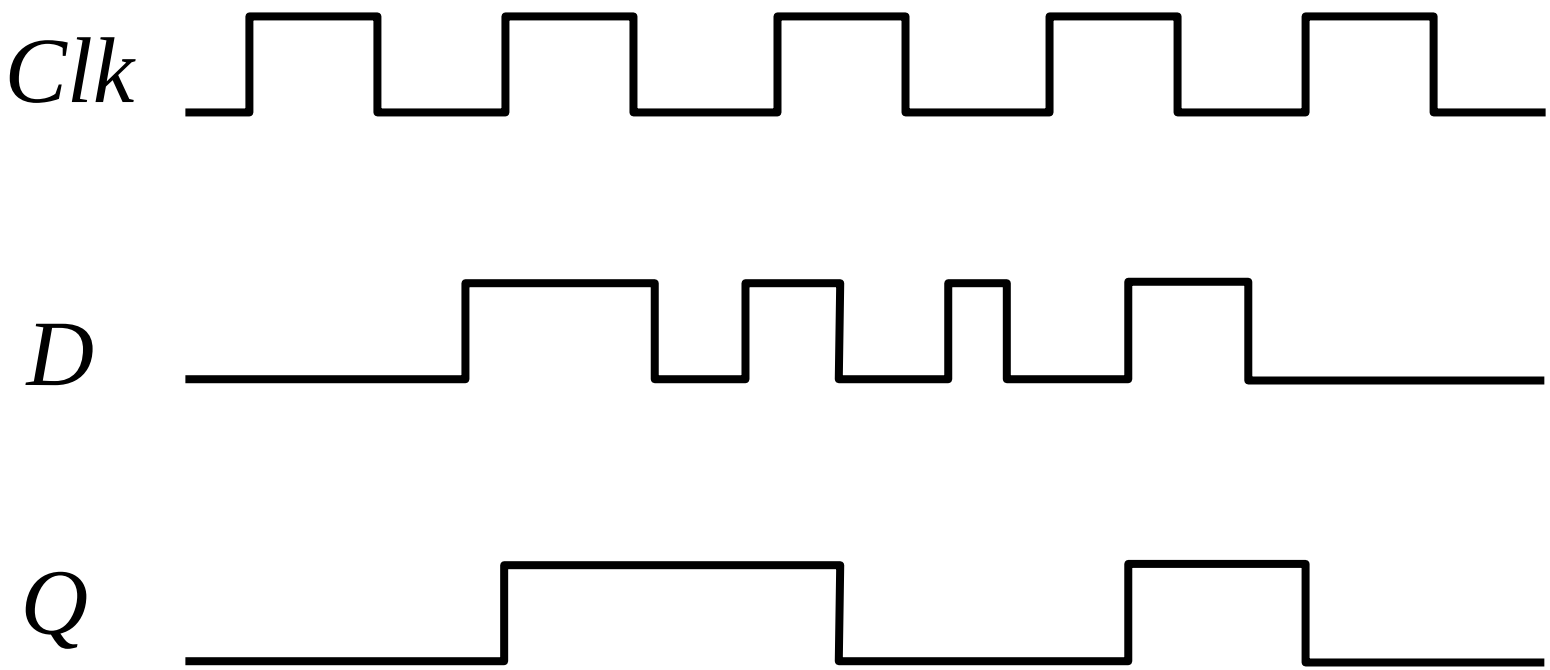


- **No feedback devices**
- **Data stored on input capacitance of inverter I0**
- **Dynamic logic issues apply: leakage, capacitive coupling, charge sharing**

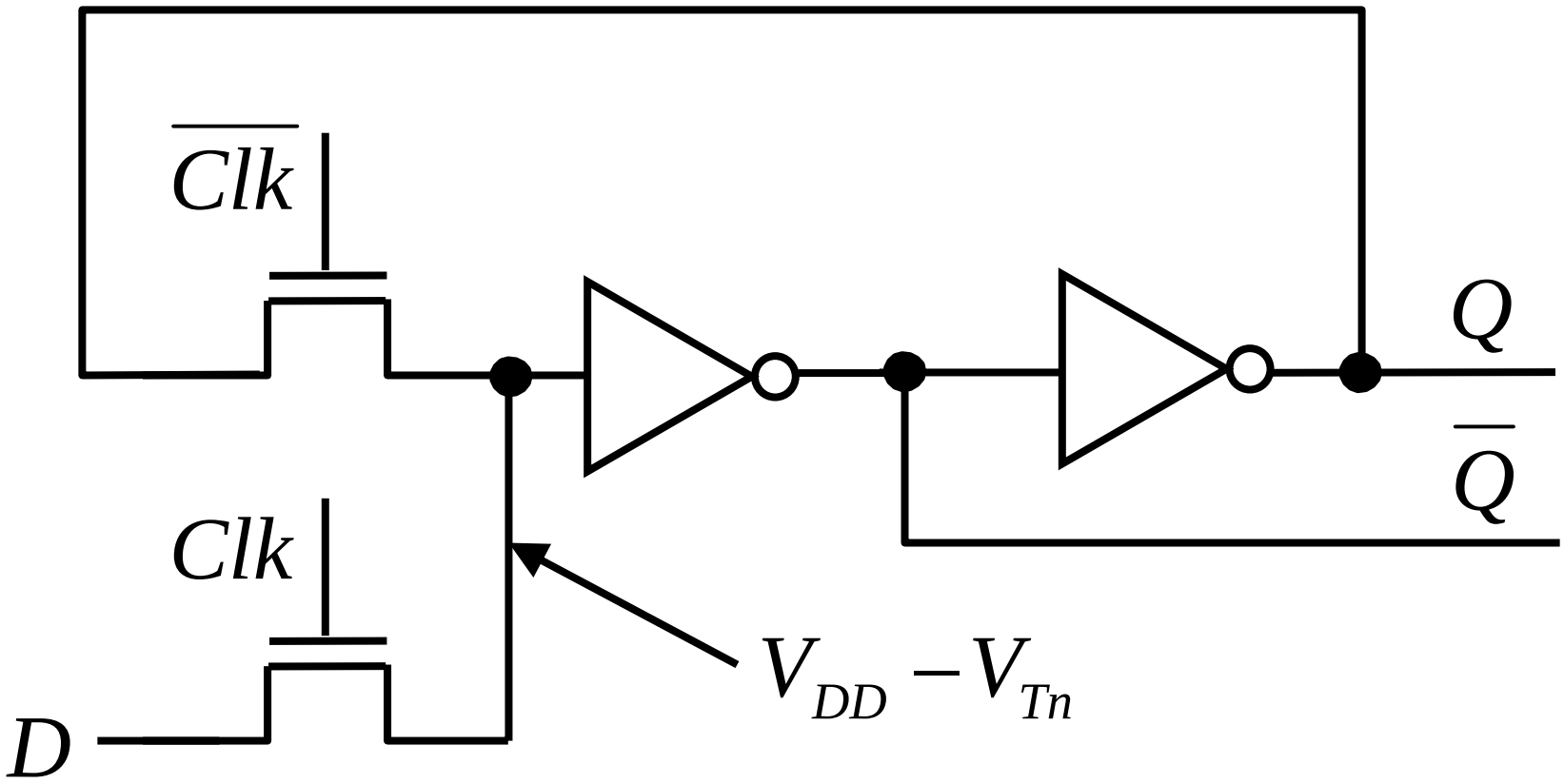
Transmission Gate Positive Static Latch



Positive Static Latch Timing Diagram



NMOS Pass Gate Positive Static Latch



- **Fewer devices, less area, lower clock load**
- **Threshold drop on internal nodes implies more static power, less noise margin**

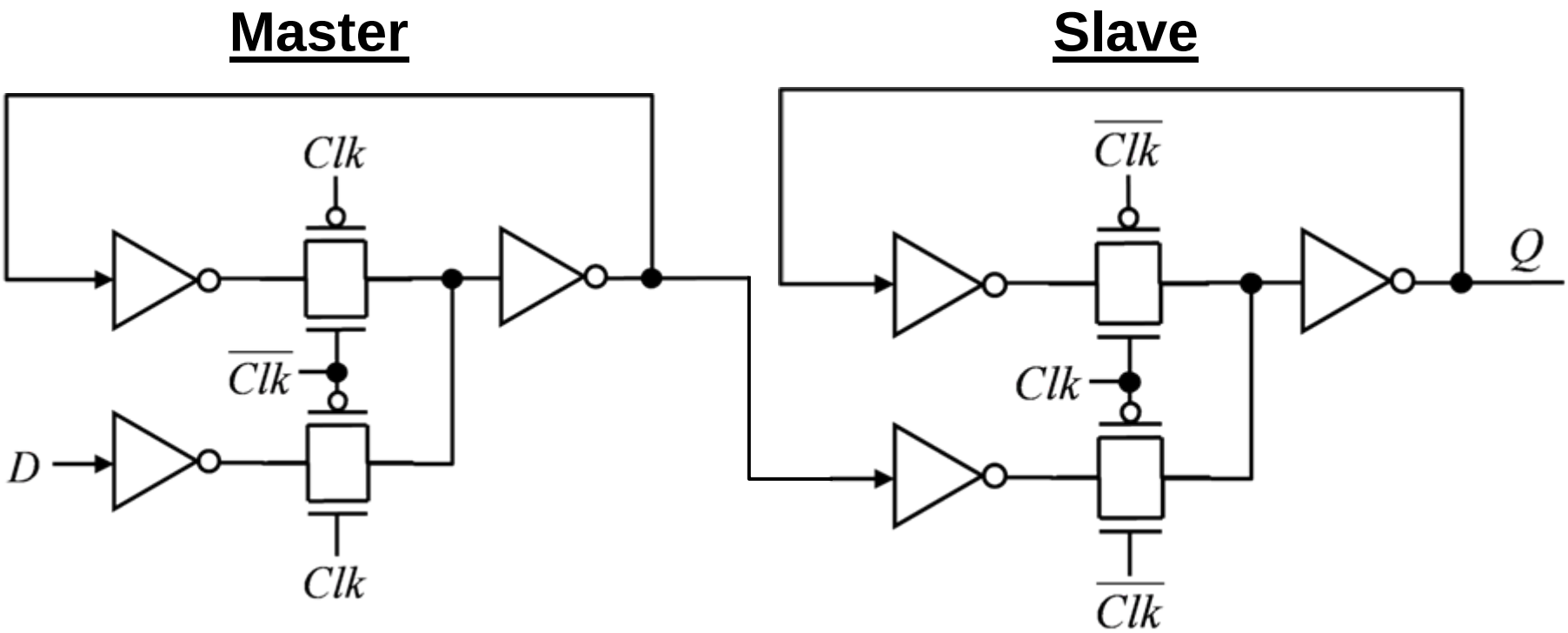
Master-Slave Flip-Flop

- **By cascading two level-sensitive latches, one type of edge triggered flip-flop is created**
- **JK latch can be used for first stage so that no input combinations are invalid**
- **SR latch is then used for the second stage because inputs cannot be invalid**
- **Rather than using logic gate-based latches, can cascade latches such as above (e.g., transmission gate dynamic or static latches)**

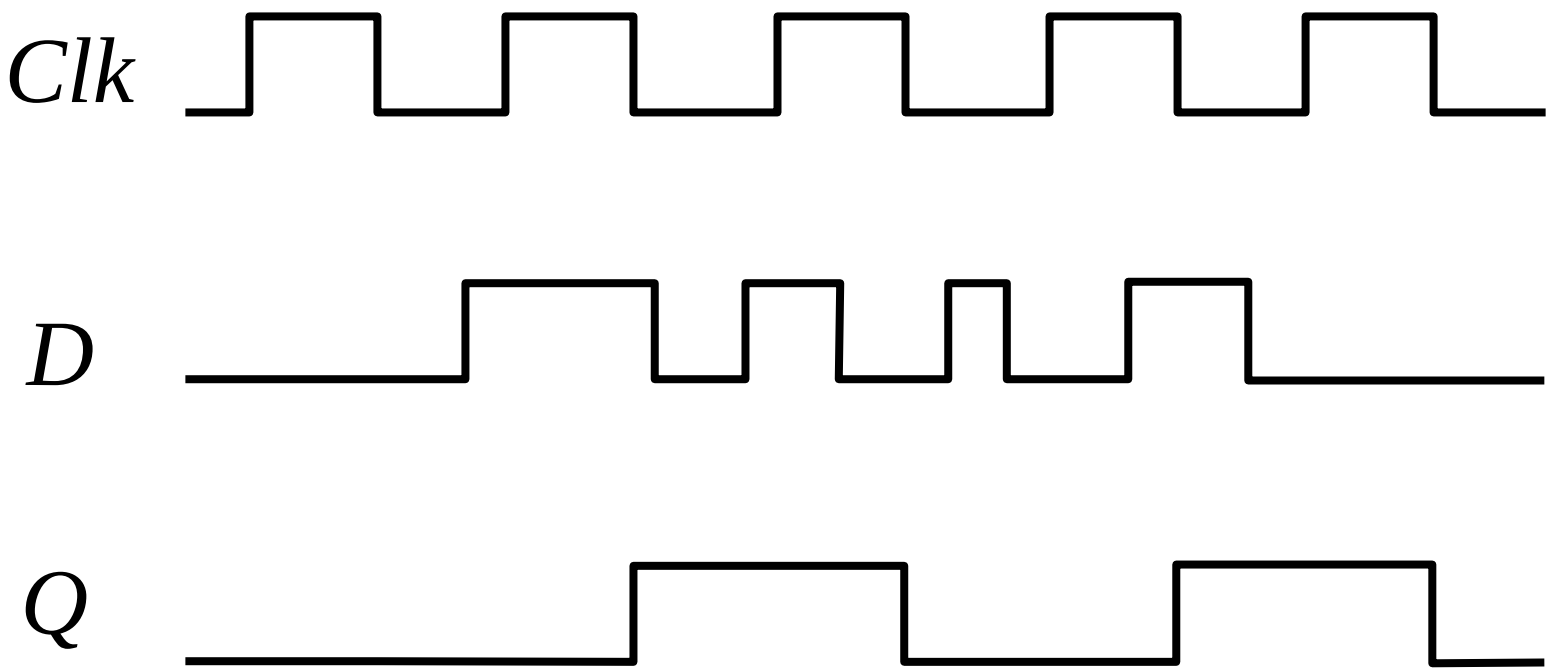
Edge-Triggered Flip-Flops

- **Types of latches/flip-flops:**
  - *Level-sensitive*: output is set when clock is a certain level (0 or 1)
  - *Edge-triggered*: output can only be set on a clock edge (rising or falling)
- **Advantages of edge-triggered flip-flops:**
  - Data only needs to be stable at clock edge
  - Reduces *race conditions*: potential errors where an input data change travels through multiple latches during their “transparent” phase

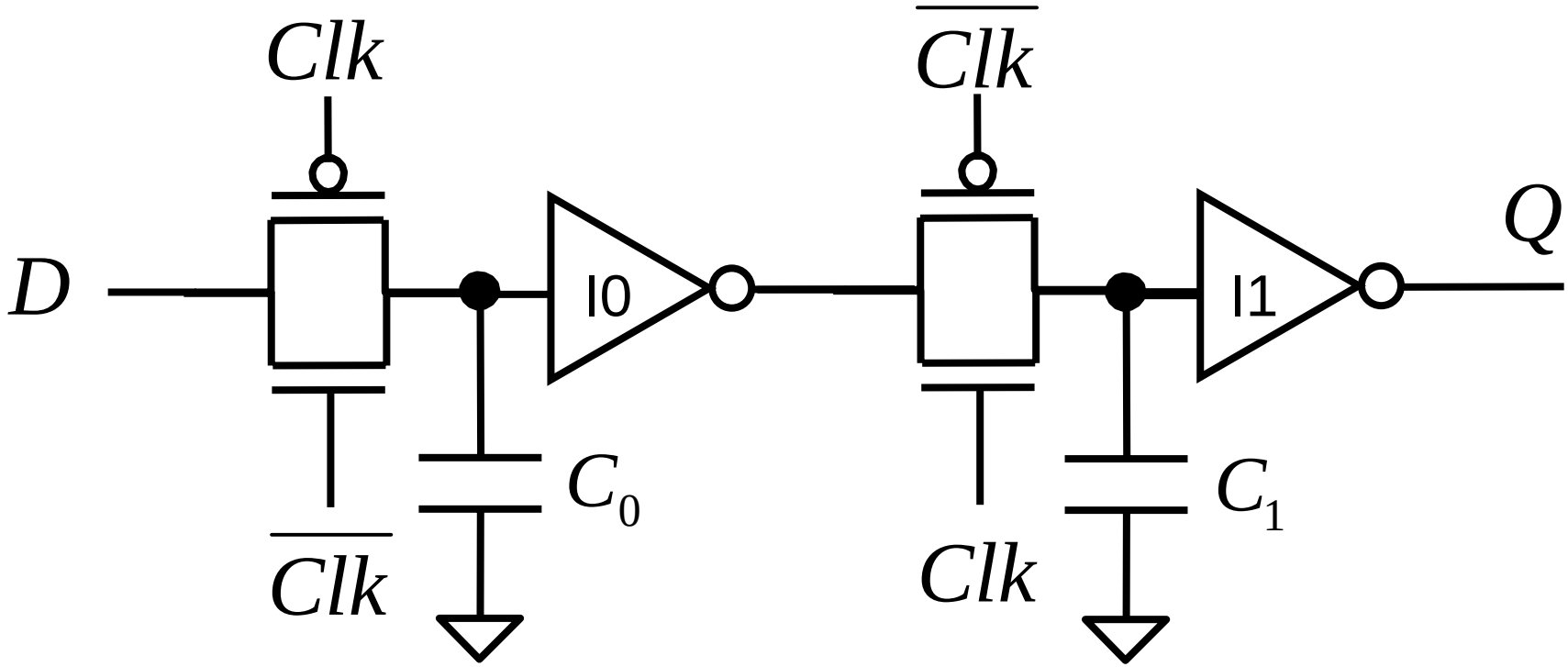
“Safest” Edge-Triggered Flip-Flop



Edge-Triggered Flip-Flop Timing Diagram



Dynamic Positive Edge-Triggered FF

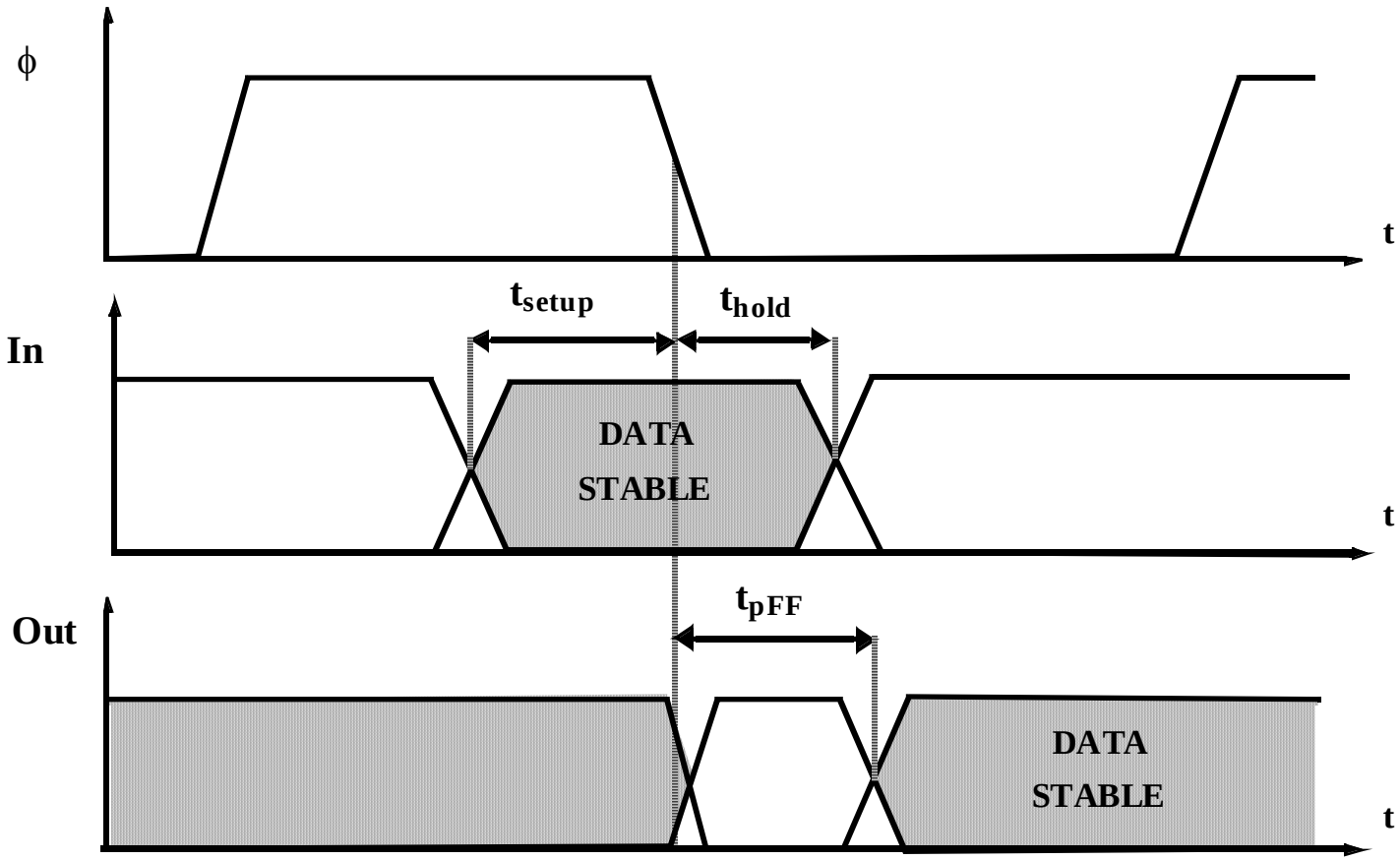


- **No feedback devices**
- **Data stored on input capacitances of inverters I0 and I1**
- **Dynamic logic issues apply: leakage, capacitive coupling, charge sharing**

Clocked Circuit Timing

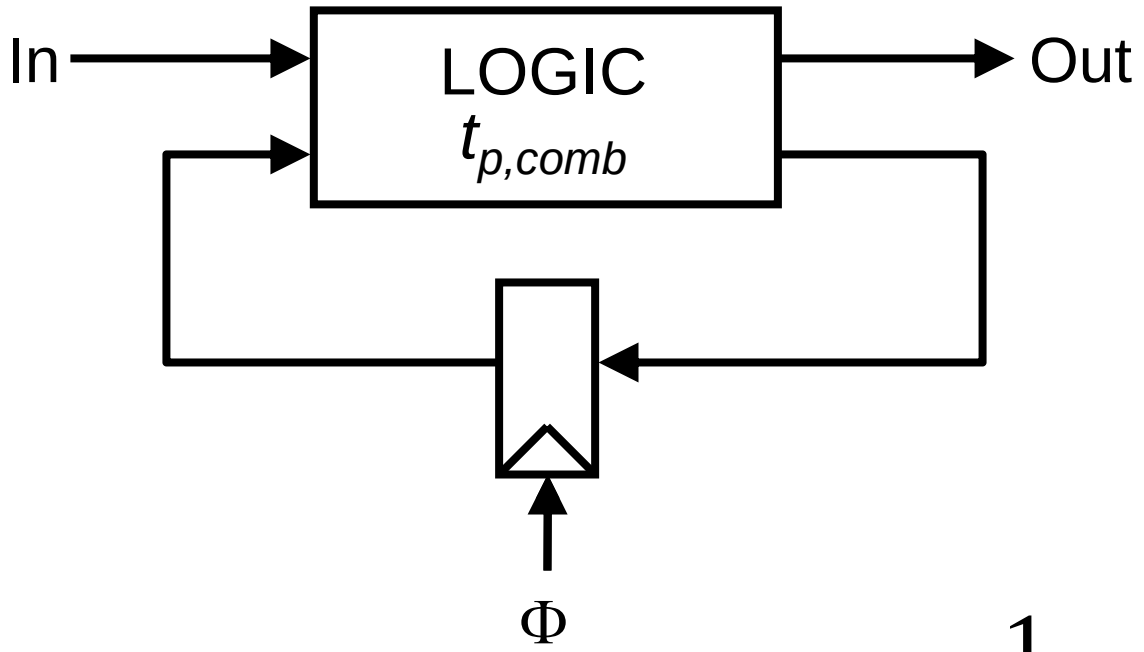
- **Timing definitions:**
  - Clock-to-Q or Propagation Delay (t_{clkQ}): delay of flip-flop from clock edge to output Q
  - Setup Time (t_{setup}): amount of time *before* clock edge that data has to be stable. If data arrives after this time, it will not be latched correctly.
  - Hold Time (t_{hold}): amount of time *after* clock edge that data has to be stable.
- **It is possible to trade off setup and hold time with flip-flop circuit design**
  - Modify data and clock timing relationship by delaying one of the two signals

Flip-Flop: Timing Definitions



From Digital Integrated Circuits – Jan Rabaey Notes

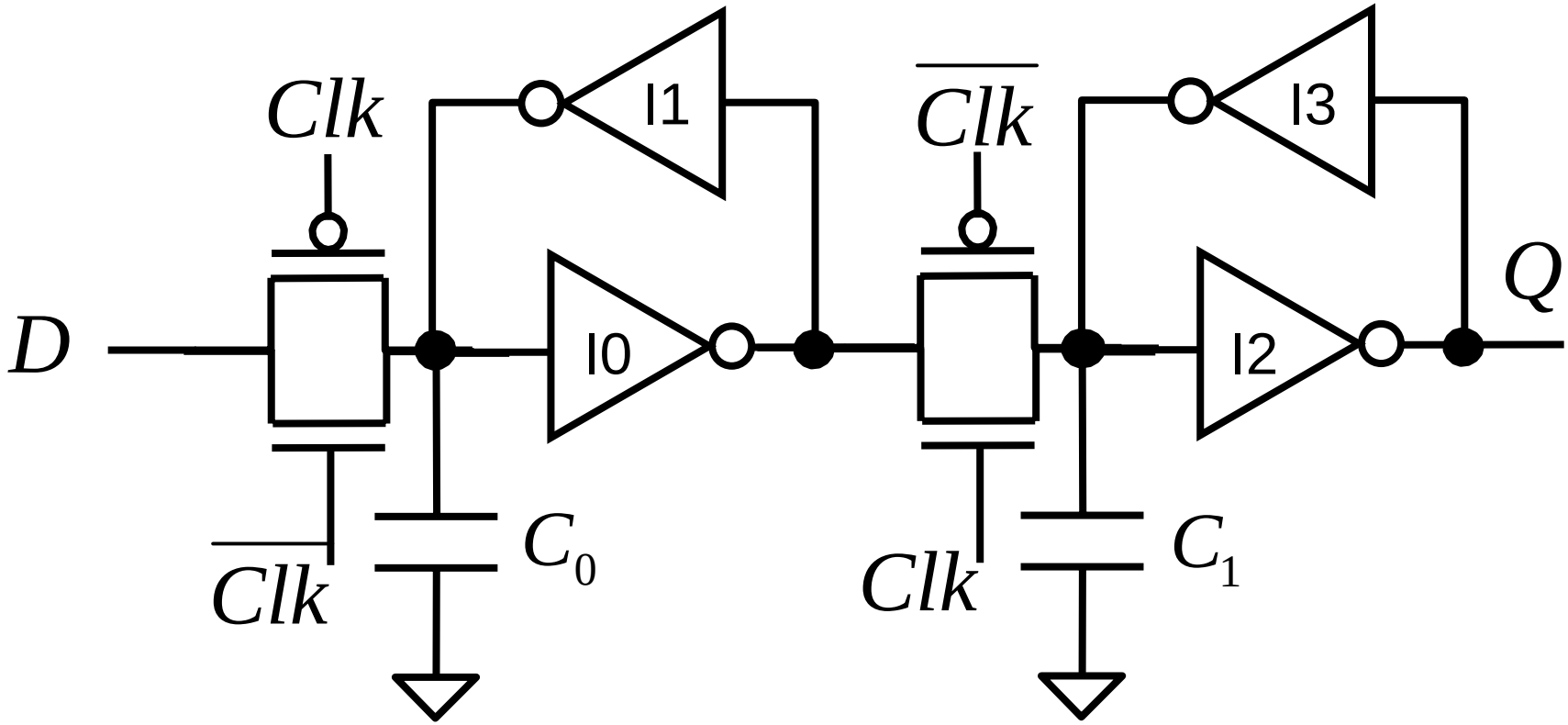
Maximum Clock Frequency



$$t_{pFF} + t_{p,comb} + t_{setup} < T = \frac{1}{f}$$

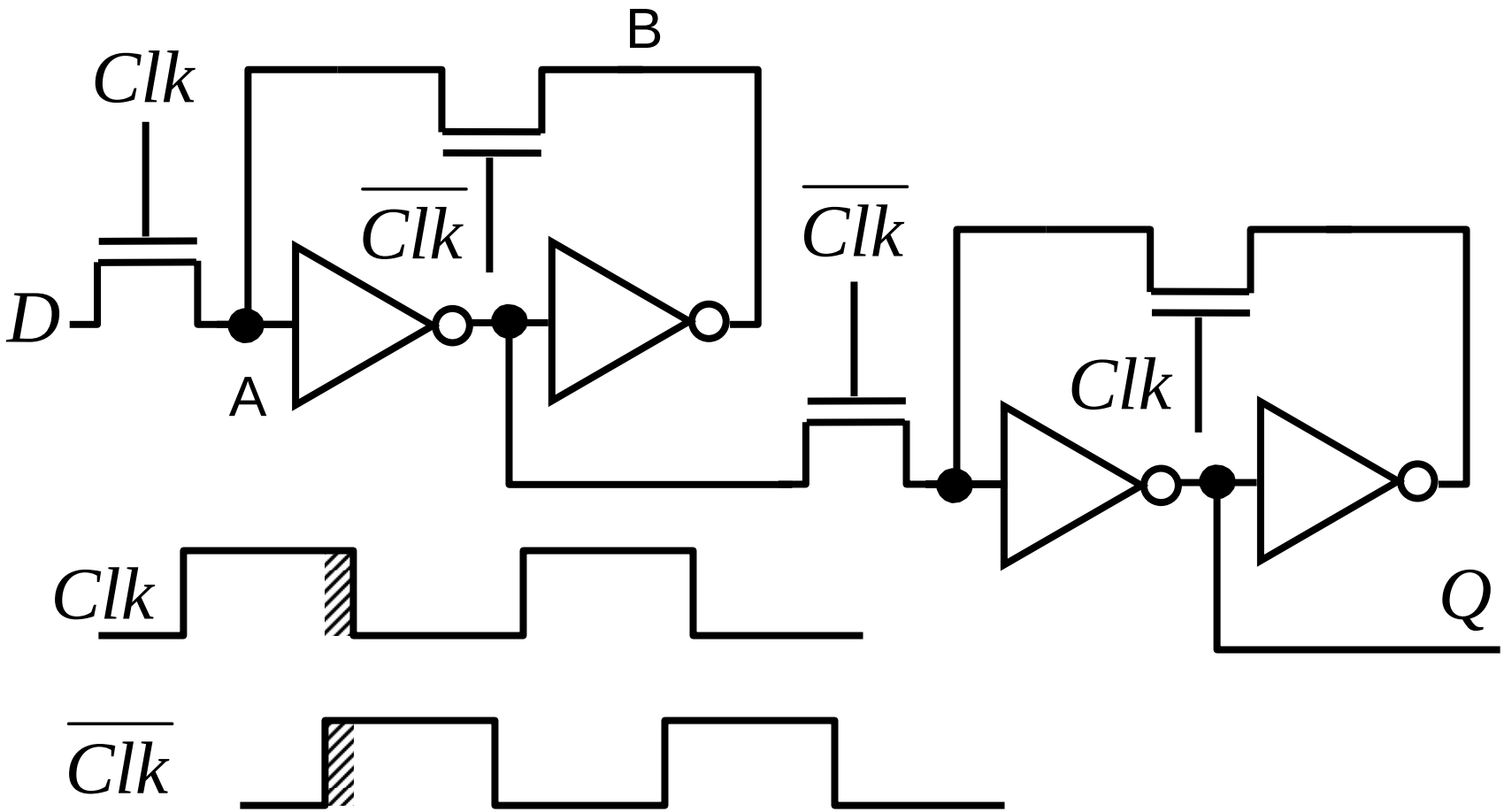
- **Signals must propagate out of flip-flop, through combinational logic, and be stable before next clock edge (clock period = T, clock frequency = f)**

Staticized Dynamic Positive Edge-Triggered FF



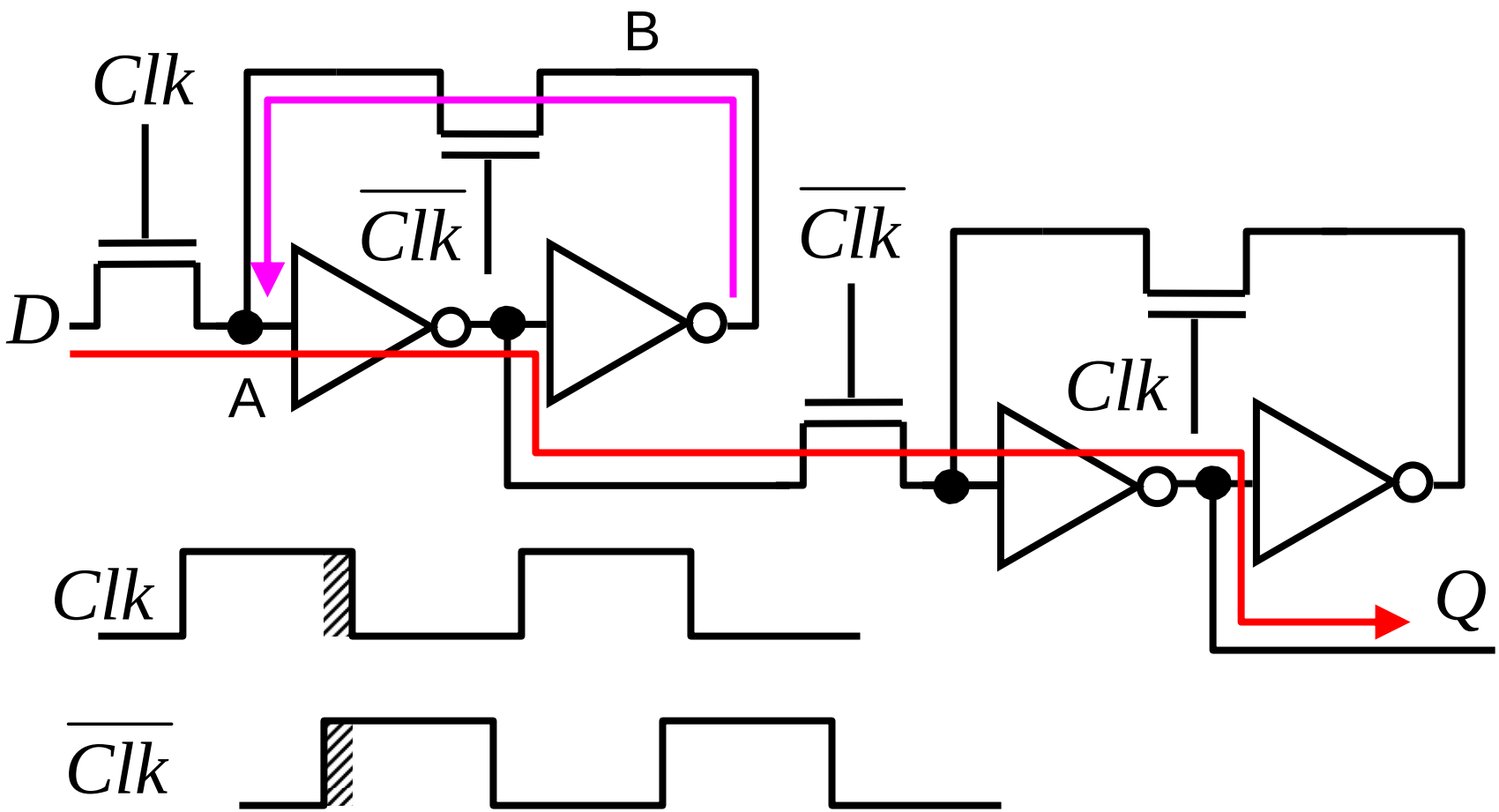
- **Use weak feedback inverters to enhance robustness**
- **Returns to reduced clock load static flip-flop with same sizing issues**

Clock Overlap Failures



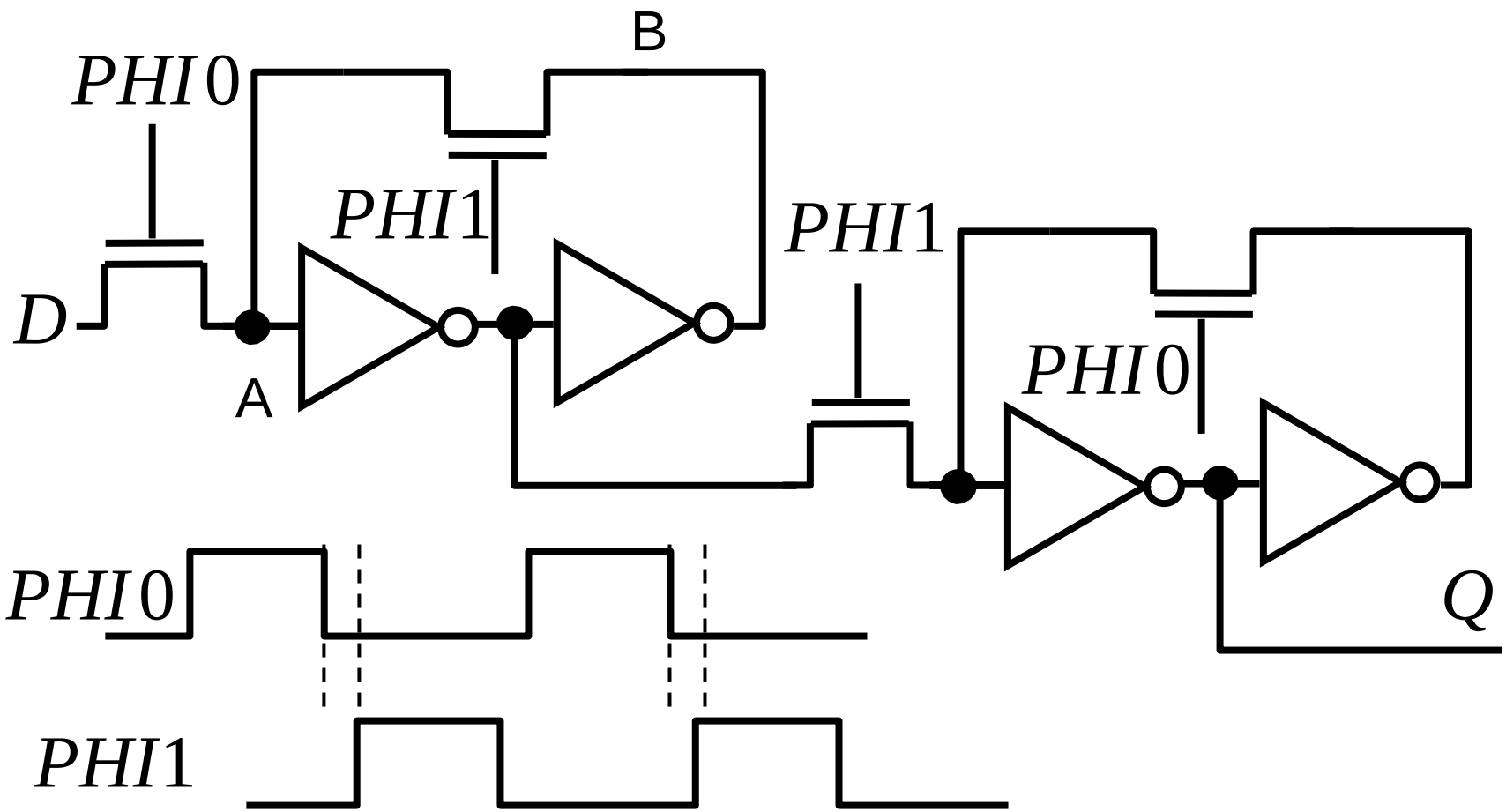
1. **Both high simultaneously, race condition from *D* to *Q***
2. **Node A can be driven simultaneously by *D* and B**

Race Through and Feedback Paths



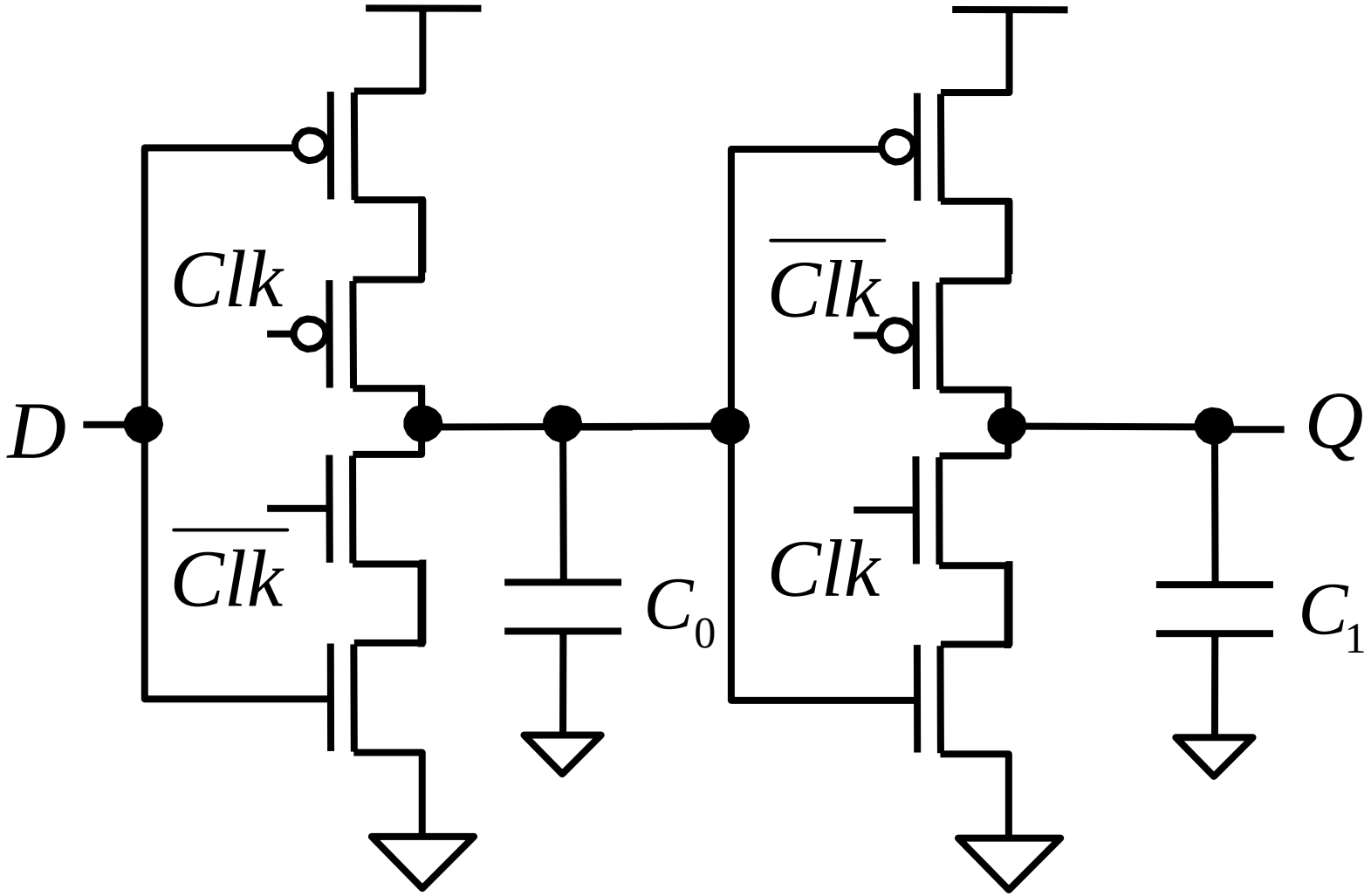
1. **Both high simultaneously, race condition from *D* to *Q***
2. **Node A can be driven simultaneously by *D* and B**

Nonoverlapping Clocks Methodology



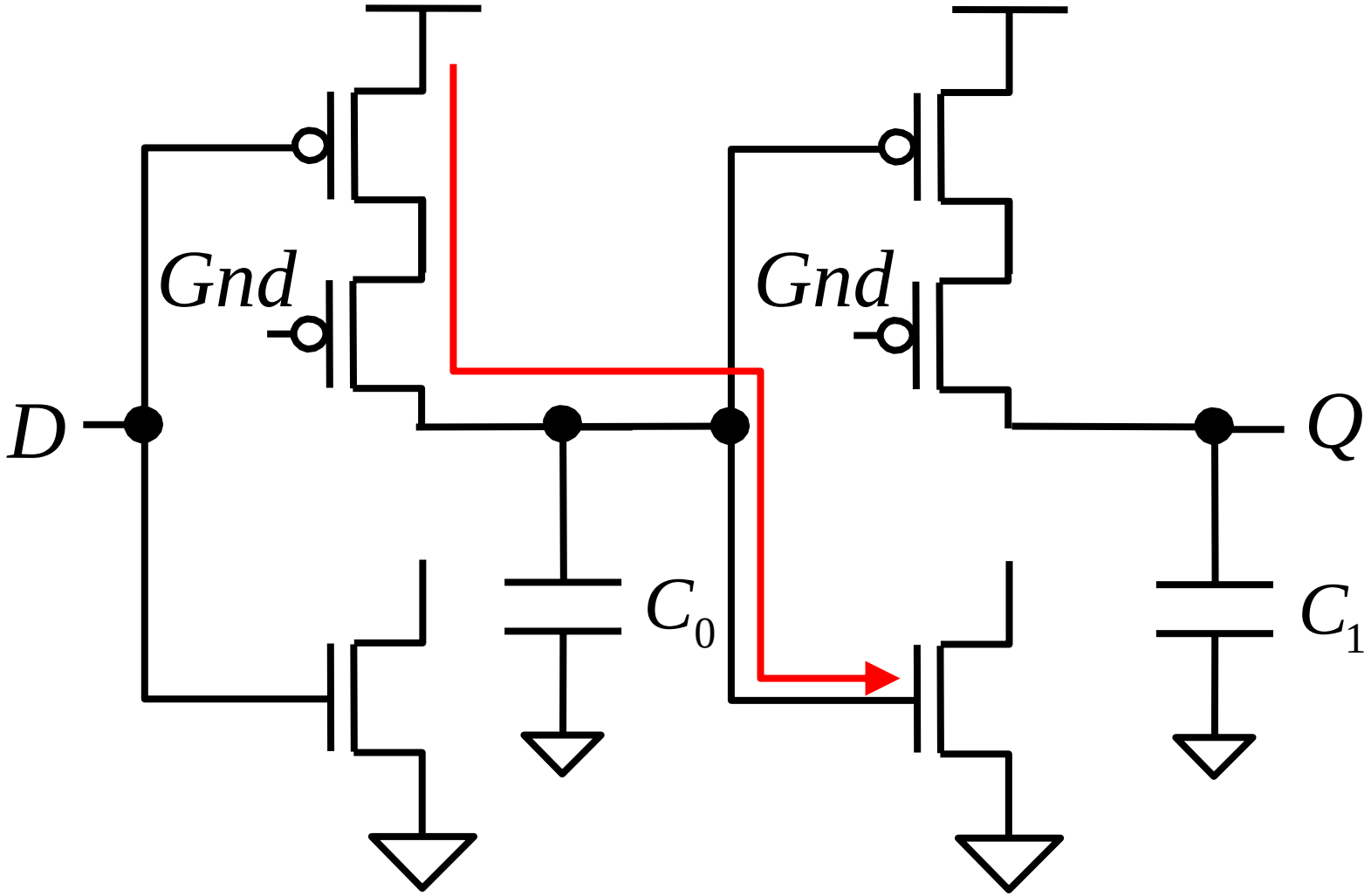
- **Guarantee nonoverlap period long enough**
- **Note: internal nodes left high Z during nonoverlap**

C²MOS Edge Triggered Flip-Flop



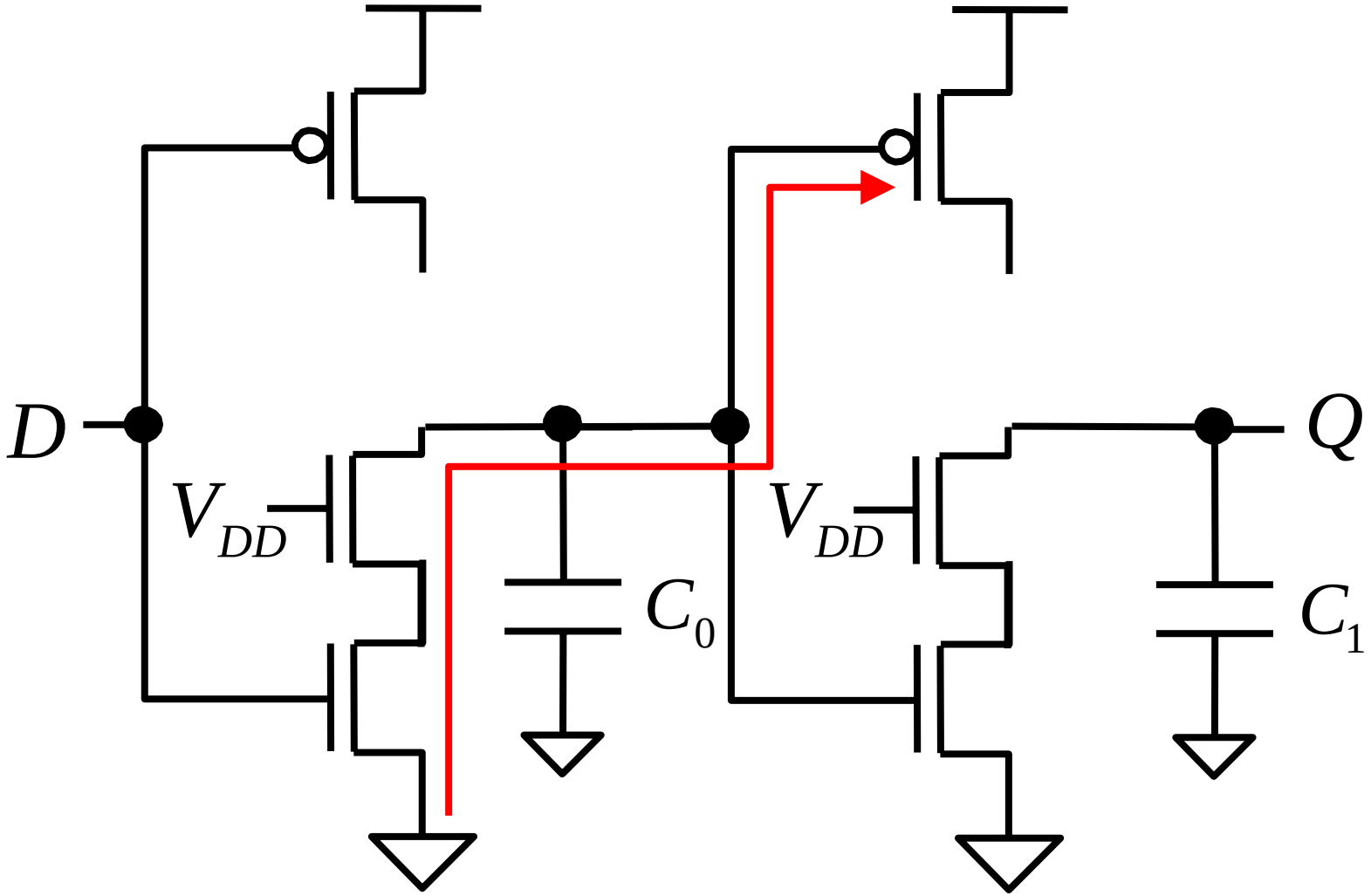
- **Tristate inverters eliminate clock overlap race condition**

Zero-Zero Overlap Condition



- **Both phases low simultaneously enables opposite nets**

High-High Overlap Condition

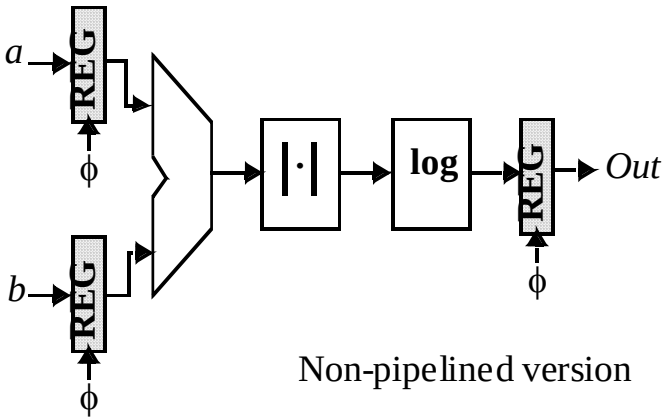


- **Both phases high simultaneously enables opposite nets**

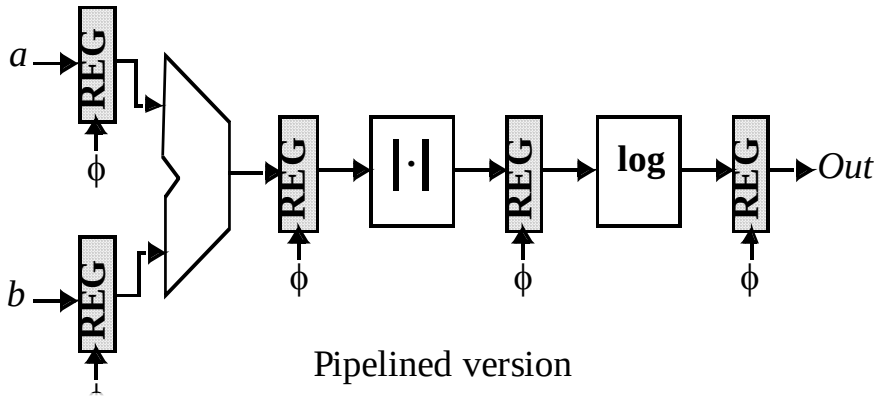
C^2MOS Design

- **Clock overlap problems eliminated as long as rise and fall times remain fast**
  - Slow rise / fall times imply pullup and pulldown nets on simultaneously resulting in potential errors, static power
- **Dynamic flip-flop style leaves output high Z**
  - Must take care when using since output wire could be exposed to many more noise sources than internal nodes
- **Mix and match styles by using C^2MOS as master and other types of latch as slave**
- **Clock load small, but potentially larger than transmission gate dynamic latches due to PMOS sizing**

Pipelining



Non-pipelined version



Pipelined version

Clock Period	Adder	Absolute Value	Logarithm
1	$a_1 + b_1$		
2	$a_2 + b_2$	$\lvert a_1 + b_1 \rvert$	
3	$a_3 + b_3$	$\lvert a_2 + b_2 \rvert$	$\log(\lvert a_1 + b_1 \rvert)$
4	$a_4 + b_4$	$\lvert a_3 + b_3 \rvert$	$\log(\lvert a_2 + b_2 \rvert)$
5	$a_5 + b_5$	$\lvert a_4 + b_4 \rvert$	$\log(\lvert a_3 + b_3 \rvert)$

From Digital Integrated Circuits – Jan Rabaey Notes

Next Topic: Latchup and Layout Guidelines

- **Latchup**
- **Minimizing area**
- **Managing complexity**