DEPARTMENT OF COMPUTER SYSTEM ENGINEERING

Digital Integrated Circuits - ENCS333

Dr. Khader Mohammad

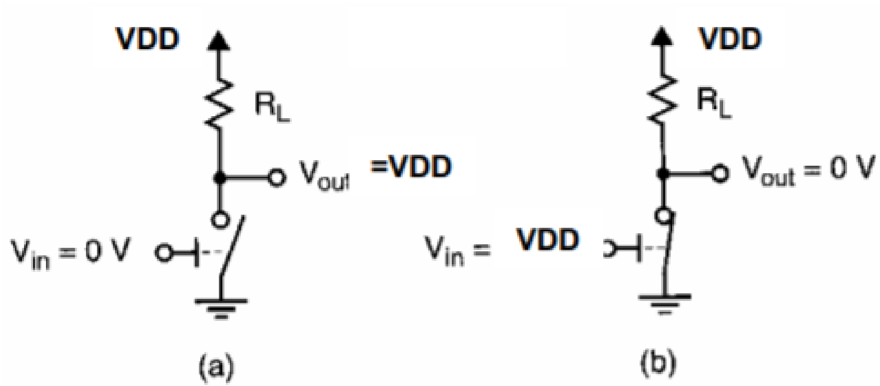
Lecture #10 Dynamic and Sequential circuit

Integrated-Circuit Devices and Modeling

nMOS Logic Gates

- We will look at nMOS logic first, simpler than CMOS
- nMOS Logic (no pMOS transistors)
  - assume a resistive load to VDD
  - nMOS switches pull output low based on inputs

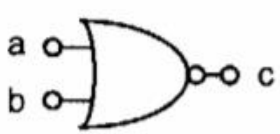
nMOS Inverter



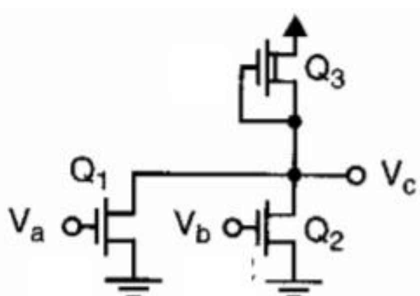
(a) nMOS is off
→ output is high (1)

(b) nMOS is on
→ output is low (0)

nMOS NOR

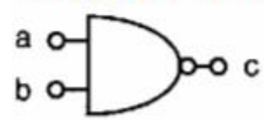


$c = \overline{a+b}$

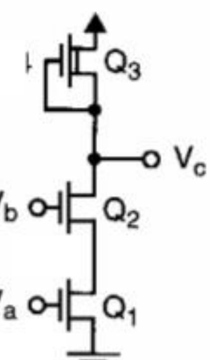


- parallel switches = OR function
- nMOS pulls low (NOTs the output)

nMOS NAND



$c = \overline{ab}$



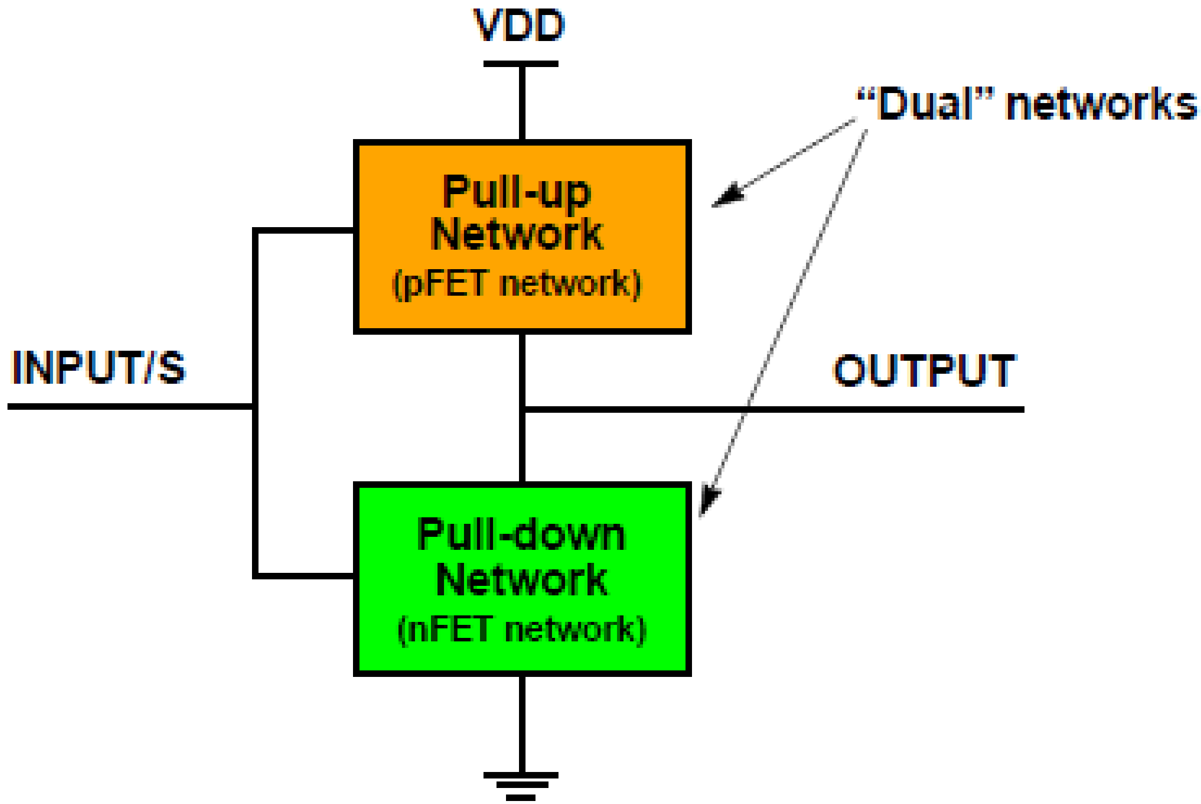
- series switches = AND function
- nMOS pulls low (NOTs the output)

DEPARTMENT OF COMPUTER SYSTEM ENGINEERING

Digital Integrated Circuits - ENCS333

Dynamic Gate

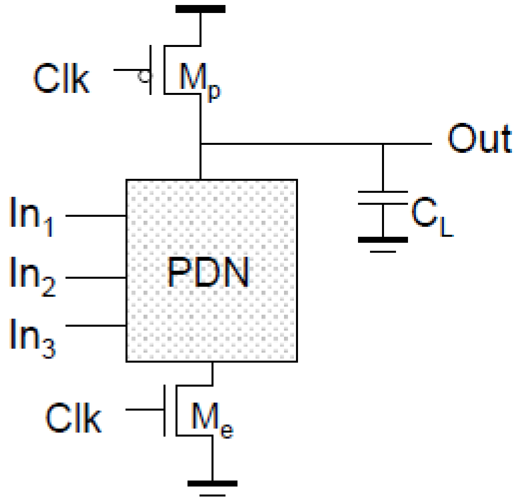
Complementary Design

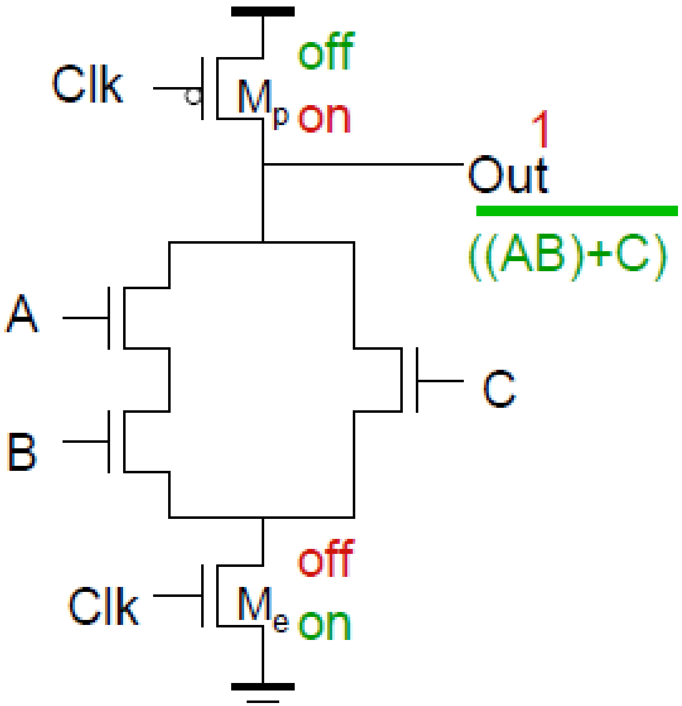


- PMOS "Pull-Up" Network (PUN)
- NMOS "Pull-Down" Network (PDN)

Dynamic Gate

Two phase operation
Precharge (CLK = 0)
Evaluate (CLK = 1)





Conditions on Output

- Once the output of a dynamic gate is discharged, it cannot be charged again until the next precharge operation.
- Inputs to the gate can make at most one transition during evaluation.

- Output can be in the high impedance state during and after evaluation (PDN off), state is stored on C_L

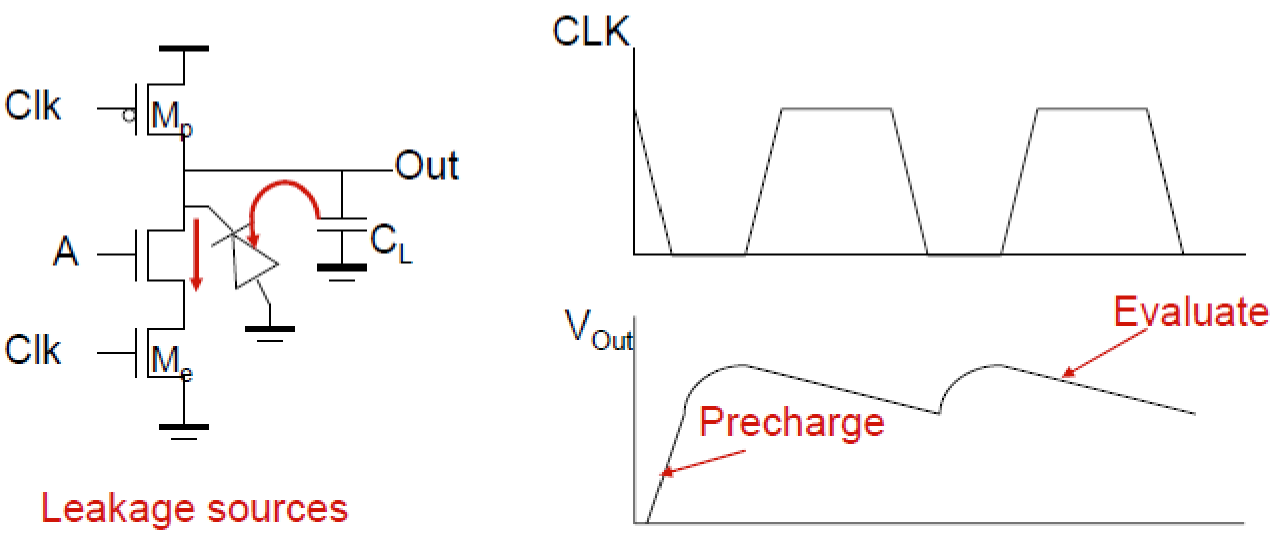
Properties of Dynamic Gates

- Logic function is implemented by the PDN only
  - number of transistors is N + 2 (versus 2N for static complementary CMOS)
- Full swing outputs (V_{OL} = GND and $V_{OH} = V_{DD}$)
- Non-ratioed - sizing of the devices does not affect the logic levels
- Faster switching speeds
  - reduced load capacitance due to lower input capacitance (C_{in})
  - reduced load capacitance due to smaller output loading (Cout)
  - no I_{sc}, so all the current provided by PDN goes into discharging C_L

Properties of Dynamic Gates

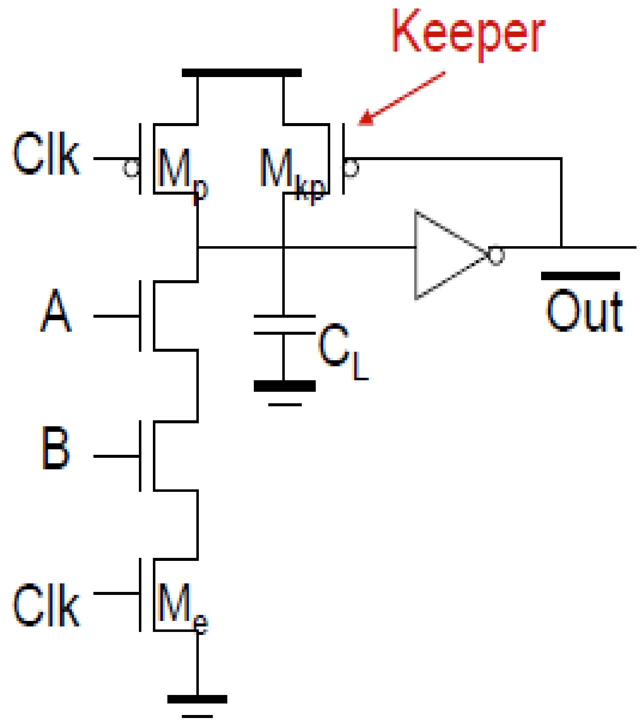
- Overall power dissipation usually higher than static CMOS
  - no static current path ever exists between V_{DD} and GND (including P_{sc})
  - no glitching
  - higher transition probabilities
  - extra load on Clk
- PDN starts to work as soon as the input signals exceed V_{Tn}, so V_M, V_{IH} and V_{IL} equal to V_{Tn}
  - low noise margin (NM_L)
- Needs a precharge/evaluate clock

Issues in Dynamic Design 1: Charge Leakage



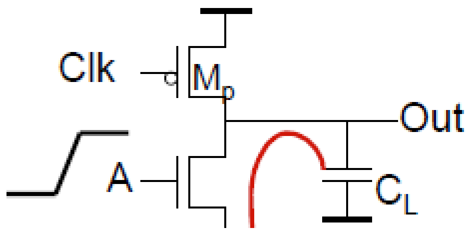
Dominant component is subthreshold current

Solution to Charge Leakage

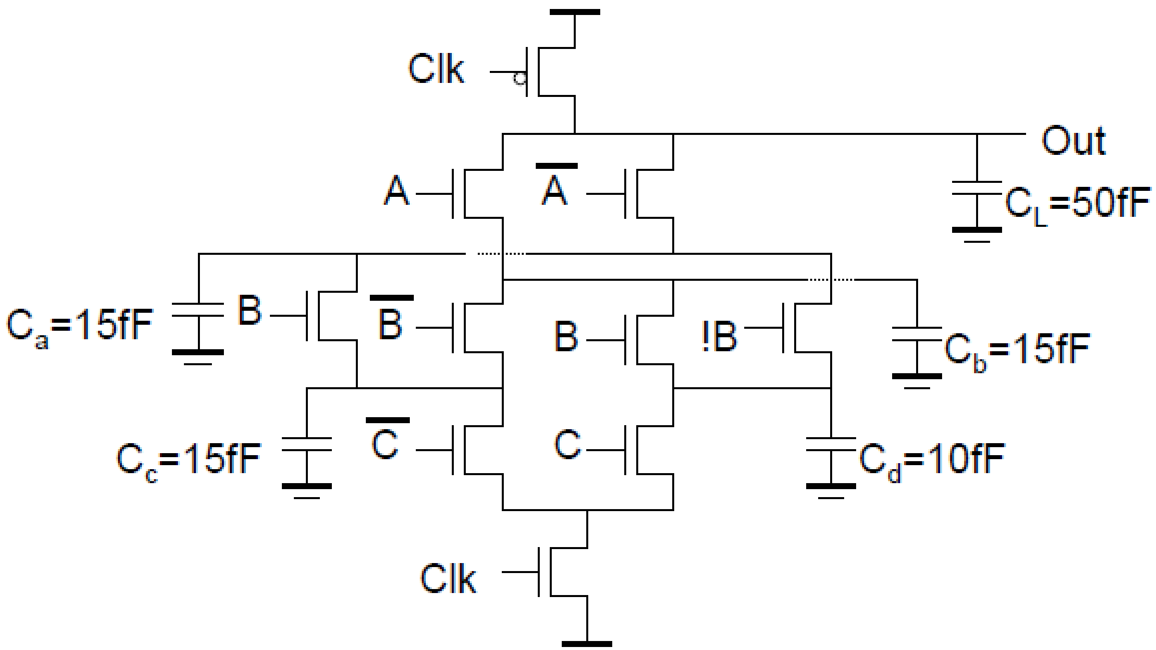


Same approach as level restorer for pass-transistor logic

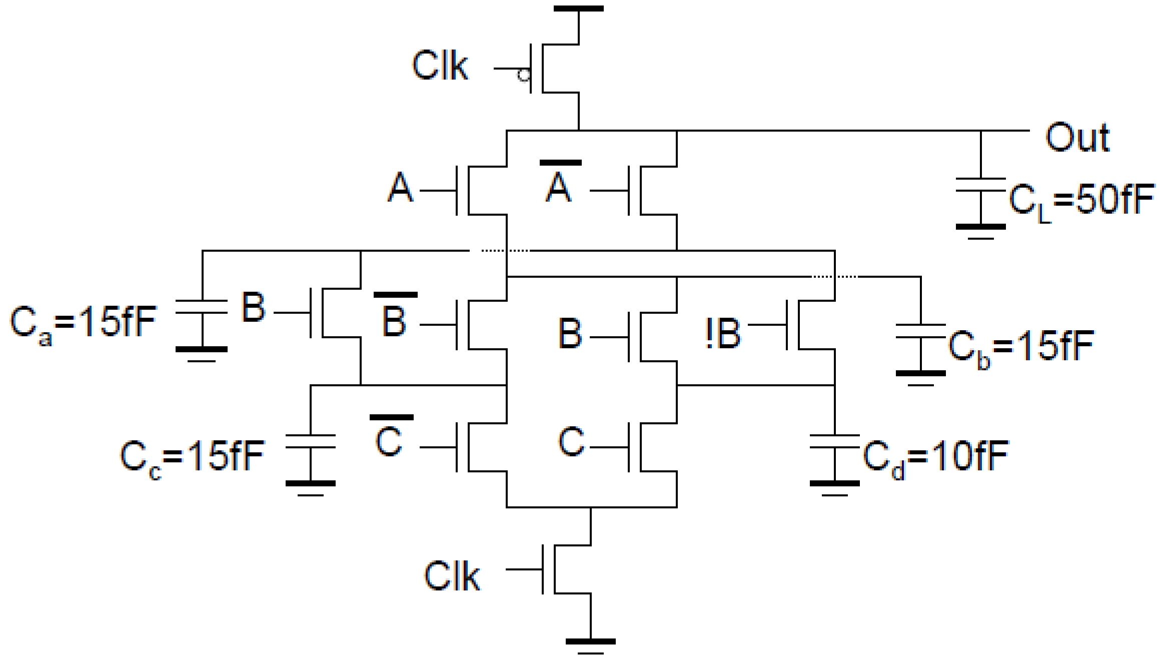
Issues in Dynamic Design 2: Charge Sharing



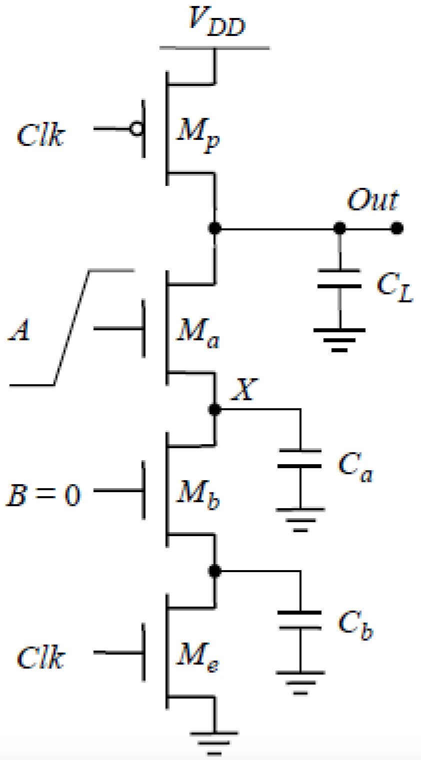
Charge stored originally on C_L is redistributed (shared) over C_L and C_A leading to reduced robustness



Issues in Dynamic Design 2: Charge Sharing



Charge Sharing



case 1) if $\Delta V_{out} < V_{Tn}$

$$C_L V_{DD} = C_L V_{out}(t) + C_a(V_{DD} - V_{Tn}(V_X))$$

or

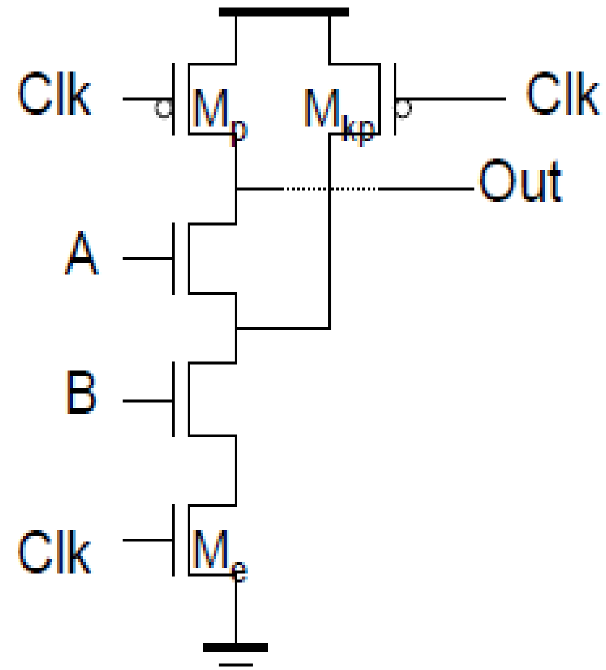
$$\Delta V_{out} = V_{out}(t) - V_{DD} = -\frac{C_a}{C_L}(V_{DD} - V_{Tn}(V_X))$$

case 2) if $\Delta V_{out} > V_{Tn}$

$$\Delta V_{out} = -V_{DD}\left(\frac{C_a}{C_a + C_L}\right)$$

Solution to Charge Redistribution

Precharge internal nodes using a clock-driven transistor
(at the cost of increased area and power)

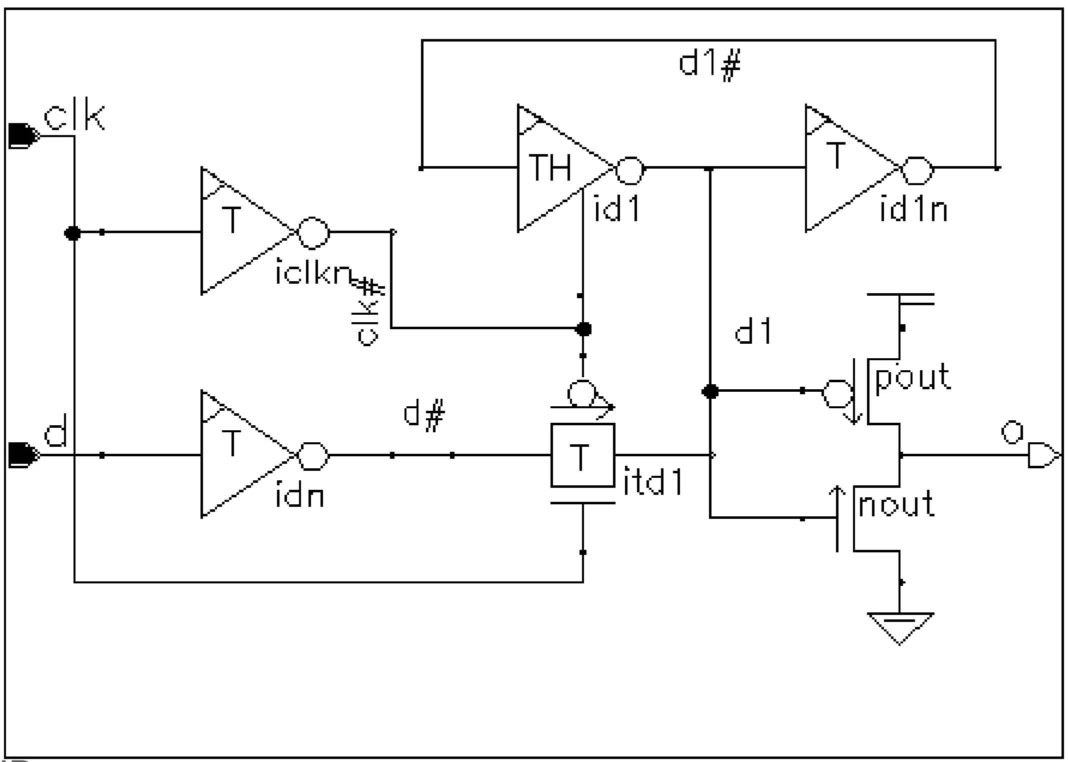


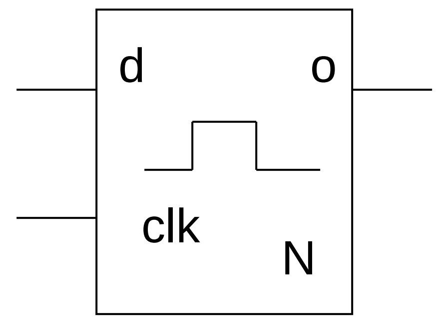
Sequential logic gates; Latches and Flip-Flops

- Basics of sequentials
- Latch and flop details
- Flop based design
- Other types of sequentials
- Understanding the power implications of flops

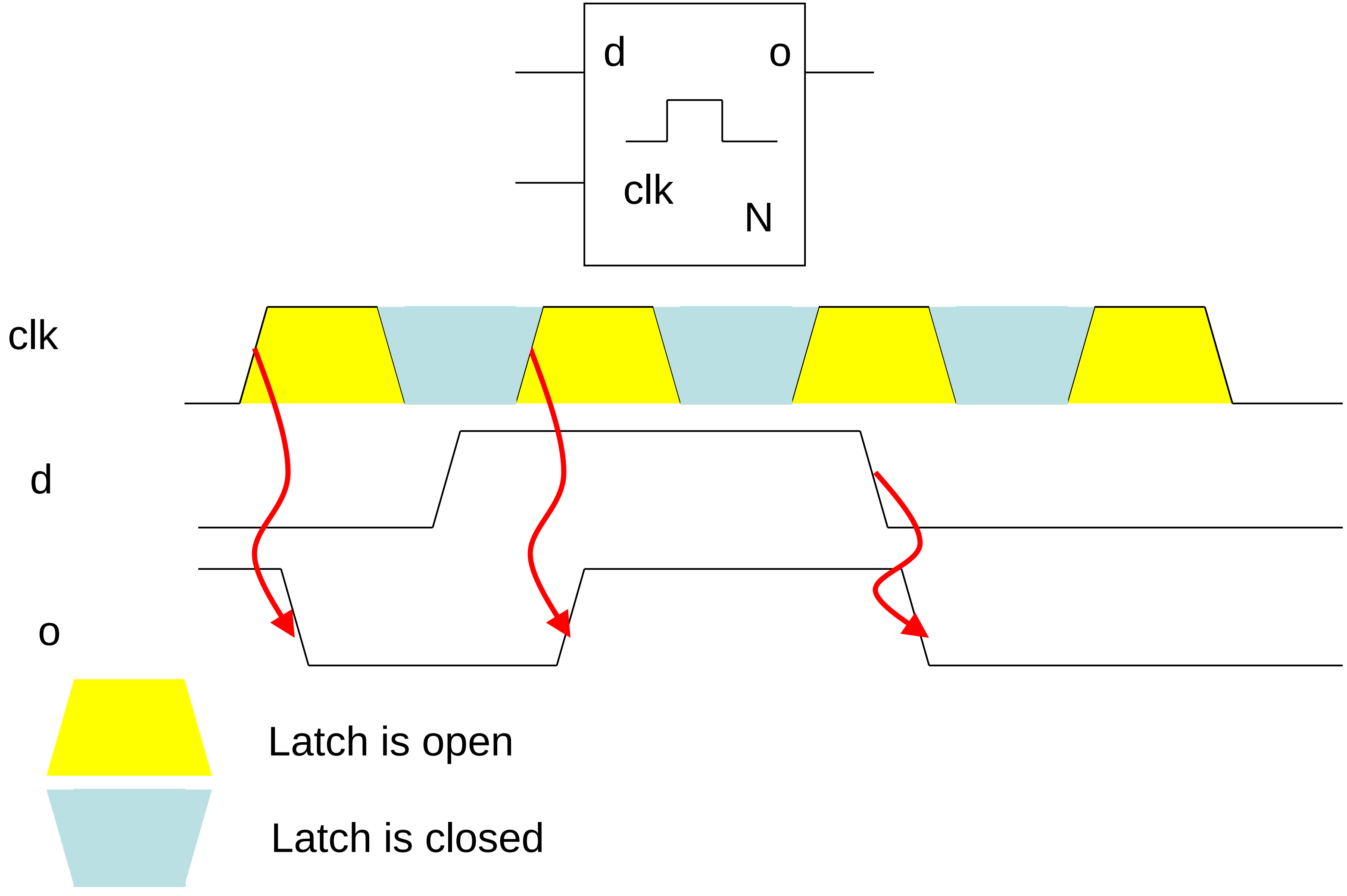
Sequential Gates

- Several Uses
  - Used to store state of the machine (Like a register)
  - Used in finite state machine to represent different states of the machine
  - Also used in pipelined machine to designate pipestages
- Always accept clock as an input to synchronize pipestages
- Types:
  - Latch (Phase 1/Phase 2)
  - Flip-flop (Rising/Falling edge)
- Flavors
  - Enabled
  - Synchronous Set/Reset
  - Asynchronous Set/Reset


clk
d1#
TH
id1
T
id1n
T
iclkn
clk#
d1
pout
d
T
idn
d#
T
itd1
nout
o

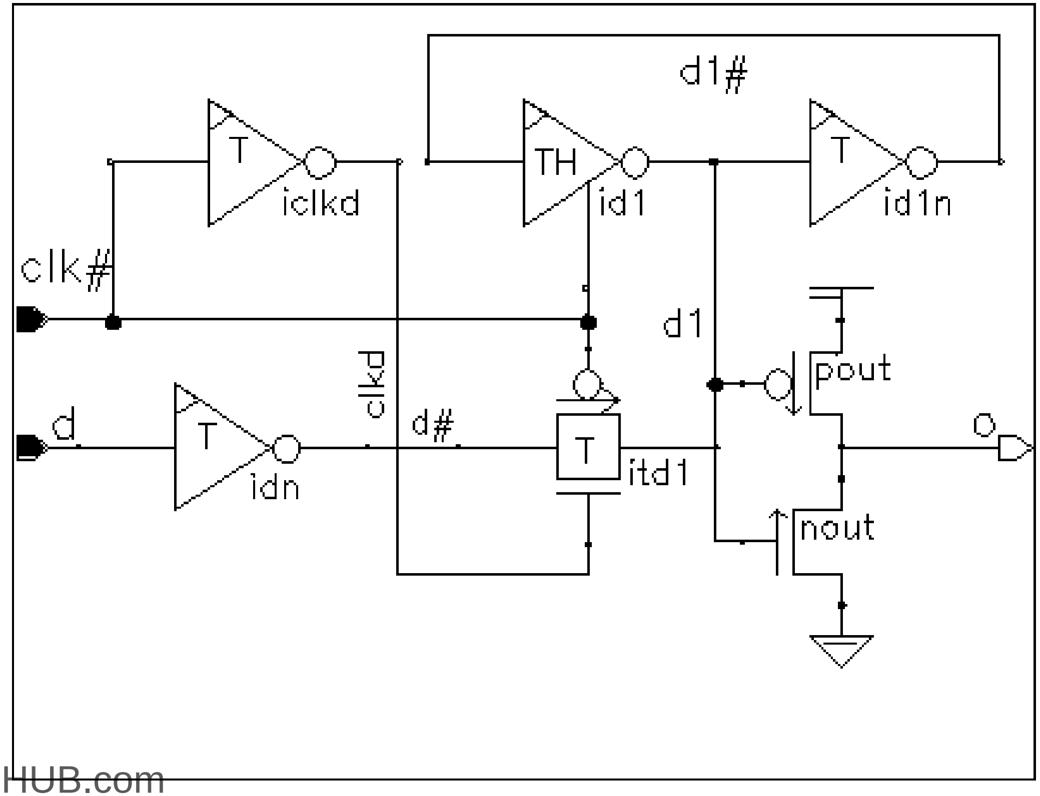

d
o
clk
N

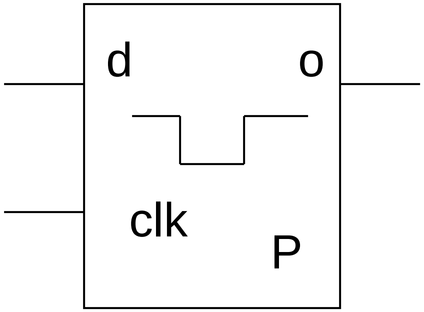
Traditional N-Latch



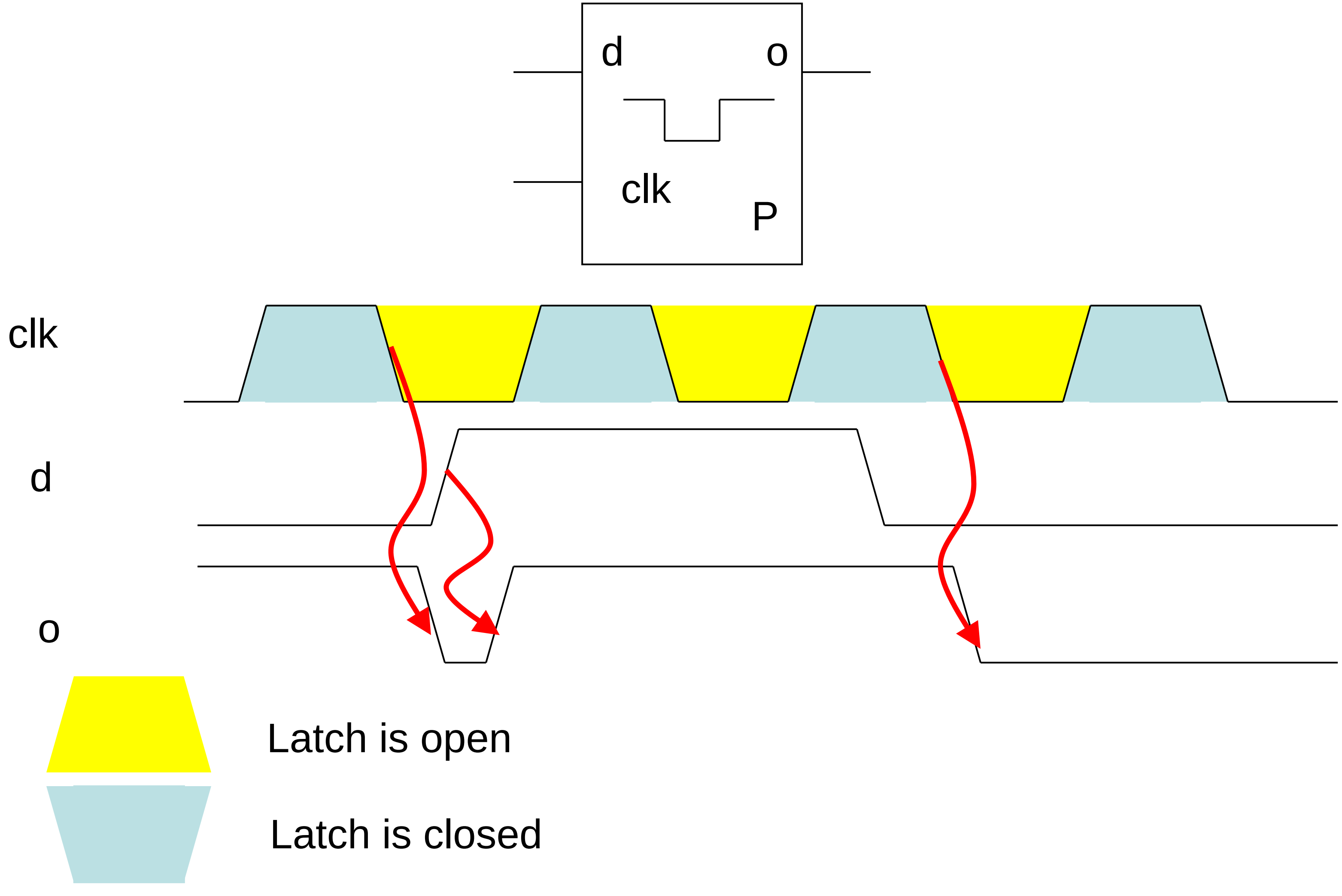
Traditional P-Latch

- Level sensitive (Phase 2 open when clk is low)
- Also called P first or Phase 2 Latch



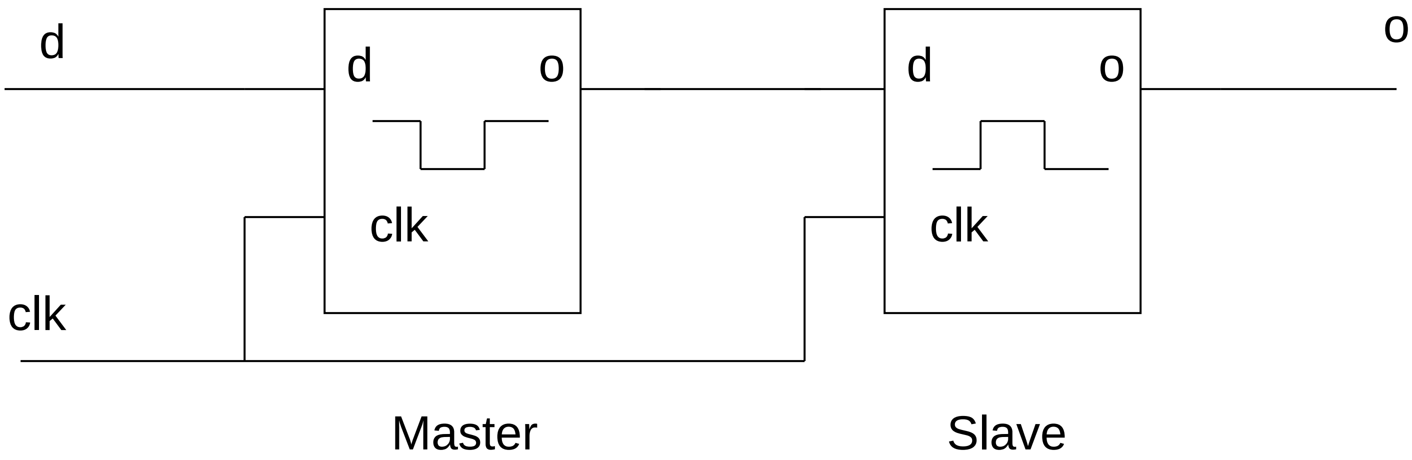


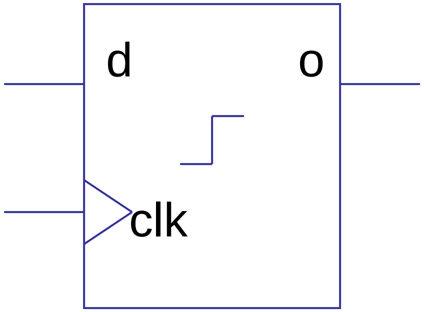
Traditional P-Latch



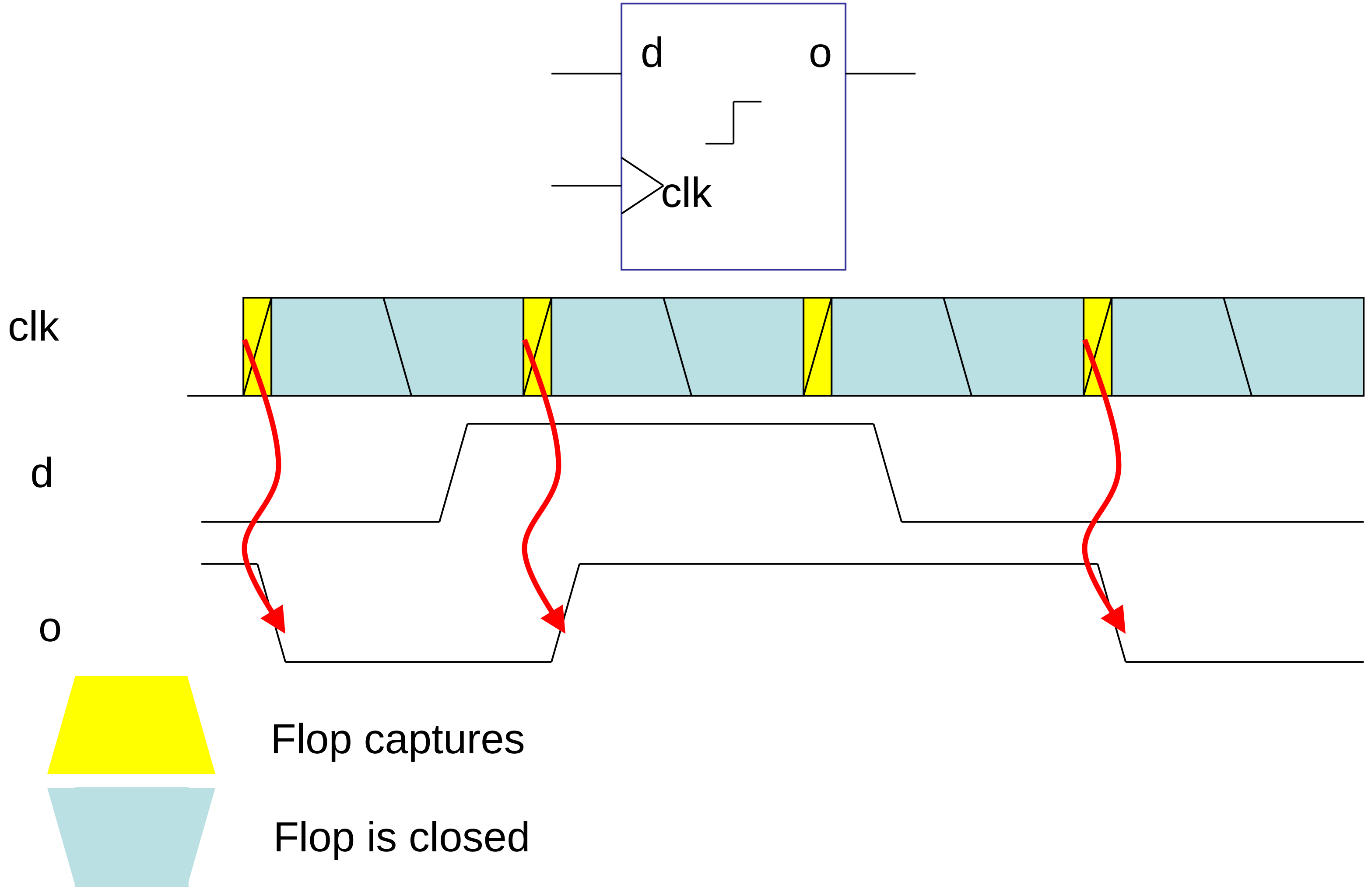
Positive Edge Flip-flop

- Edge sensitive
  - FF captures and drives data on rising edge of clk.
  - When clk rises, P latch shuts off, N latch turns on



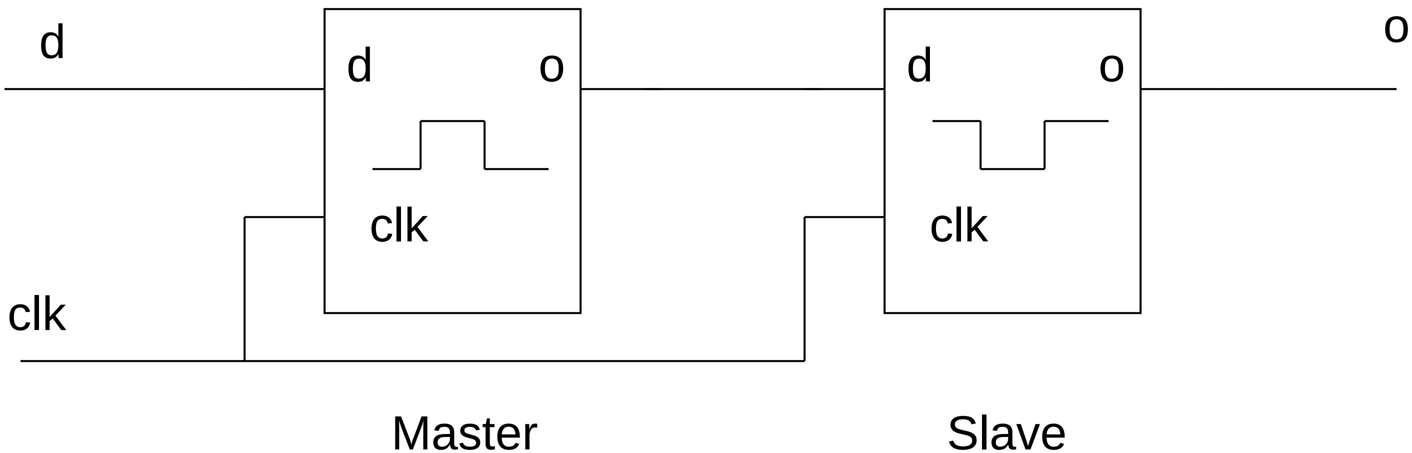


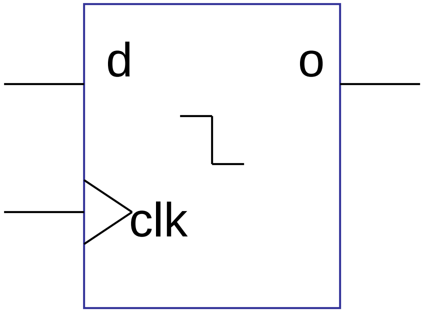
Positive Edge Flip-flop



Negative Edge Flip-flop

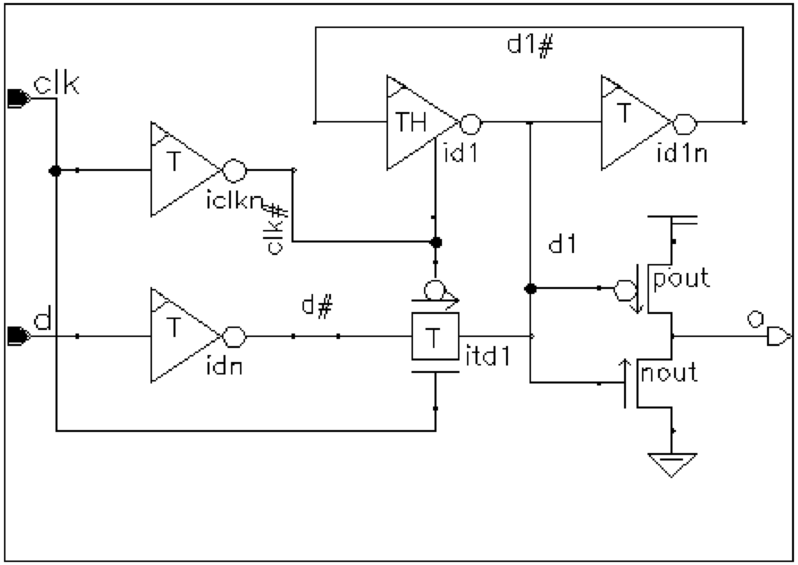
- Edge sensitive (FF captures and drives data on falling edge of clk)



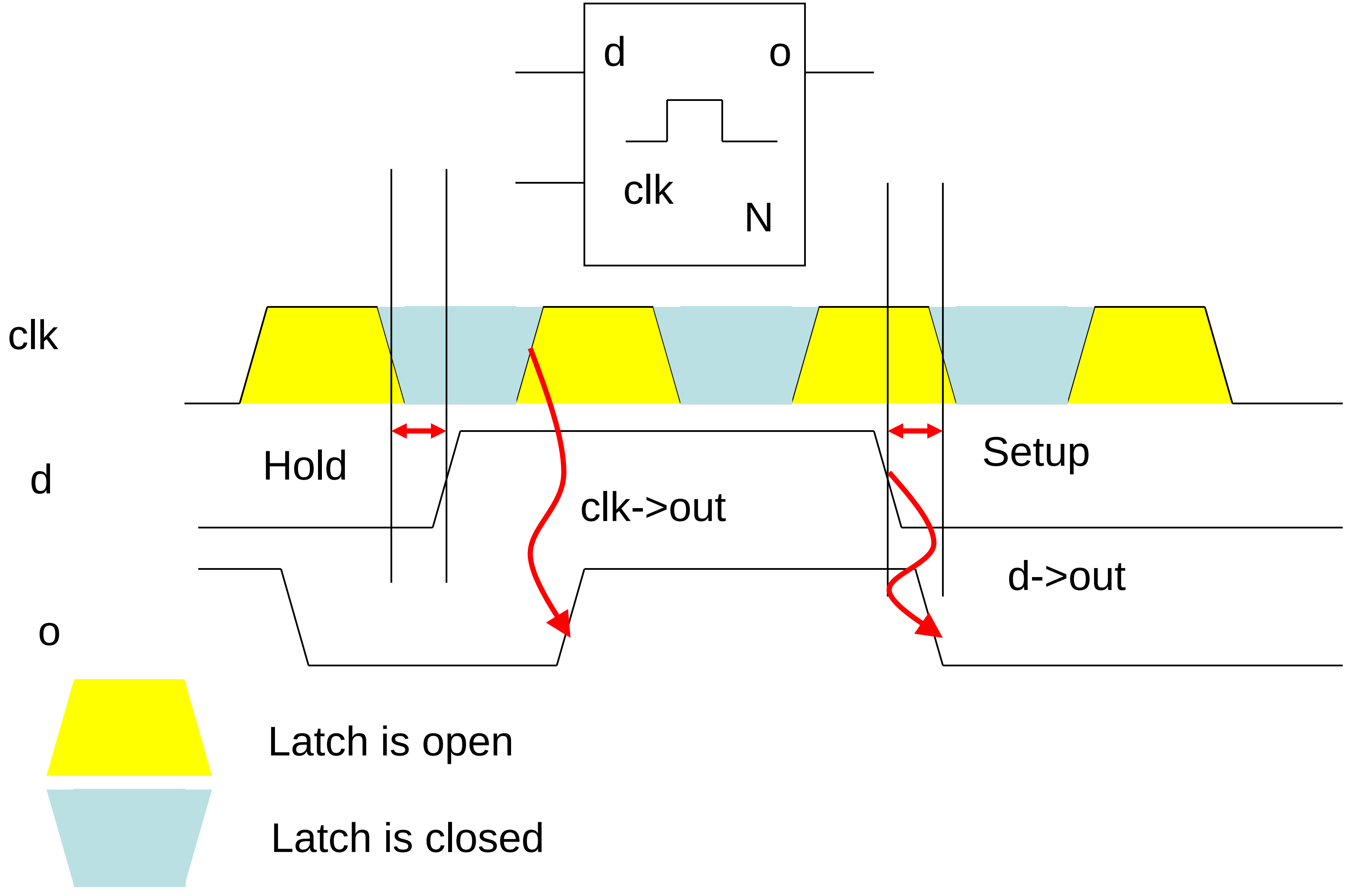


Latch parameters

- Setup time
  - Time, before the latch closes, that the data must arrive to guarantee the data is captured correctly after the latch closes
- Hold time
  - Time, after the latch closes, that the data can not switch to guarantee the data is captured correctly when the latch closes
- Clk to out delay
  - Delay from clk to out when the data is setup before the clk
- Data to out delay
  - Delay from data to out when the data arrives after the clk



Traditional N-Latch



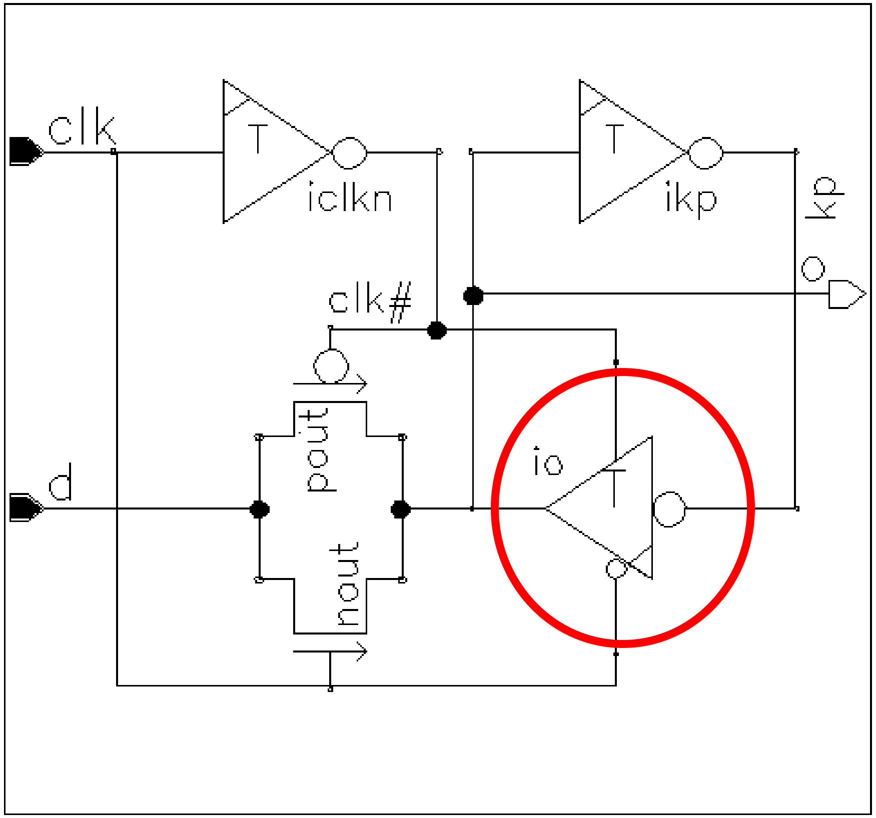
Flip-flop parameters

- Setup time
  - Time, before the opening clk edge, that the data must arrive to guarantee the data is captured correctly
- Hold time
  - Time, after the clk edge, that the data can not switch to guarantee the data is captured correctly
- Clk to out delay
  - Delay from clk to out when the data is setup before the clk.

- Note that there is no data to out delay since flops must have data setup to the edge.

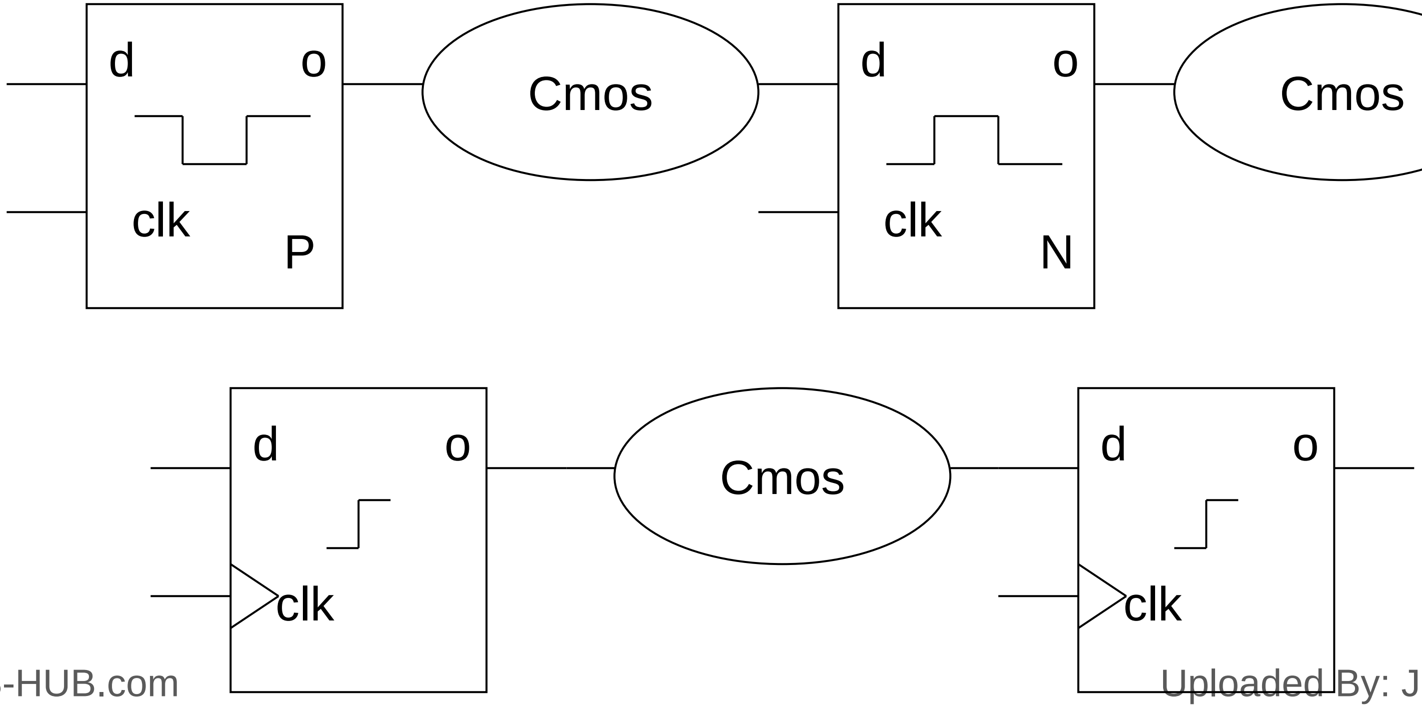
Full keeper

- Feedback device has tristated both N and P.
- Writing of output through passgate has no opposition



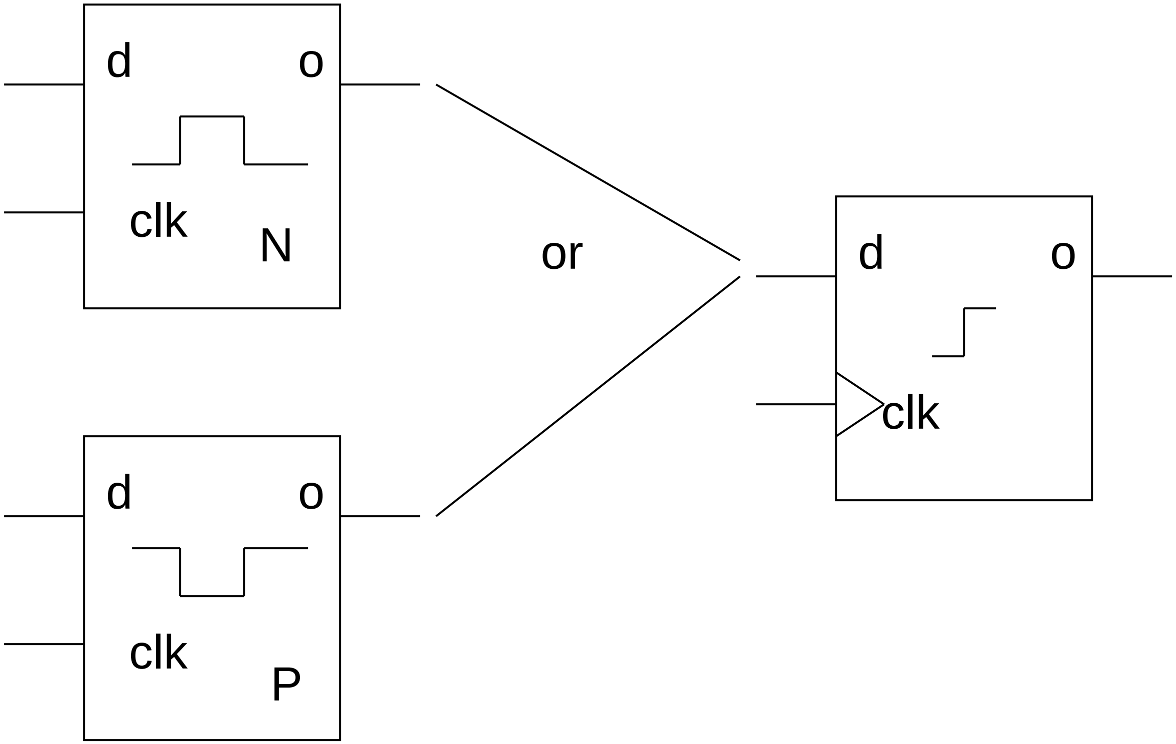
Pipelining Sequence

- For proper pipelining, must always alternate N-latch and P-latch.
- Flops must be consecutive rising edge flops



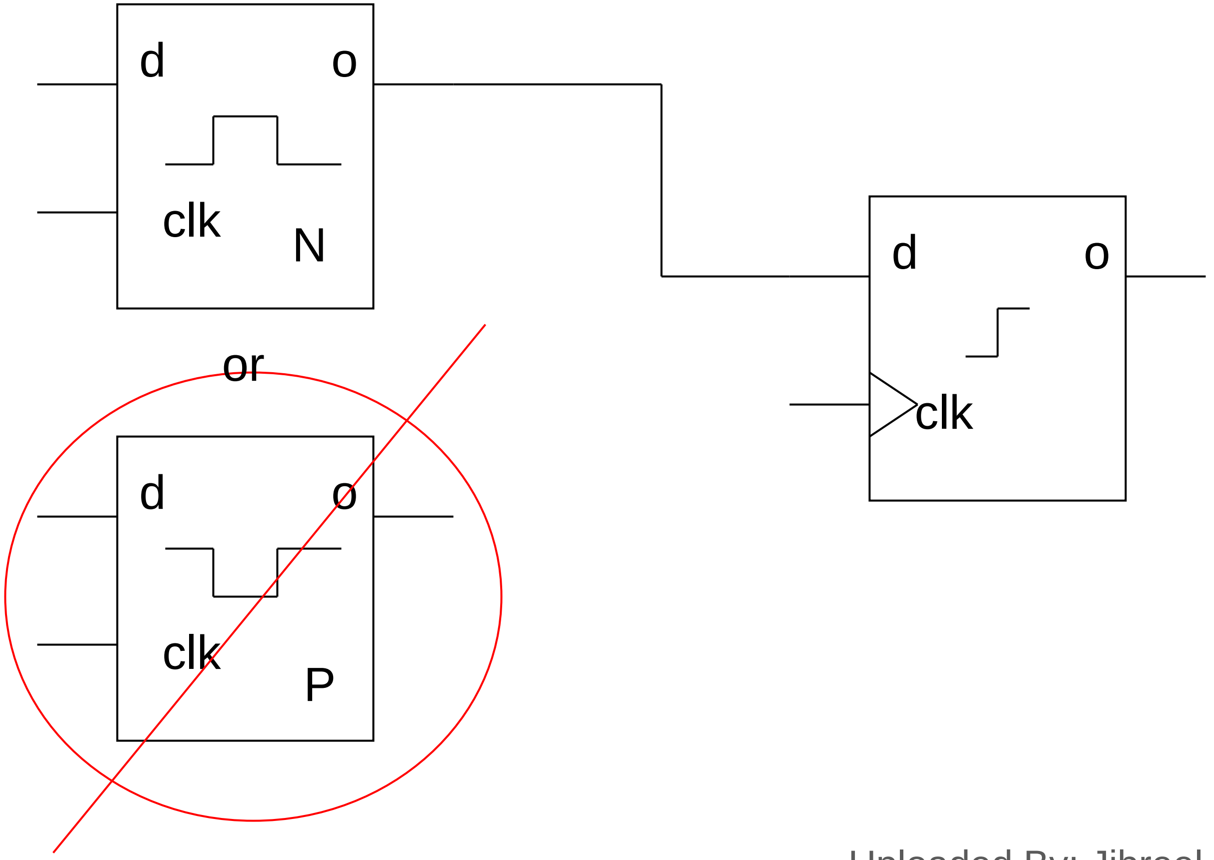
What about Latch/Flop boundary?

- What type of latch should be in front of a rising edge flop?



What about Latch/Flop boundary?

- Since in a flop, the master is a P-latch, we MUST have a N-latch in front of a rising edge flop.

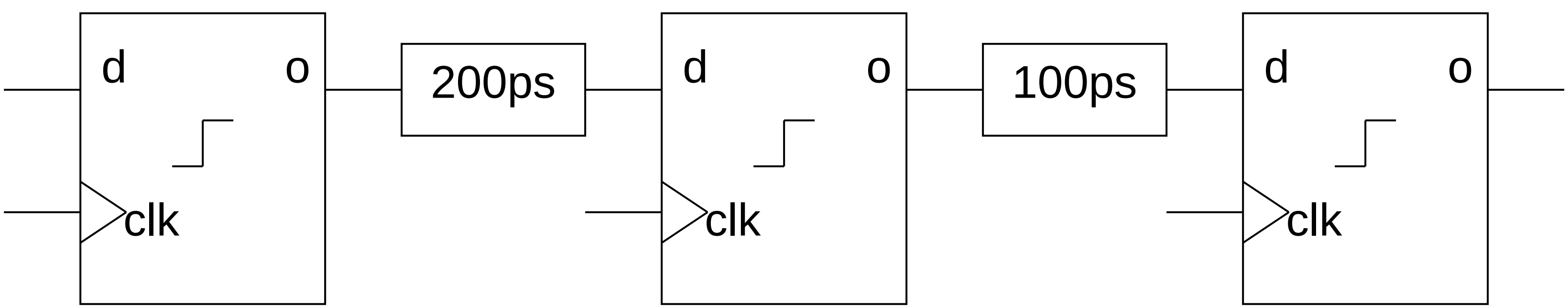


Likewise, Flop must drive to a P-latch

- On the other hand, since the slave is an n-latch, a rising edge flop must be followed by a p-latch

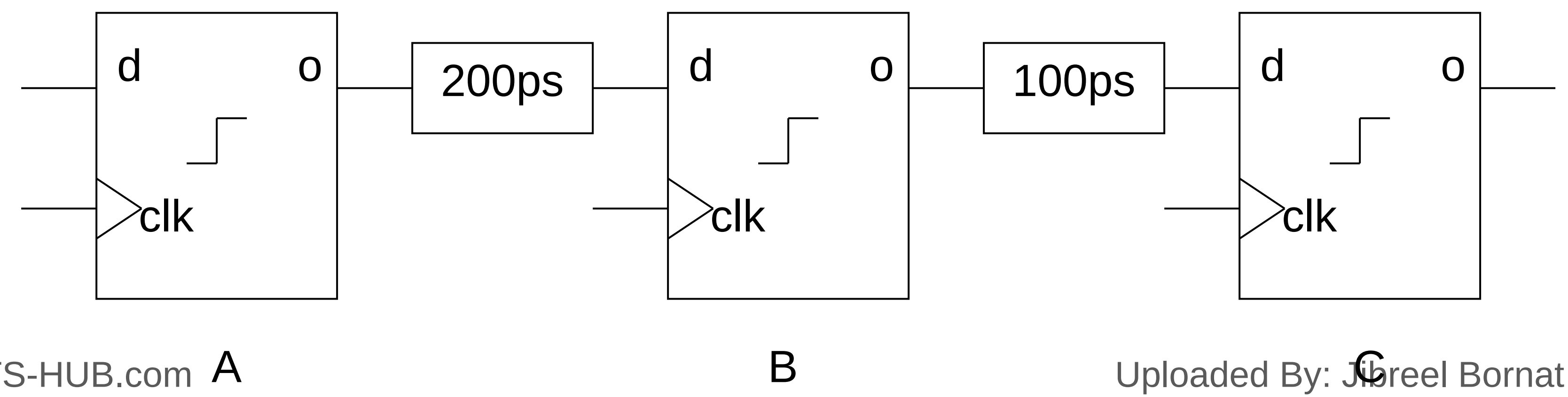
Flop issues

- Assuming 100ps setup time, skew and clk-out delay
- How many paths are there and how many cycles?
- What is max frequency this circuit could run?



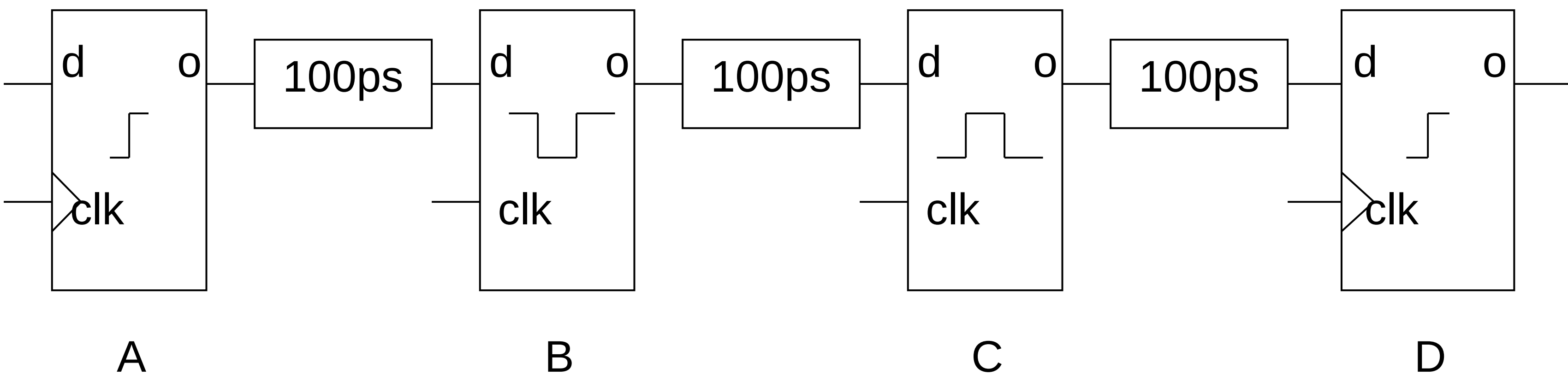
Flop Answers

- 2 paths (A->B and B->C) each 1 cycle
- 200ps + 100ps = 300ps. 1/300ps = 3.33GHz
- 100ps + 100ps = 200ps. 1/200ps = 5.0GHz
- Max frequency = 3.33GHz



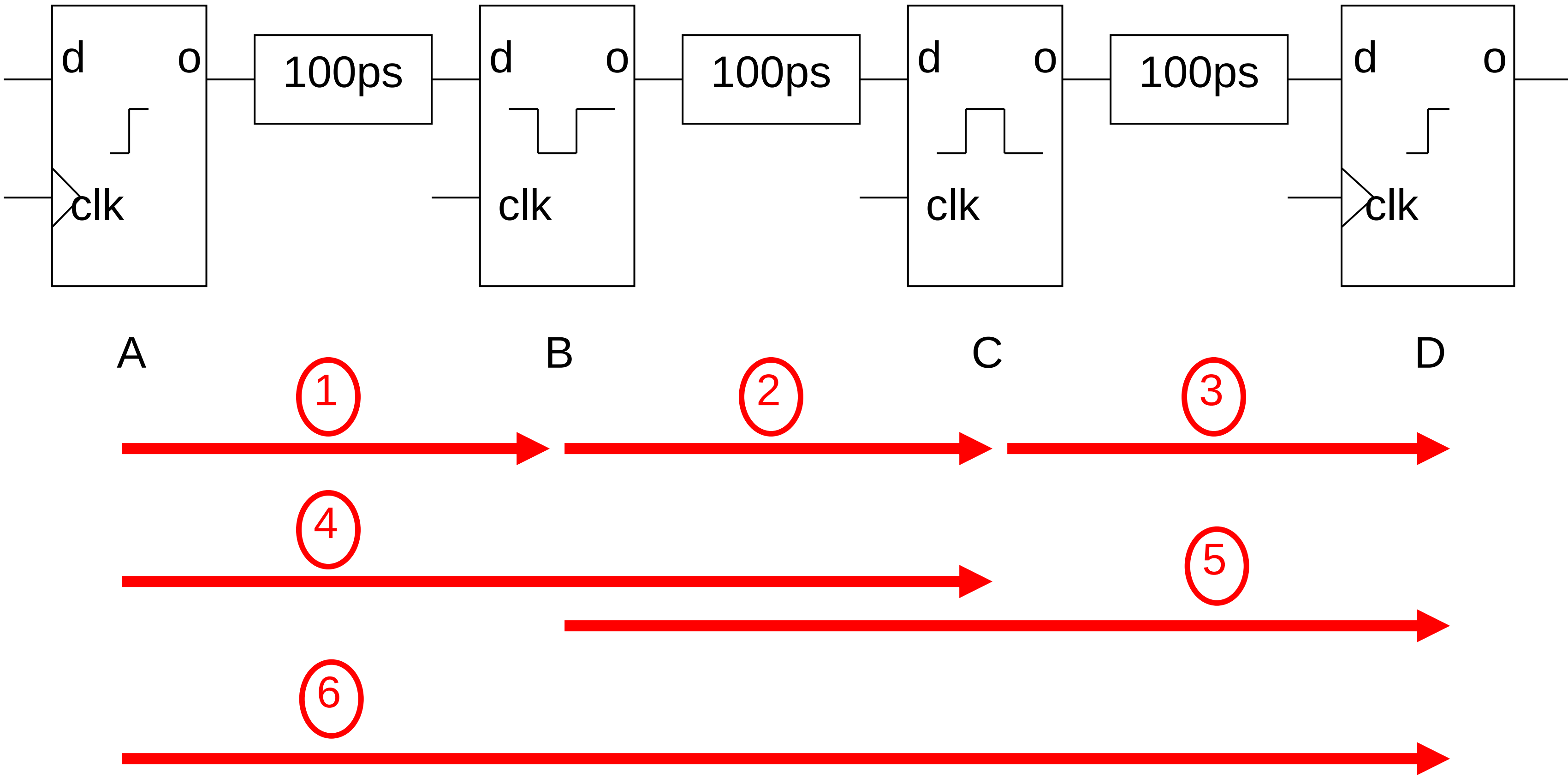
Time borrowing

- Time borrowing is a technique to increase frequency by converting flops to latches.
  - Allows amortizing skew, jitter, clk to out and data delays across more than 1 cycle
- How many paths are there and how many cycles?
- What is the new frequency assuming data-out delay is 50ps?



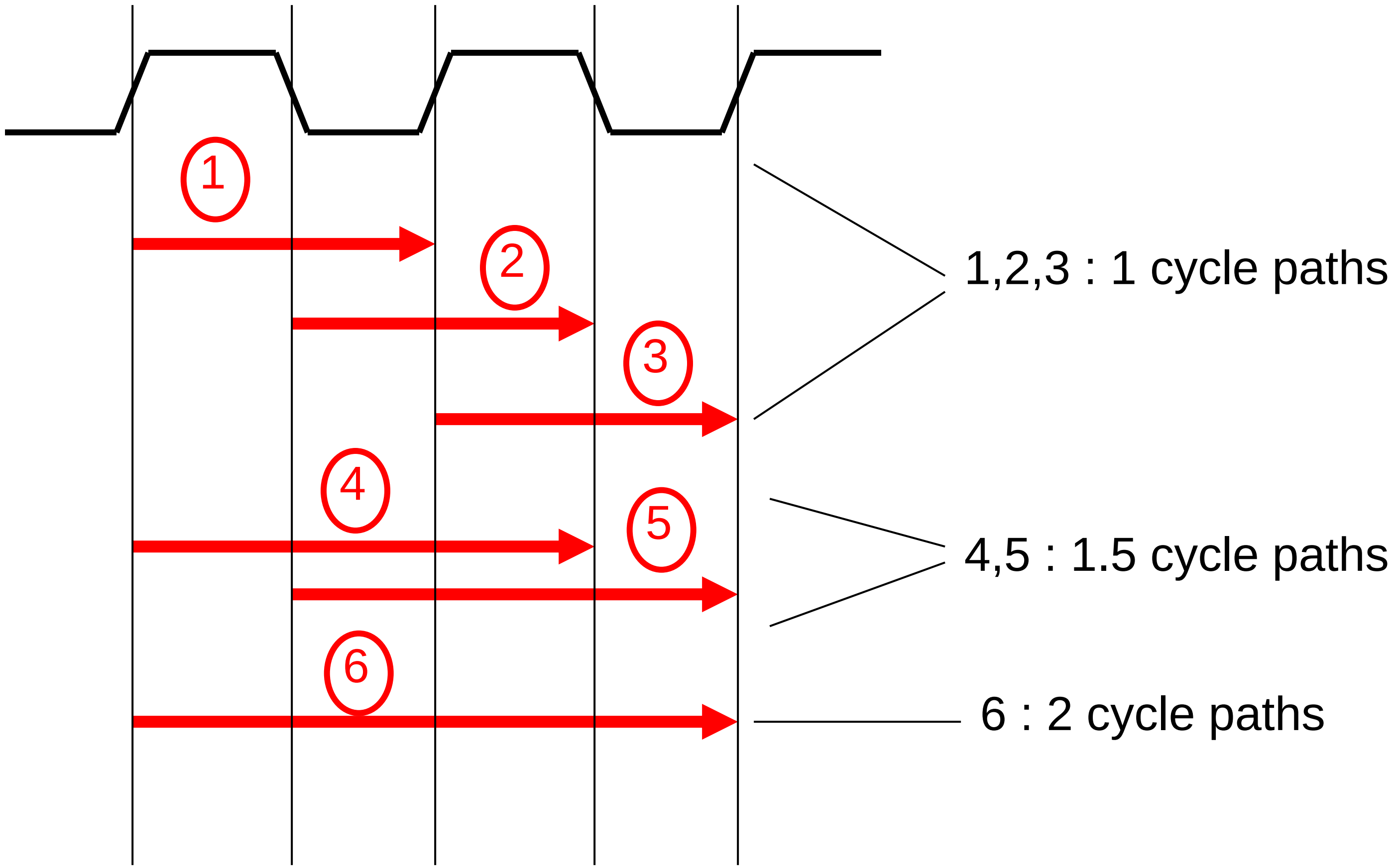
6 paths

- 6 paths (A-B, A-C, A-D, B-C, B-D, C-D).



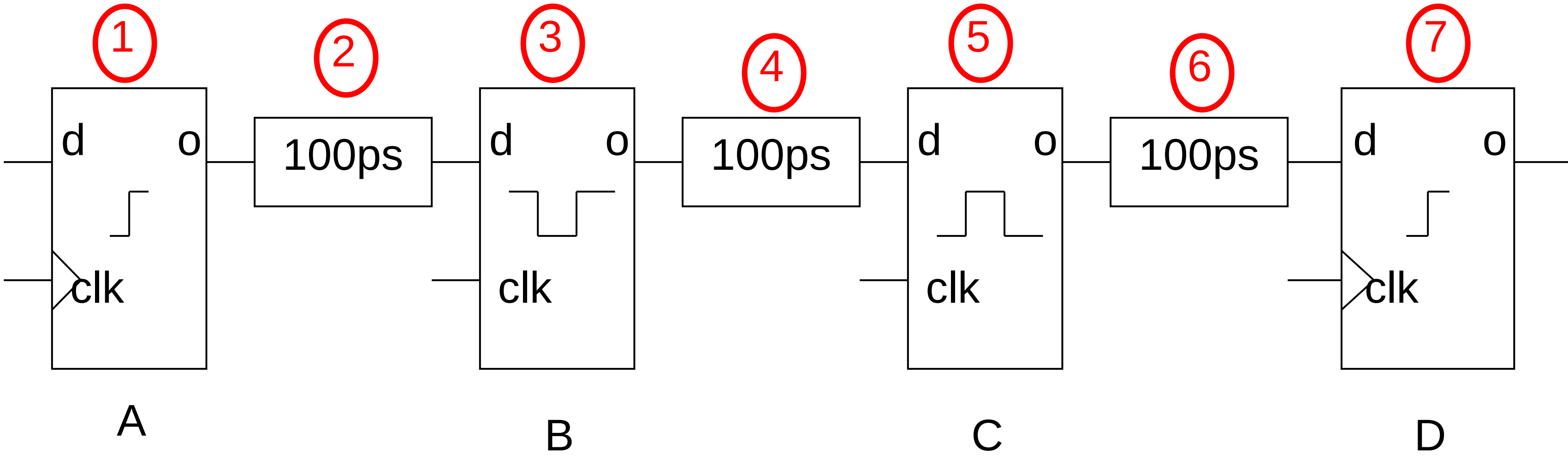
6 paths

- 6 paths (A-B, A-C, A-D, B-C, B-D, C-D).



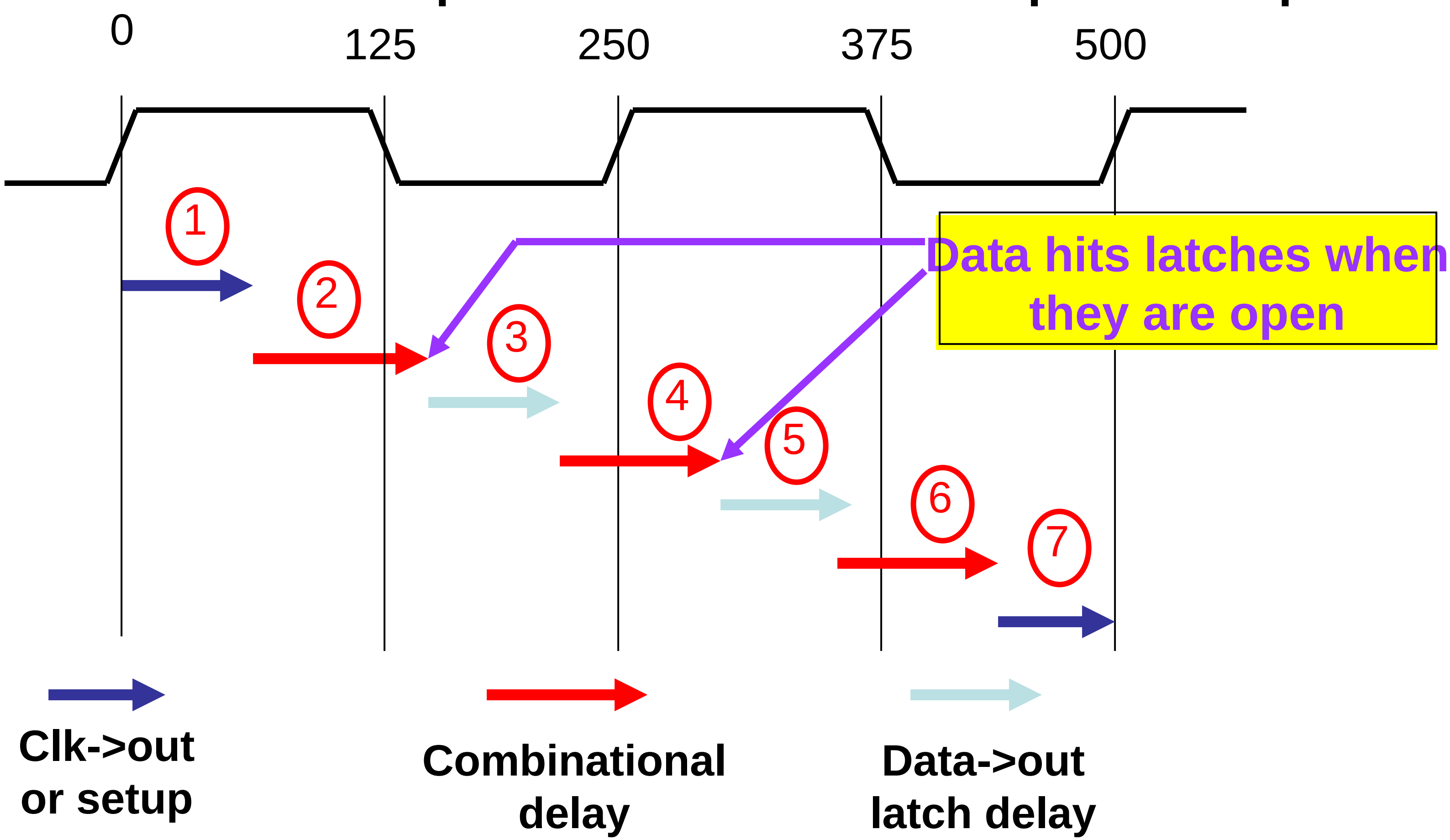
Time borrowing Frequency calculations

- 2 paths -> 6 paths (A-B, A-C, A-D, B-C, B-D, C-D).
- A-B, B-C, C-D: 1 cycle path (100ps + 100ps) = 200ps = 5GHz
- A-C, B-D: 1.5 cycle path (100 + 50 + 100 + 100) = 350ps : 1.5/350ps = 4.28GHz
- A-D: 2 cycle path (100 + 50 + 100 + 50 + 100 + 100) = 500ps : 2/500ps = 4GHz
- New frequency = 4GHz



A-D path explained

- Assume 50ps clk-out and 50ps setup



When/Why time borrowing works

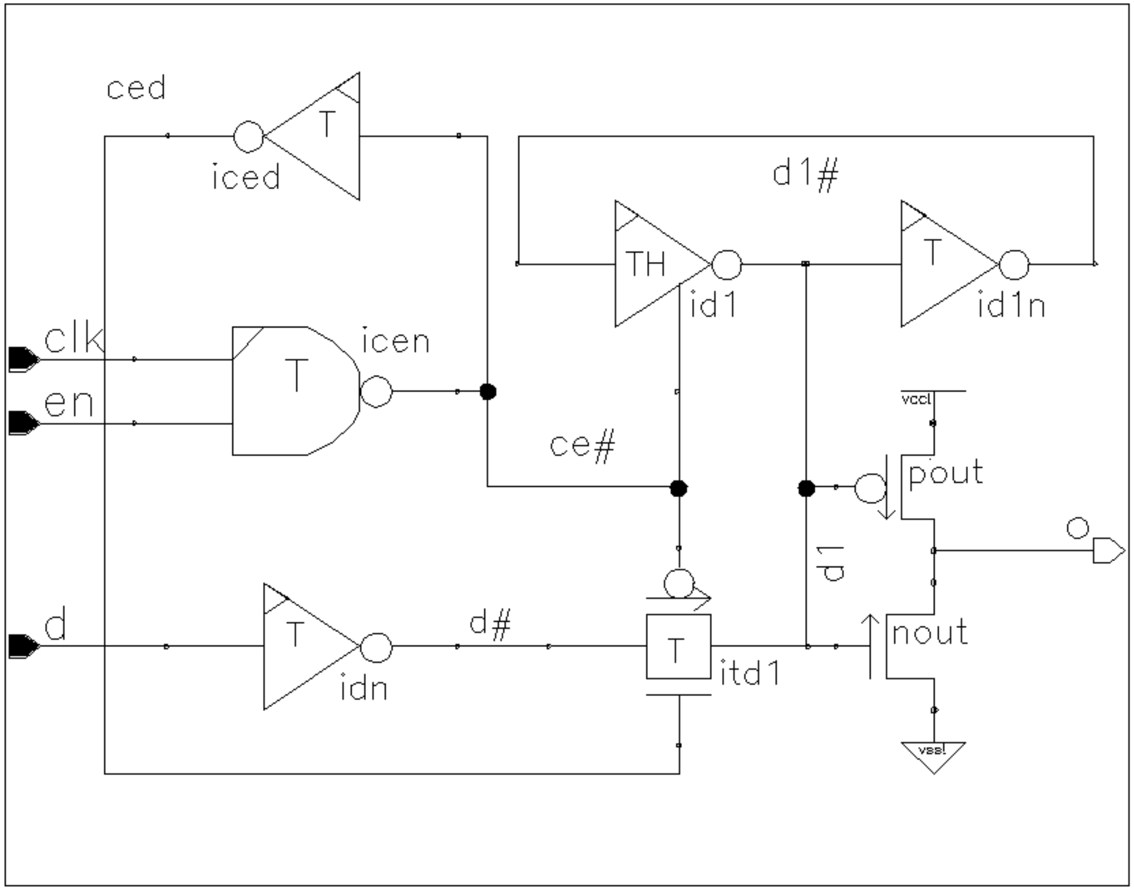
- 2 advantages of time borrowing
  - Allows “borrowing” time from a cycle that has extra margin to a cycle that doesn’t
    - In our example, the cycle that ran at 3.33GHz borrows time from the cycle that had 5GHz
  - Allows amortization of setup time and skew over several cycles
    - In our example, the 100ps overhead penalty is now over 2 cycles so the per cycle penalty is 50ps

Time borrowing

- Positives:
  - Allows some amount of time borrowing without some of the negative of time borrowing
    - Very little extra clock load
    - No latch explosion
    - No RTL change
- Negatives:
  - Only allows only a buffer delay of borrowing
  - Min delay of the first path has been worsened by 1 buffer delay.
  - Skew has increased

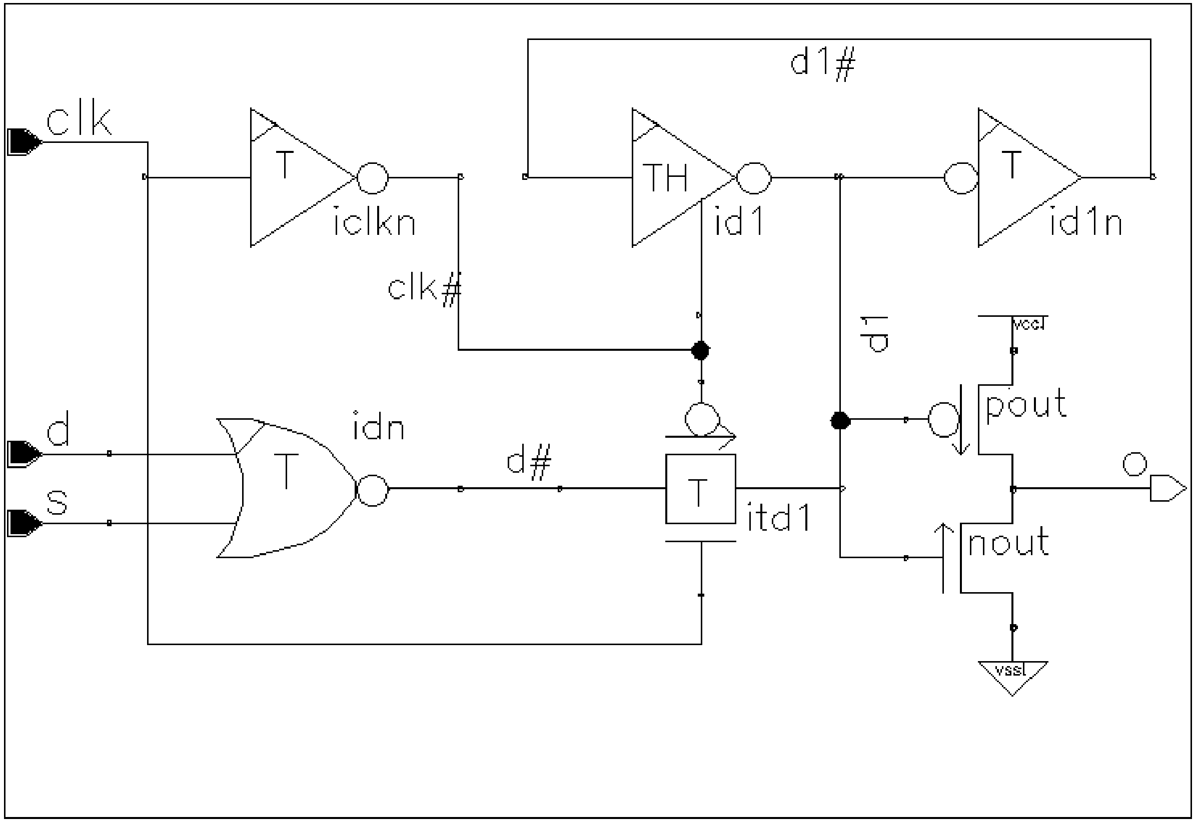
Enabled latches

- When enable is a 0, latch does keeps passgate closed



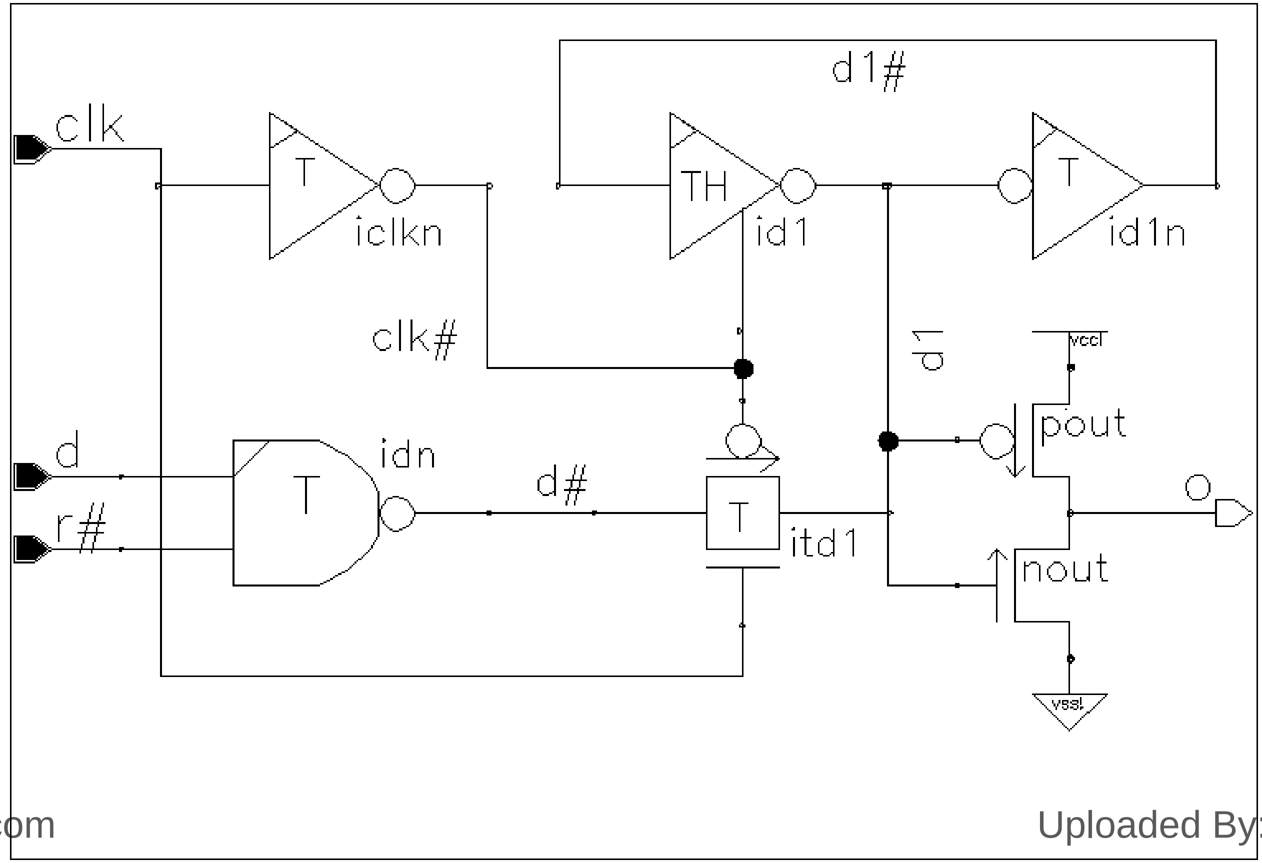
Synchronous Set latches

- When set is a 1, when clk is open, latch stores a 1



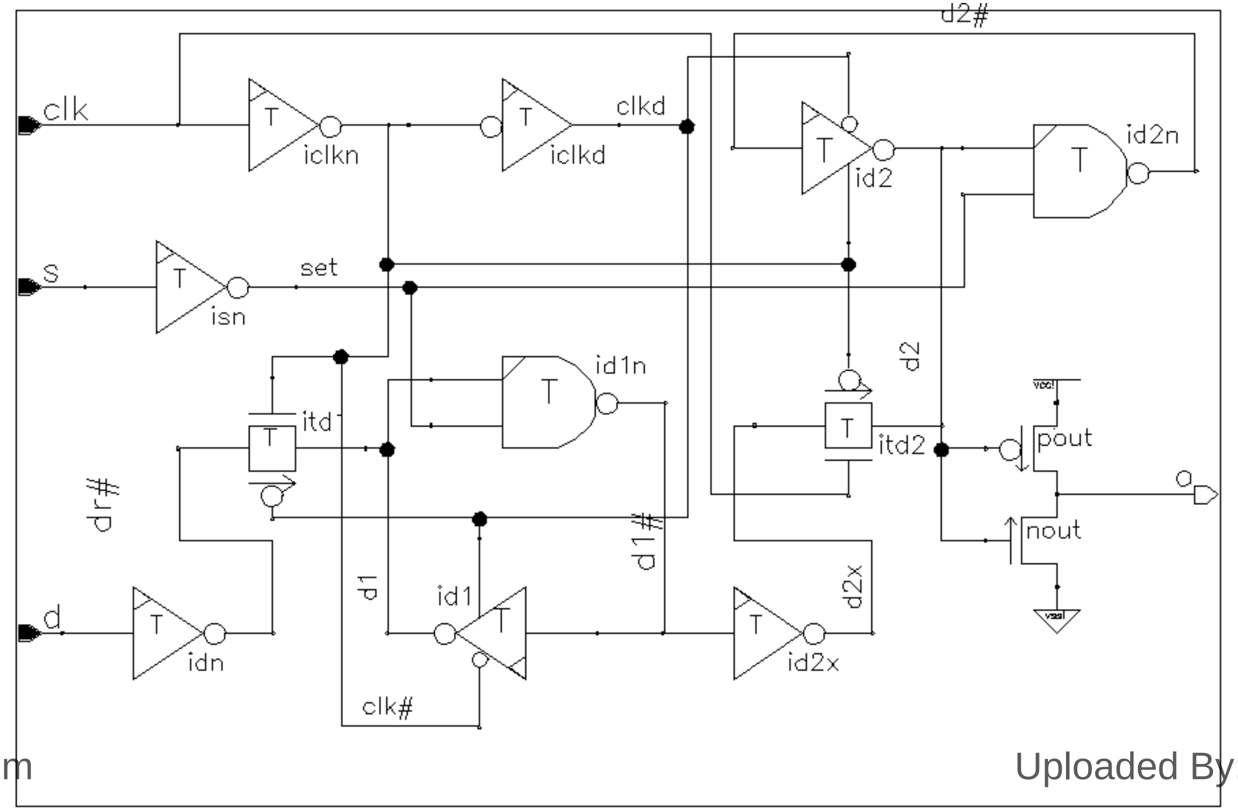
Synchronous Reset latches

- When r# is a 0, when clk is open, latch stores a 0



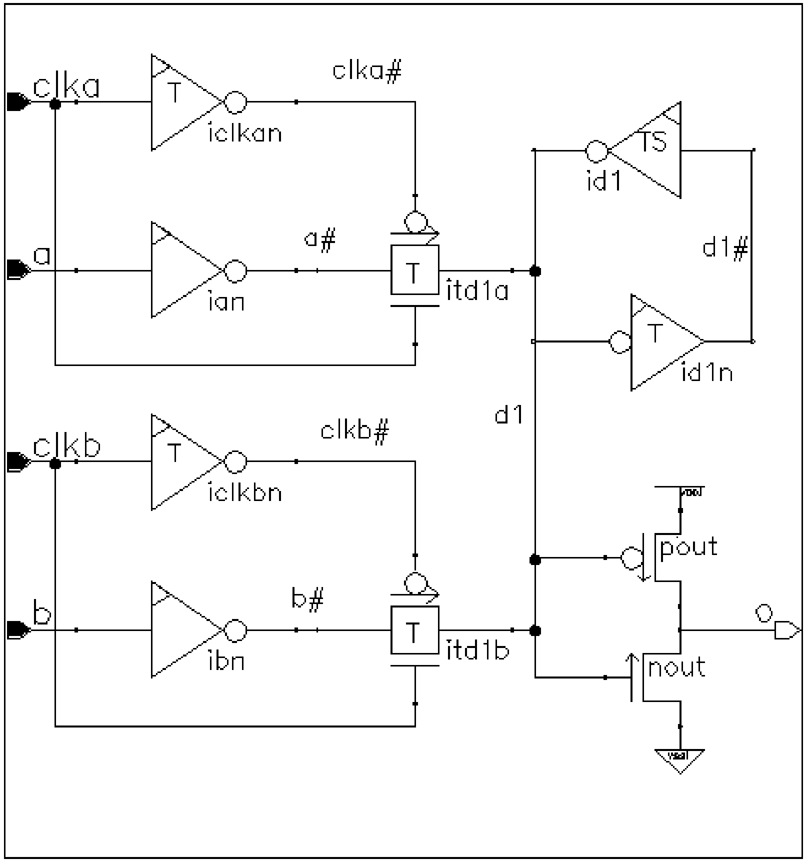
Asynchronous Set Flops

- When set is a 1, output immediately changes to a 1
- Does not wait for rising edge of a clock



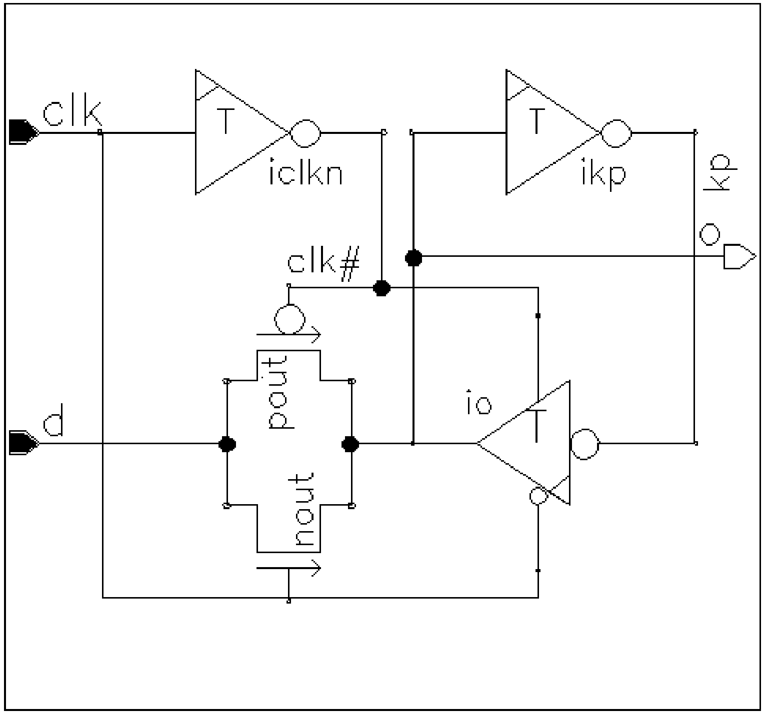
Mux latches

- Note selects are mutexed qualified clocks



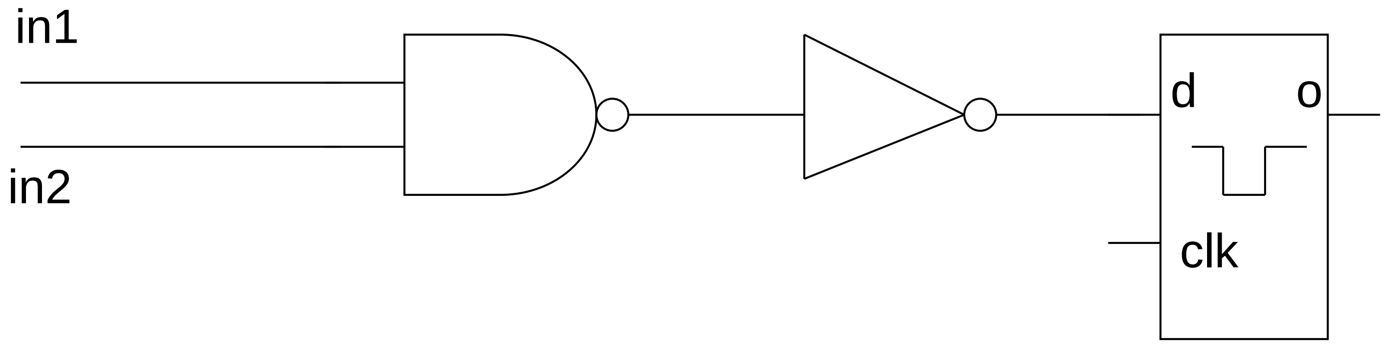
Unprotected latches

- Either no input protection, output protection or both
- Allows converting the inverter to logic gates or customize the inverters normally in a latch
- Loss of output protection require extra caution not to disturb latch node
- To guarantee latch writability, should use a full keeper



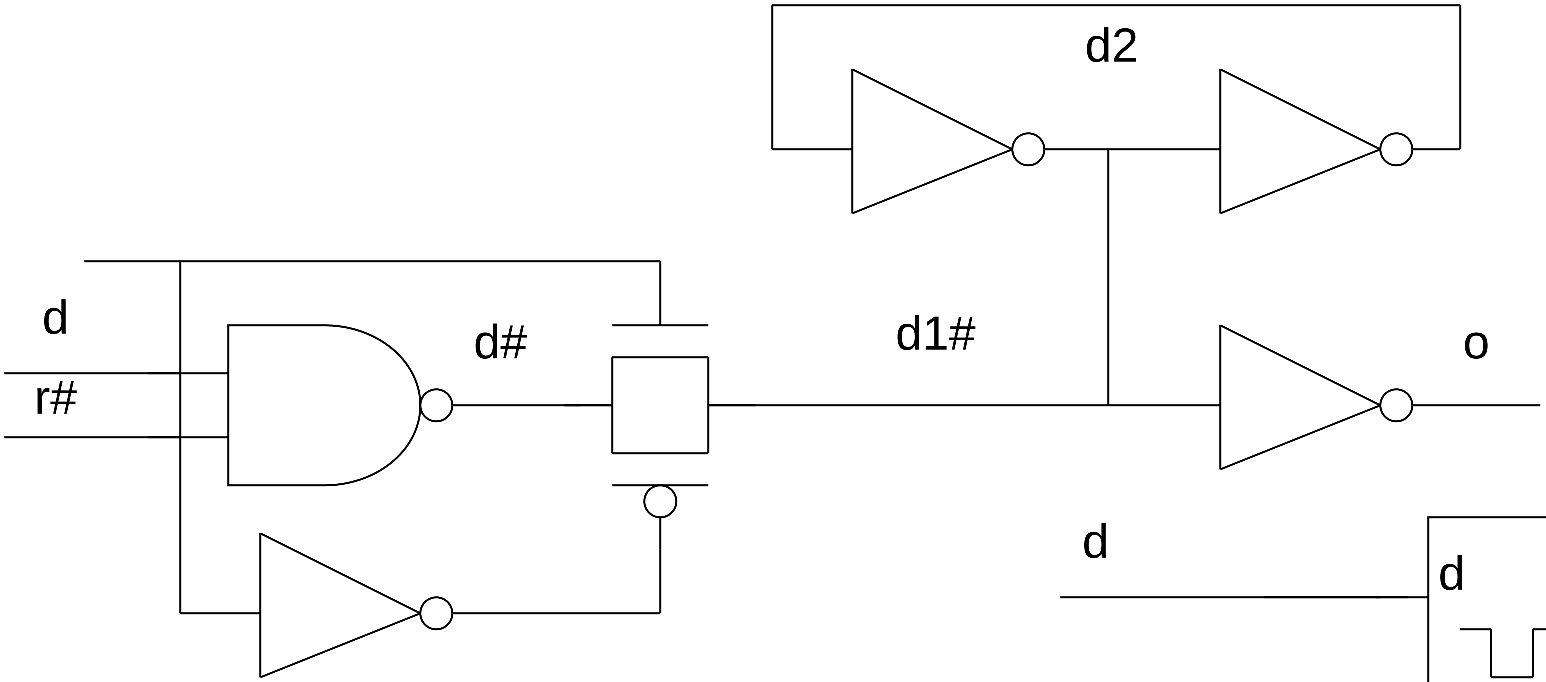
Using the Enabled/Set/Reset sequentials

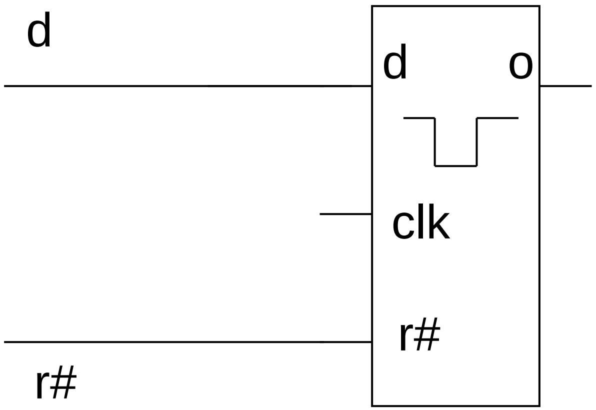
- Problem: How can we improve this?



Use a Reset Latch to reduce logic

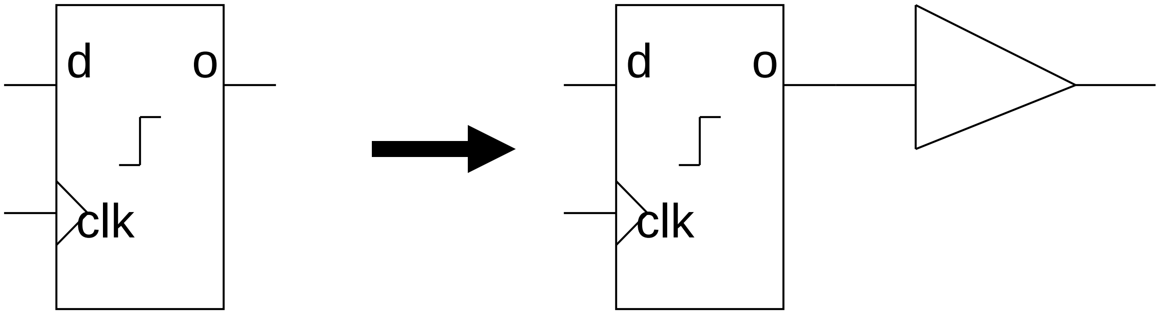
- Converts the first inverter in a latch to a nand gate to eliminate 2 gates





Power issues

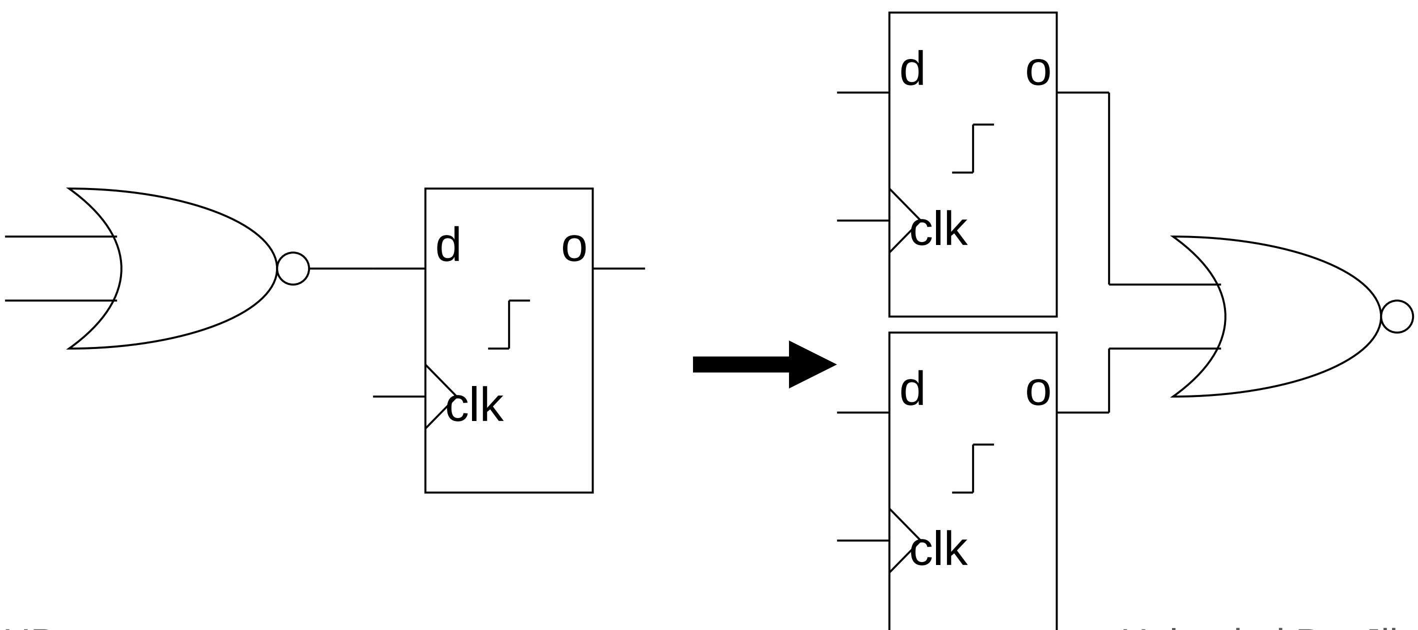
- Sequentials mean clock power!
  - Keep sequentials as small as possible by buffering outputs when timing allows to keep the sizing of the flip-flop small.



  - Avoid duplicating/unnecessary flops (Look for opportunities for flop reduction)

Power issues

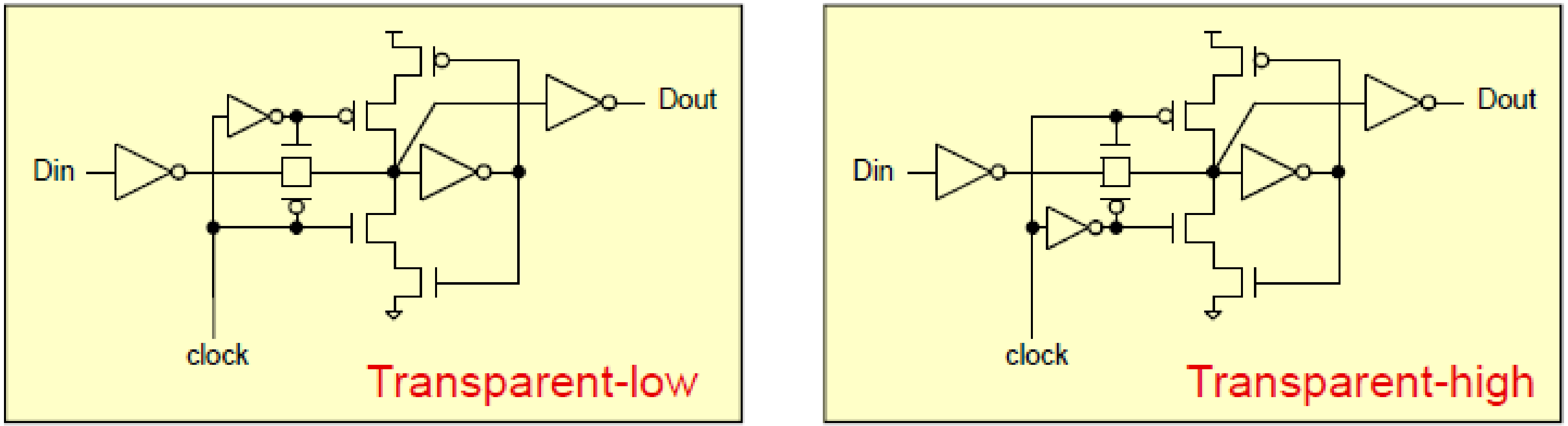
- Be careful where to select flops
  - When fixing speed paths by moving logic from 1 stage to another, watch for flop explosion
  - Same issue with flop->latch conversion

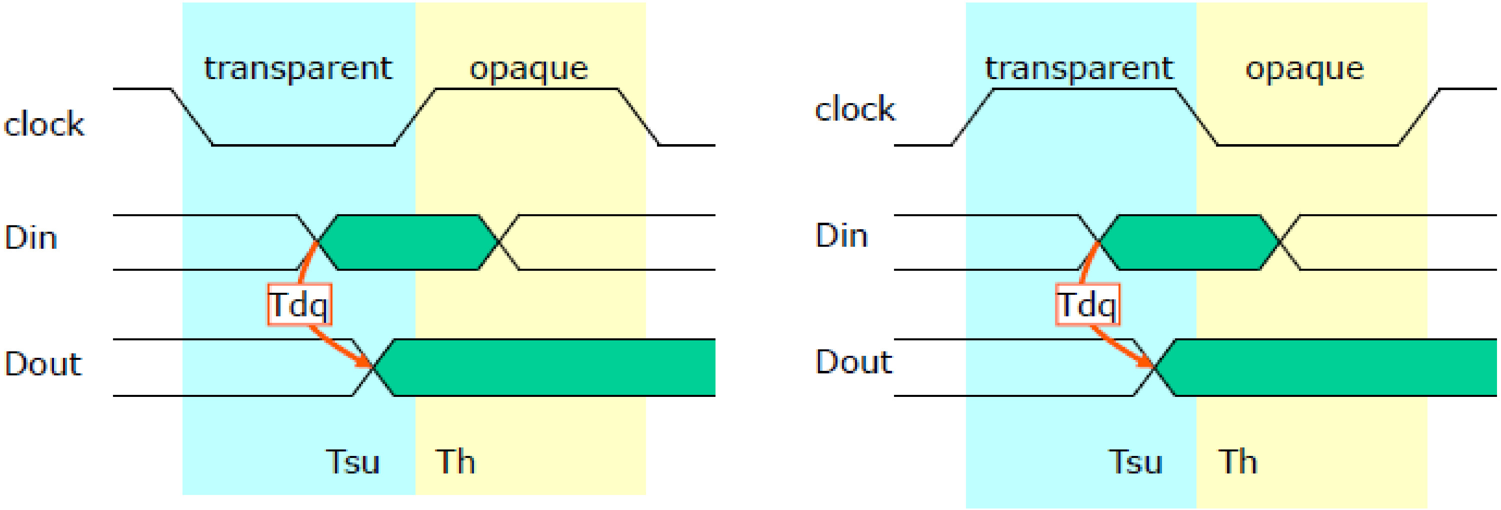


Summary

- Know the different types of sequential and the order they need to be used.
- Understanding time borrowing can help solve speed paths in a path at the expense of complexity, clock power
  - Select flop to latch conversion or time borrowing flop where appropriate
- ALWAYS think about power. Keep sequential count as low as possible and keep sequential sizes small by buffering outputs.

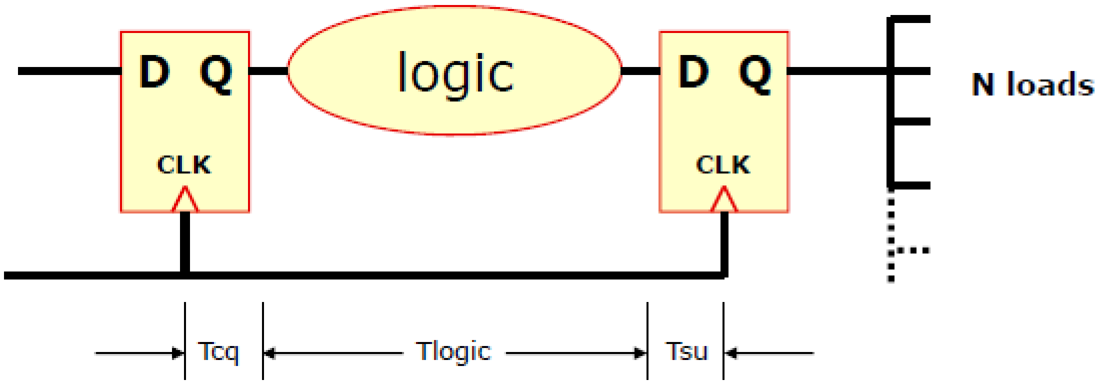
Basic LATCH Operation



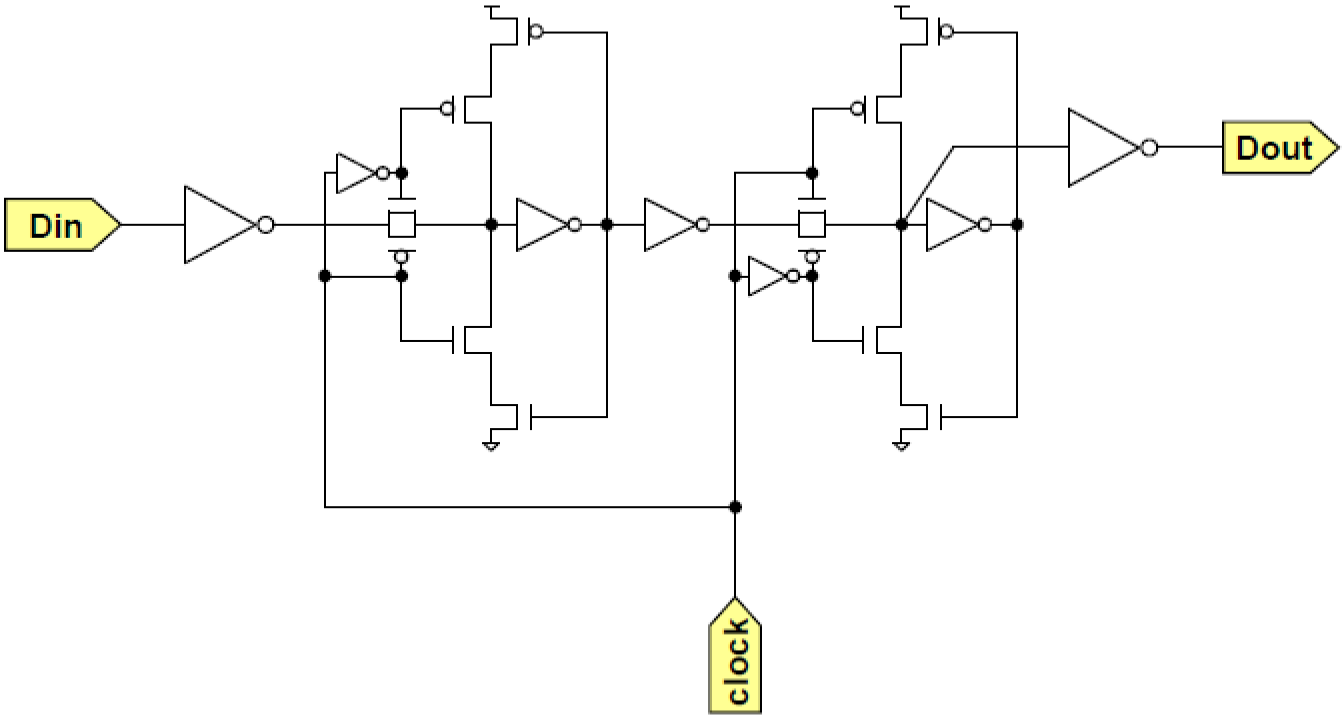


FLOP Delay

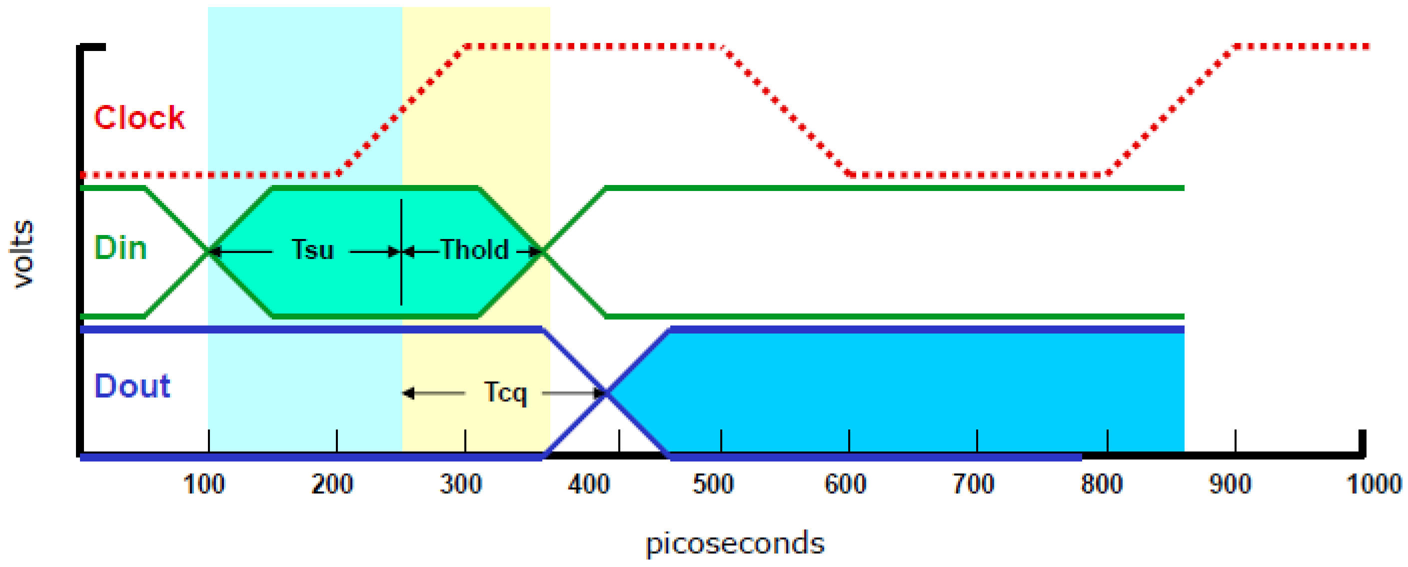
- **Sum of setup time and Clk-output delay is the only true measure of the performance with respect to the system speed (MAXDELAY)**
- ***Tcycle = Tcq + Tlogic + Tsu + Tskew***
- ***Tlogic* contains interconnect delay**



Building a FLOP with Two Latches

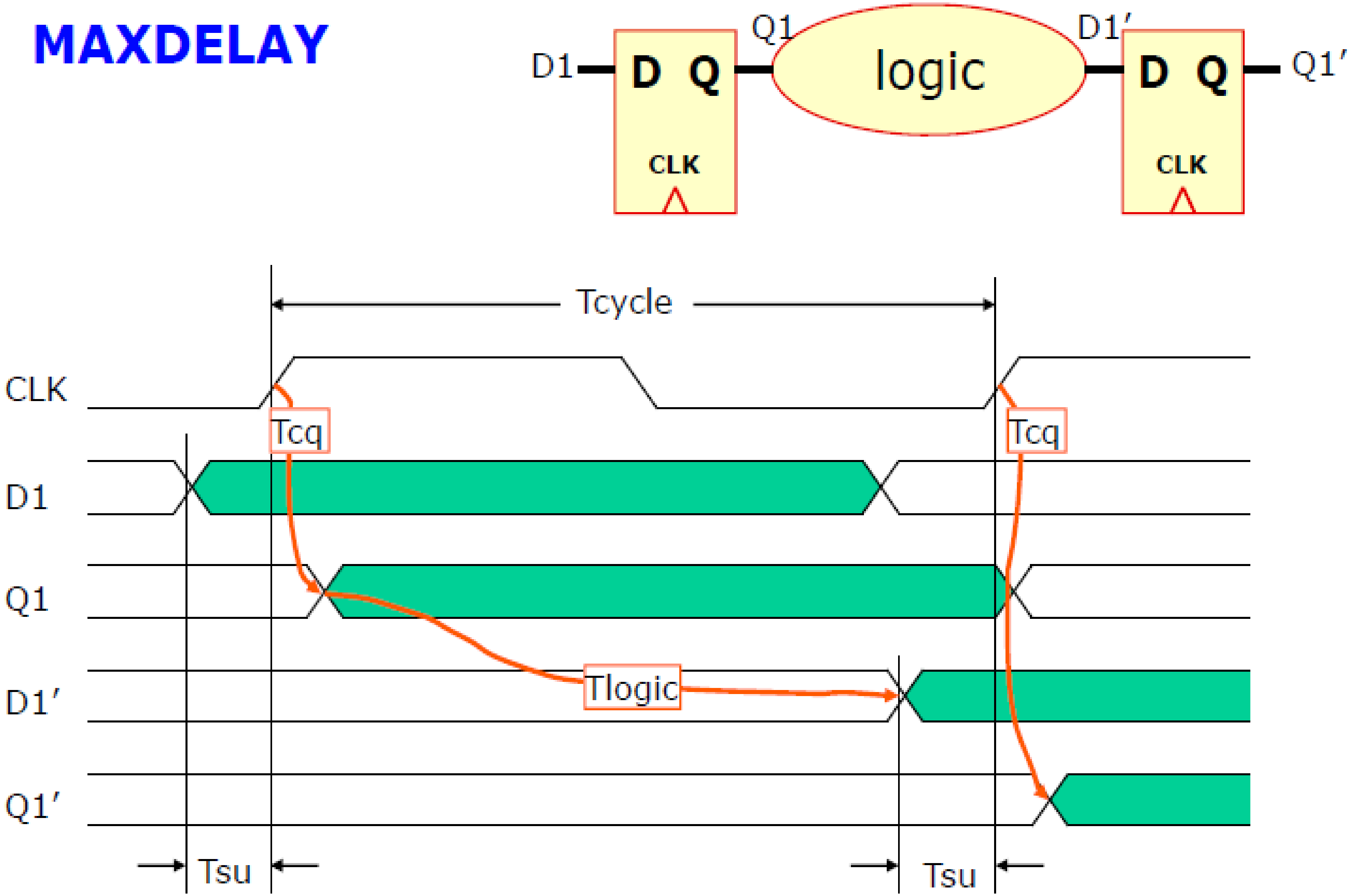


FLOP Timing Diagrams



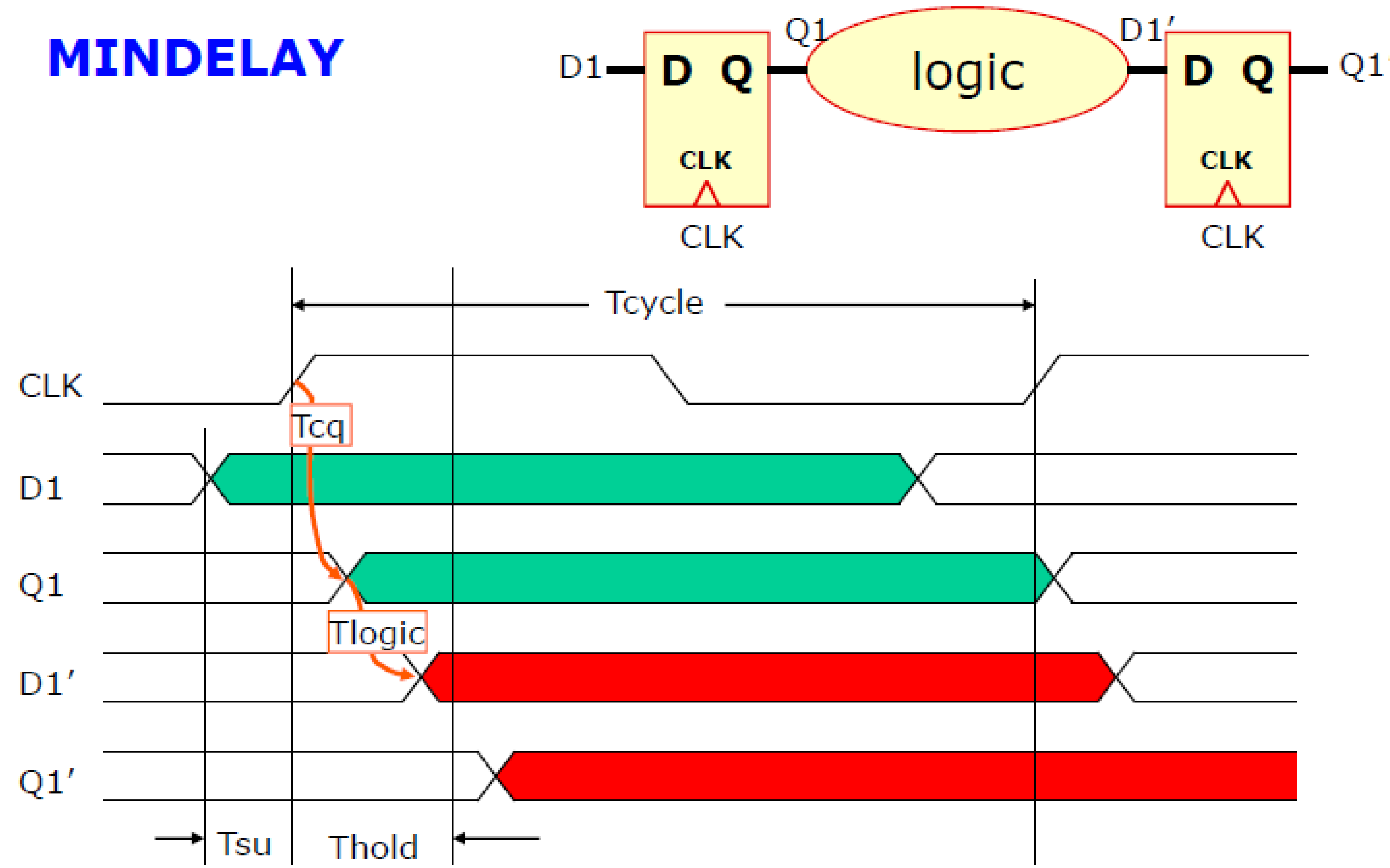
Tsu : input setup time
Thold : input hold time
Tcq : clock to out
Tdata to out = Tsu + Tcq

MAXDELAY



Tlogic < Tcycle – (Tcq + Tsu) or **Tcycle <= Tlogic + Tcq + Tsu**

MINDELAY



Tlogic > Thold – Tcq + Tskew

QZ

- What is this circuit ?