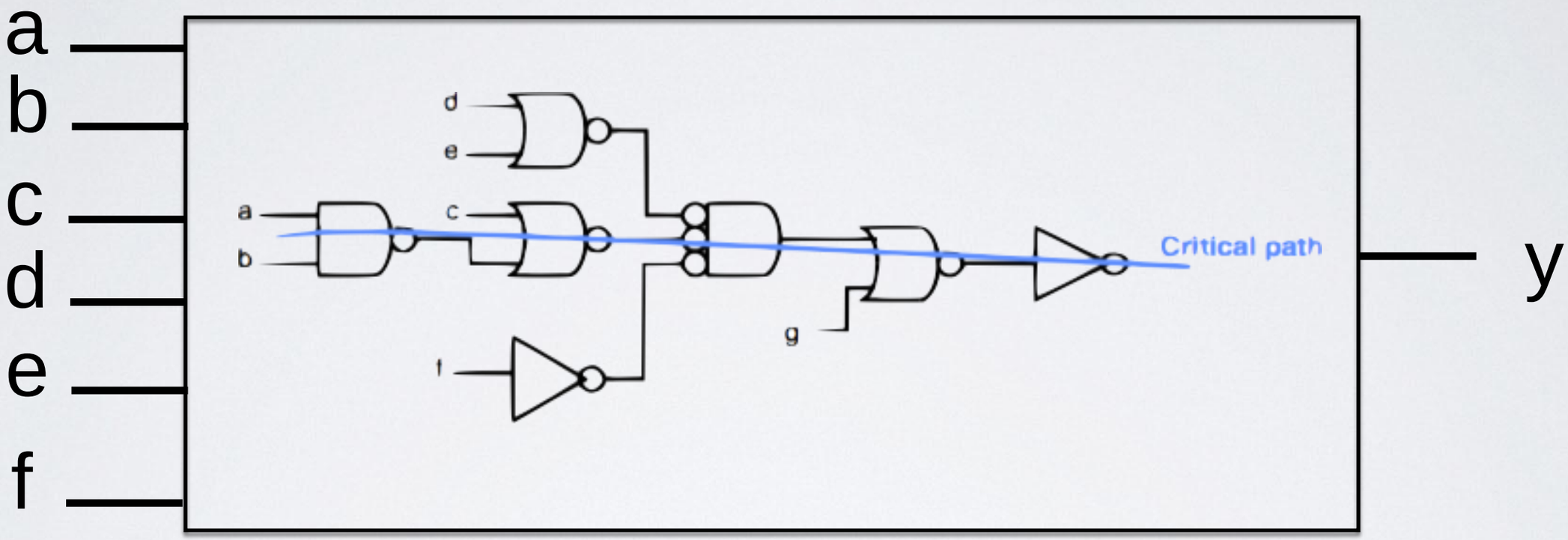
SEQUENTIAL CMOS DESIGN

Ronald Valenzuela (ronaldv@synopsys.com)
Corporate Application Engineer @ Synopsys

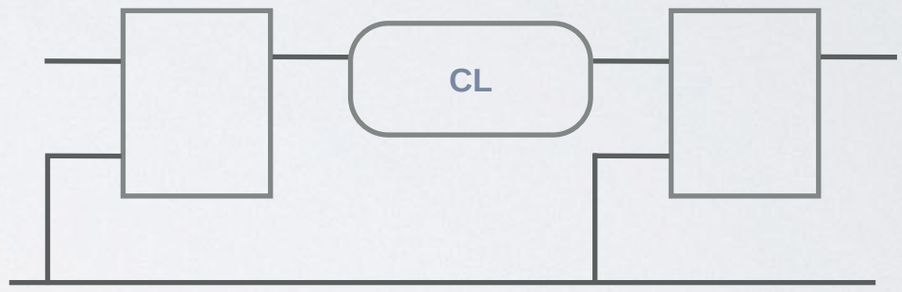
WE HAVE LONG PATHS

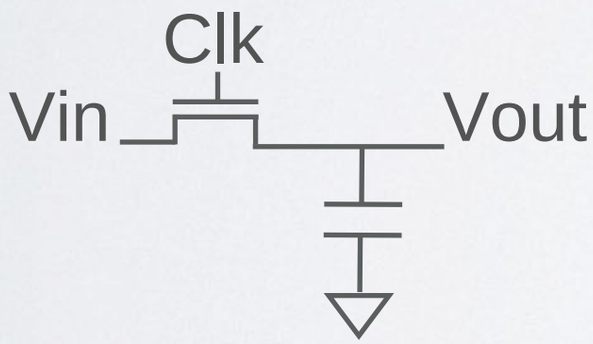


Most gates will be done changing way before y is finally computed.
By storing intermediate values, we can trade wasted area over time.
We still need similar amount of time to calculate a single output, but we can increase throughput.

STATE RETENTION

- To store a state, we store charge

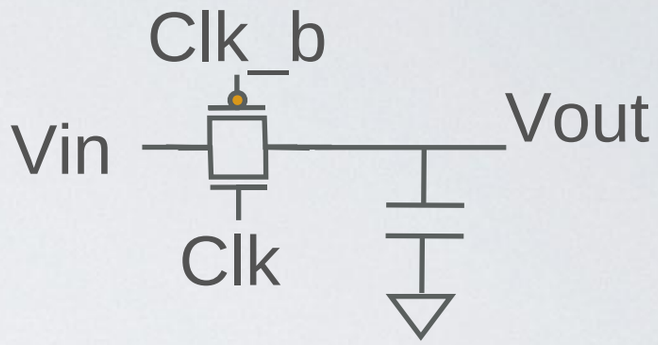


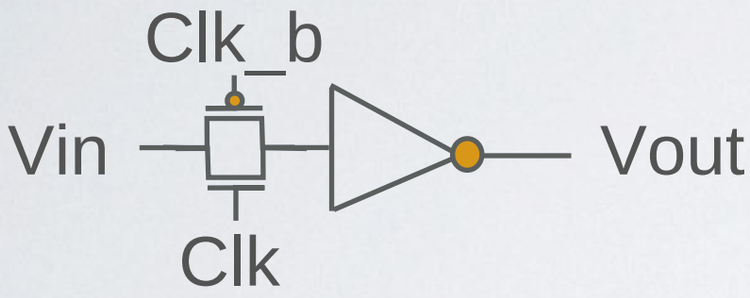


This is the simplest Latch you'll get, but violates every principle of a good gate.

TRADE COMPLEXITY FOR RELIABILITY

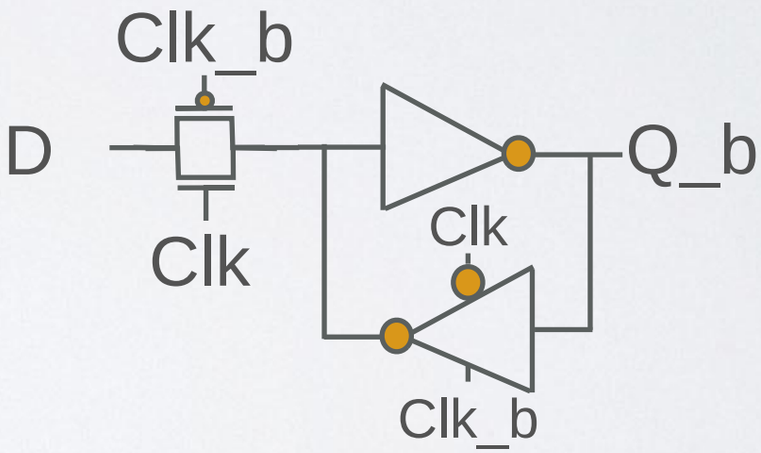
Rail to rail

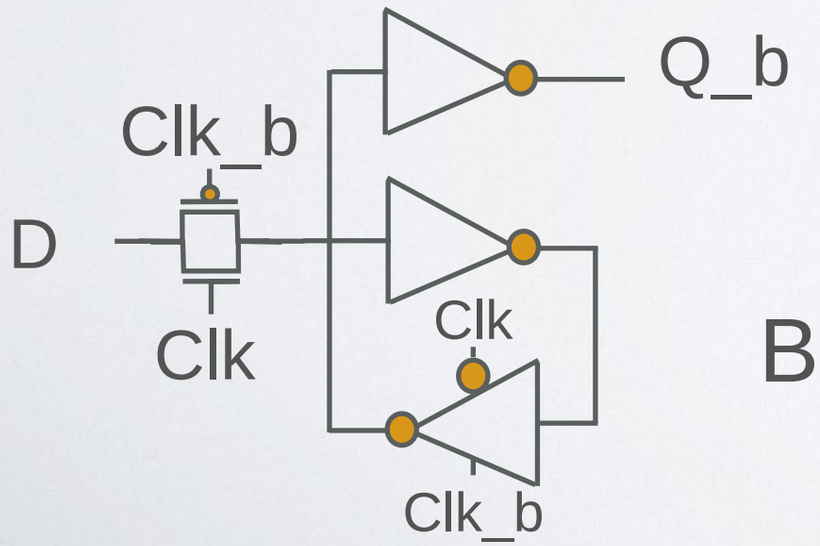




Unidirectional: State is isolated (Dynamic)

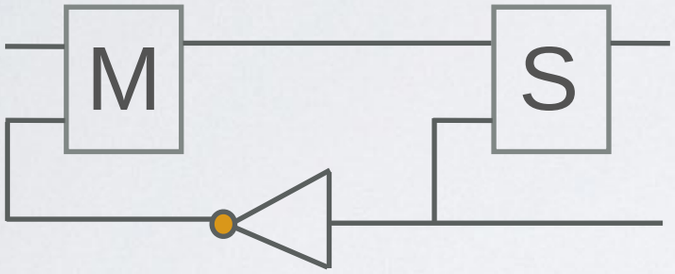
Balloon latch: Restore signal (static)

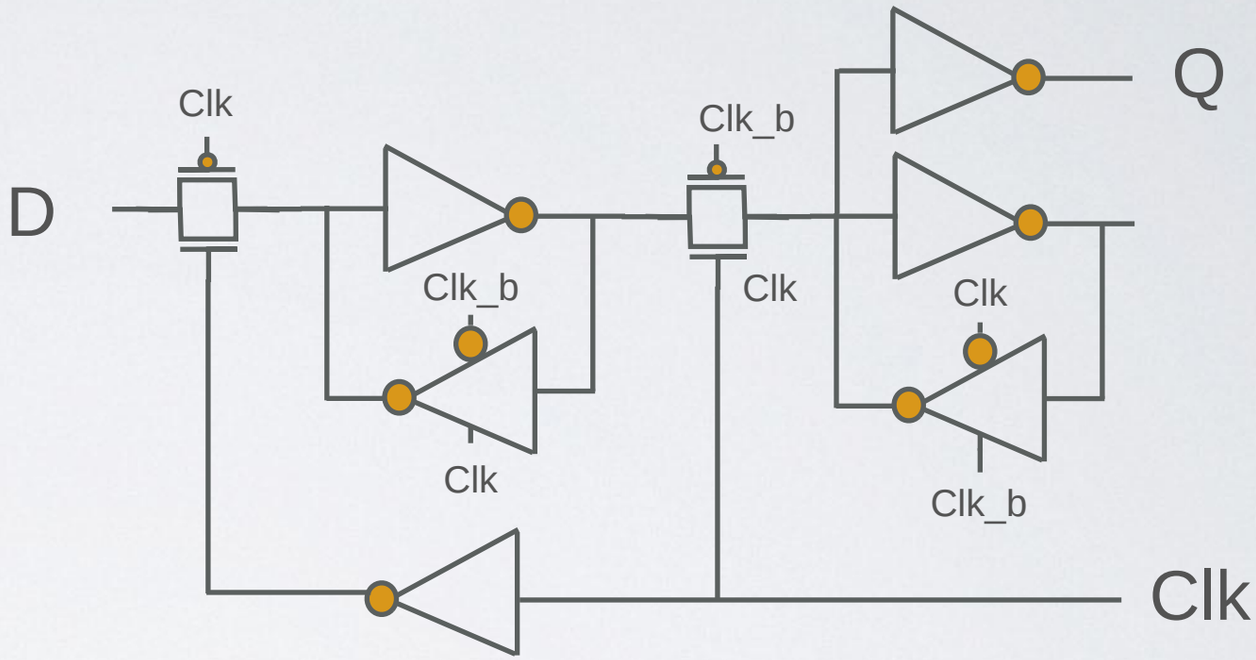




Balloon latch: Output isolation (static)

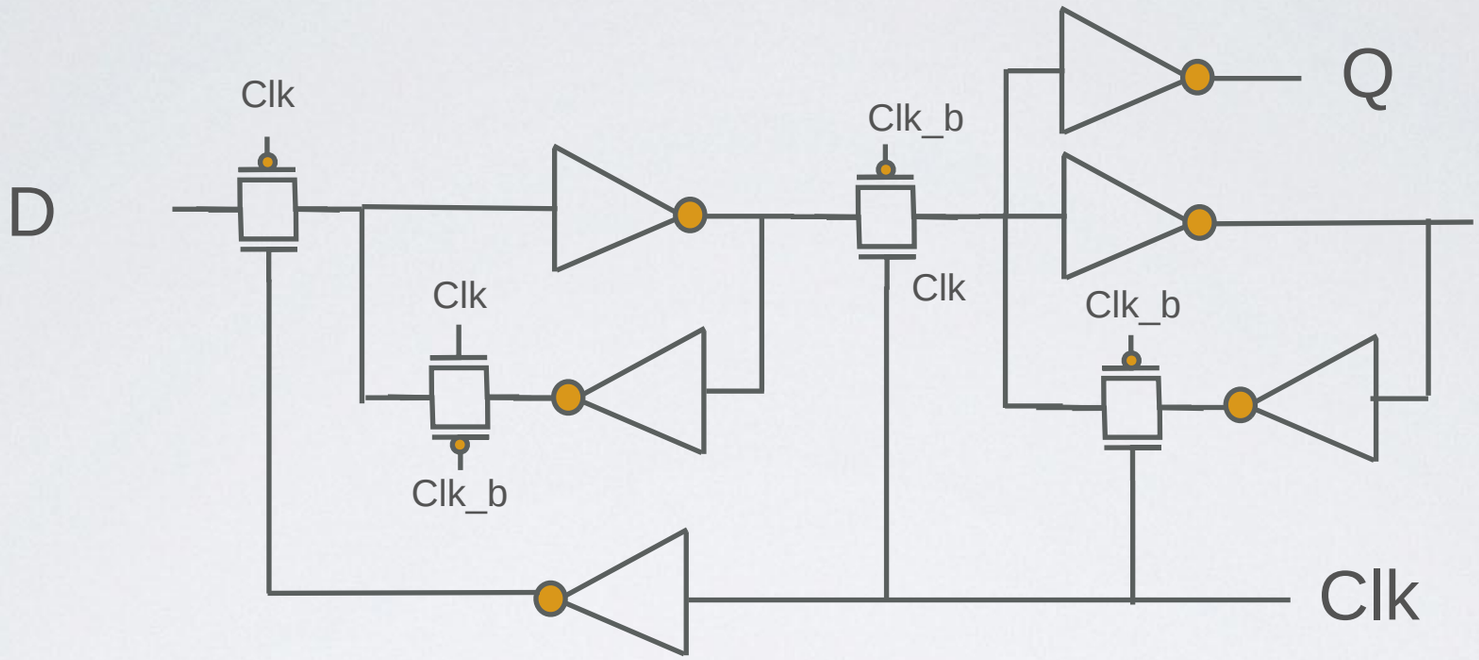
THE FLIP FLOP



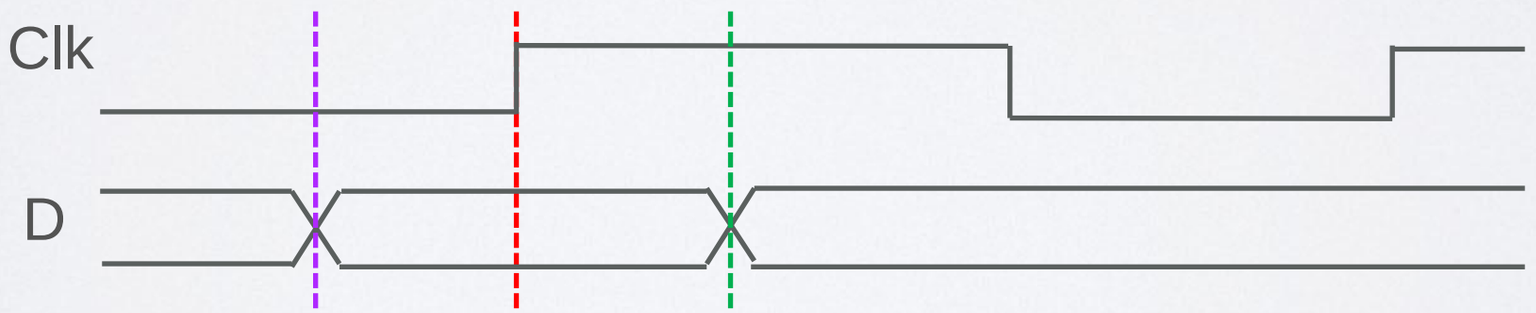


Clock	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**1**	**1**	**1**
D	0	1	2	3	4	5	6	7	8	9	A	B
MASTER LATCH	0	1	2	2	2	2	6	7	8	8	8	8
SLAVE LATCH	X	X	X	2	2	2	2	2	2	8	8	8

DATA PROPAGATION ON FLIP-FLOP



Just before and just after the clock edge, there is a critical time region where the D input must not change.

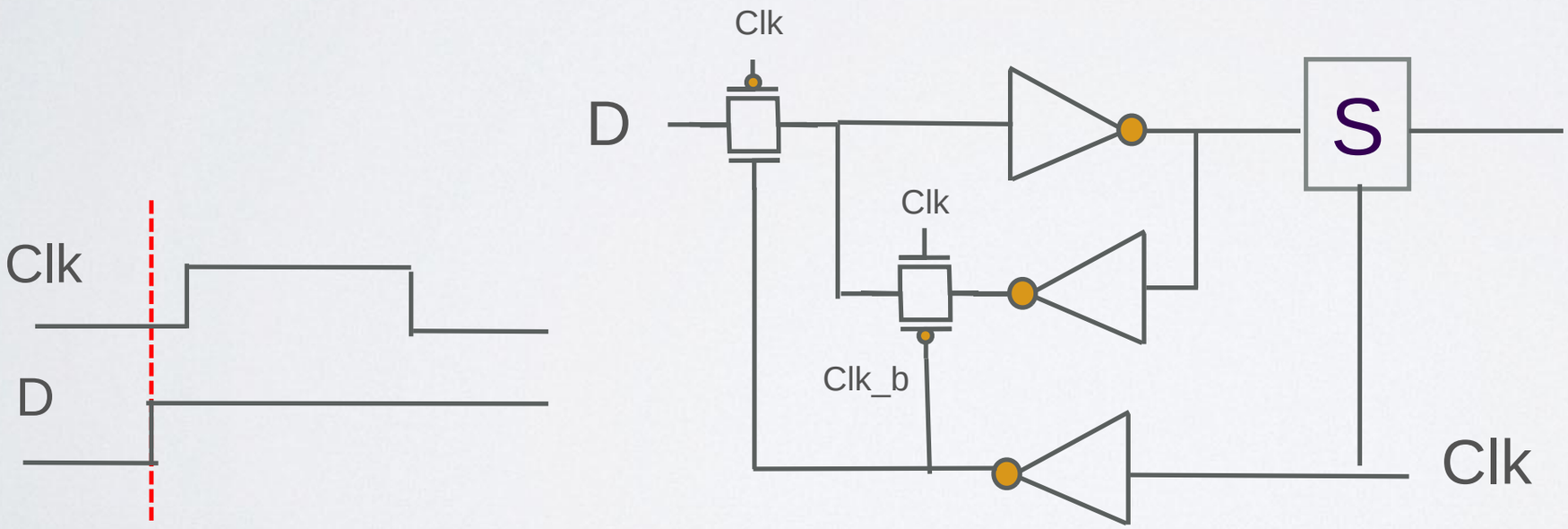


The region just before the clock edge is called setup time
The region just after the clock edge is called hold time

SETUP TIME

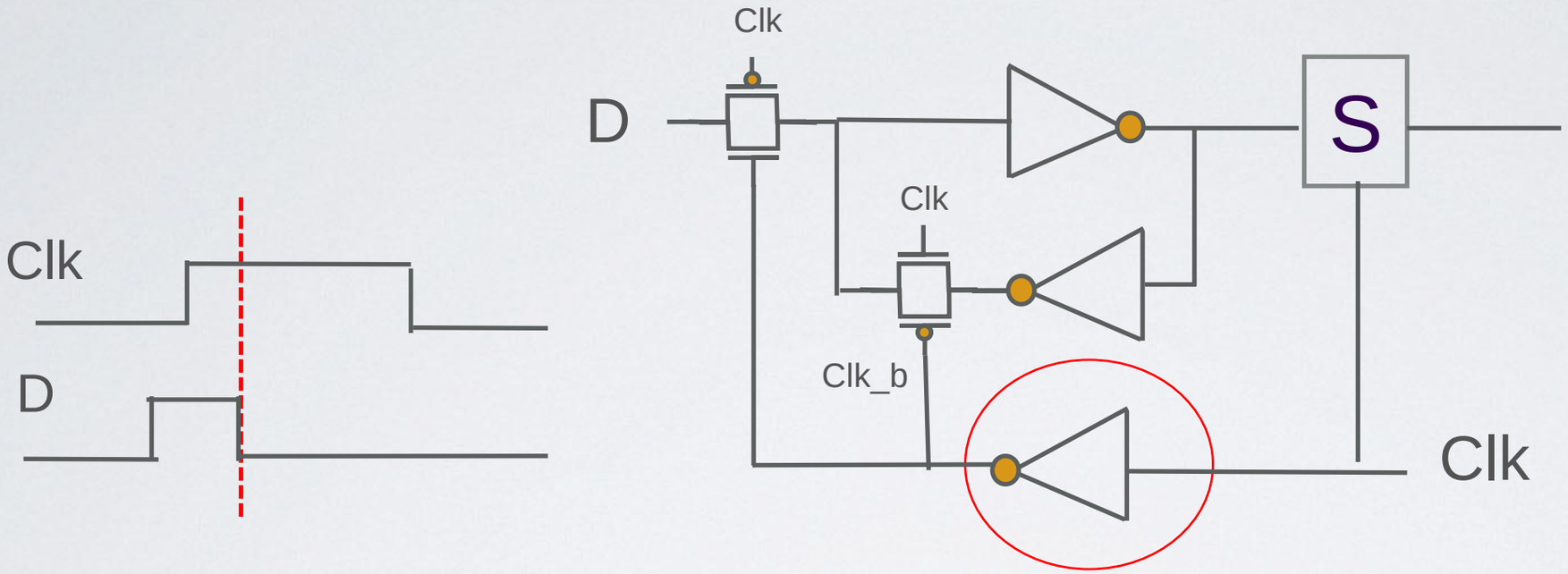
To understand setup time, let's make the following analysis.

1. Imagine that D is changing from 0 → 1 and the output of the latch has a value of 1. Input pass gate is active (Clk = 0)
2. To flip the master latch, data need to propagate all the way to the input of the feedback inverter and make it flip.
3. If the input pass gate isolate too early, the feedback inverter may win and the stored value won't flip *(metastability).*



Data must arrive some time EARLIER than clock edge

HOLD TIME



Note that Clk needs to inverted in order to trigger the isolation mode on the pass gate.

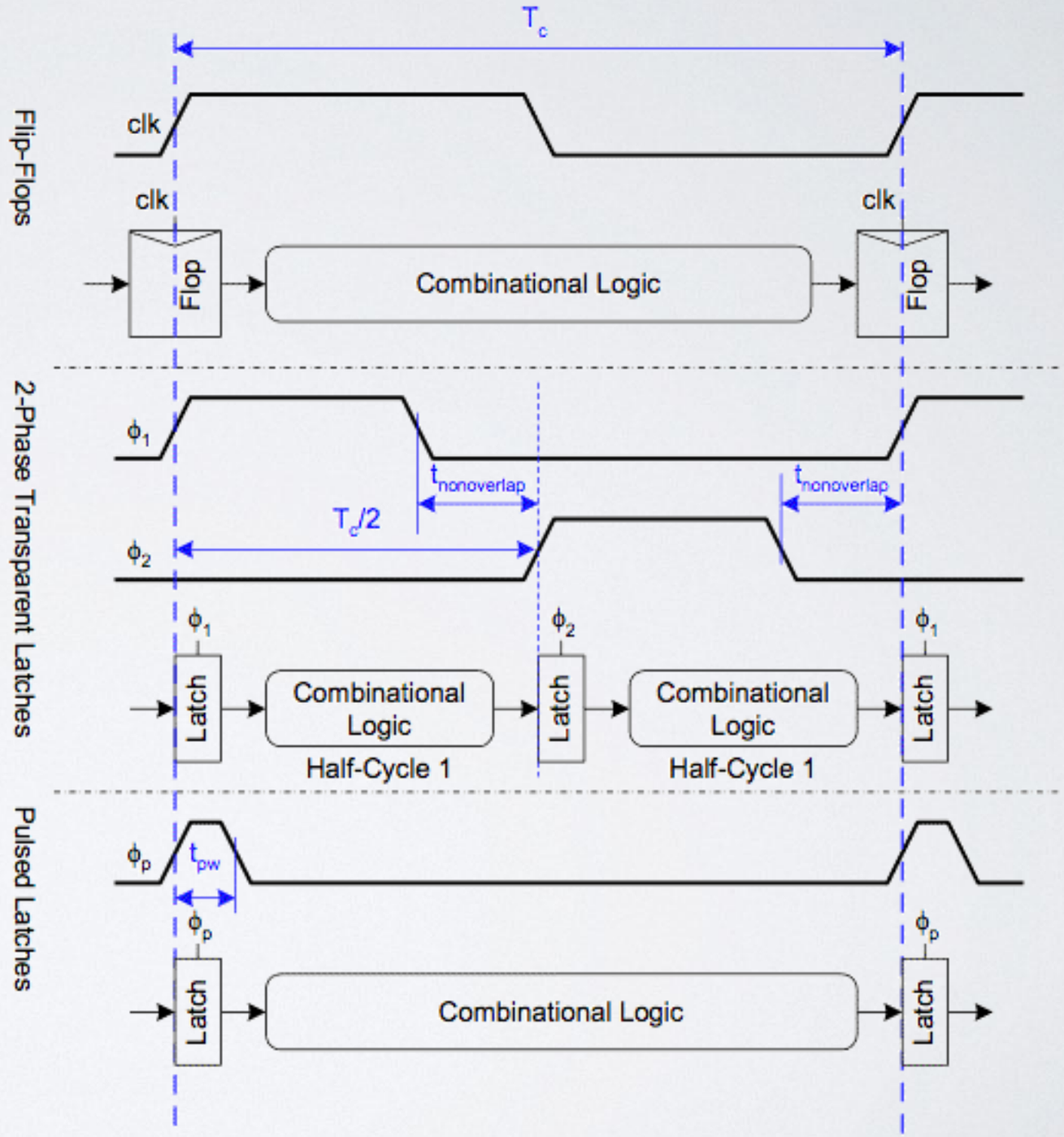
The inverter (in red) introduces a delay. Therefore, input pass gate will isolate after feedback pass gate starts to conduct

If D changes during this time, a new value may slip through (*metastability*)

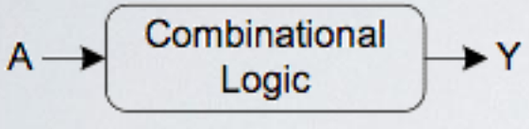
Data must stay PAST some time the clock edge

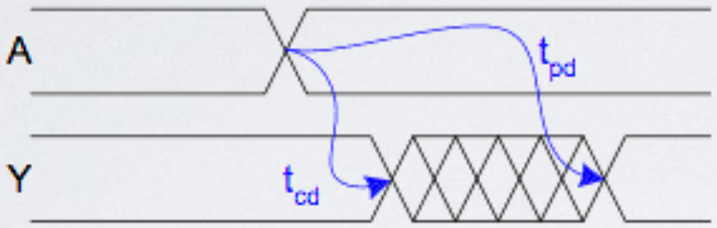
SEQUENCING METHODS

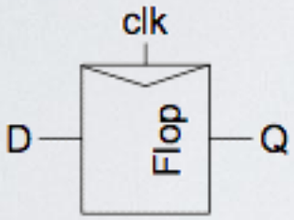
- Flip - Flop
- 2- Phase Latches
- Pulsed Latches

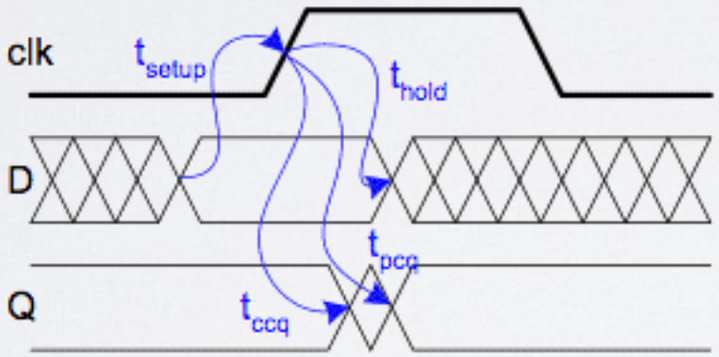


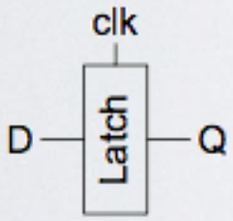
TO STUDY SEQUENTIAL CIRCUITS WE USE TIMING DIAGRAMS (MAX/MIN)

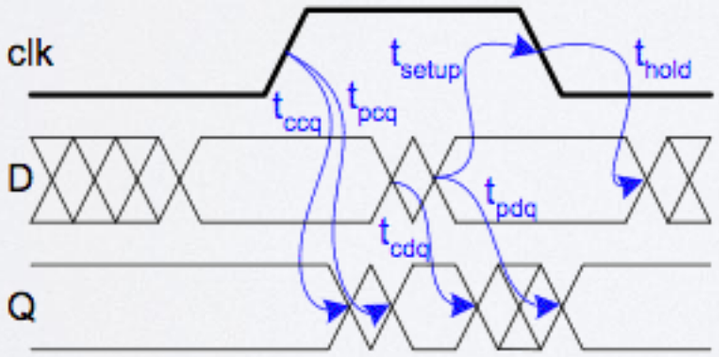






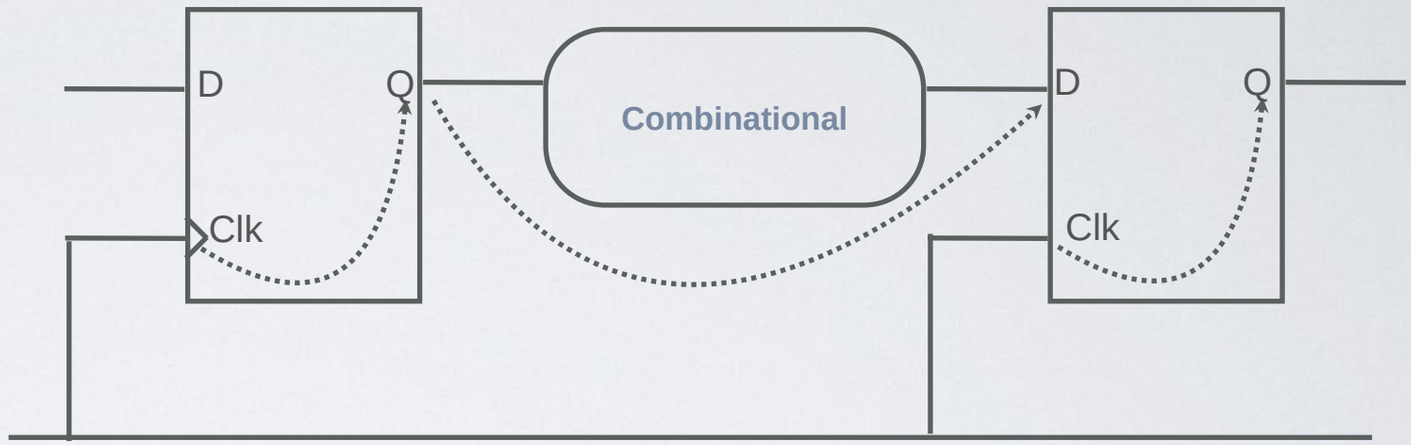






t_{pd}	Logic Propagation Delay
t_{cd}	Logic Contamination Delay
t_{pcq}	Latch/Flop Clk-Q Prop. Delay
t_{ccq}	Latch/Flop Clk-Q Cont. Delay
t_{pdq}	Latch D-Q Prop. Delay
t_{cdq}	Latch D-Q Cont. Delay
t_{setup}	Latch/Flop Setup Time
t_{hold}	Latch/Flop Hold Time

LOT OF TIMES: LETS RECAP



Time requirements:

- Data can’t take too long (Setup)
- Data can’t change faster than is captured (Hold)

Time overhead:

- Data propagates inside the sequential element

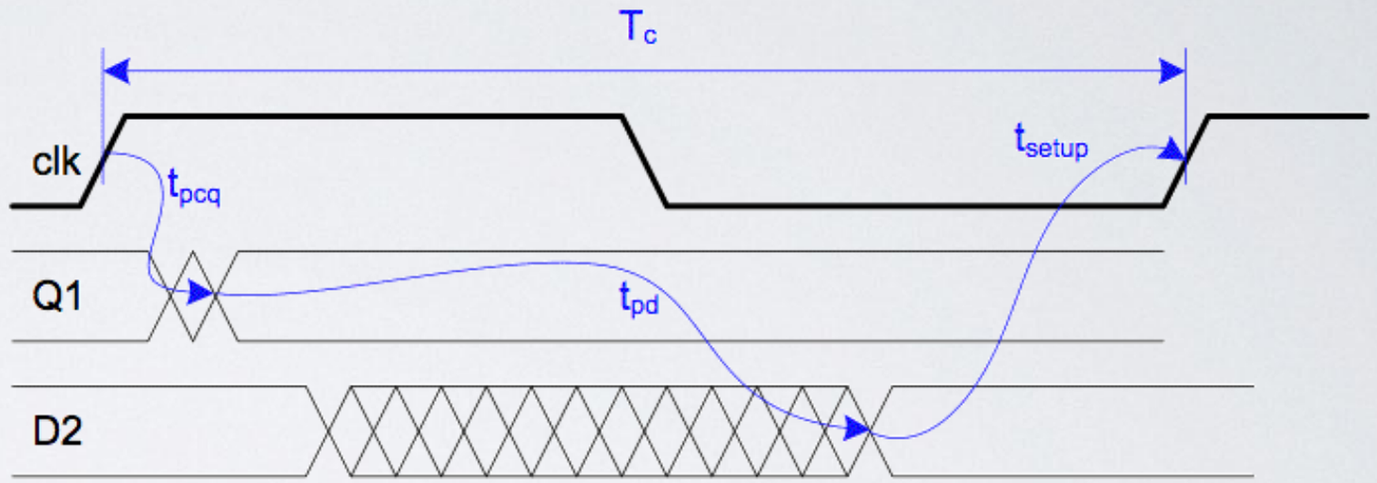
Time Definitions:

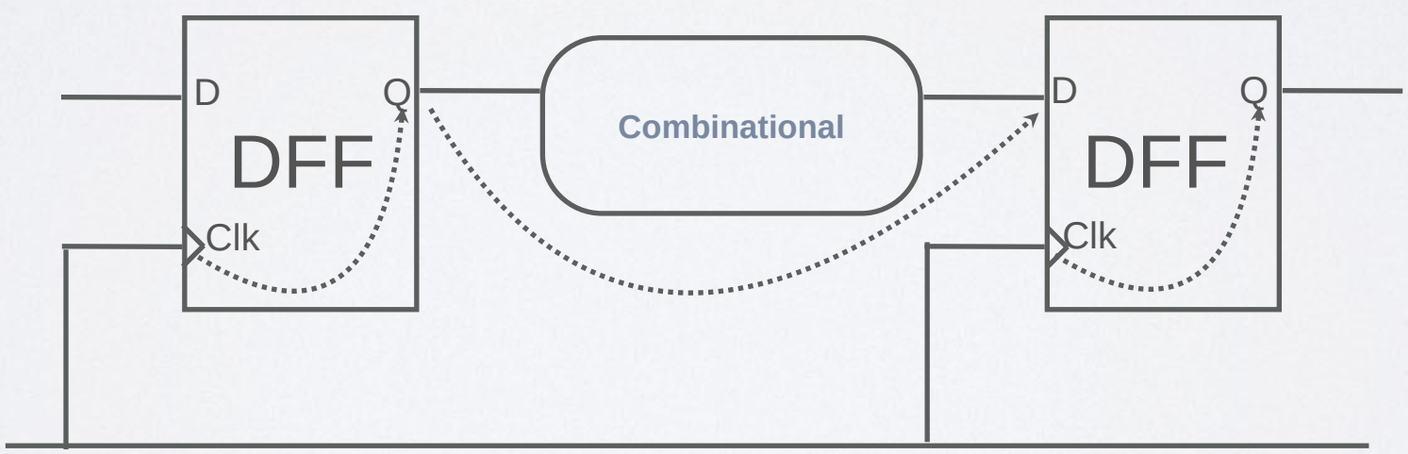
- Some times we need a value not to change, some times we need the actual value to arrive (cont. / prop.)

TIMING CONSTRAINTS: MAX DELAY

Flip-Flop

$$t_{pd} + t_{pcq} \leq T_c - t_{setup}$$





*Overhead: (t_{pcq} + t_{setup})

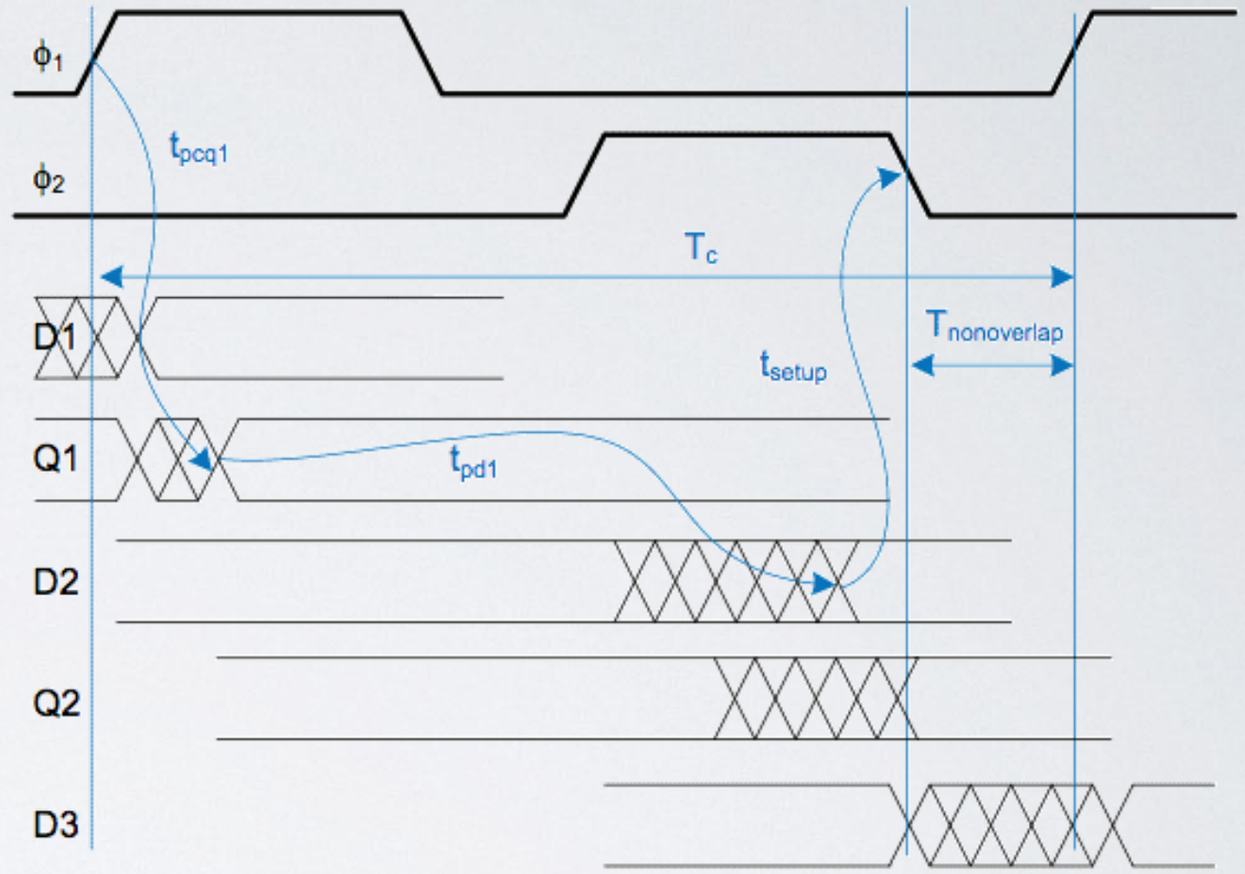
TIMING CONSTRAINTS: MAX DELAY

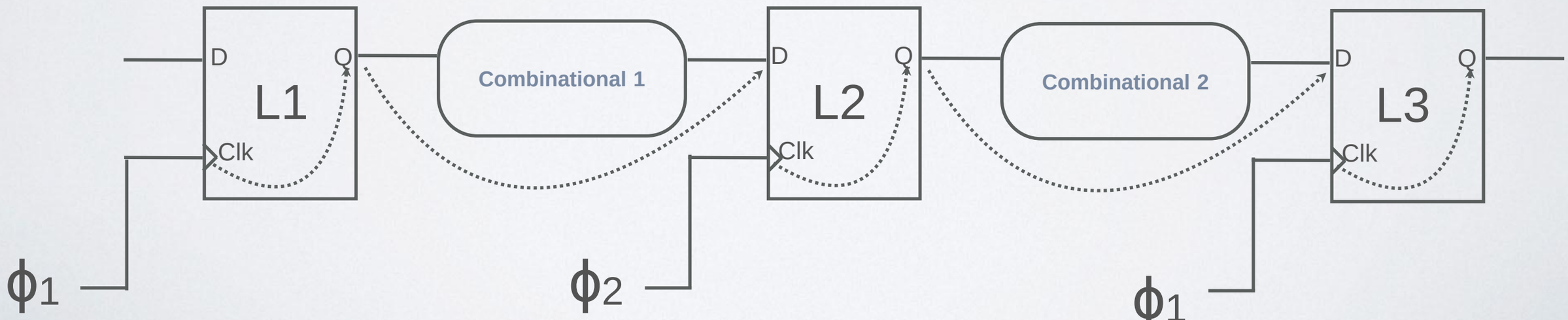
2 Phase Latches

$$t_{pd} = t_{pd1}+t_{pd2} \leq T_c - (2t_{pdq})$$

$$t_{pd1} \leq T_c - t_{nonoverlap} - t_{setup} - t_{pcq}$$

$$t_{pd2} \leq T_c - t_{nonoverlap} - t_{setup} - t_{pcq}$$

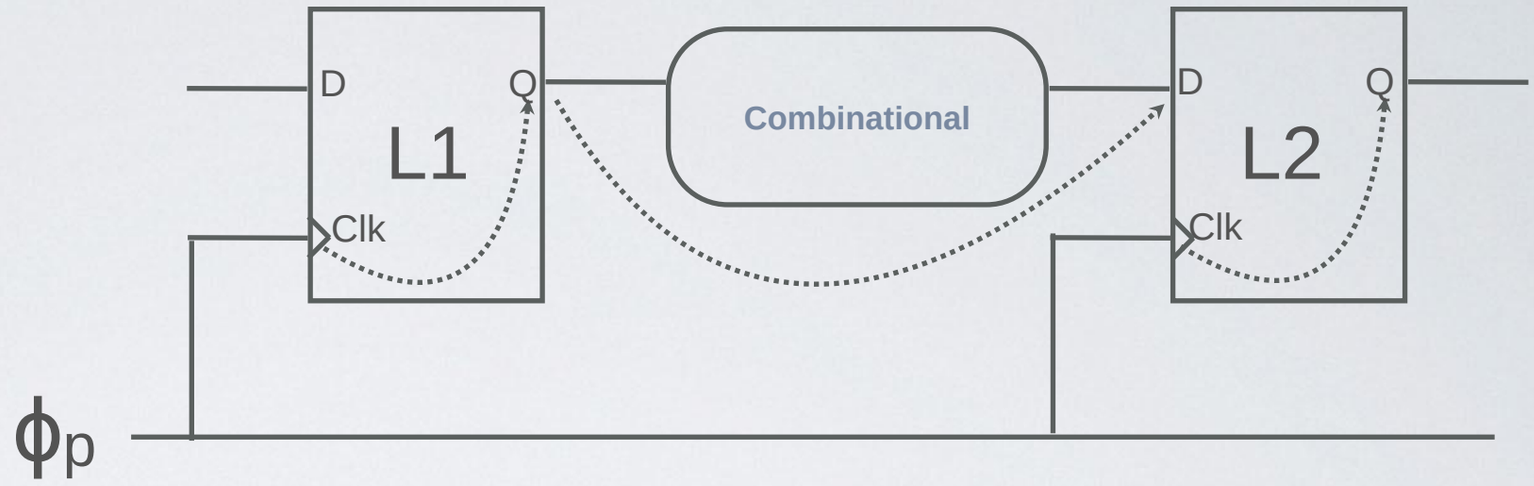


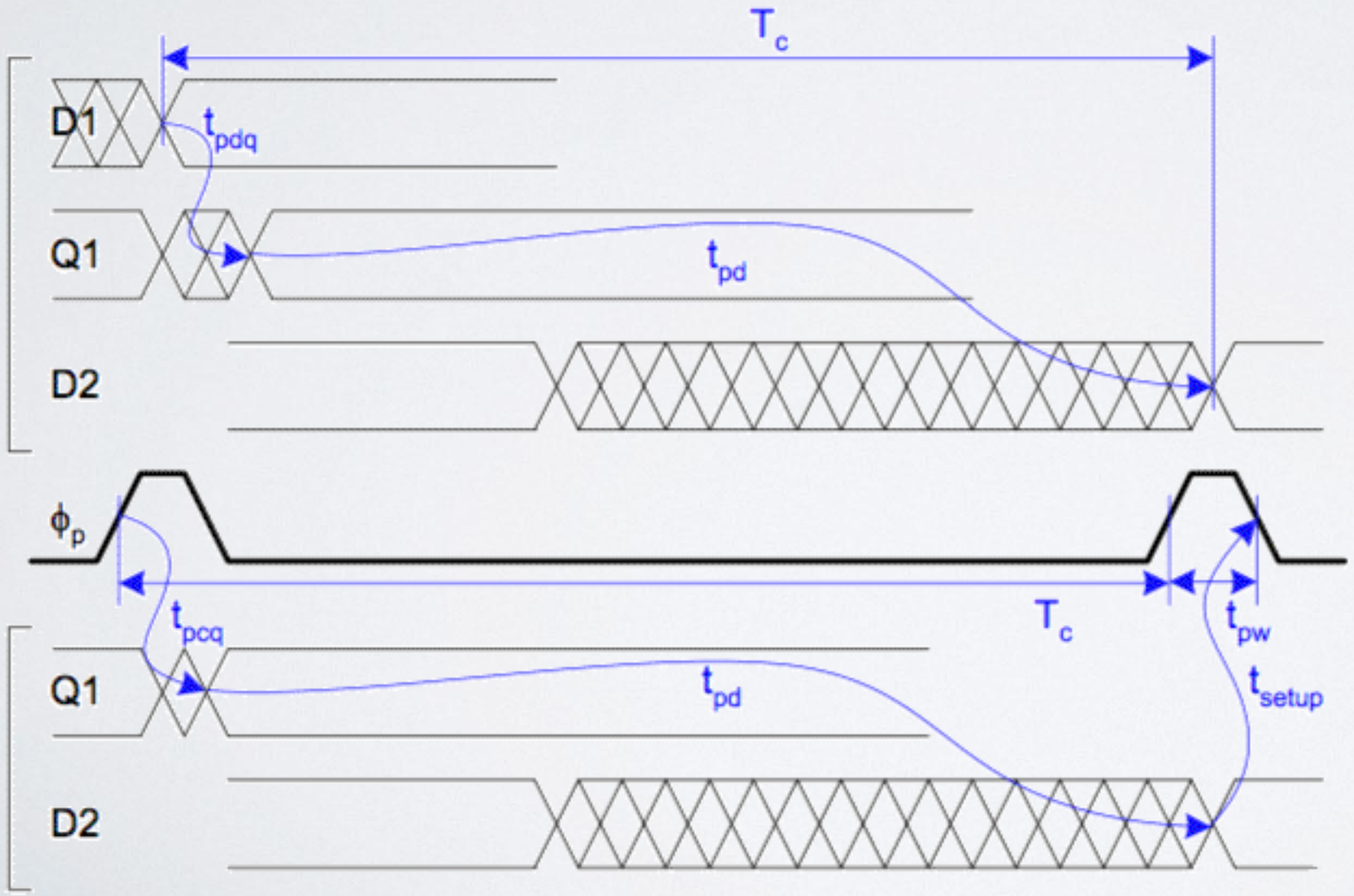


*Overhead: $(2t_{pdq})$

TIMING CONSTRAINTS: MAX DELAY

Pulsed Latch





$\max(d_1,d_2) \leq Tc$

$d_1 = t_{pdq} + t_{pd}$

$d_2 = t_{pcq} + t_{pd} + t_{setup} - t_{pw}$

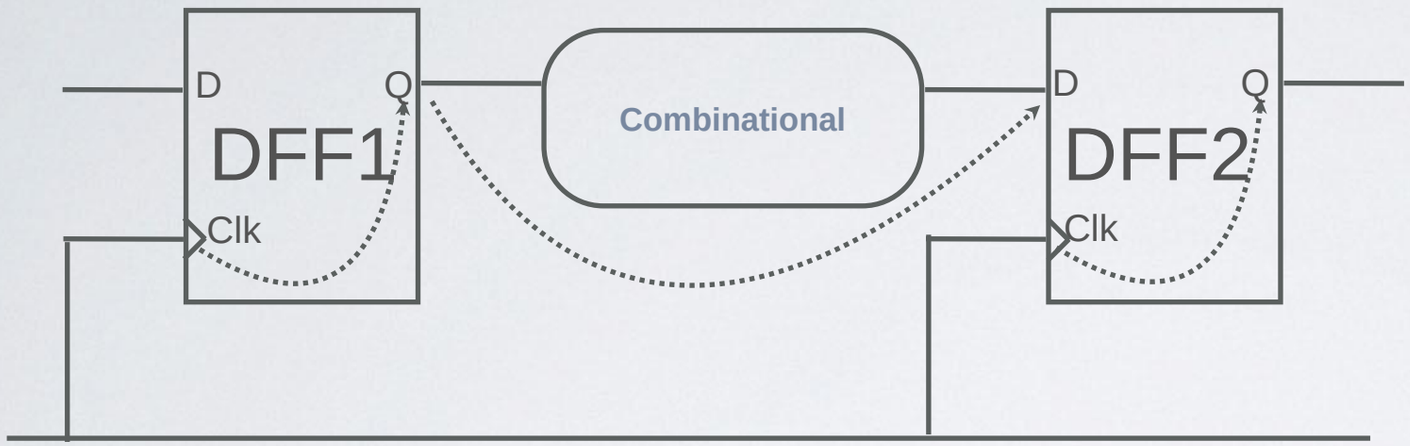
* Overhead:

d1: tpdq

d2: $t_{pcq} + t_{setup} - t_{pw}$

TIMING CONSTRAINTS: MIN DELAY

Flip-Flop

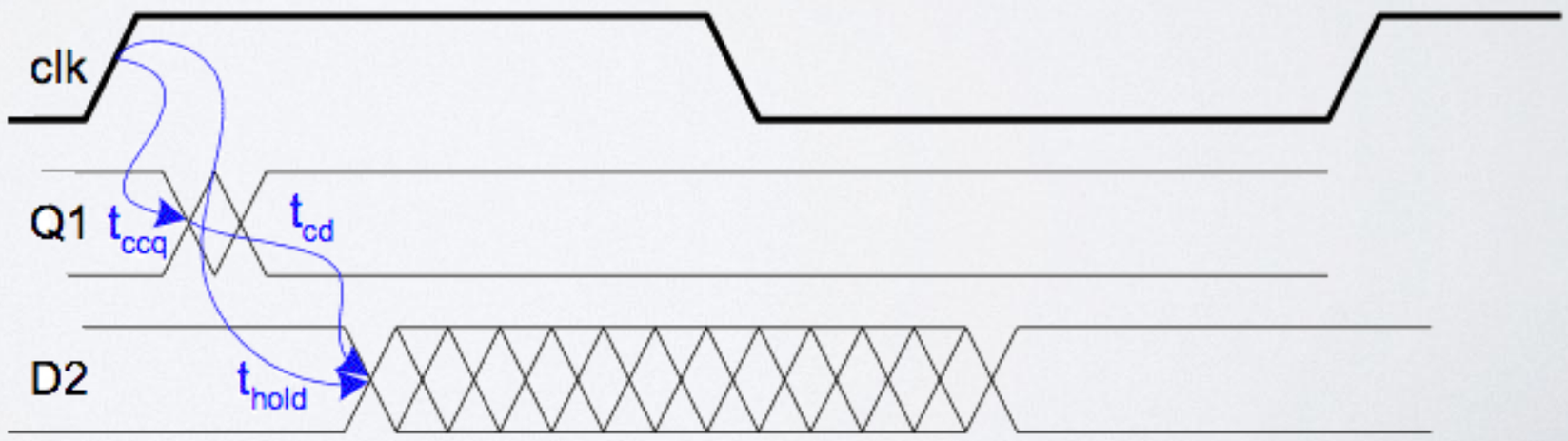


For Hold you think on the launching flop, it should not cause contamination at node DFF2/D before t_{hold}.

A MAX violation can be solved after tape out by increasing Tc.

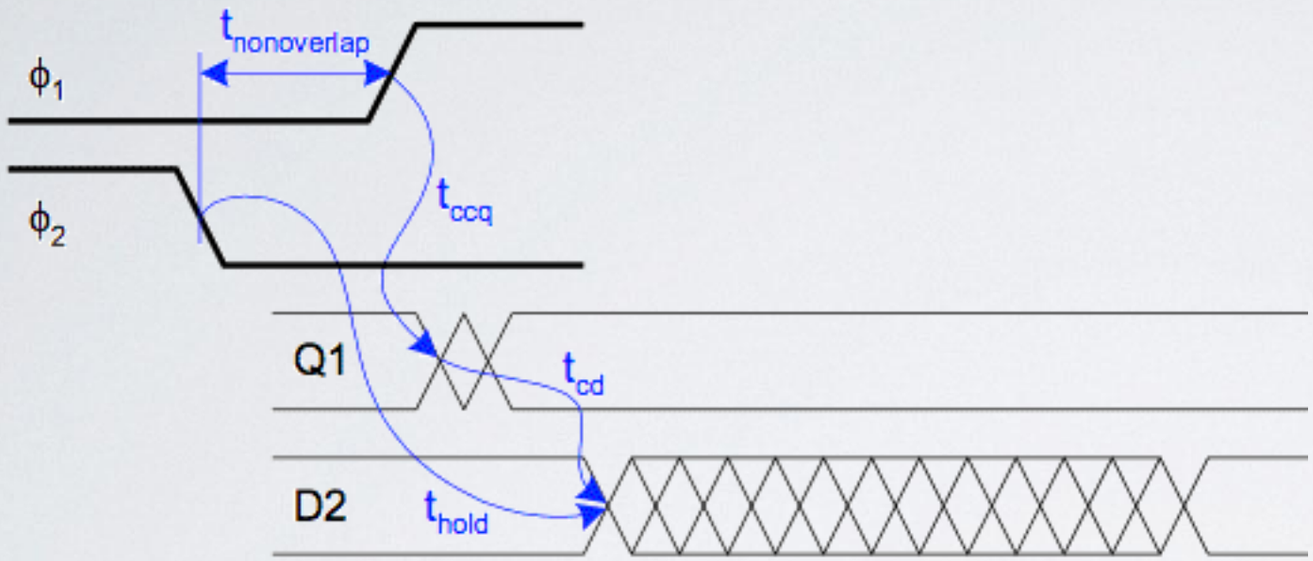
Note that Tc is not a knob of the MIN constraint, this mean it can't be solved after tape out.

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



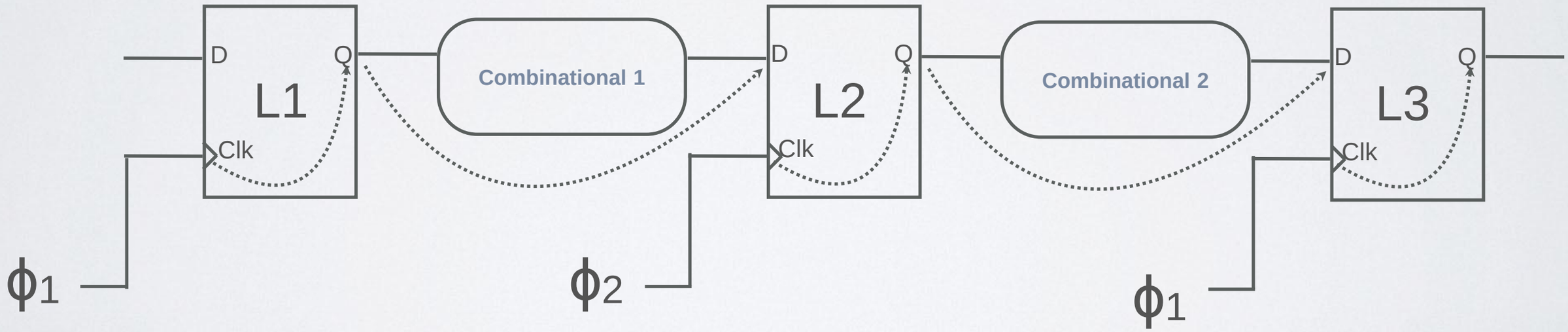
TIMING CONSTRAINTS: MIN DELAY

2 - phase latches



Now ϕ_2 needs to close before ϕ_1 corrupts the signal.

Typically ϕ_2 is a function of ϕ_1

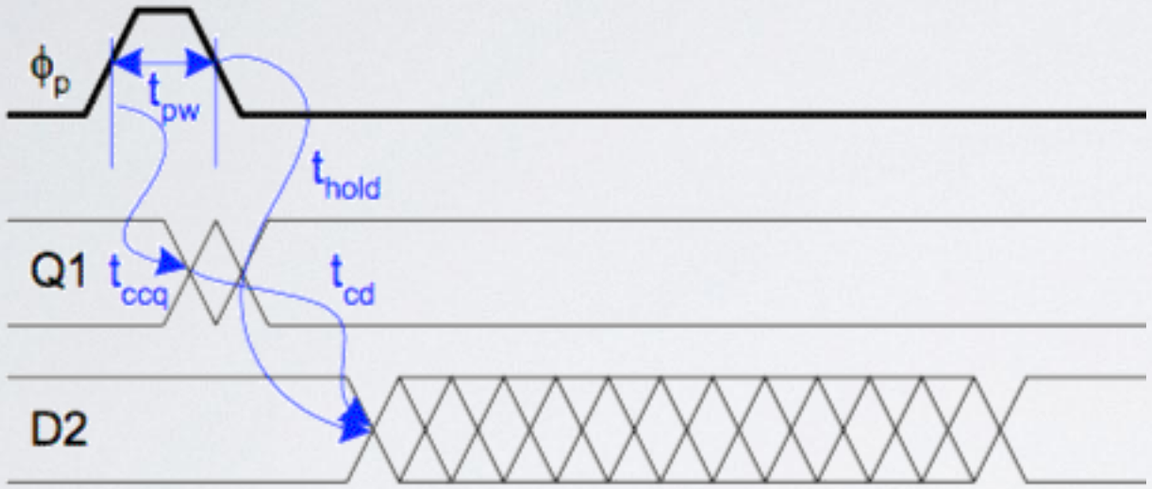


$$(t_{cd1}, t_{cd2}) + t_{ccq} + t_{nonoverlap} \geq t_{hold}$$

<u>Can Hold be fixed after tape out now?</u>

TIMING CONSTRAINTS: MIN DELAY

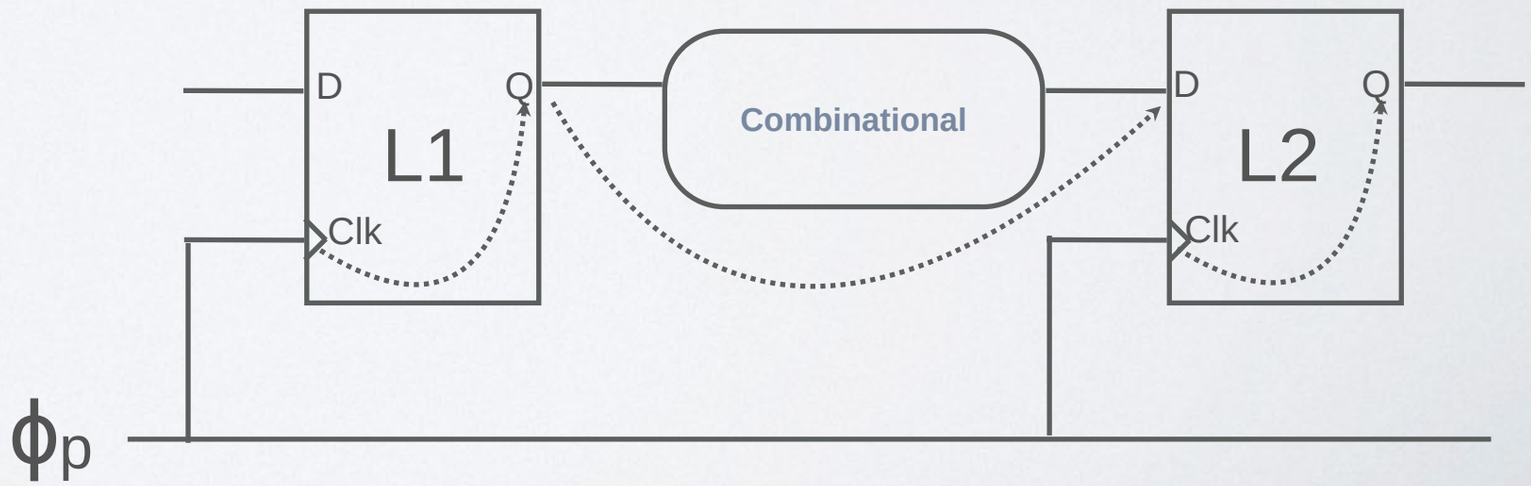
Pulsed latches



Here positive edge triggers contamination, while falling edge is the closing event.

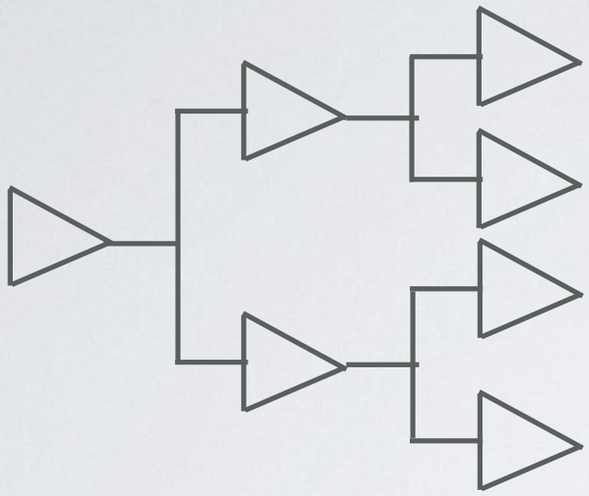
Pulse is typically narrow, can be generated as a glitch like signal inside the latch

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

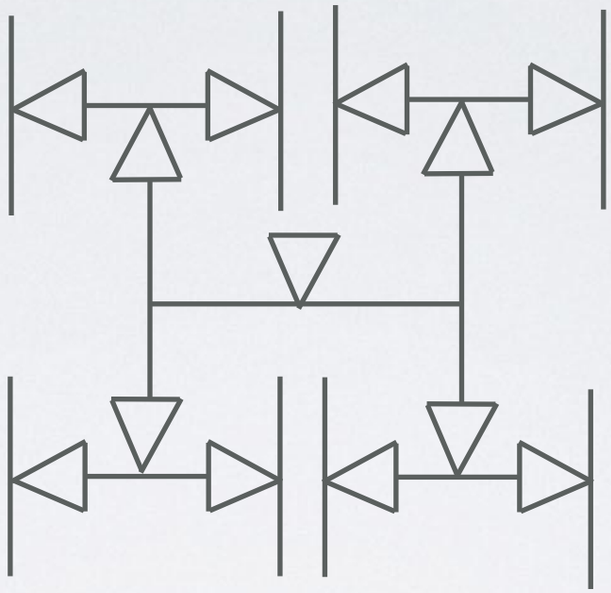


CLOCK DISTRIBUTION

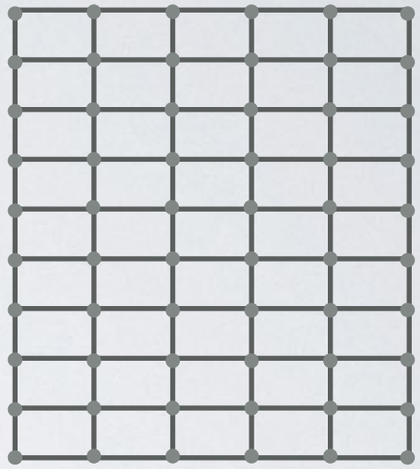
Tree



H-Tree



Grid



- We wish the clock to arrive to every point at the same time
- May want to stop the clock feeding a bank of registers (clock gating)
- Can combine to get the best balance/power trade off

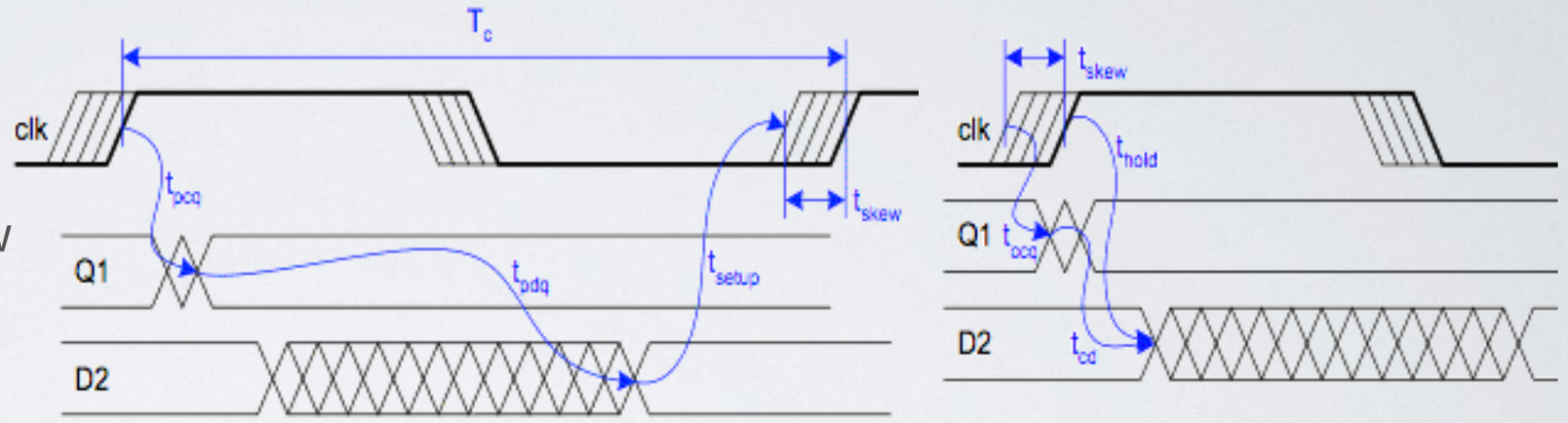
The time difference between the actual arrival times of a clock at its sinks is called **clock skew**

SKEW IMPACT ON CONSTRAINTS

- Flip - Flop

$t_{pd} + t_{pcq} \leq T_c - t_{setup} - t_{skew}$

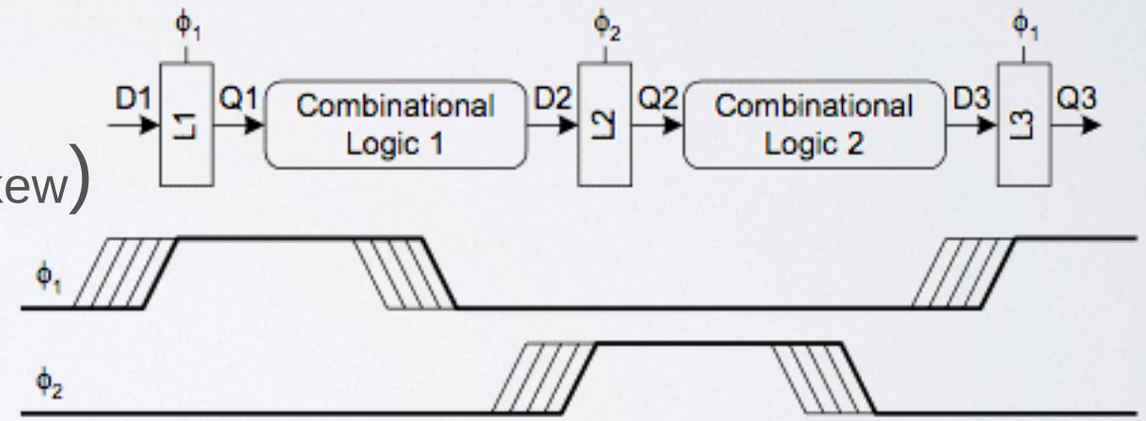
$t_{cd} + t_{ccq} \geq t_{hold} + t_{skew}$



- 2- Phase Latches

$t_{borrow} \leq 0.5*T_c - (t_{nonoverlap} + t_{setup} + t_{skew})$

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



- Pulsed Latches

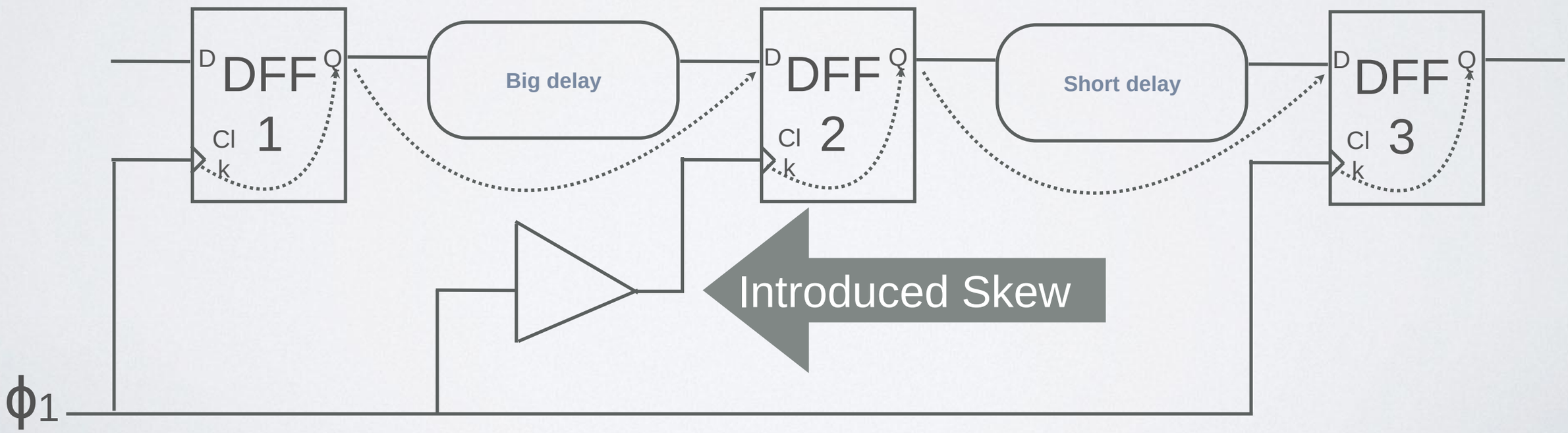
$t_{cd} \geq t_{hold} + t_{pw} - t_{ccq} + t_{skew}$

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

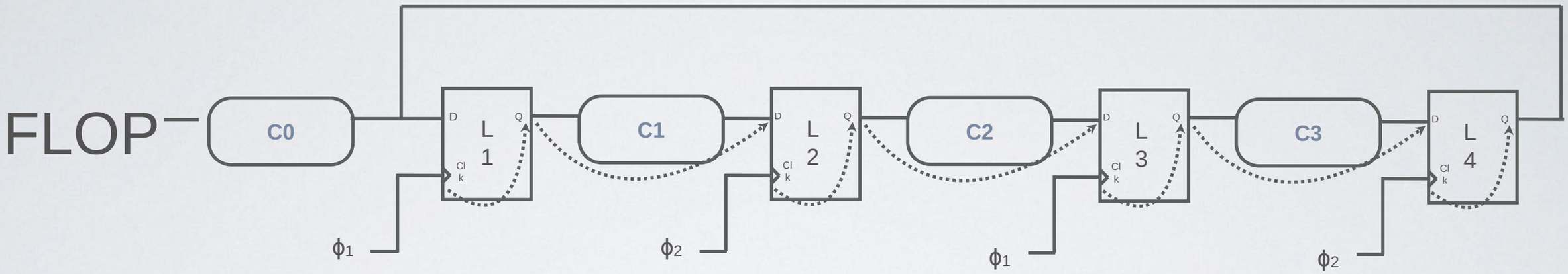
As we can predict from Murphy's law.
…skew makes all constraints worst

GOOD SKEW

Manage skew to balance delay



TIME BORROWING



Stage	Min Check	Loop Slack	Stage Slack
TB0	0+550-500	-50	-50
TB1	50+580-500	-130	-80
TB2	130+450-500	-80	50
TB3	80+200-500	220	300

Here we relax the constraint that each path needs to fit in half a cycle, but overall still need to fit on T

Still: $T_{borrow} \leq T/2 - (t_{nonoverlap} + t_{setup})$

Assume no skew, setup, PCQ and PDQ are included in C#, then:

T = 1000ps

C0 = 550ps

C1 =580ps

C2 = 450ps

C3 = 200ps

STATIC TIMING ANALYSIS

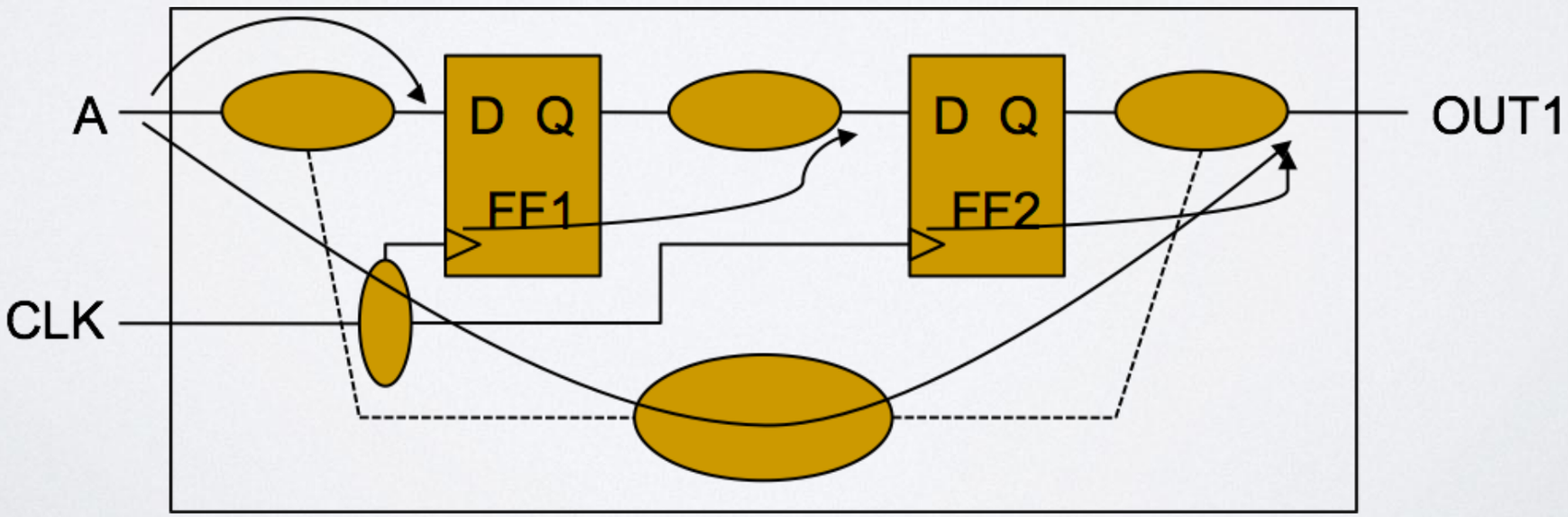
Method of computing the expected timing of a digital circuit without requiring simulation

GENERAL STEPS OF STA

- Three general steps in Static Timing Analysis
  - Circuit is broken down into sets of timing paths
  - Delay of each path is calculated
  - Path delays are checked to make sure timing constraints have been met

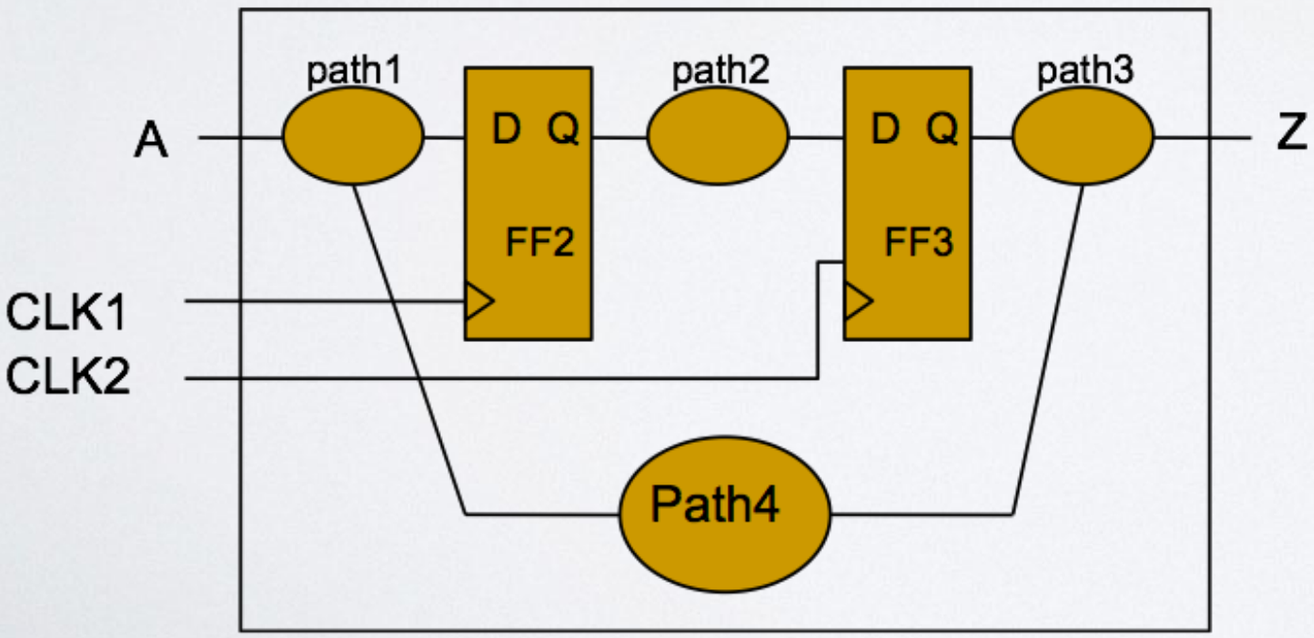
DEFINITION OF “TIMING PATH”

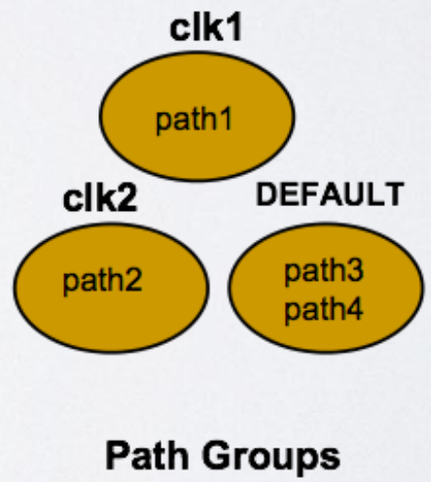
- A “Timing Path” is a point-to-point path in a design which can propagate data from one flip-flop to another
- Each path has a start point and an endpoint
- Start point:
  - Input ports, clock pins of flip-flops
- Endpoints:
  - Output ports, data input pins of flip-flops



GROUPED TIMING PATHS

- Timing paths are grouped into path groups by the clocks controlling their endpoints
- STA tools like PimeTime generates timing reports by path groups

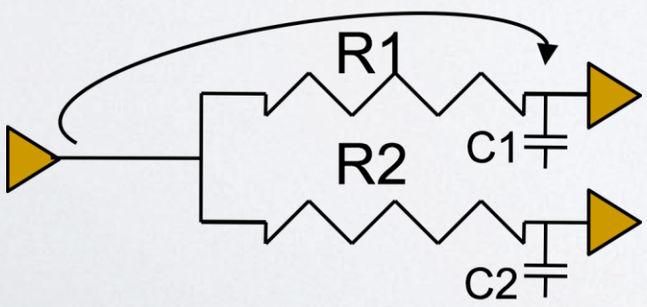




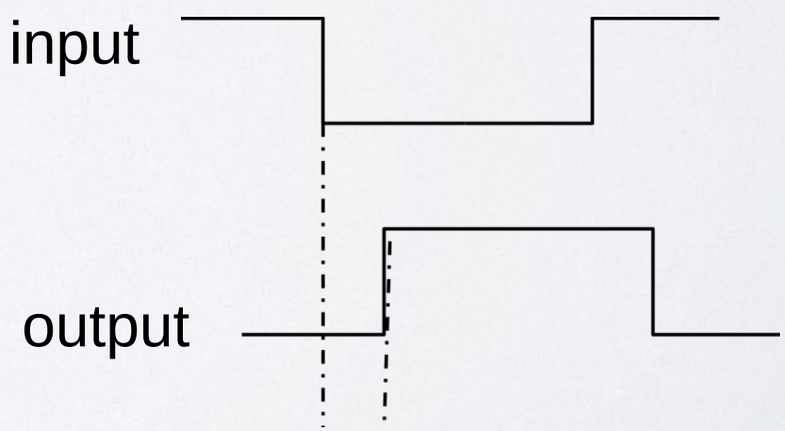
TIMING PATH DELAY

The path delay is the sum of net and cell delays along the timing path

Net Delay is the total time which needs to charge or discharge all parasitic capacitances of a given net

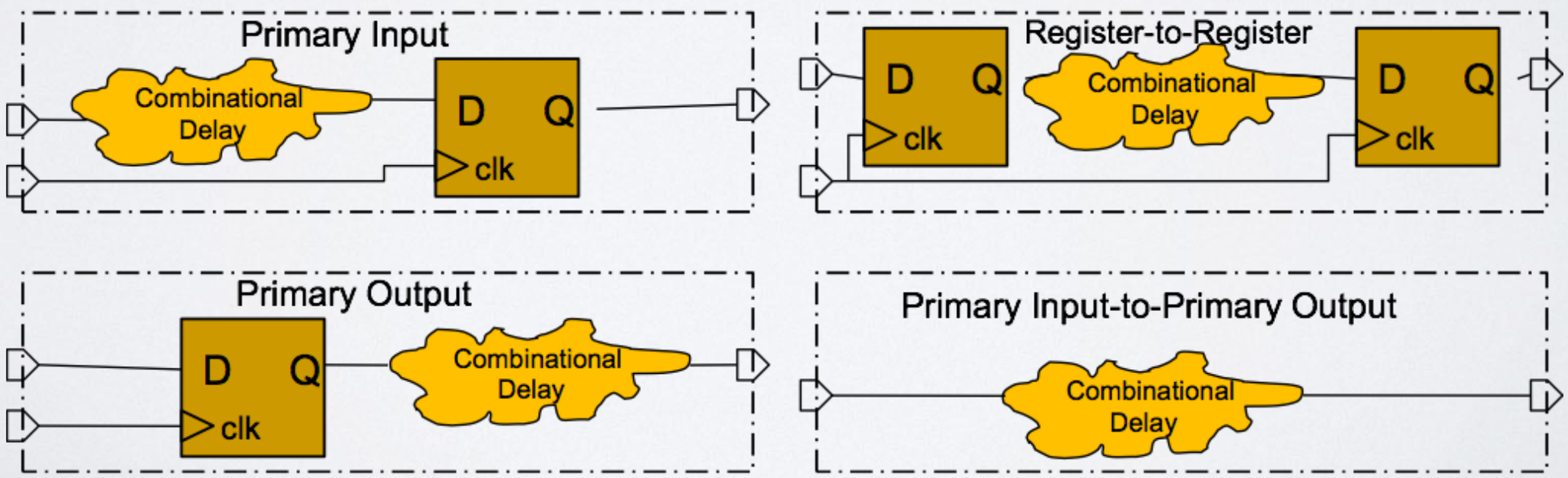


Cell Delay shows a timing difference between output change and input change



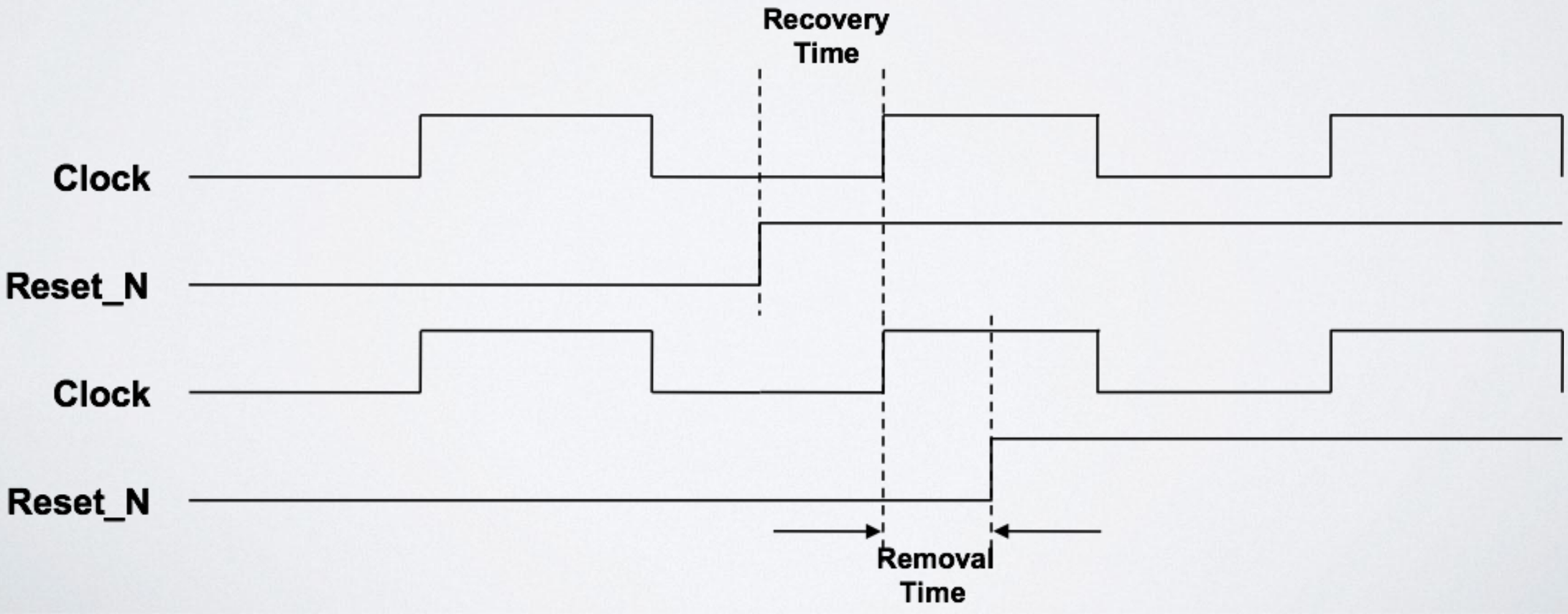
SLACK CALCULATION

- Primary input to register path
  - Delays of input nets and combinational logic, up to the first sequential device
- Register to primary output paths
  - Start at a sequential device and CLK to Q transition delay + the combinational logic delay + external delays
- Register-to register paths
  - Delay and setup/hold times between sequential devices for synchronous clocks + source and destination clock propagation times
- Primary input to primary output paths
  - Delays of input nets + combinational logic delays + external delay



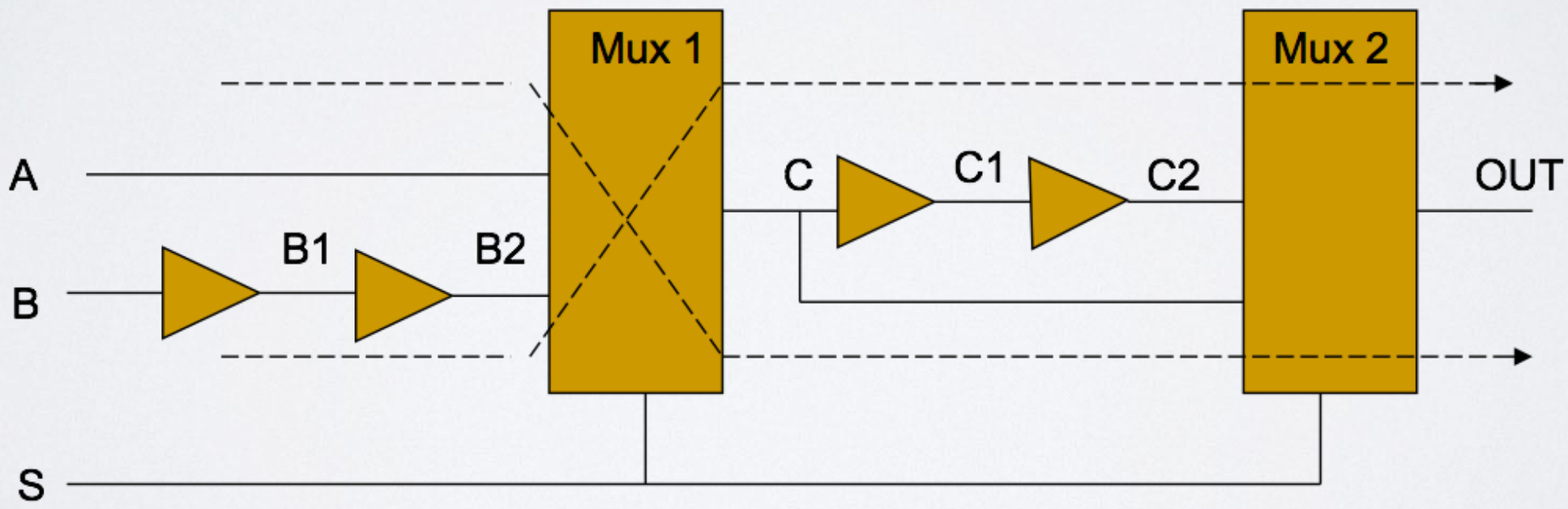
RECOVERY AND REMOVAL TIMES

- Recovery time
  - The time available between the asynchronous signal going inactive to the active clock edge (like setup time for set/reset)
- Removal time
  - The time between active clock edge and asynchronous signal going inactive (like hold time for set reset)



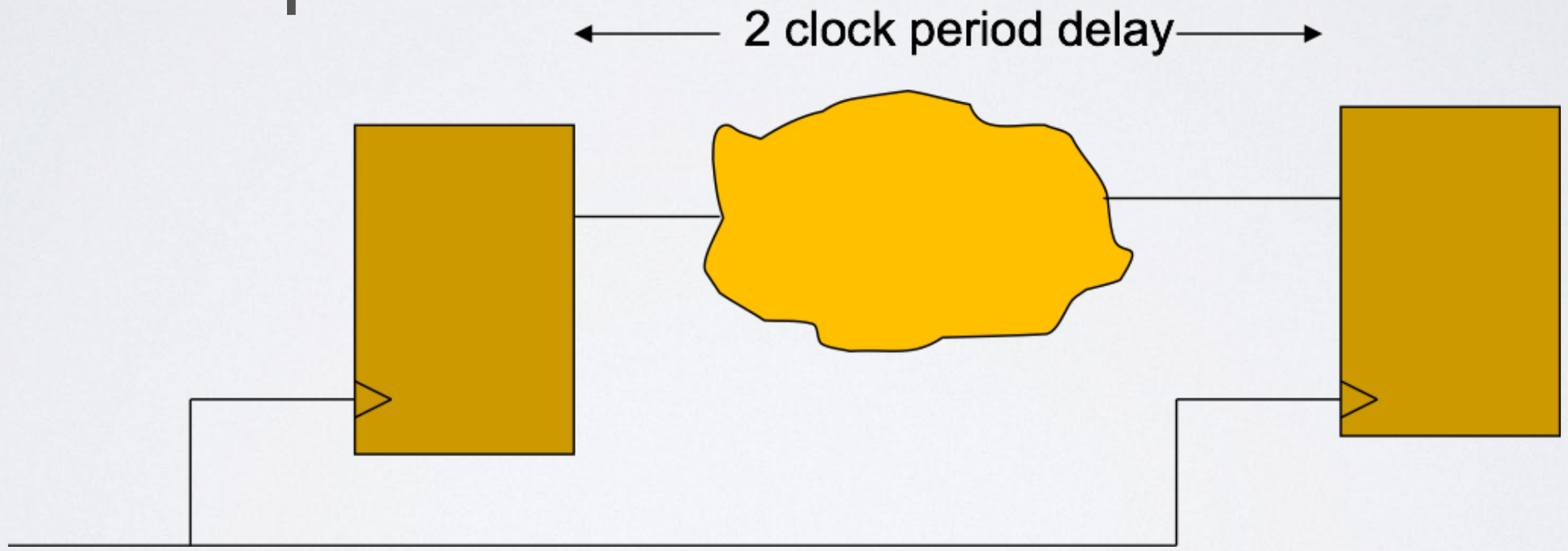
FALSE PATHS

- Paths which physically exist in a design but are not logic paths. These paths never get sensitized under any input condition
  - In other words, a timing path not required to meet its timing constraints for the design to function properly

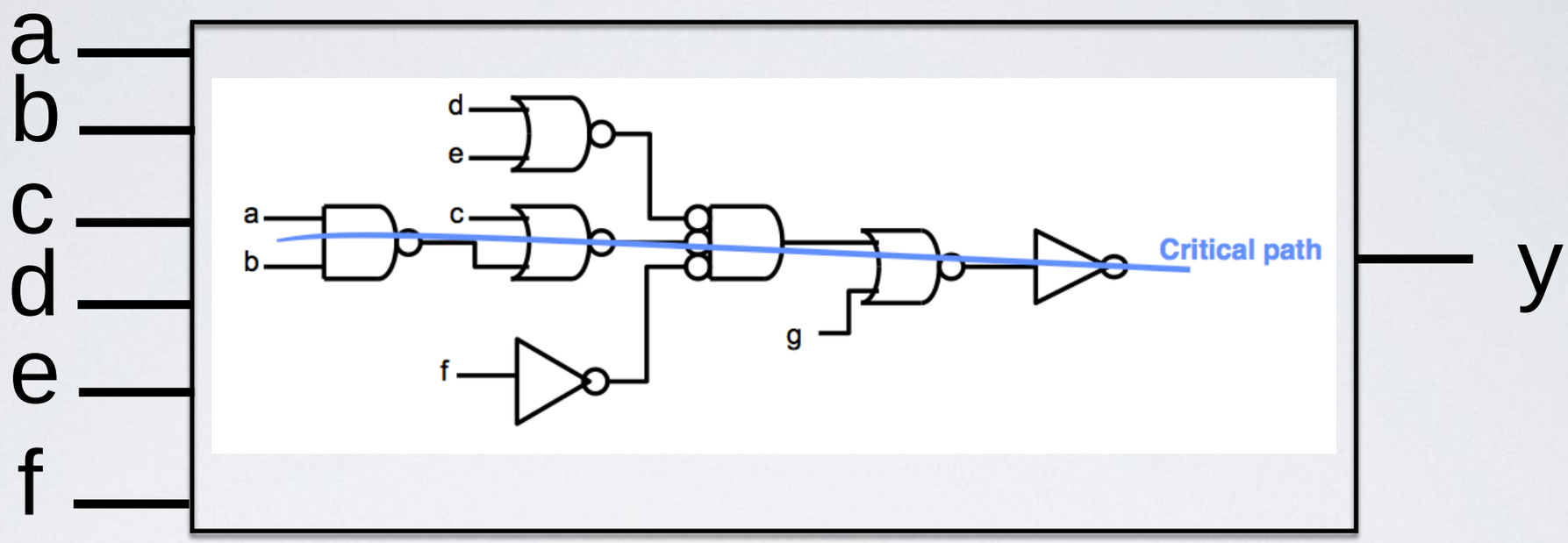


MULTI-CYCLE PATHS

- There are data paths that require more than one clock period for execution



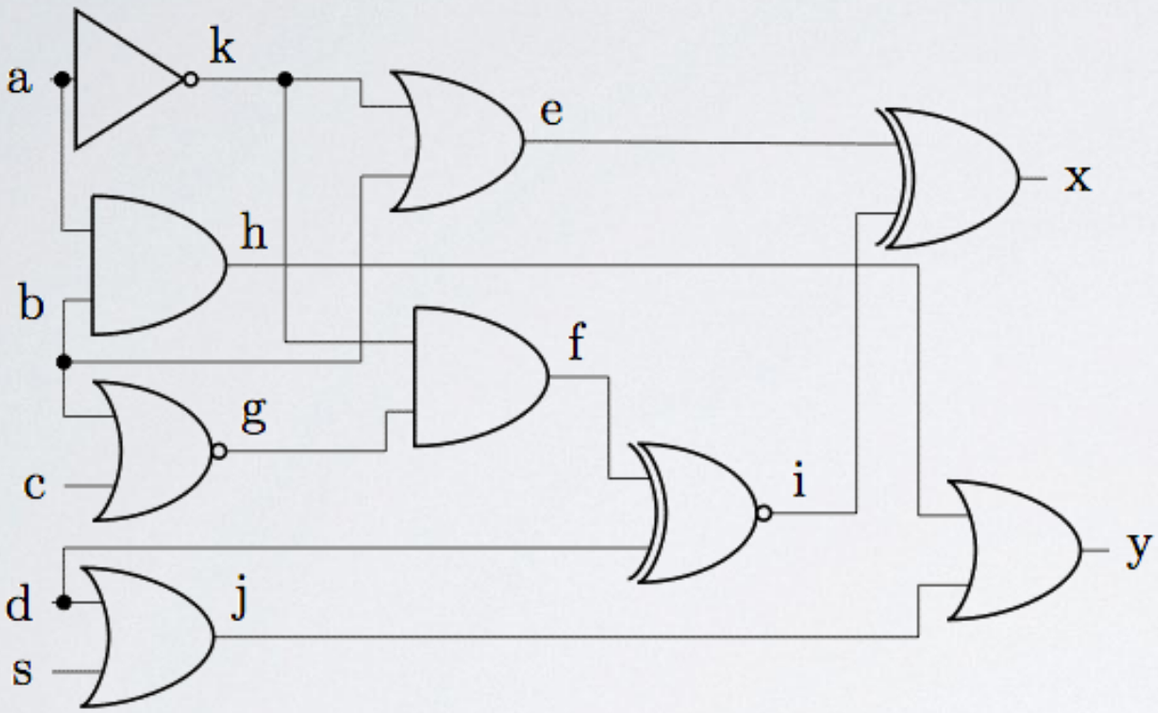
STATIC TIMING ANALYSIS



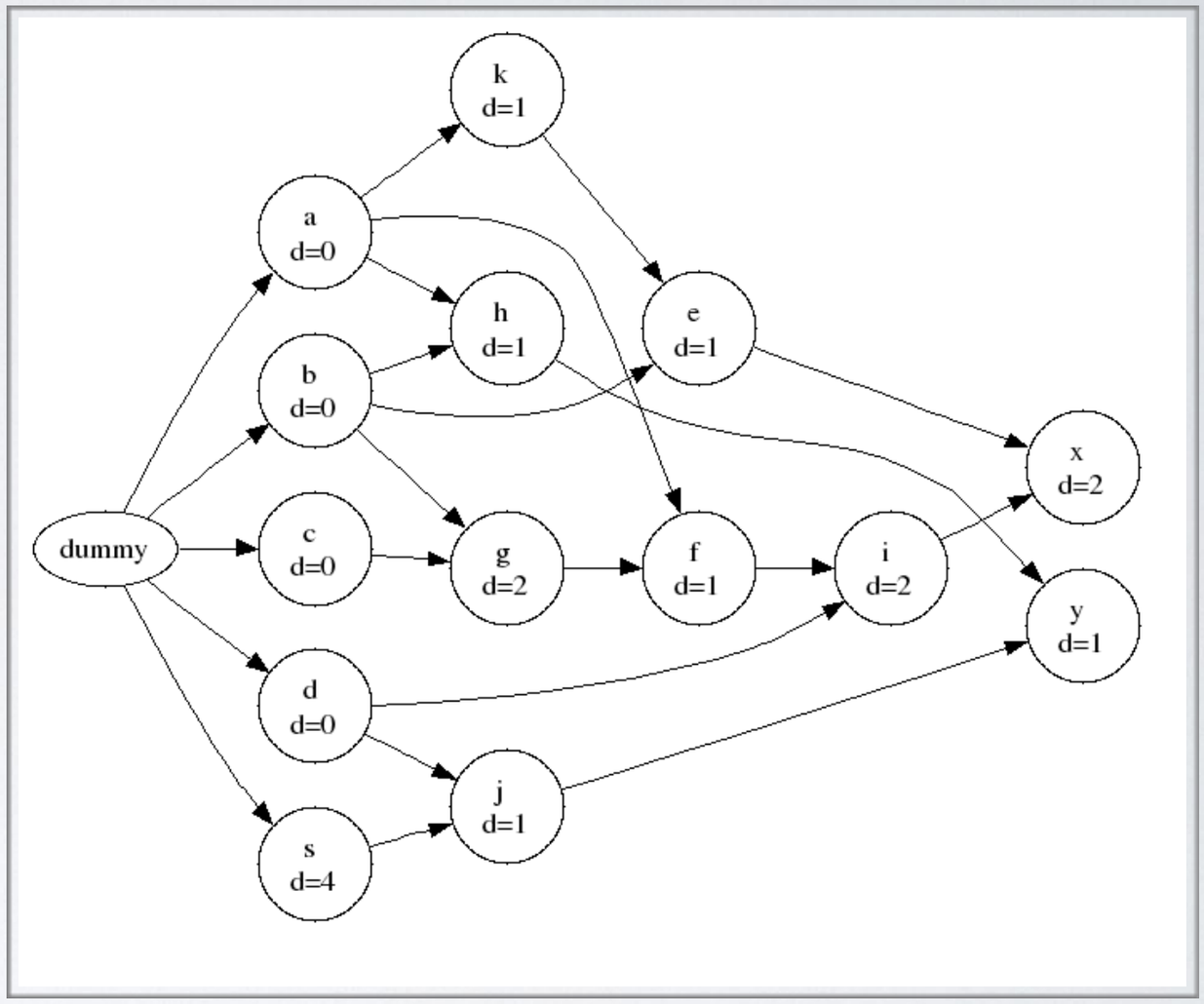
Many paths to y, but one is worst case

- Called Critical Path
- Clock is fixed and max period is set by worst case. Other paths just wait for this one.
- Focus optimization on critical path.

STATIC TIMING ANALYSIS

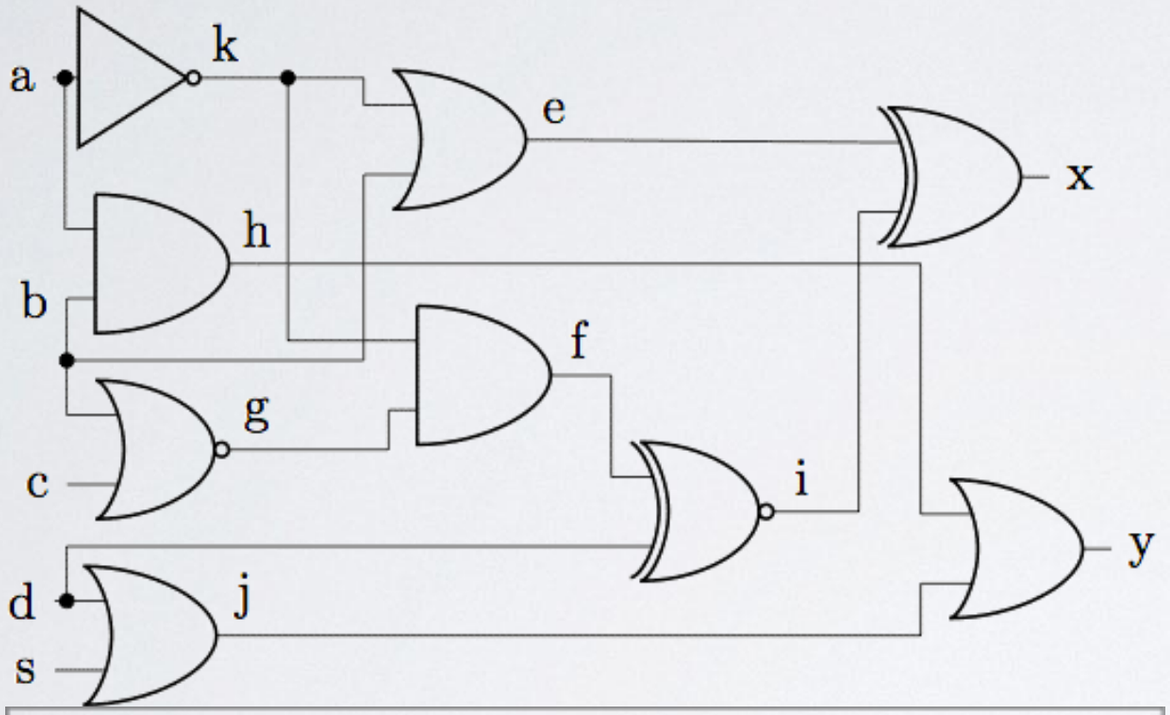


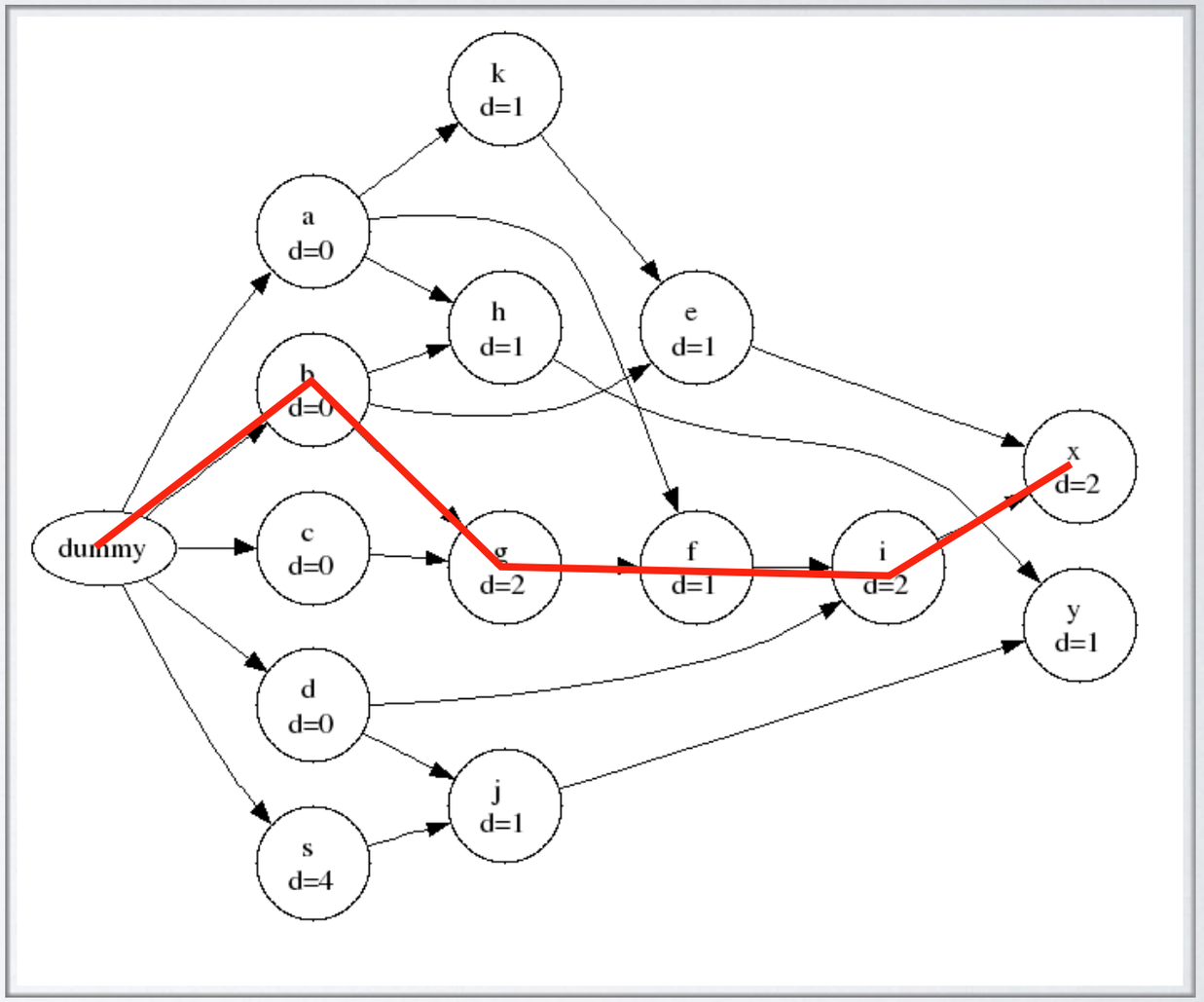
Each stage adds:
Gate Delay + Interconnect delay

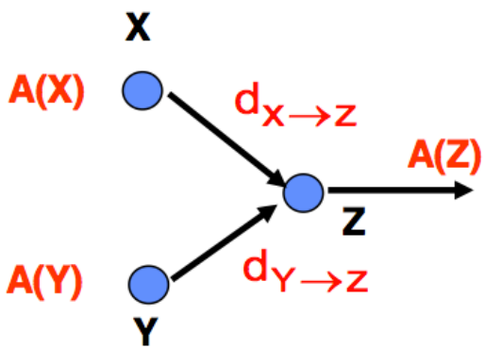


directed acyclic graph (DAG)

STATIC TIMING ANALYSIS
CRITICAL PATH



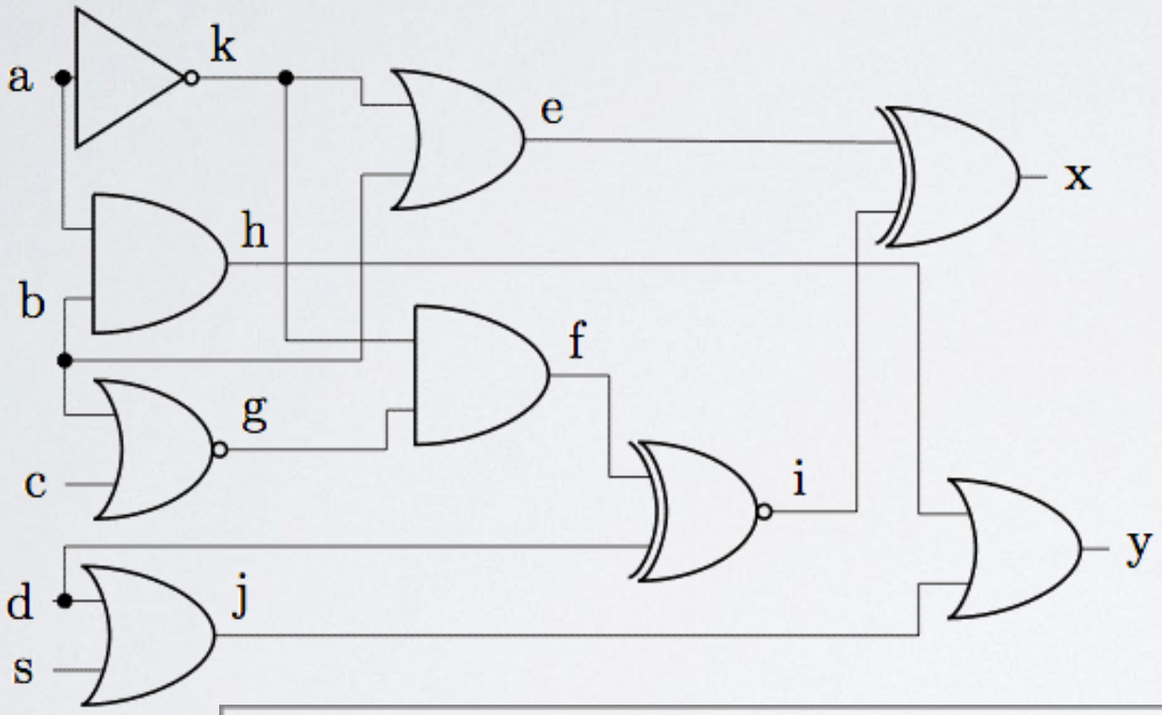


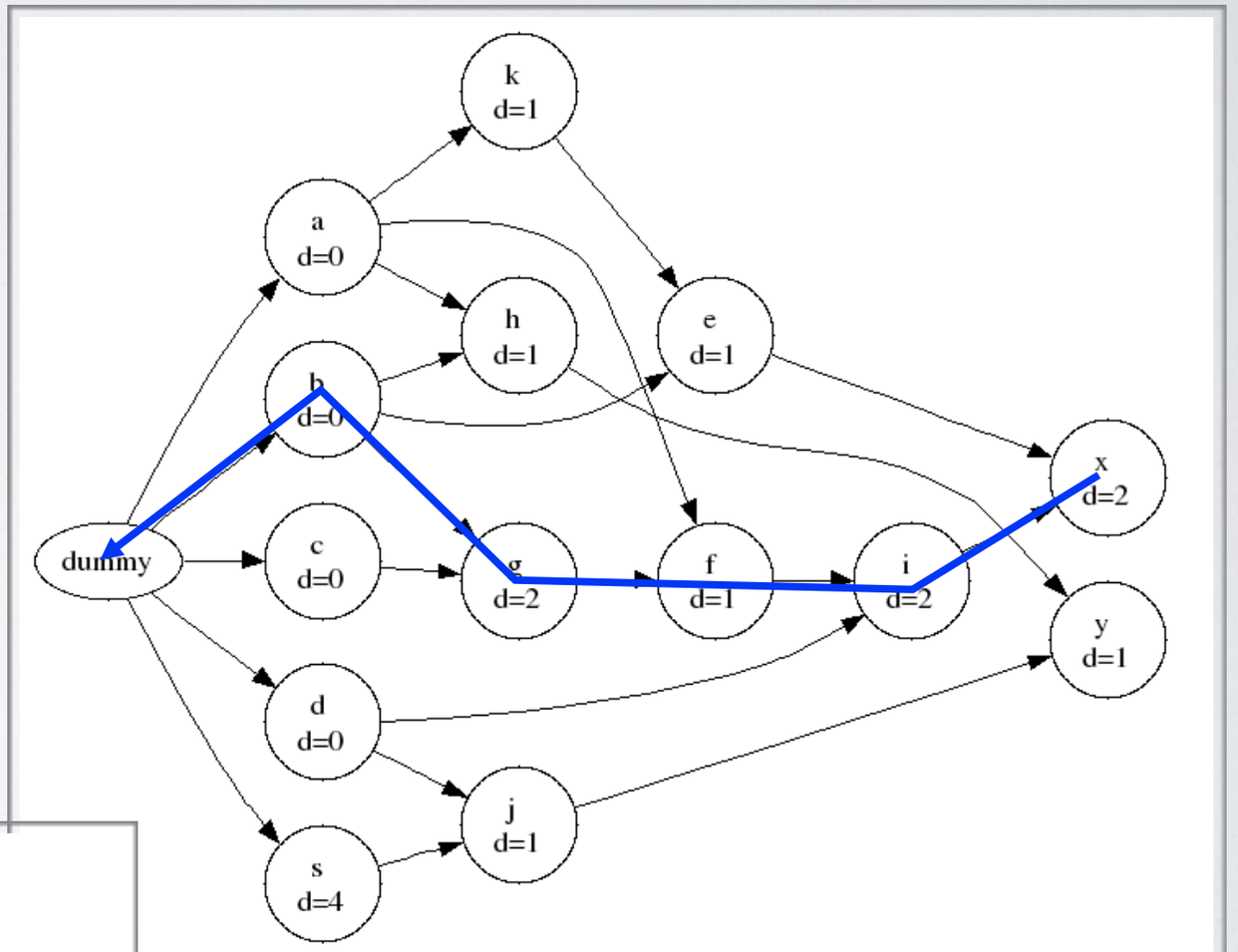


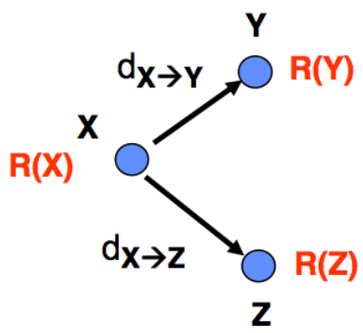
$A(Z) = \text{Delay}(Z) + \text{MAX}(\{A(X) + d_{X\to Z}\}, \{A(Y) + d_{Y\to Z}\})$

This recursion allows linear computation time:
O(|gates|+|edges|)

STATIC TIMING ANALYSIS REQUIRED TIME







R(X) = MIN ({R(Y) – Delay(Y) – $d_{X\to Y}$ }, {R(Z) – Delay(Z) – $d_{X\to Z}$})

Slack(x) = R(x) - A(x)

REFERENCES

- Weste, Harris - CMOS VLSI Design 4th Edition
- Kahng, Lienig, Markov, Hu - VLSI Physical Design: “From Graph Partitioning to Timing Closure”
- Synopsys University Courseware by Vazgen Melikyan