EECS 151/251A Spring 2019 Digital Design and Integrated Circuits

Instructor:
John Wawrzynek

Lecture 10

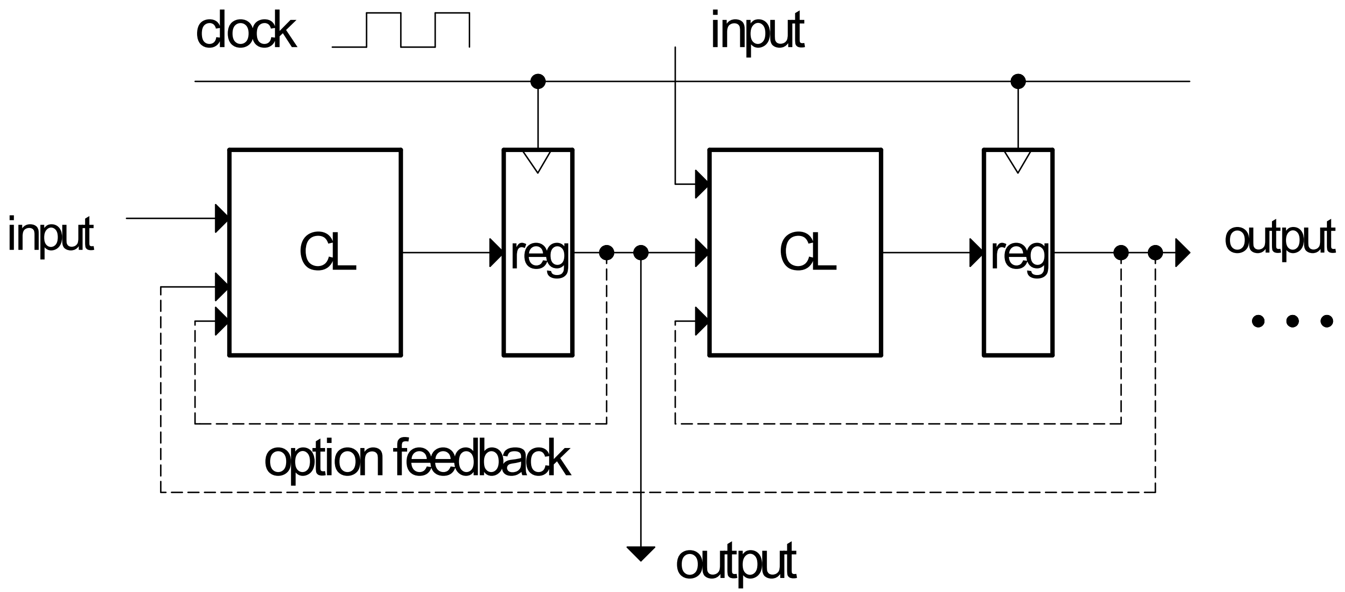
What do ASIC/FPGA Designers need to know about physics?

- Physics effect:

Area ⇒ cost
Delay ⇒ performance
Energy ⇒ performance & cost

- Ideally, zero delay, area, and energy. However, the physical devices occupy area, take time, and consume energy.
- CMOS process lets us build transistors, wires, connections, and we get capacitors, inductors, and resistors whether or not we want them.

Performance, Cost, Power



- How do we measure performance?

  operations/sec? cycles/sec?

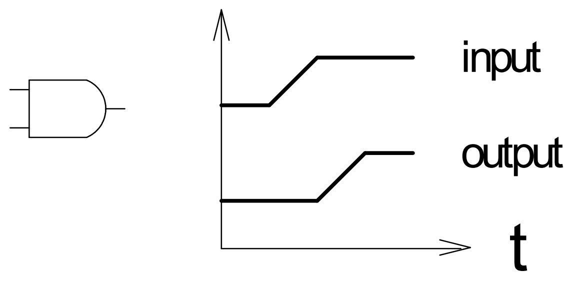
- Performance is directly proportional to clock frequency. Although it may not be the entire story:

  Ex: CPU performance

  = # instructions X CPI X clock period

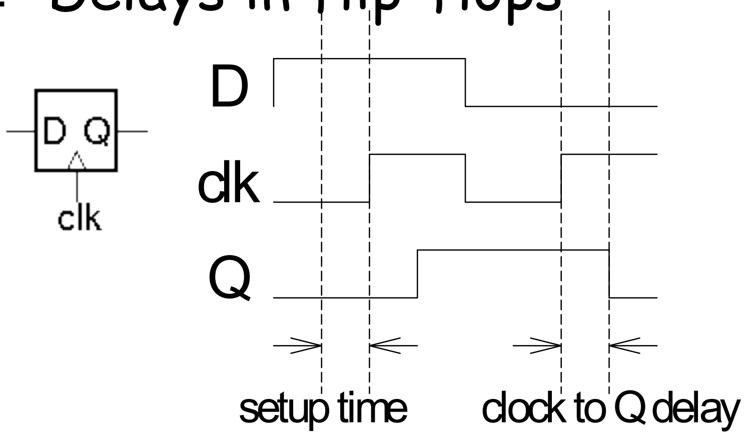
Limitations on Clock Rate

1 Logic Gate Delay



What are typical delay values?

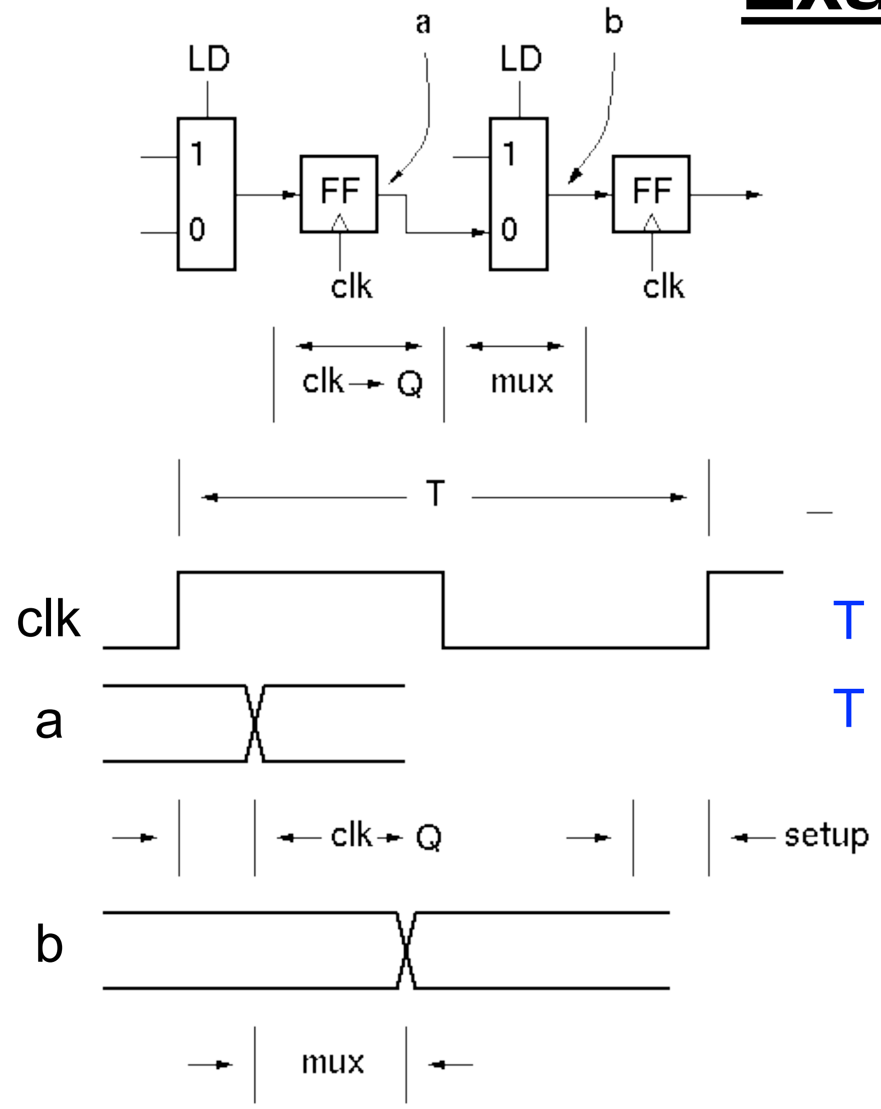
2 Delays in flip-flops



Both times contribute to limiting the clock period.

- What must happen in one clock cycle for correct operation?
  - All signals connected to FF (or memory) inputs must be ready and "setup" before rising edge of clock.
  - For now we assume perfect clock distribution (all flip-flops see the clock at the same time).

Example

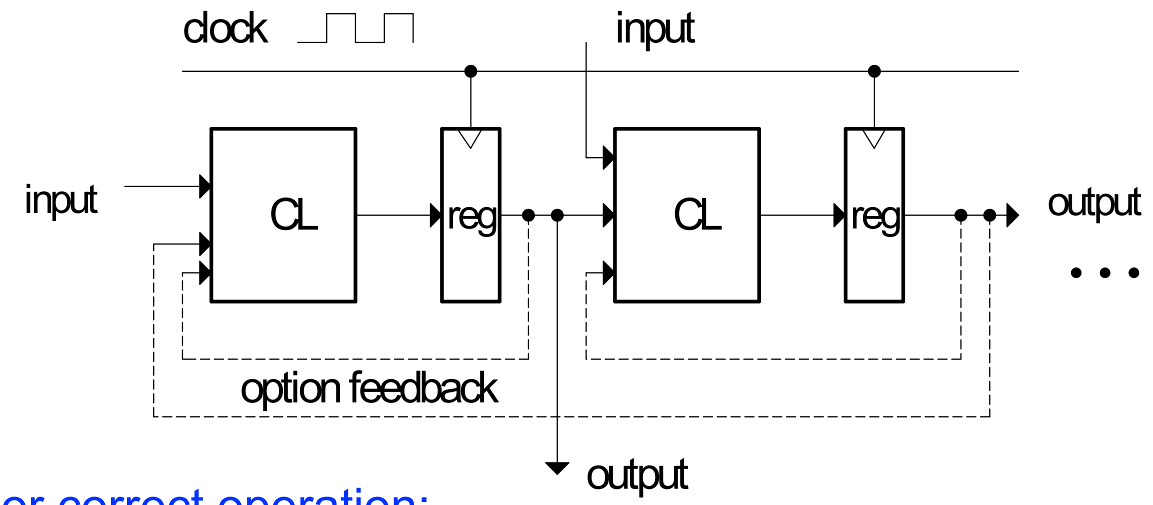


Parallel to serial converter circuit

$T \geq$ time(clk→Q) + time(mux) + time(setup)

$T \geq \tau_{clk \rightarrow Q} + \tau_{mux} + \tau_{setup}$

In General ...

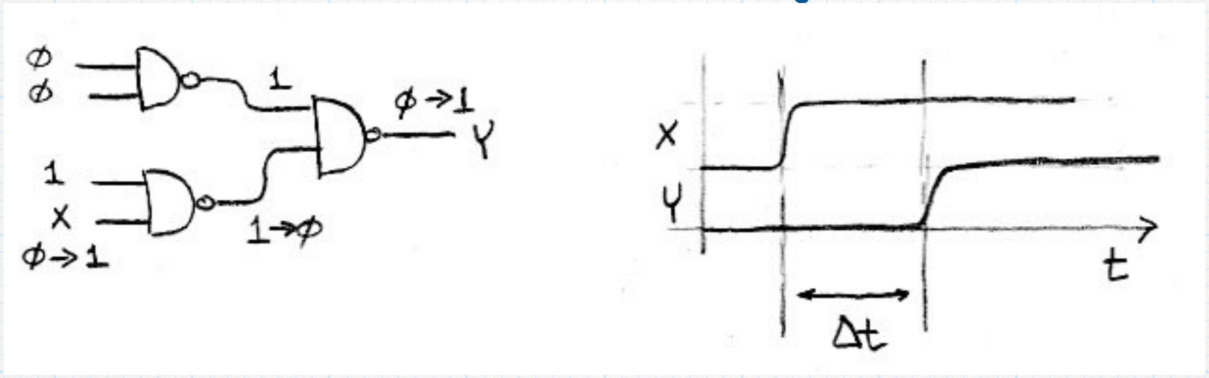


For correct operation:

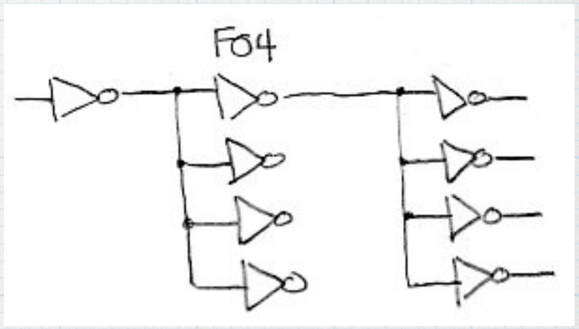
$$\boxed{T \geq \tau_{clk\rightarrow Q} + \tau_{CL} + \tau_{setup}}$$ for all paths.

- How do we enumerate **all** paths?
  - Any circuit input or register output to any register input or circuit output?
- Note:
  - "setup time" for outputs is a function of what it connects to.
  - "clk-to-q" for circuit inputs depends on from where it comes.

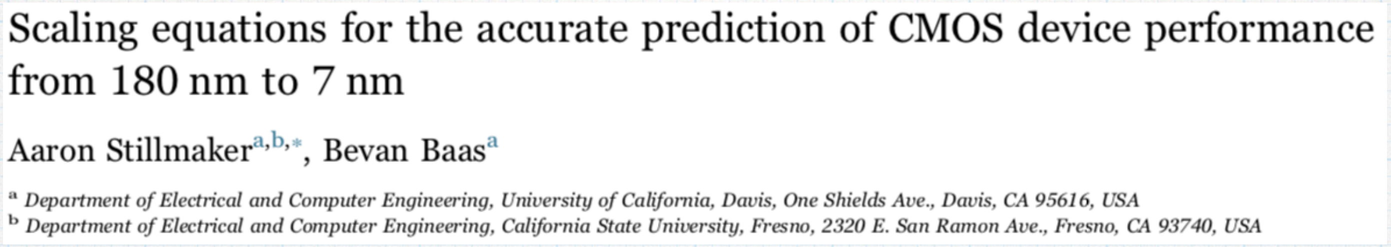
"Gate Delay"



- Modern CMOS gate delays on the order of a few picoseconds. (However, highly dependent on gate context.)
- Often expressed as FO4 delays (fan-out of 4) - as a process dependent delay metric:
  - the delay of an inverter, driven by an inverter 4x smaller than itself, and driving an inverter 4x larger than itself.
  - For a 7nm process FO4 is around 2.5ps. Less than 10ps for a 32nm process.



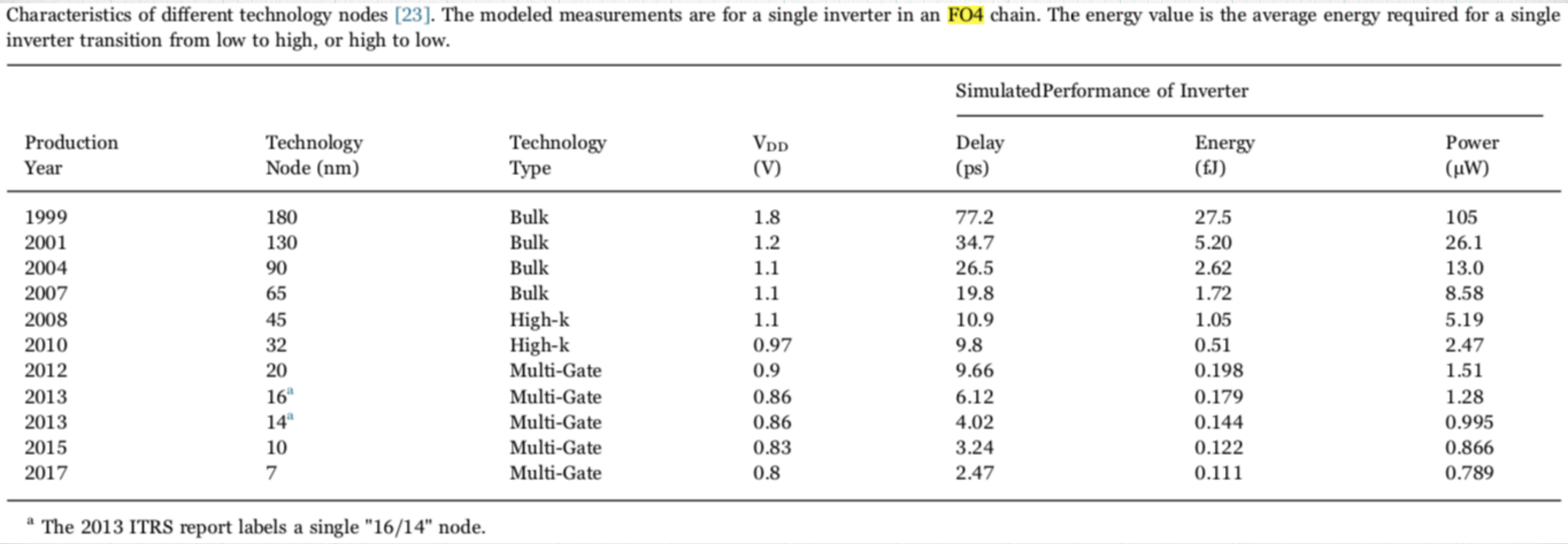
Process Dependent FO4 Delay



Scaling equations for the accurate prediction of CMOS device performance from 180 nm to 7 nm

Aaron Stillmaker[a,b,*], Bevan Baas[a]

[a] *Department of Electrical and Computer Engineering, University of California, Davis, One Shields Ave., Davis, CA 95616, USA*
[b] *Department of Electrical and Computer Engineering, California State University, Fresno, 2320 E. San Ramon Ave., Fresno, CA 93740, USA*

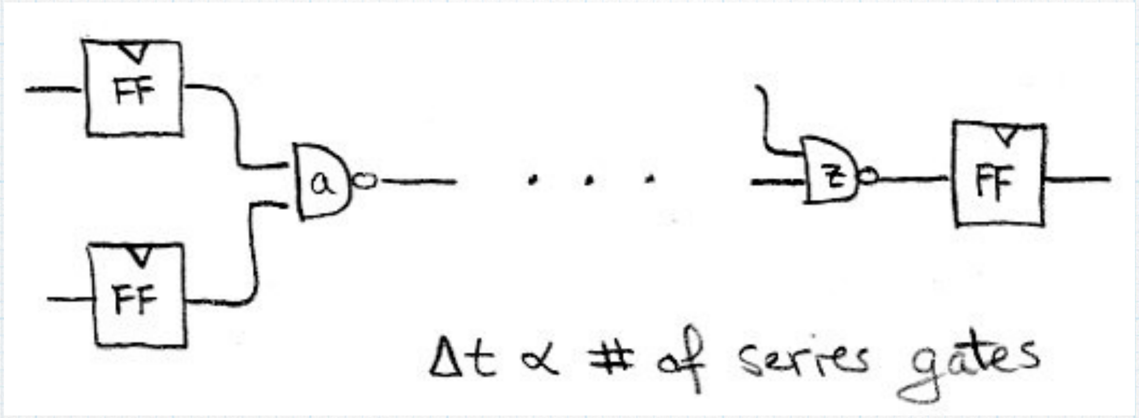


Characteristics of different technology nodes [23]. The modeled measurements are for a single inverter in an FO4 chain. The energy value is the average energy required for a single inverter transition from low to high, or high to low.

				Simulated Performance of Inverter		
Production Year	Technology Node (nm)	Technology Type	V_{DD} (V)	Delay (ps)	Energy (fJ)	Power (μW)
1999	180	Bulk	1.8	77.2	27.5	105
2001	130	Bulk	1.2	34.7	5.20	26.1
2004	90	Bulk	1.1	26.5	2.62	13.0
2007	65	Bulk	1.1	19.8	1.72	8.58
2008	45	High-k	1.1	10.9	1.05	5.19
2010	32	High-k	0.97	9.8	0.51	2.47
2012	20	Multi-Gate	0.9	9.66	0.198	1.51
2013	16[a]	Multi-Gate	0.86	6.12	0.179	1.28
2013	14[a]	Multi-Gate	0.86	4.02	0.144	0.995
2015	10	Multi-Gate	0.83	3.24	0.122	0.866
2017	7	Multi-Gate	0.8	2.47	0.111	0.789

[a] The 2013 ITRS report labels a single "16/14" node.

"Path Delay"

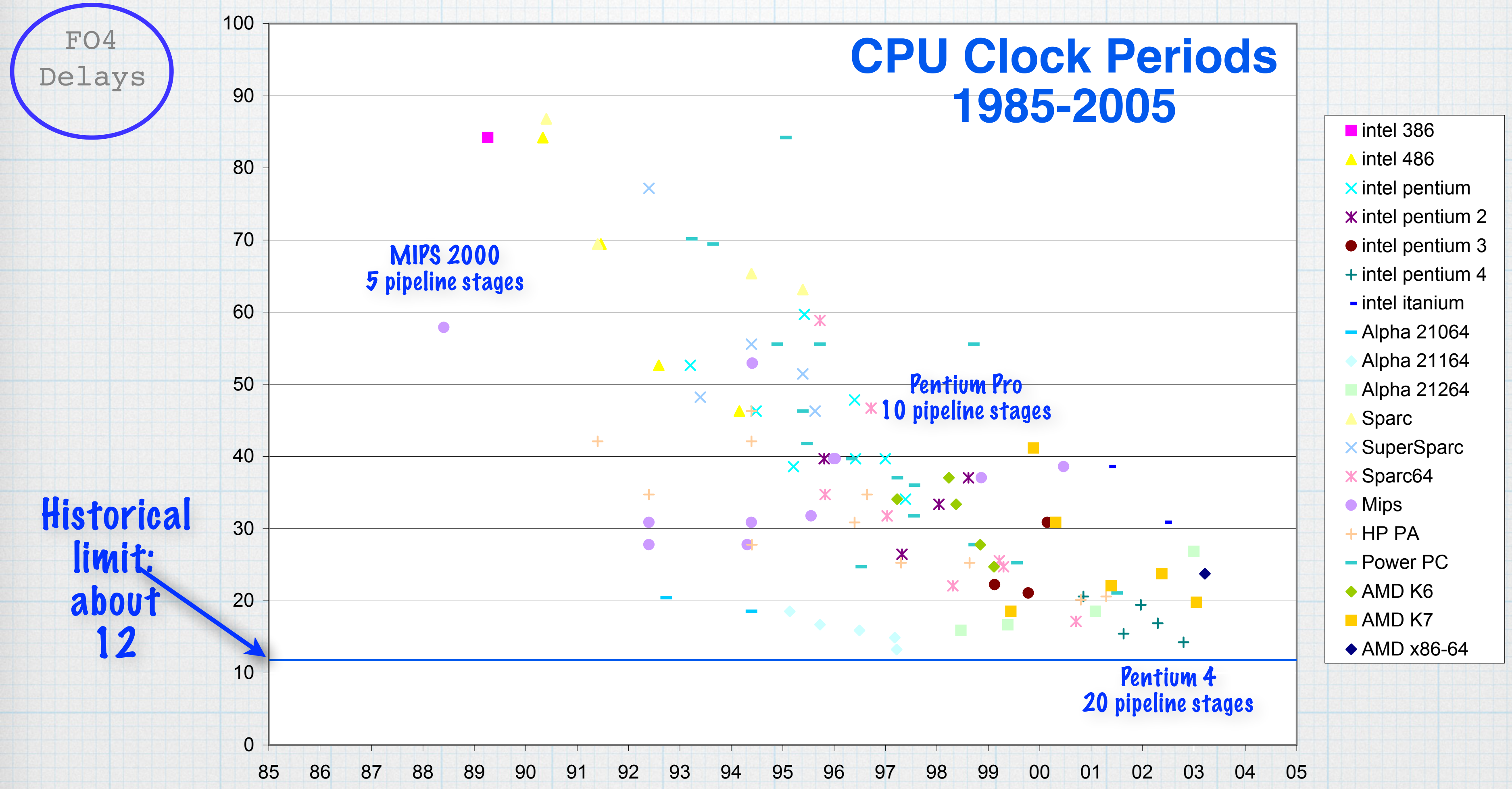


- For correct operation:
  Total Delay ≤ clock_period - FF_{setup_time} - $FF_{clk\_to\_q}$
  on all paths.

- High-speed processors critical paths have around 20 FO4 delays.

F04 Delays per clock period

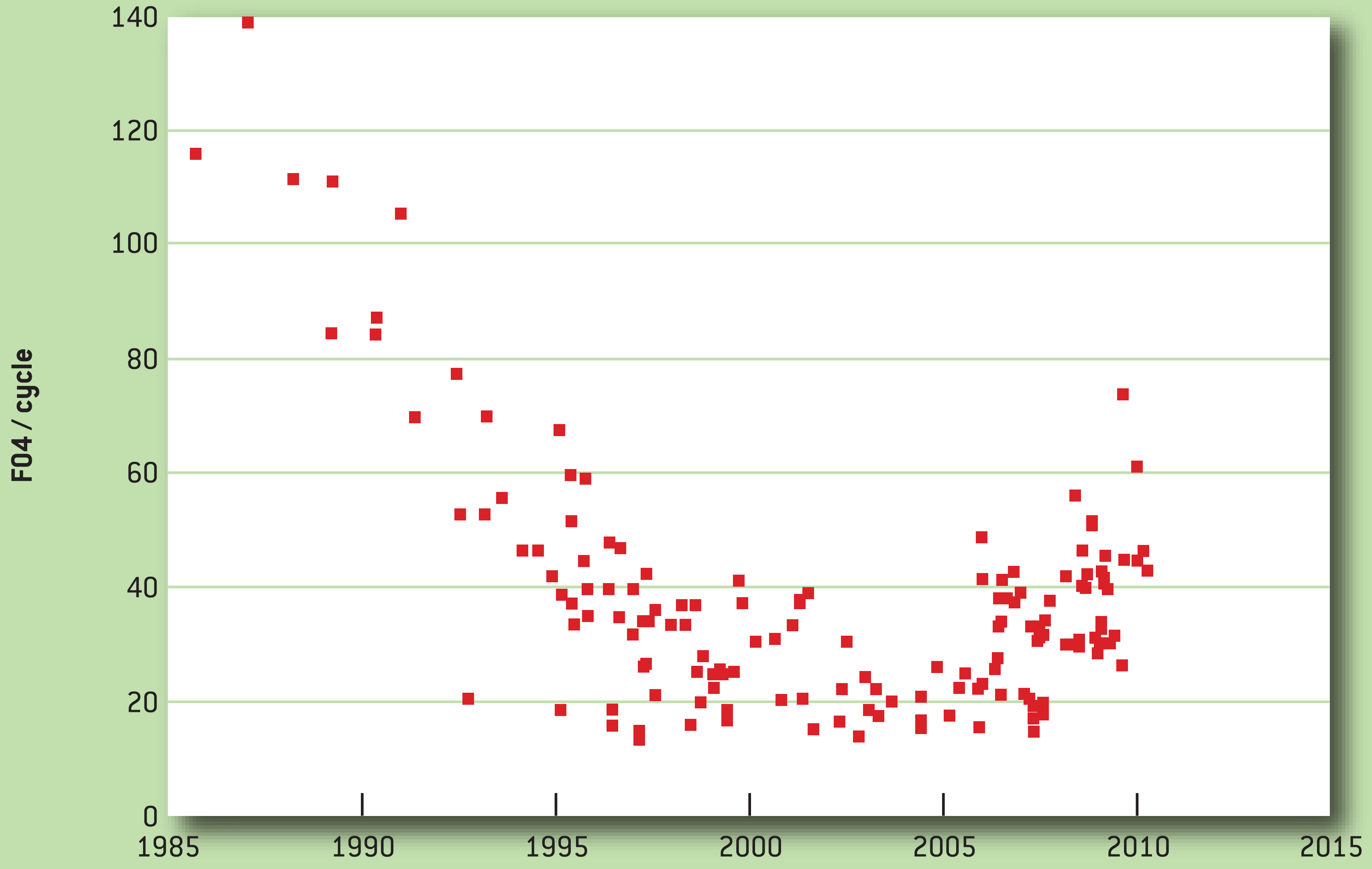
FO4 Delays



Thanks to Francois Labonte, Stanford

Andrew Danowitz, Kyle Kelley, James Mao, John P. Stevenson, Mark Horowitz, Stanford University

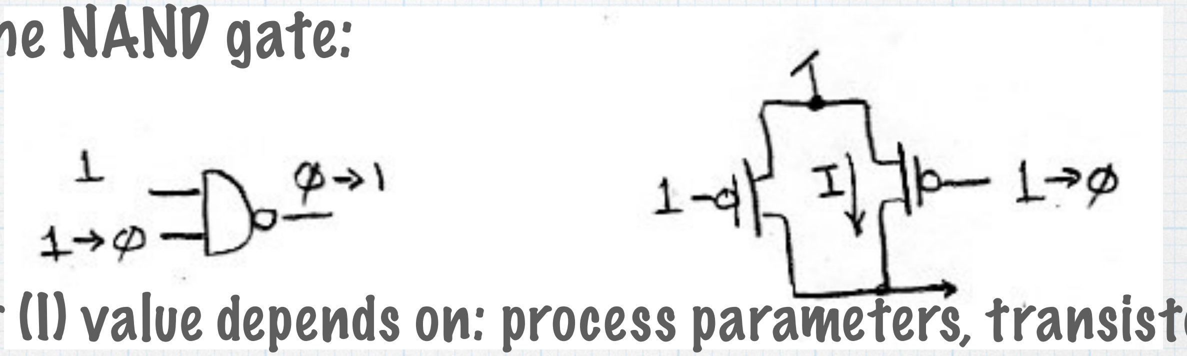
F04 Delays Per Cycle for Processor Designs



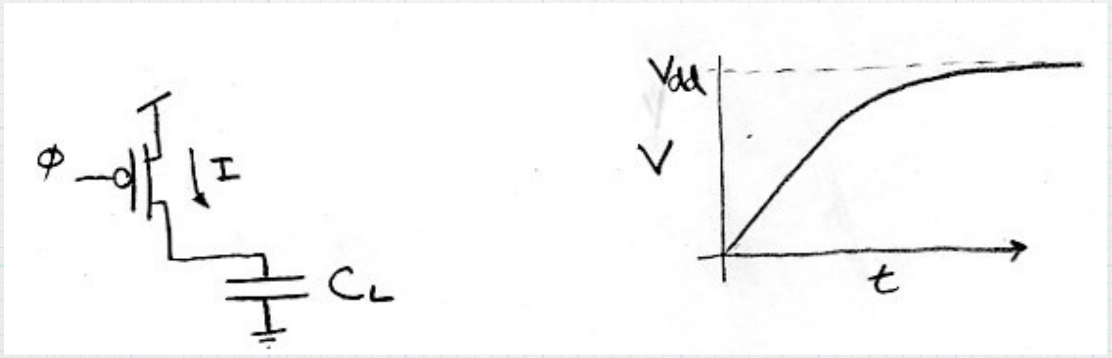
F04 delay per cycle is roughly proportional to the amount of computation completed per cycle.

"Gate Delay"

- What determines the actual delay of a logic gate?
- Transistors are not perfect switches - cannot change terminal voltages instantaneously.
- Consider the NAND gate:



  - Current (I) value depends on: process parameters, transistor size

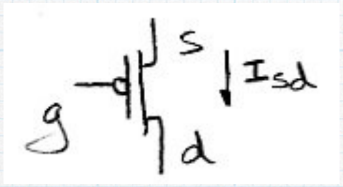


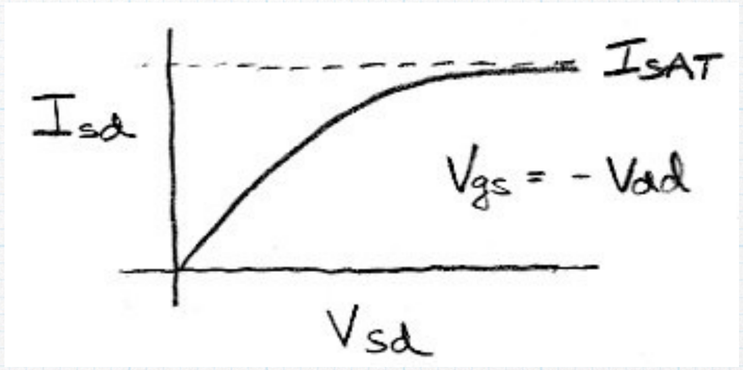
$\Delta t \propto C_L / I$

- C_L models gate output, wire, inputs to next stage (Cap. of Load)
- C "integrates" I creating a voltage change at output

More on transistor Current

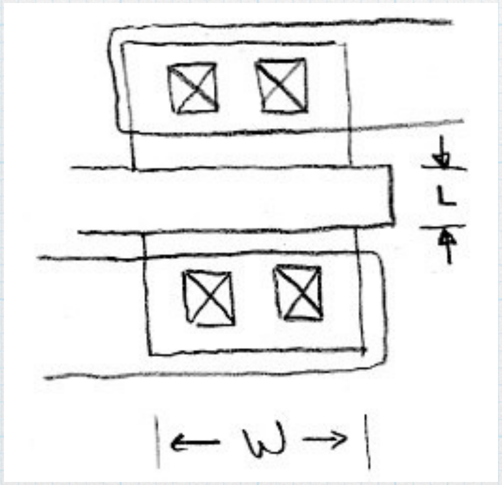
- Transistors act like a cross between a resistor and "current source"



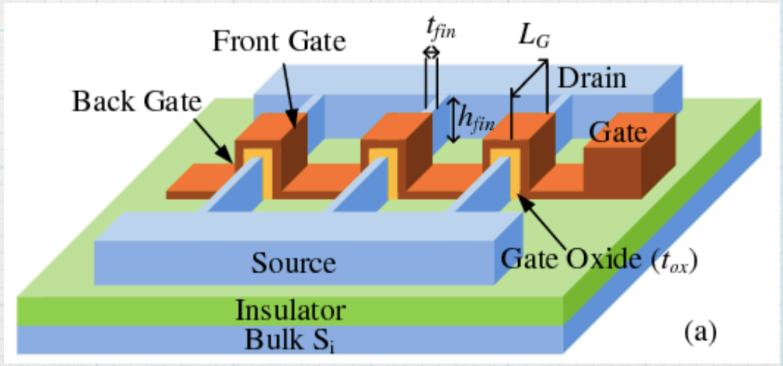


- I_{SAT} depends on process parameters (higher for nFETs than for pFETs) and transistor size (layout):

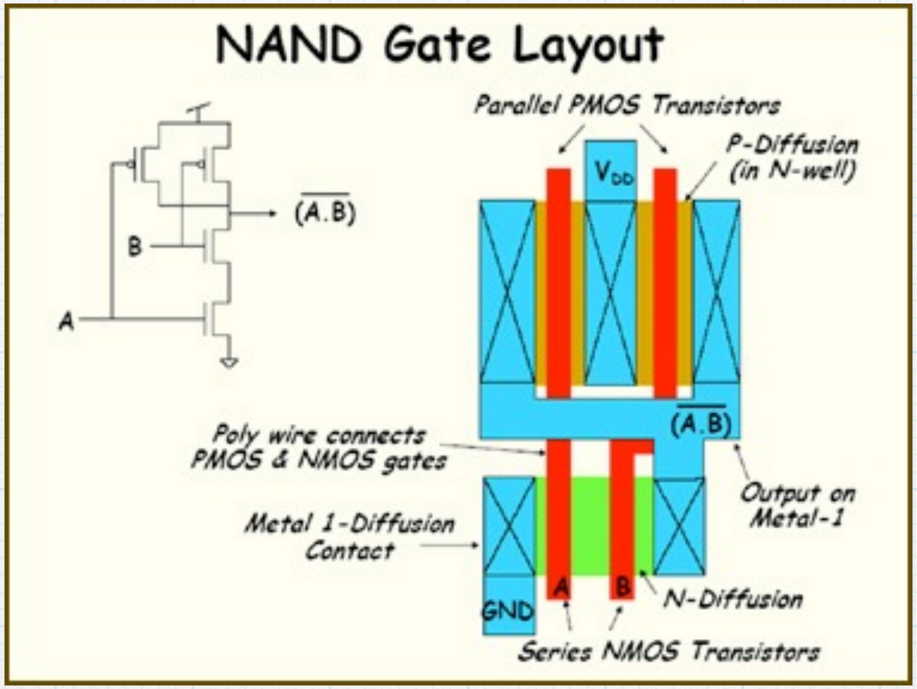
$I_{SAT} \propto W/L$



FinFets use multiple "fins" to get wider



Physical Layout determines FET strength

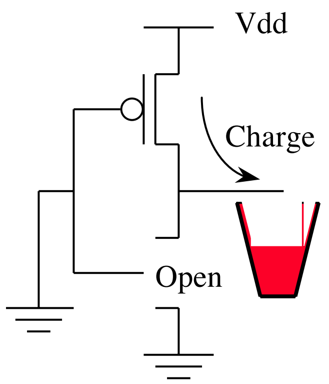


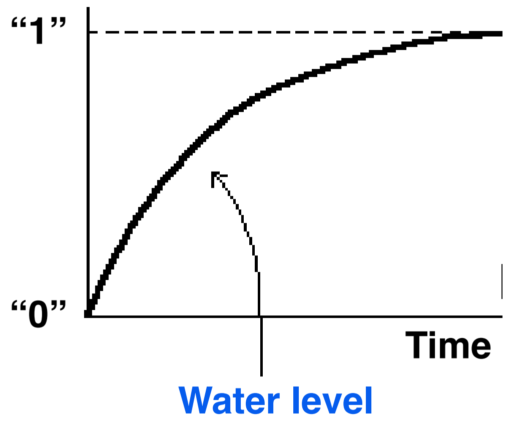
- "Switch-level" abstraction gives a good way to understand the **function** of a circuit.
  - nFET (g=1 ? short circuit : open)
  - pFET (g=0 ? short circuit : open)
- Understanding delay means going below the switch-level abstraction to transistor physics and layout details.

Transistors as water valves. (Cartoon physics)

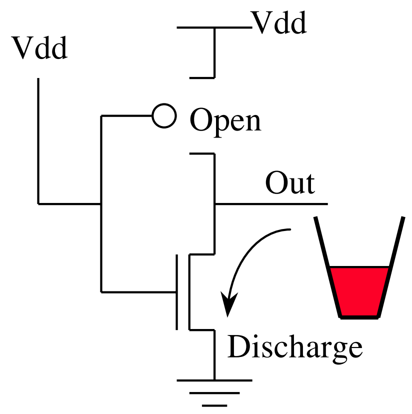
If electrons are water molecules,
transistor strengths (W/L) are pipe diameters,
and capacitors are buckets ...

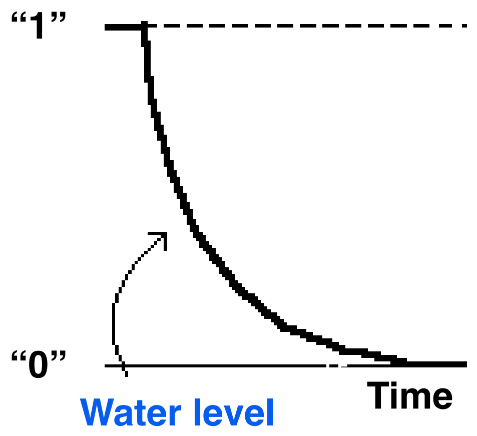
A "on" p-FET fills up the capacitor with charge.





A "on" n-FET empties the bucket.



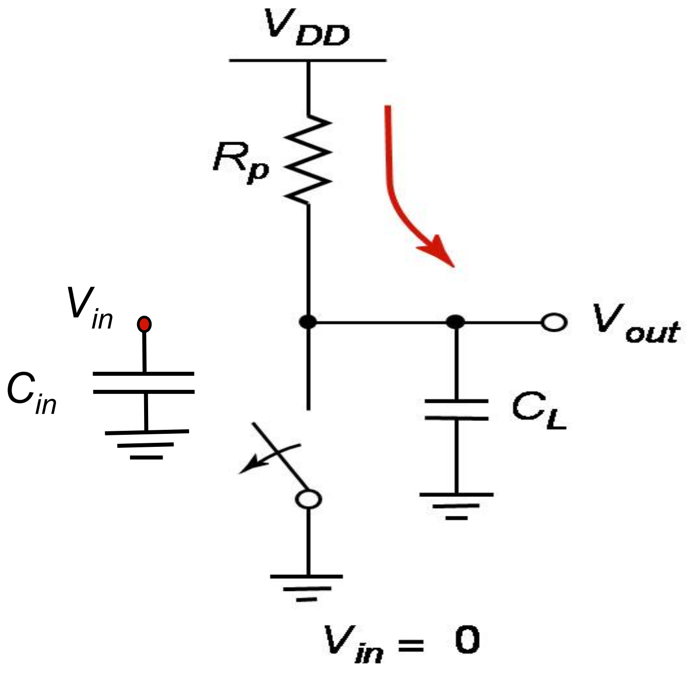


The Switch Inverter: Transient Response

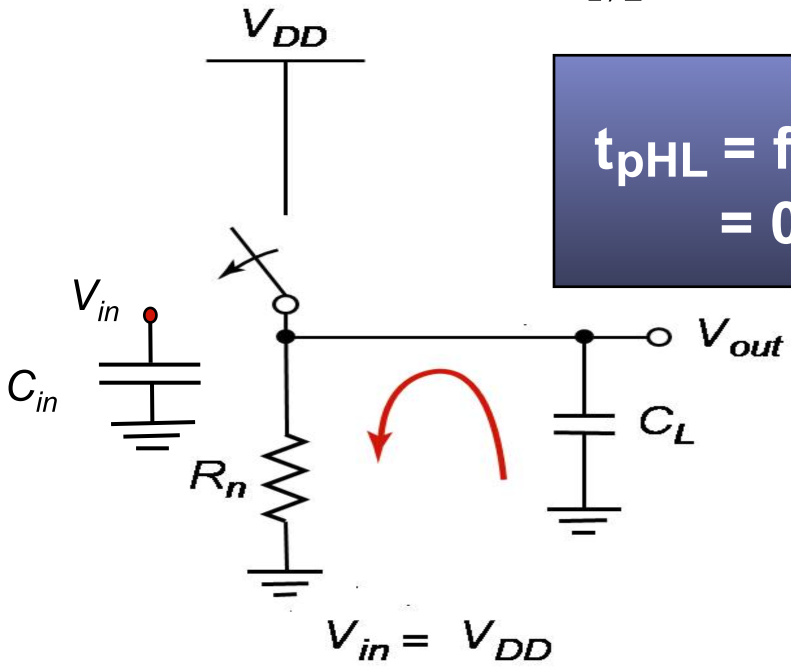
$$V(t) = V_0 \, e^{-t/RC}$$

$$t_{1/2} = \ln(2) \times RC$$

$t_{pHL} = f(R_{on}C_L)$
$= 0.69\ R_n C_L$



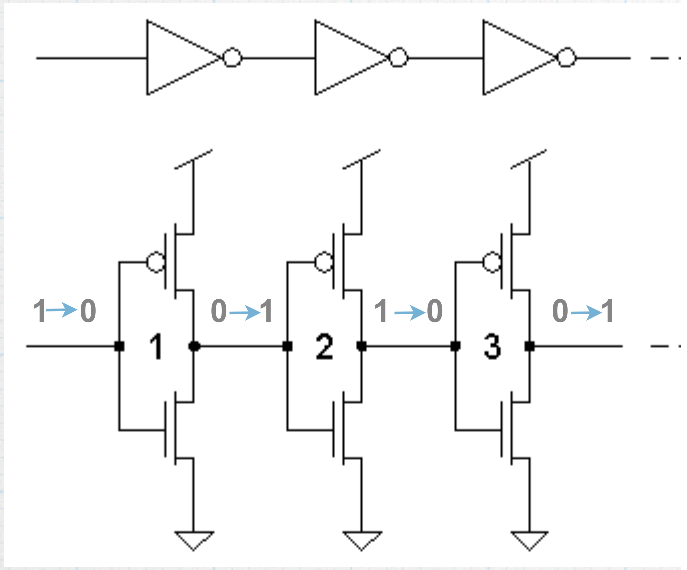
(a) Low-to-high

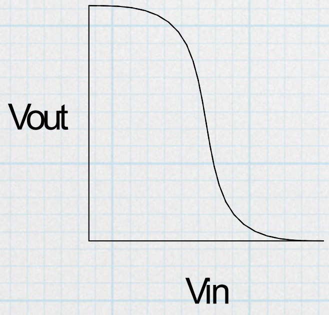


(b) High-to-low

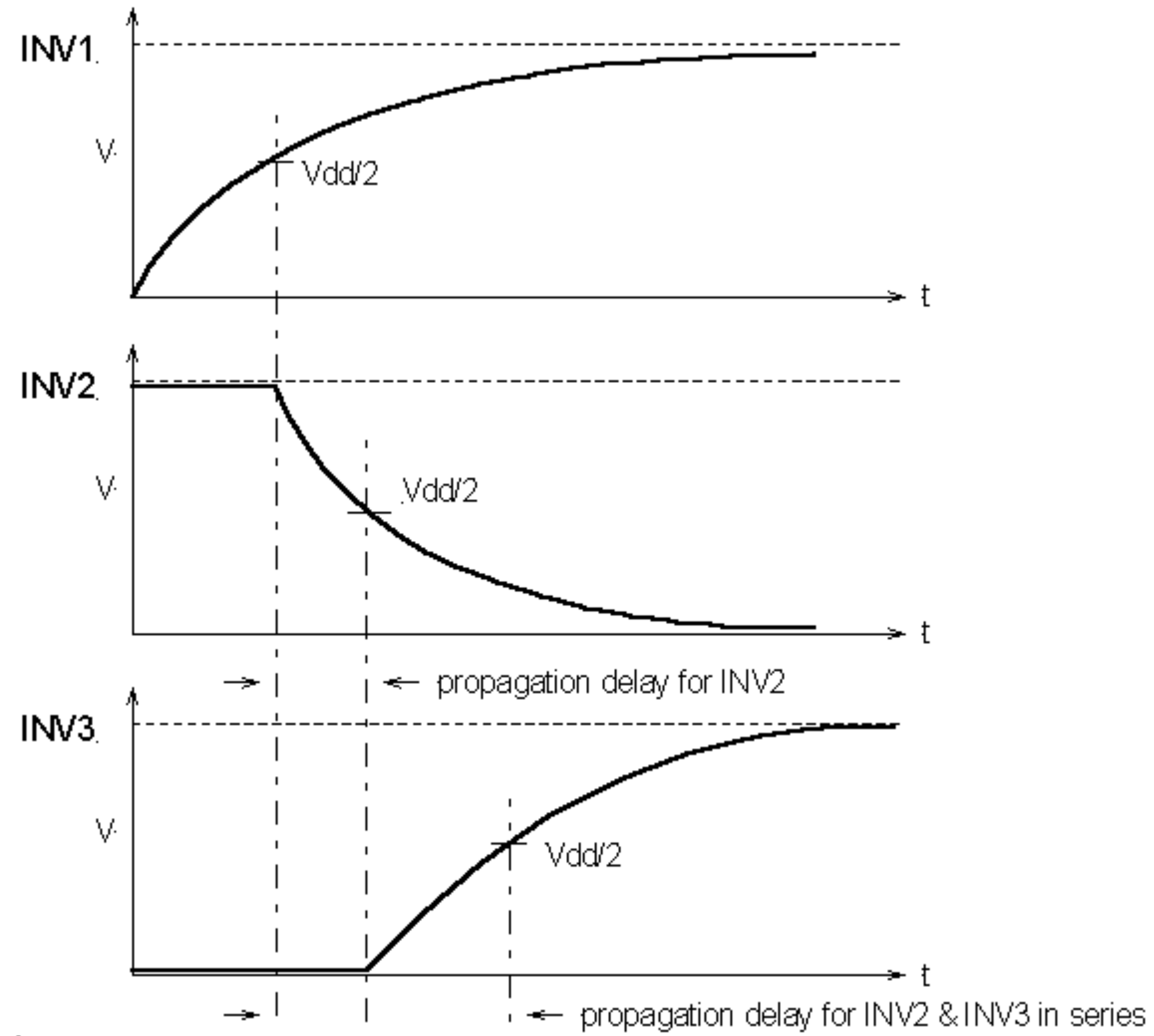
Turning Rise/Fall Delay into Gate Delay

- Cascaded gates:





"transfer curve" for inverter.

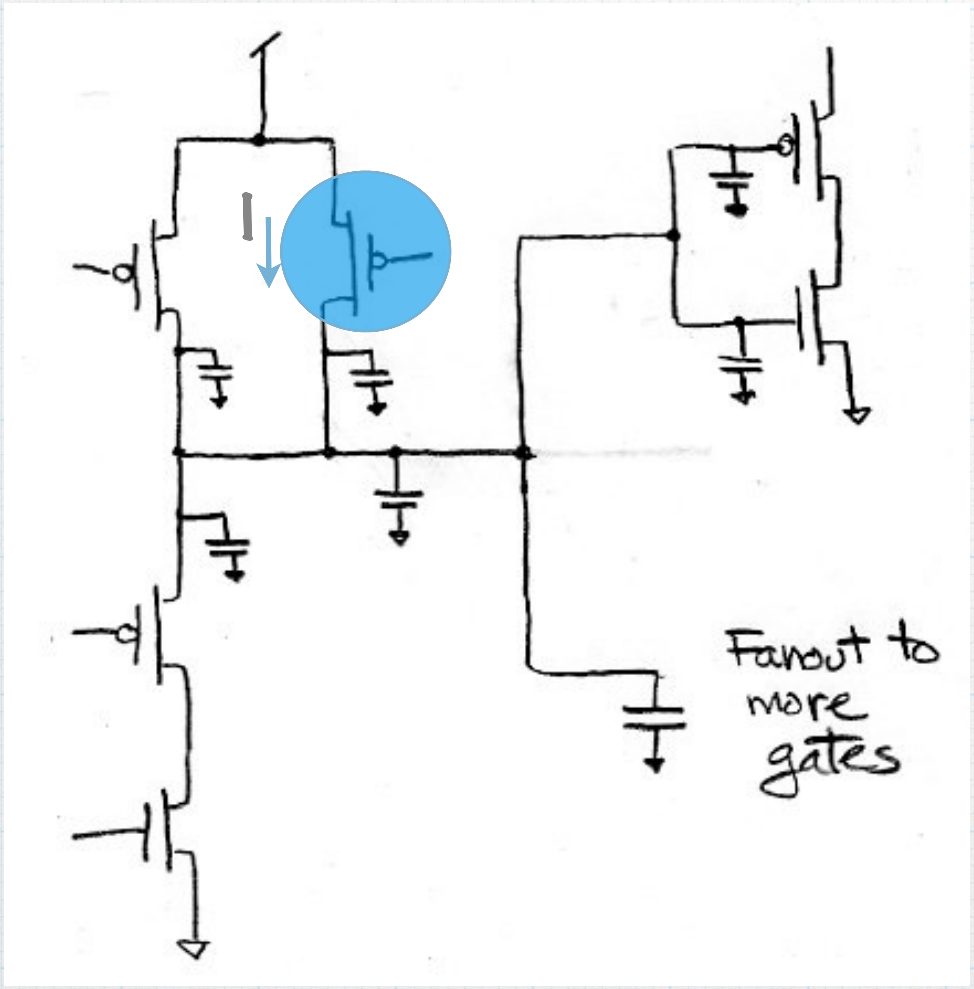


In general:

prop. delay = sum of individual prop. delays of gates in series.

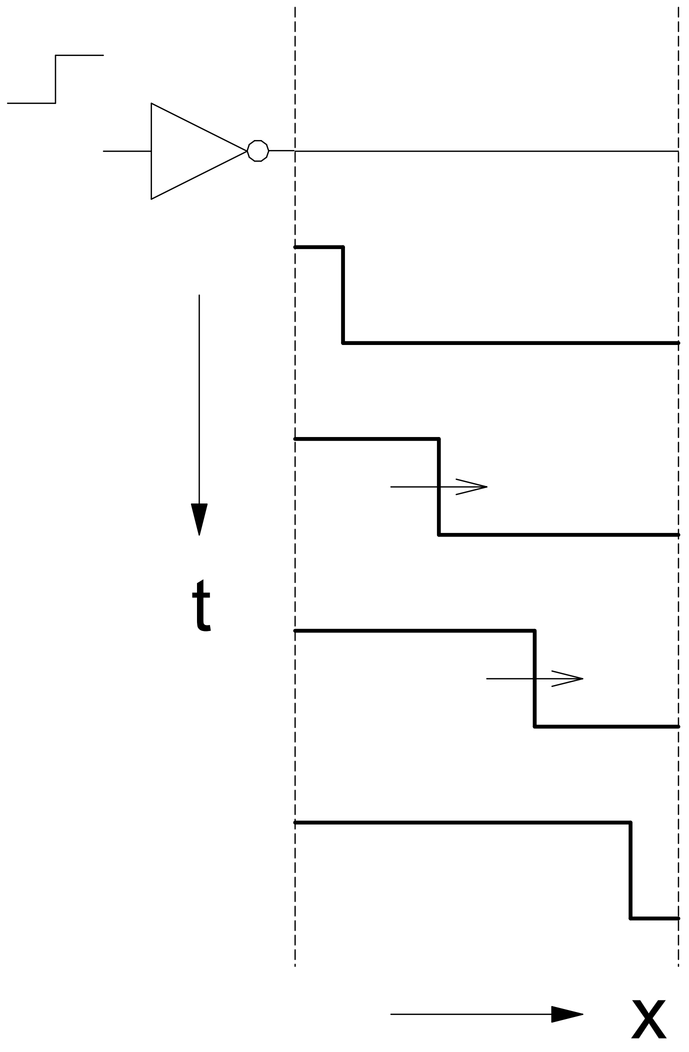
More on C_L

- Everything that connects to the output of a logic gate (or transistor) contributes capacitance:



- Transistor drains
- Interconnection (wires/contacts/vias)
- Transistor Gates

Wire Delay

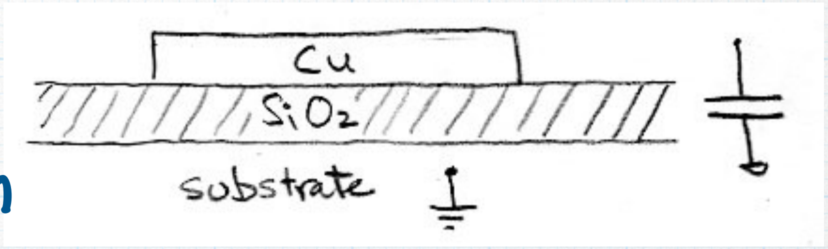


- Ideally, wires behave as "transmission lines":
  - signal wave-front moves close to the speed of light
    - ~1ft/ns
  - Time from source to destination is called the "transit time".
  - In ICs most wires are short, and the transit times are relatively short compared to the clock period and can be ignored.
  - Not so on PC boards.

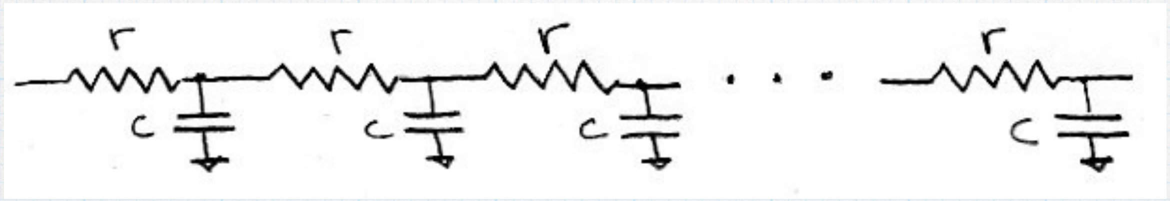
Wires

- As parallel plate capacitors:

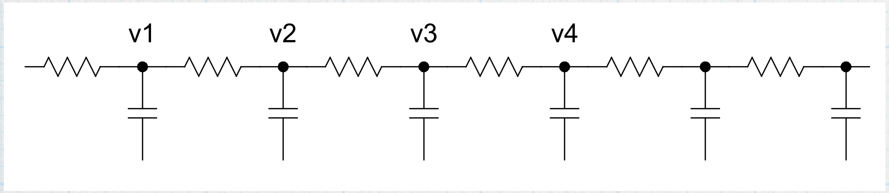
  $C \propto$ Area = width $*$ length

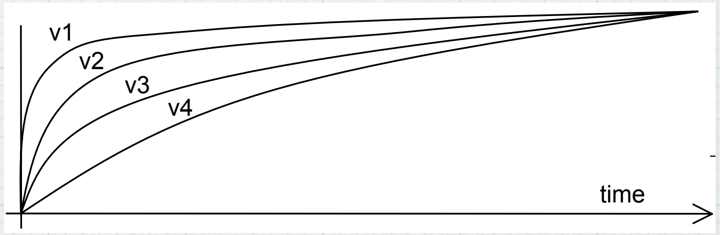


- Wires have finite resistance, so have distributed R and C:



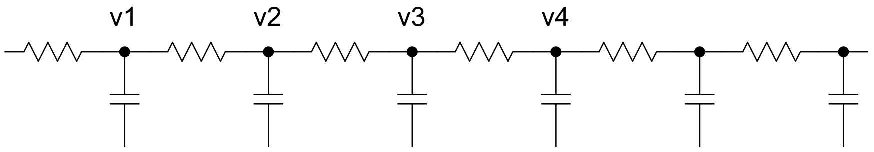
with r = res/length, c = cap/length, $\Delta t \propto rcL^2 \cong rc + 2rc + 3rc + \ldots$





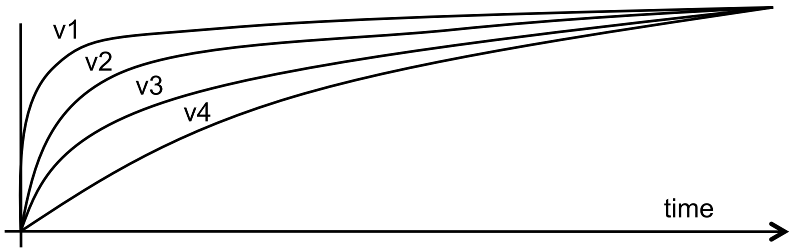
Wire Delay

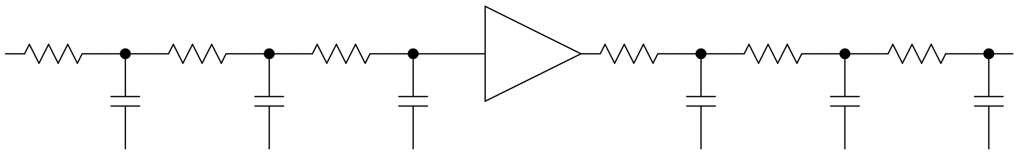
- Even in those cases where the transmission line effect is negligible:
  - Wires posses distributed resistance and capacitance



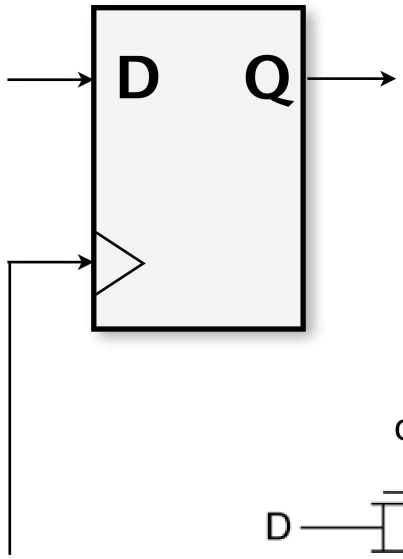
  - Time constant associated with distributed RC is proportional to the *square* of the length

- For **short wires** on ICs, resistance is insignificant (relative to effective R of transistors), but C is important.
  - Typically around half of C of gate load is in the wires.
- For **long wires** on ICs:
  - busses, clock lines, global control signal, etc.
  - Resistance is significant, therefore distributed RC effect dominates.
  - signals are typically “rebuffered” to reduce delay:

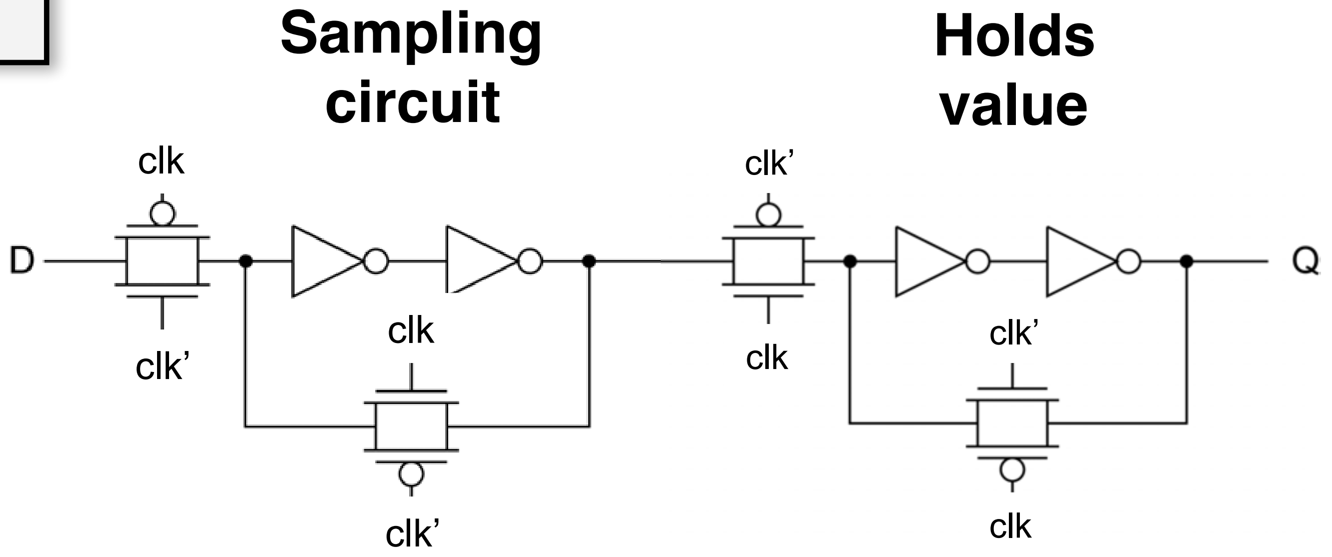




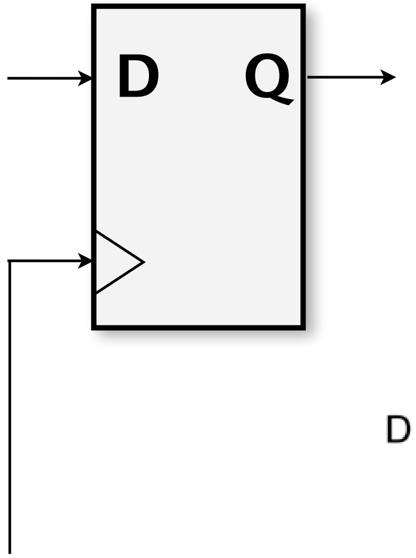
Recall: Positive edge-triggered flip-flop



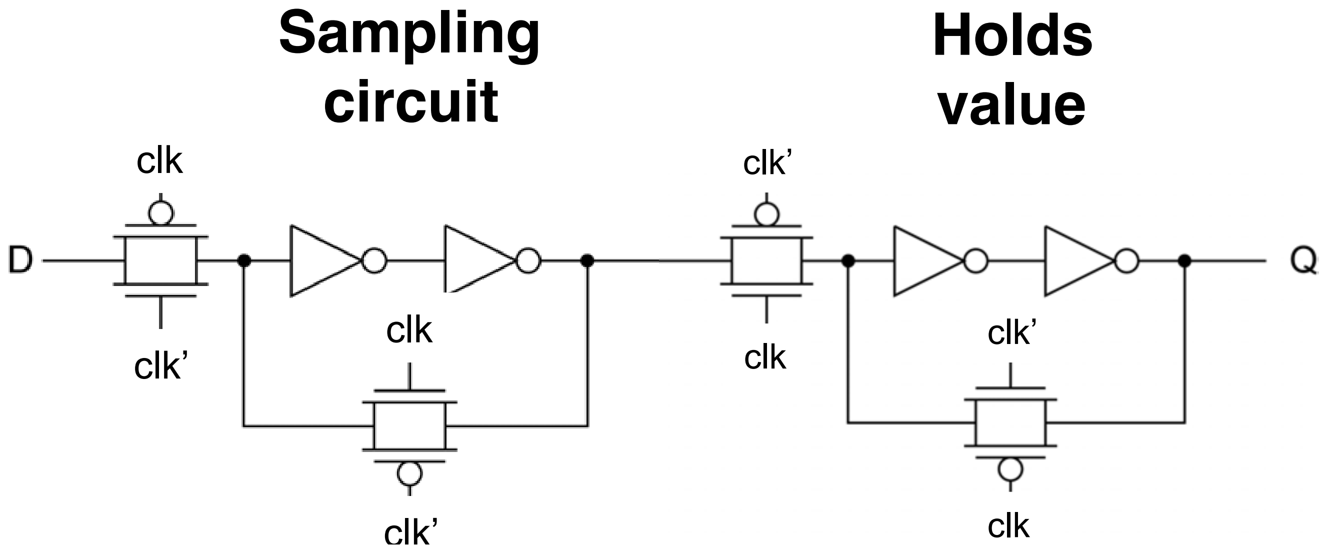
A flip-flop "samples" right before the edge, and then "holds" value.



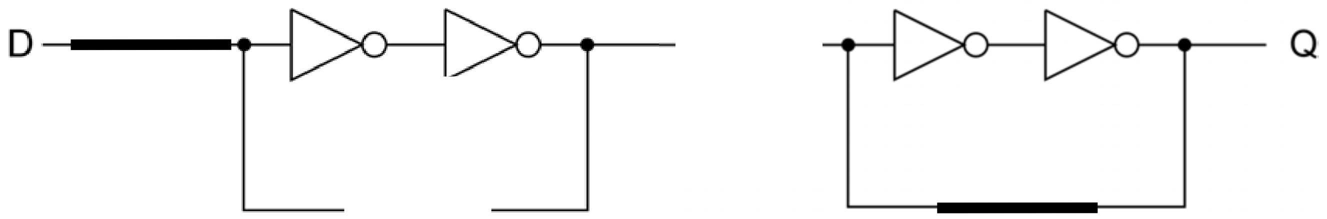
Sensing: When clock is low



A flip-flop "samples" right before the edge, and then "holds" value.



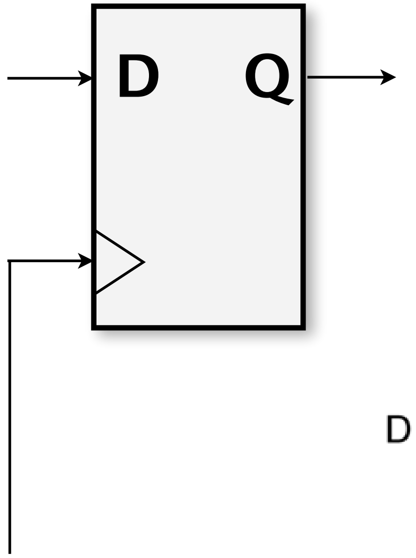
clk = 0
clk' = 1



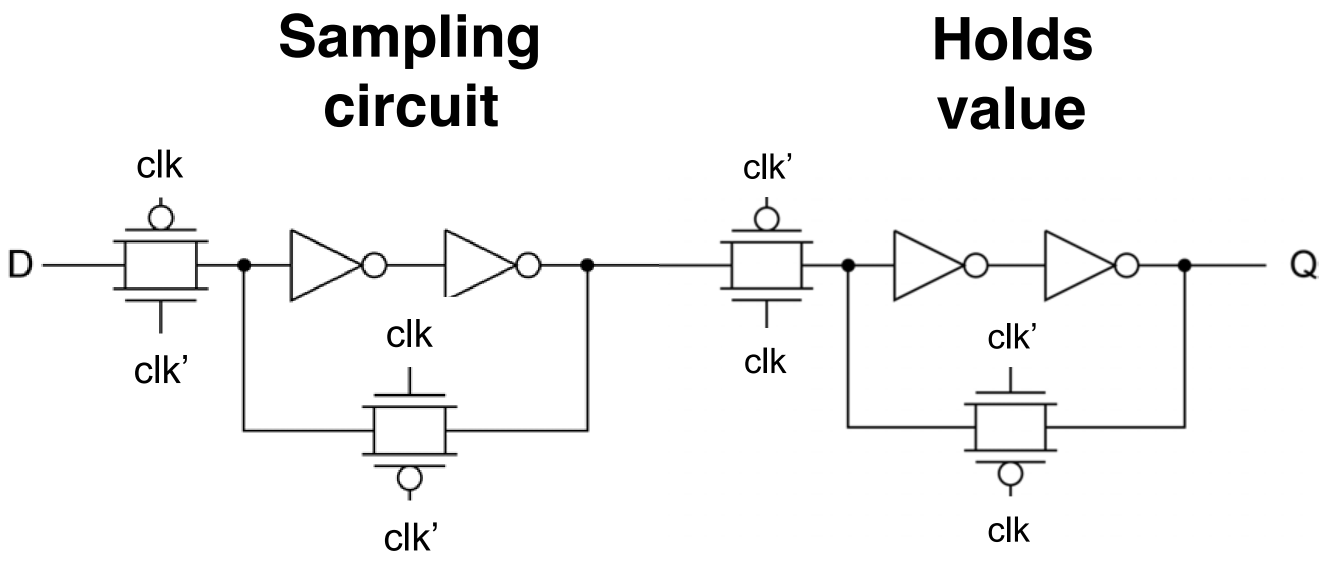
Will capture new value on posedge.

Outputs last value captured.

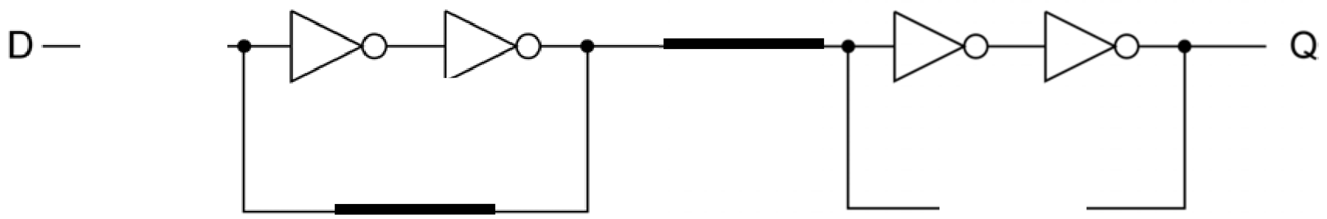
Capture: When clock goes high



A flip-flop "samples" right before the edge, and then "holds" value.



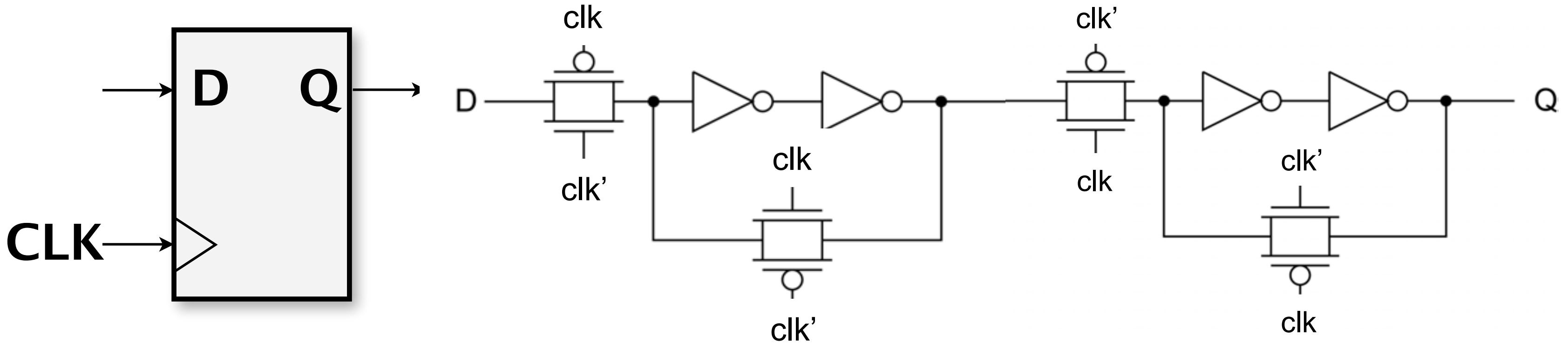
clk = 1
clk' = 0



Remembers value just captured.

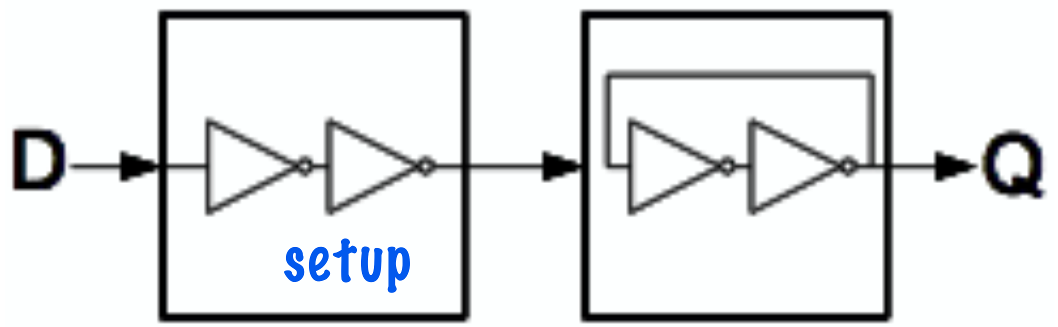
Outputs value just captured.

Flip Flop delays: clk-to-Q ? setup ? hold ?



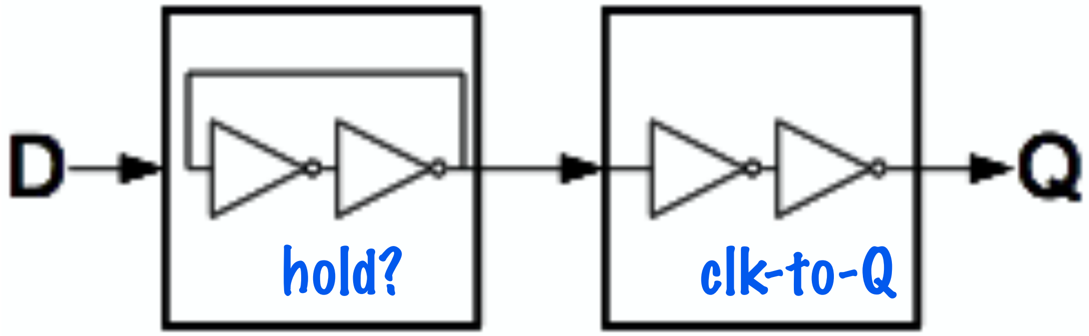
CLK == 0

Sense D, but Q outputs old value.



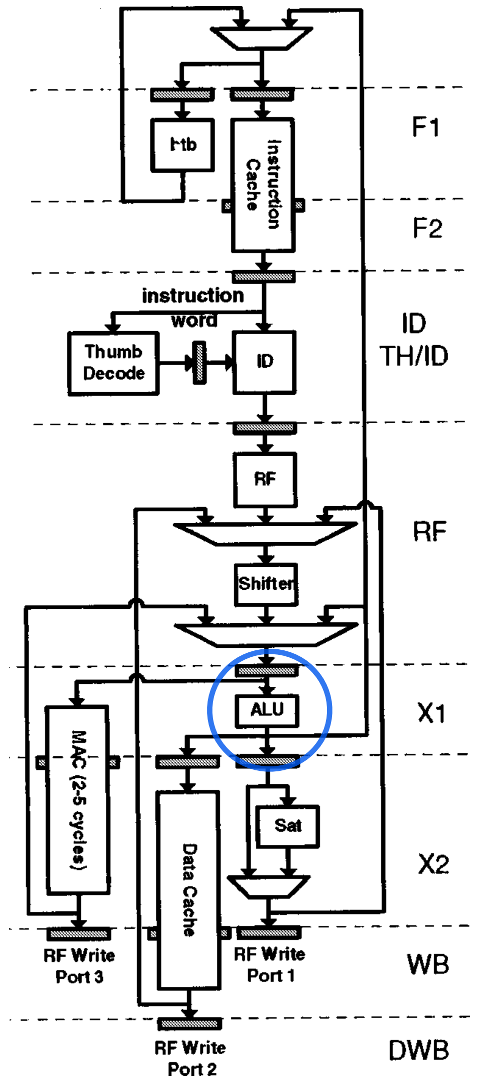
CLK 0->1

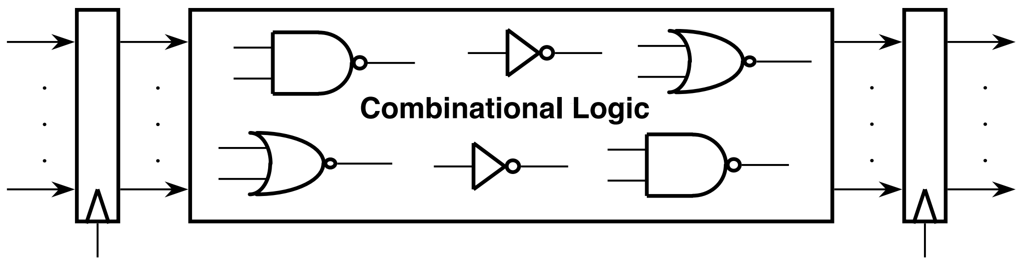
Capture D, pass value to Q

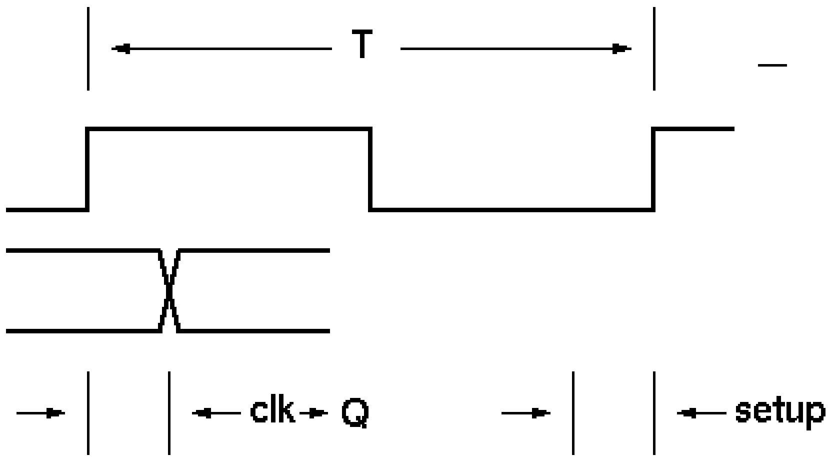


Note: with too much fanout, second stage could fail to capture data properly. Often output is rebuffered.

Flip-Flop delays eat into "time budget"



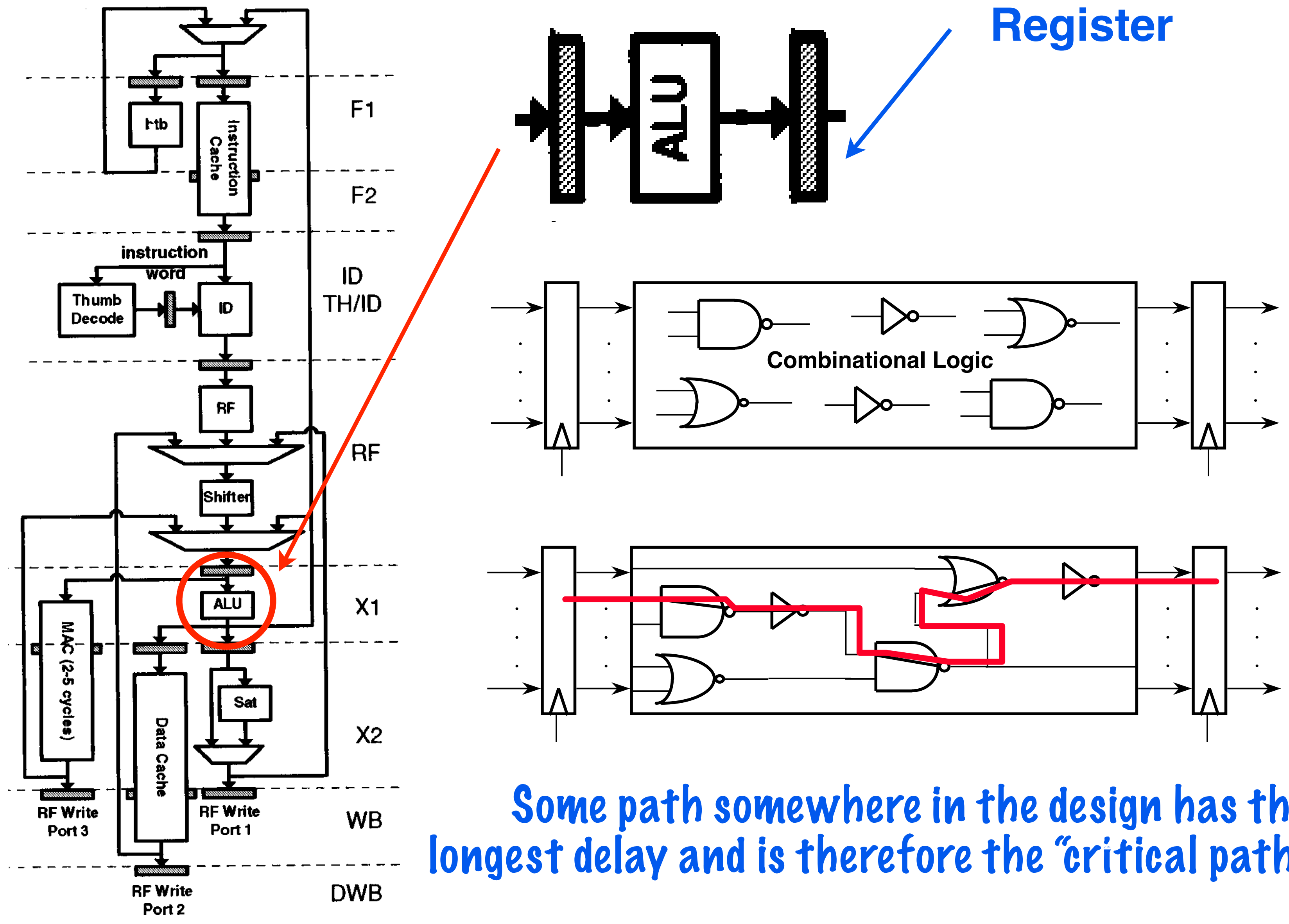




ALU "time budget"

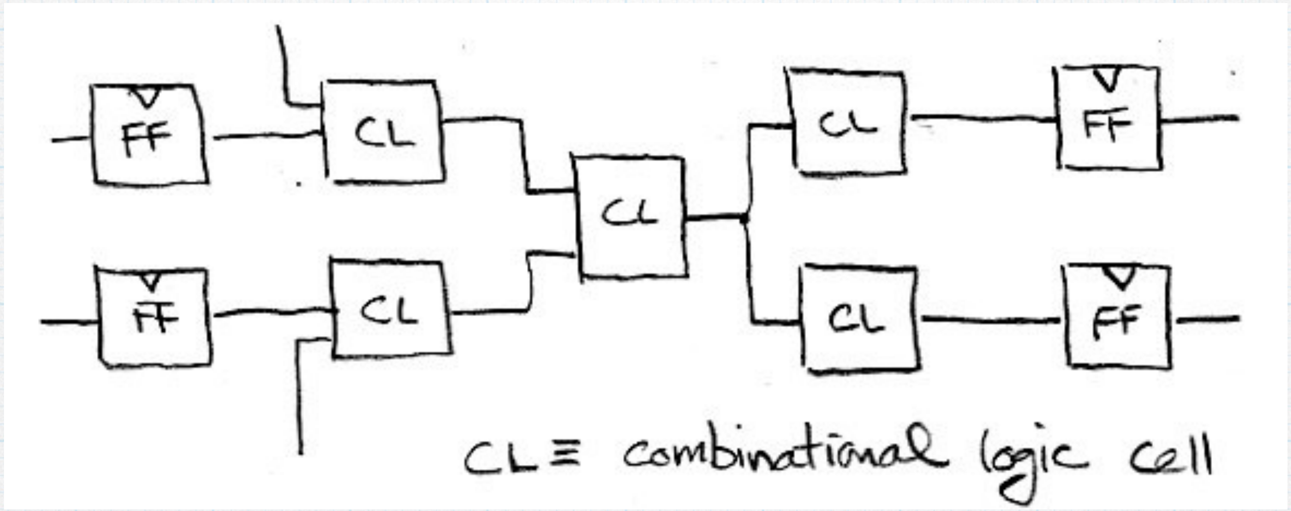
$$T \geq \tau_{clk \to Q} + \tau_{CL} + \tau_{setup}$$

Timing Analysis and Logic Delay



Some path somewhere in the design has the longest delay and is therefore the "critical path".

Components of Path Delay

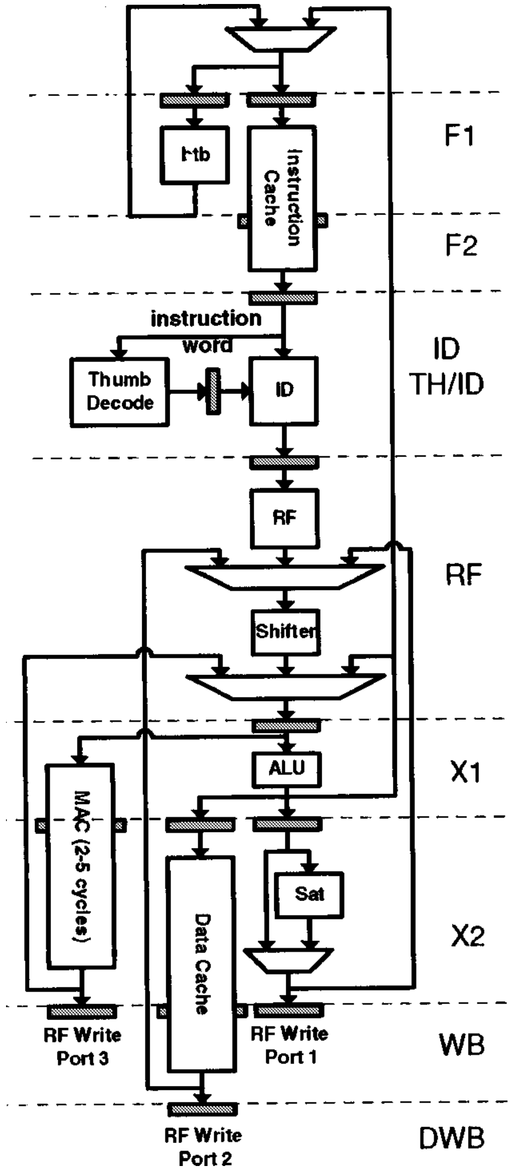


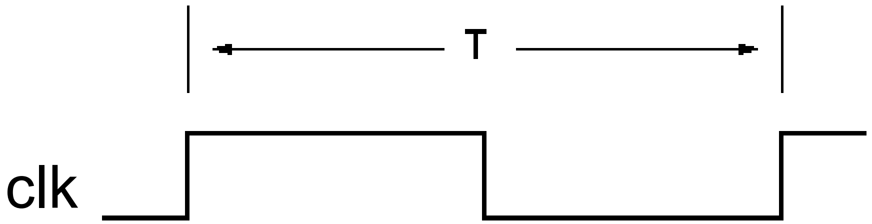
1. # of levels of logic
2. Internal cell delay
3. wire delay
4. cell input capacitance
5. cell fanout
6. cell output drive strength

Who controls the delay?

	foundary engineer (TSMC)	Library Developer (Aritsan)	CAD Tools (DC, IC Compiler)	Designer (you!)
1. # of levels			synthesis	RTL
2. Internal cell delay	physical parameters	cell topology, trans sizing	cell selection	
3. Wire delay	physical parameters		place & route	layout generator
4. Cell input capacitance	physical parameters	cell topology, trans sizing	cell selection	
5. Cell fanout			synthesis	RTL
6. Cell drive strength	physical parameters	transistor sizing	cell selection	

From Delay Models to Timing Analysis



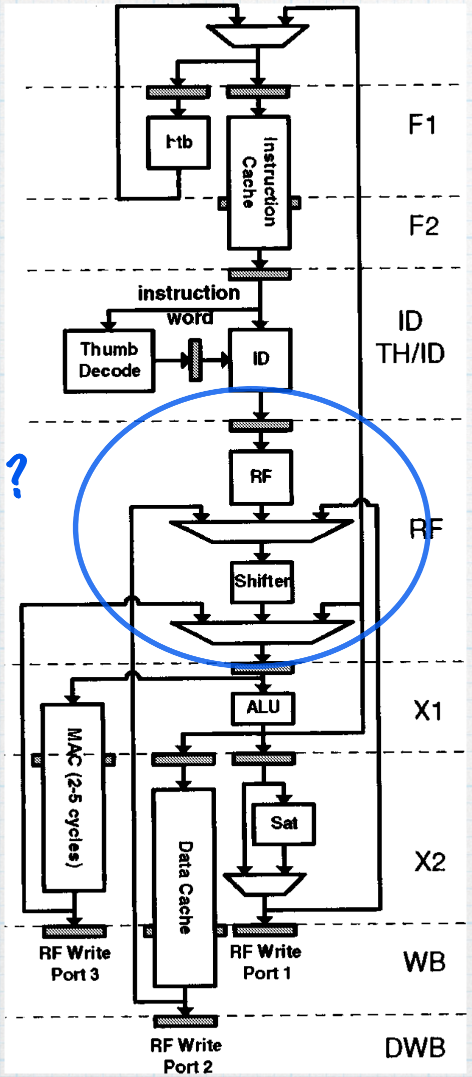


Timing Analysis

What is the smallest T that produces correct operation? Or, can we meet a target T?

f	T
1 MHz	1 µs
10 MHz	100 ns
100 MHz	10 ns
1 GHz	1 ns

Timing Closure: Searching for and beating down the critical path

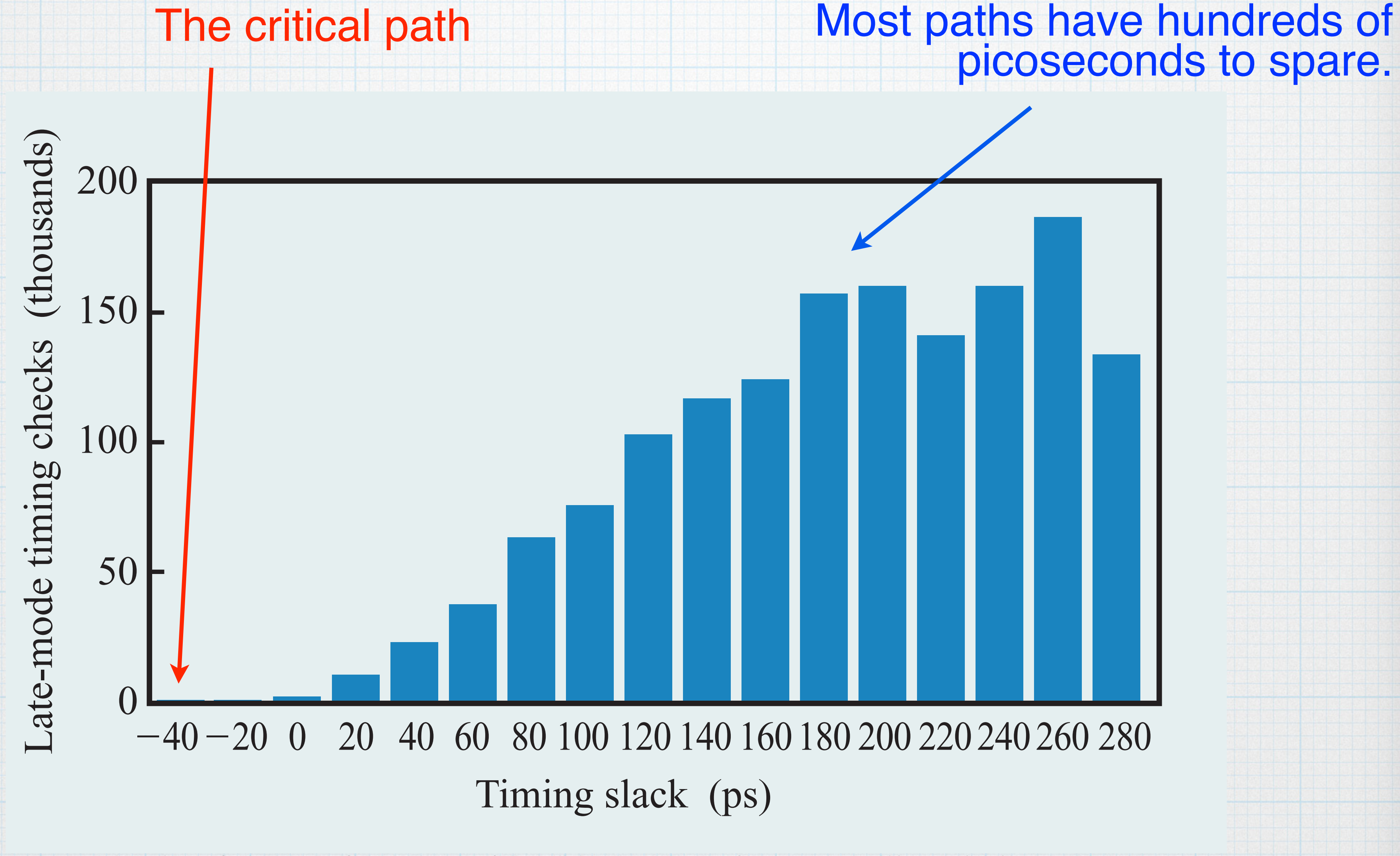


Must consider all connected register pairs, paths, plus from input to register, plus register to output.

- Design tools help in the search.
- **Synthesis tools** work to meet clock constraint, report delays on paths,
  - Special **static timing analyzers** accept a design netlist and report path delays,
  - and, of course, **simulators** can be used to determine timing performance.

Tools that are expected to **do something** about the timing behavior (such as synthesizers), also include provisions for specifying input arrival times (relative to the clock), and output requirements (set-up times of next stage).

Timing Analysis, real example



From "The circuit and physical design of the POWER4 microprocessor", IBM J Res and Dev, 46:1, Jan 2002, J.D. Warnock et al.

Timing Optimization

As an ASIC/FPGA designer you get to choose:

- The algorithm
- The Microarchitecture (block diagram)
- The RTL description of the CL blocks (number of levels of logic)
- Where to place registers and memory (the pipelining)
- Overall floorplan and relative placement of blocks

How to retime logic

Circles are combinational logic, labelled with delays.

Critical path is 5. We want to improve it without changing circuit semantics.

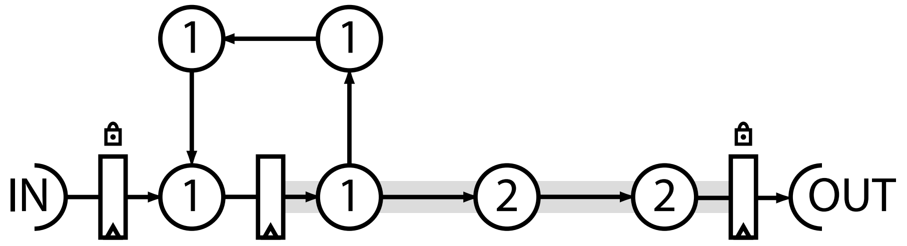


Figure 1: A small graph before retiming. The nodes represent logic delays, with the inputs and outputs passing through mandatory, fixed registers. The critical path is 5.

Add a register, move one circle. Performance improves by 20%.

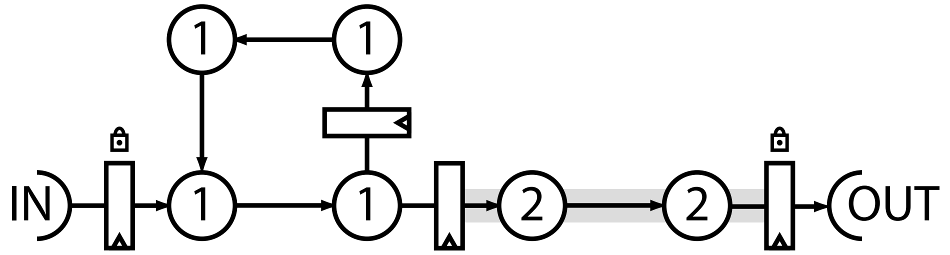
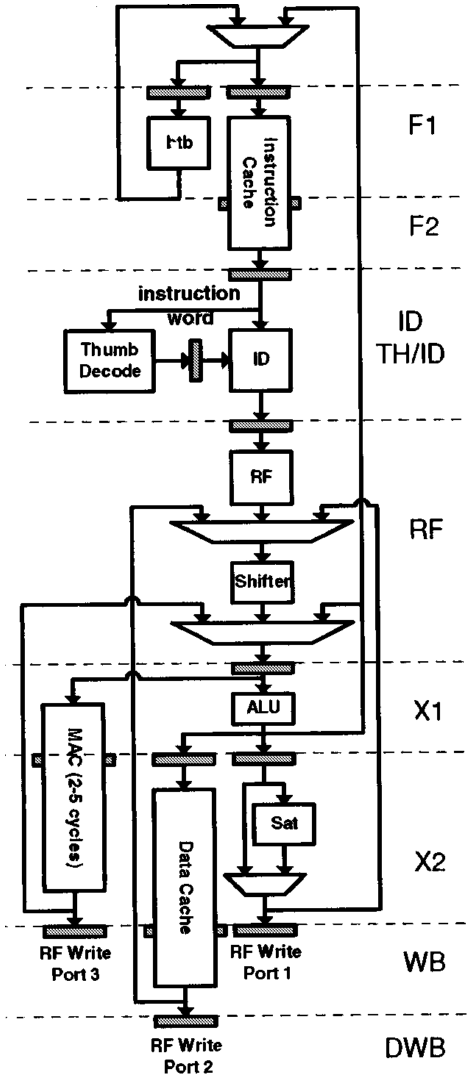
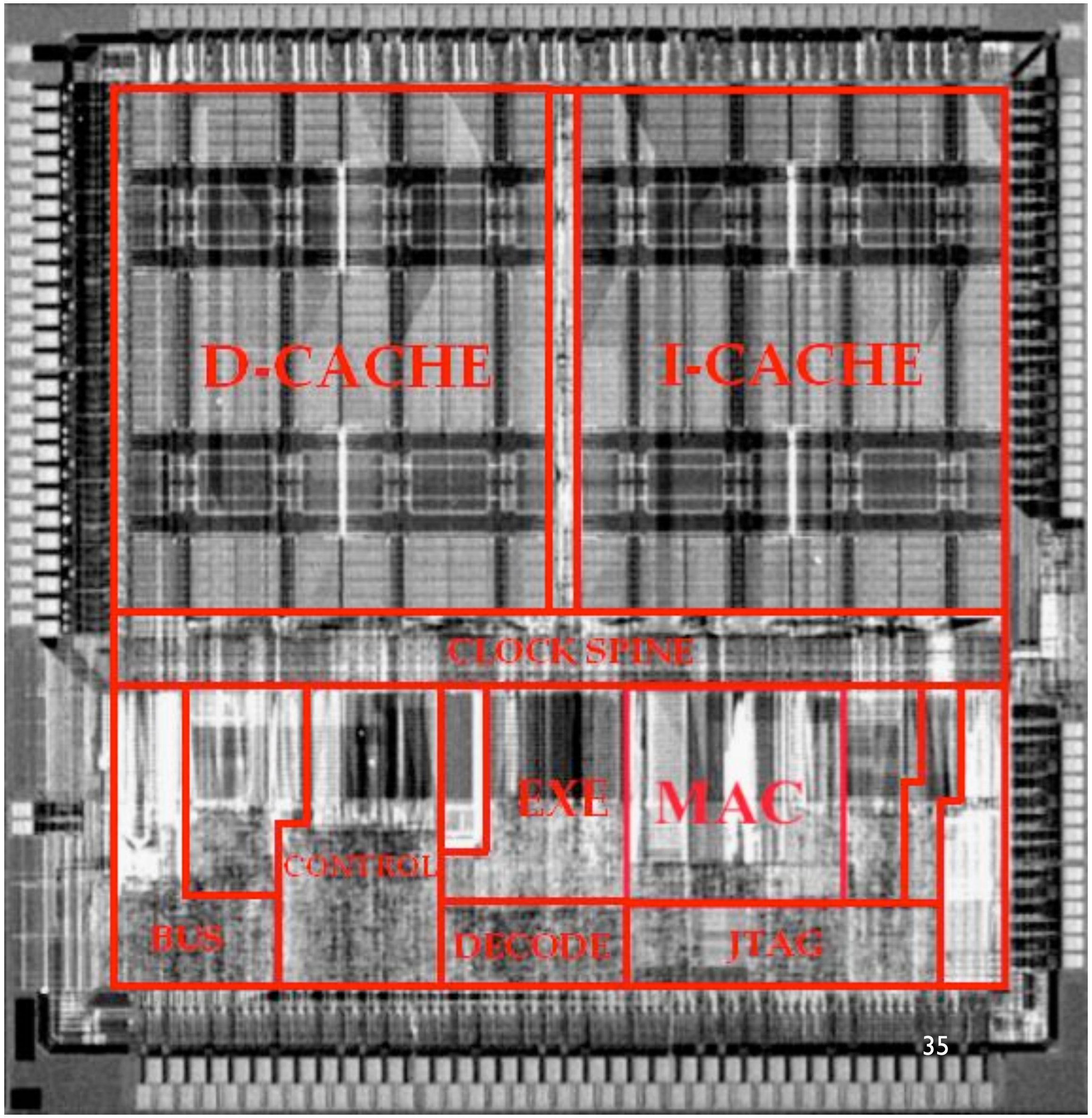


Figure 2: The example in Figure 2 after retiming. The critical path is reduced from 5 to 4.

Logic Synthesis tools can do this in simple cases.

Floorplaning: essential to meet timing.

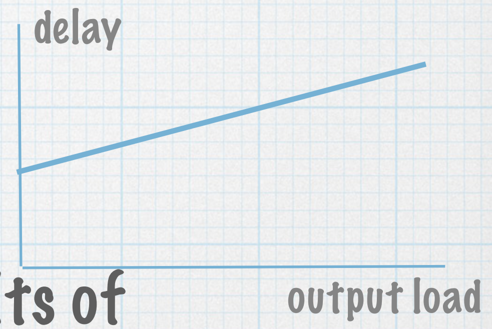




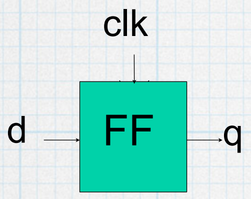
(Intel XScale 80200)

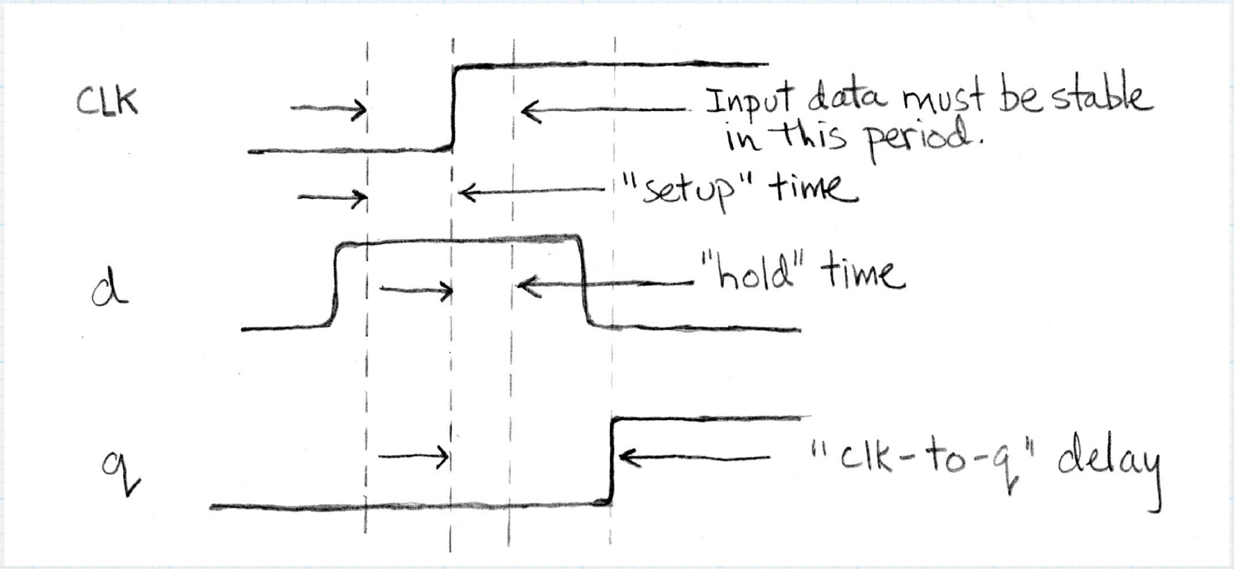
Timing Analysis Tools

- Static Timing Analysis: Tools use delay models for gates and interconnect. Traces through circuit paths.
  - Cell delay model capture
    - For each input/output pair, internal delay (output load independent)
    - output dependent delay
- Standalone tools (PrimeTime) and part of logic synthesis.
- Back-annotation takes information from results of place and route to improve accuracy of timing analysis.
- DC in "topographical mode" uses preliminary layout information to model interconnect parasitics.
  - Prior versions used a simple fan-out model of gate loading.



Hold-time Violations





- Some state elements have positive hold time requirements.
  - How can this be?
- Fast paths from one state element to the next can create a violation. (Think about shift registers!)
- CAD tools do their best to fix violations by inserting delay (buffers).
  - Of course, if the path is delayed too much, then cycle time suffers.
  - Difficult because buffer insertion changes layout, which changes path delay.

Conclusion

- Timing Optimization: You start with a target on clock period. What control do you have?
- Biggest effect is RTL manipulation.
  - i.e., how much logic to put in each pipeline stage.
  - We will be talking later about how to manipulate RTL for better timing results.
- In most cases, the tools will do a good job at logic/circuit level:
  - Logic level manipulation
  - Transistor sizing
  - Buffer insertion
  - But some cases may be difficult and you may need to help
- The tools will need some help at the floorpan and layout