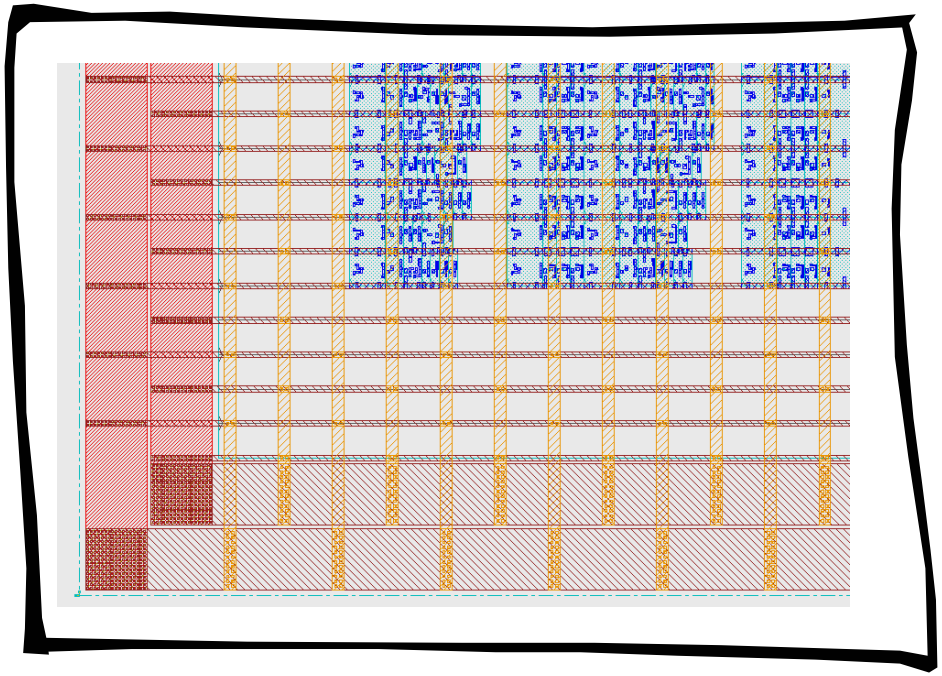
Managing Physical Design Issues in ASIC Toolflows



6.375 Complex Digital Systems

Christopher Batten

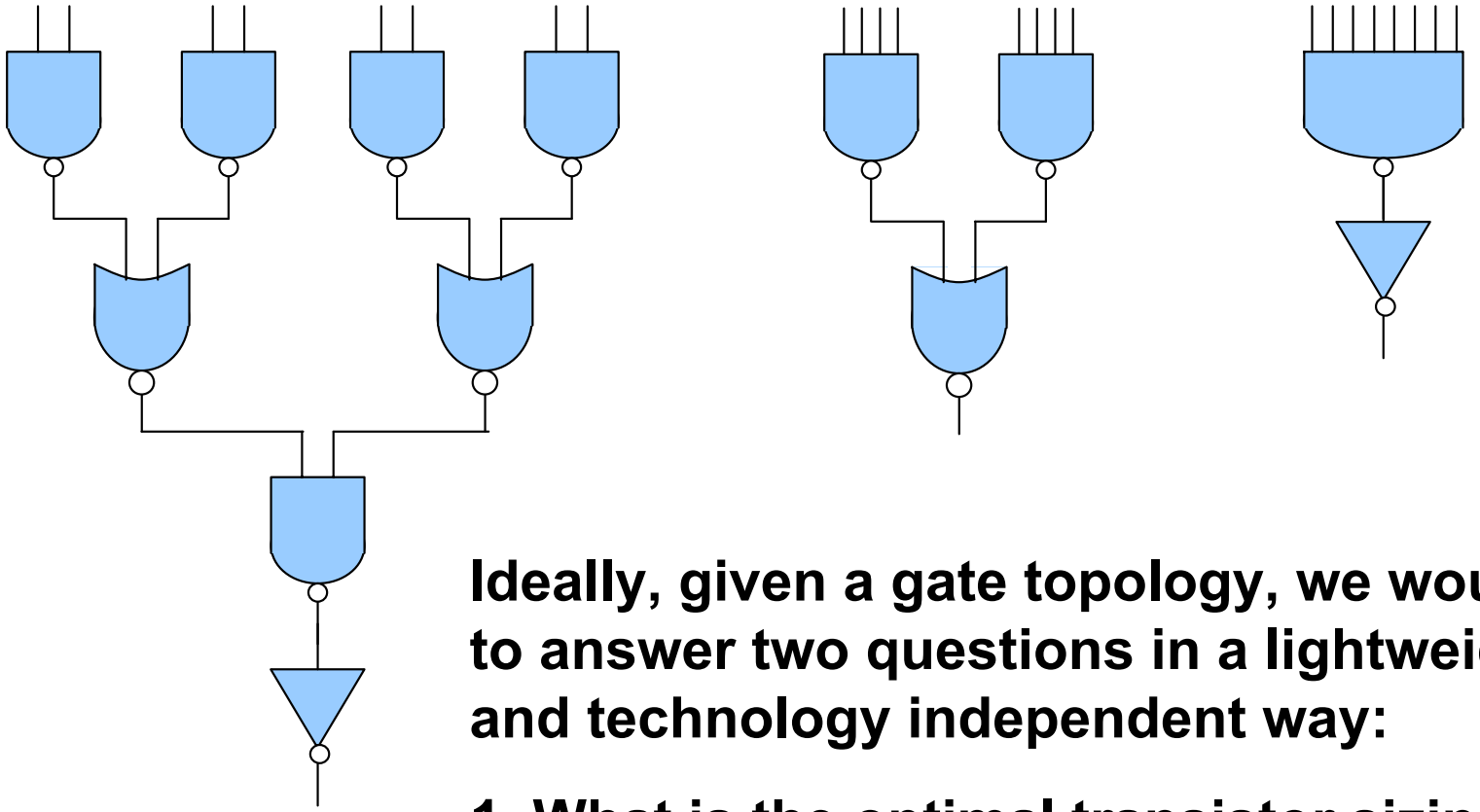
February 21, 2006

Managing Physical Design Issues in ASIC Toolflows

- Logical Effort
- Physical Design Issues
  - Clock Distribution
  - Power Distribution
  - Wire Delay
  - Power Consumption
  - Capacitive Coupling

1. **What is the issue?**
2. **How do custom designers address the issue?**
3. **How can we approximate these approaches in an ASIC toolflow?**

Which gate topology and transistor sizing is optimal?

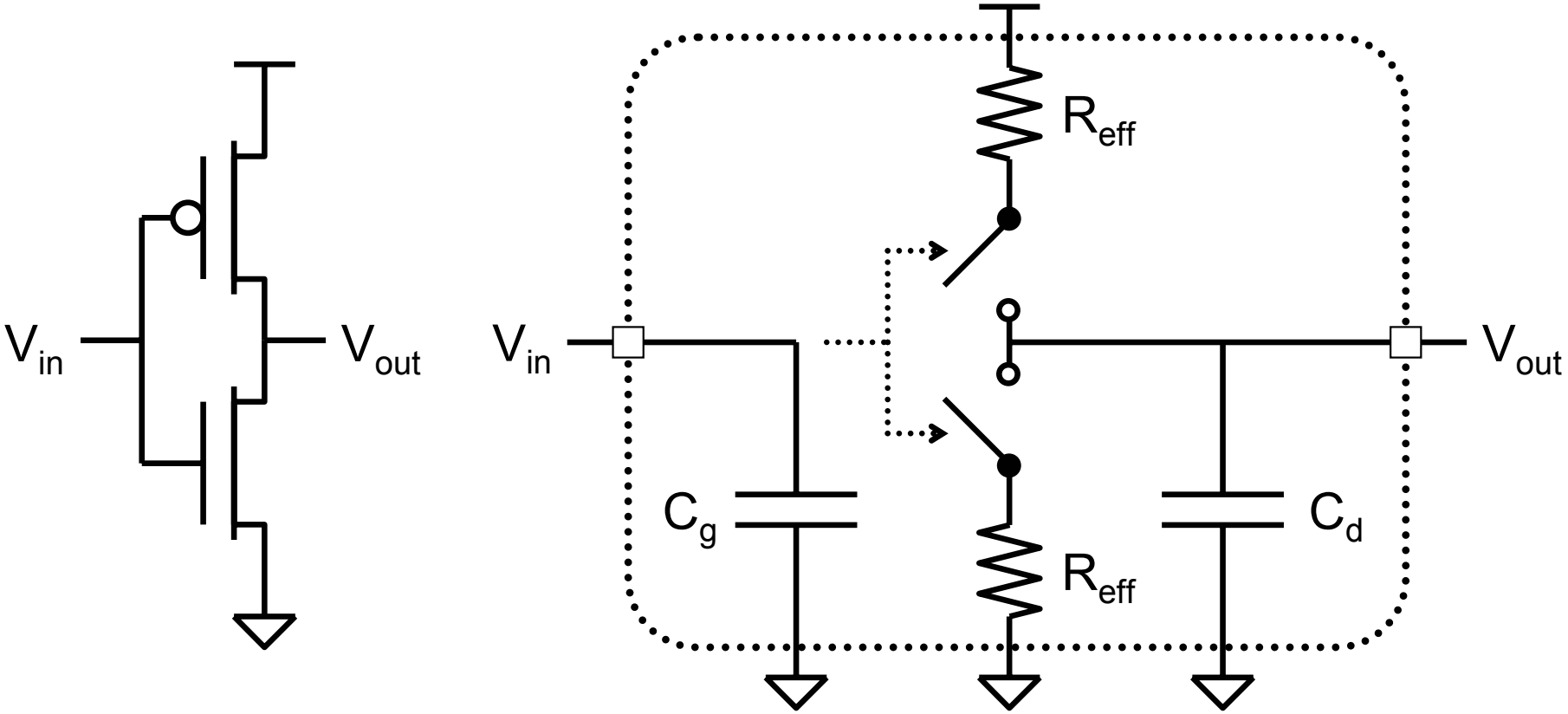


Ideally, given a gate topology, we would like to answer two questions in a lightweight and technology independent way:

1. What is the optimal transistor sizing?

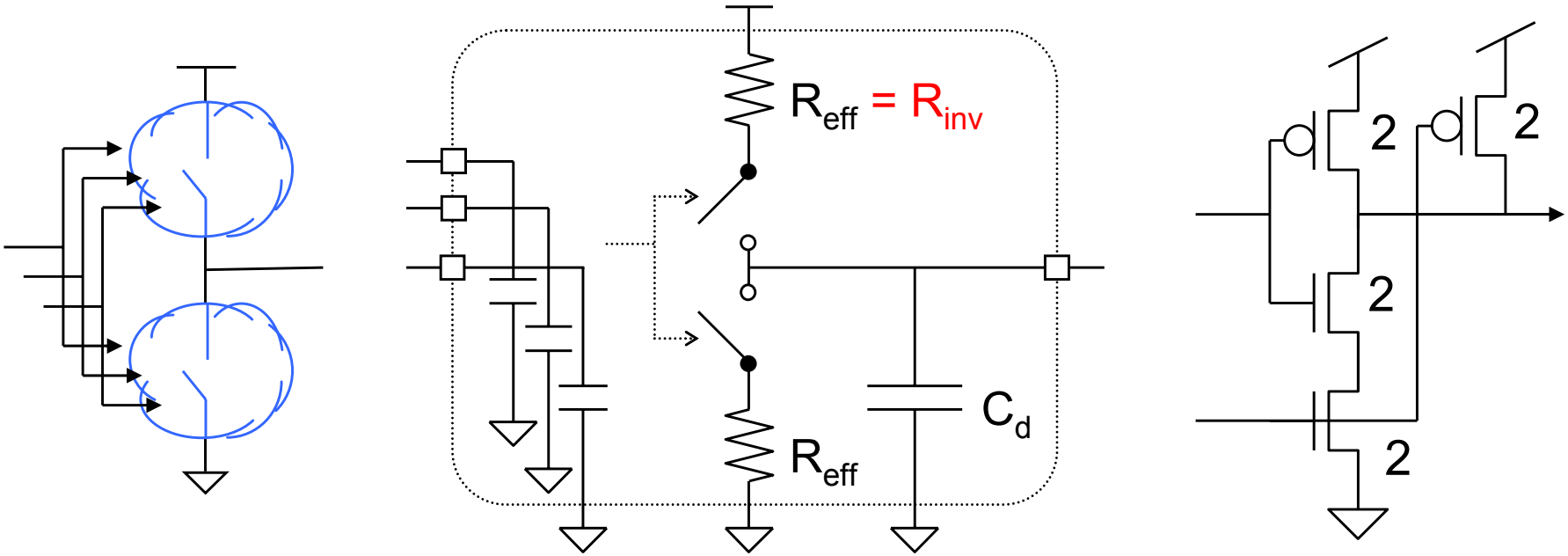
2. What is the optimal number of stages?

Review of the simple RC model for the CMOS inverter



$$R_{eff} = R_{eff,N} = R_{eff,P}$$
$$C_g = C_{g,N} + C_{g,P}$$
$$C_d = C_{d,N} + C_{d,P}$$

A gate template is gate with same drive current as minimum sized inverter



We begin by deriving an equation for unitless delay in terms of a template

Determine RC for an actual gate relative to the template

$$C_{in} = \alpha \cdot C_{in,T} \quad C_p = \alpha \cdot C_{p,T} \quad R_{EFF} = \frac{R_{EFF,T}}{\alpha}$$

Derive absolute delay in terms of the template

$$d_{abs} = K \cdot R_{EFF}\left(C_{out} + C_p\right) = K \cdot R_{EFF} \cdot C_{out} + K \cdot R_{EFF} \cdot C_p = K \cdot R_{EFF} \cdot C_{in} \frac{C_{out}}{C_{in}} + K \cdot R_{EFF} \cdot C_p$$

$$= K \cdot \frac{R_{EFF,T}}{\alpha} \cdot \alpha C_{in,T} \frac{C_{out}}{C_{in}} + K \cdot \frac{R_{EFF,T}}{\alpha} \cdot \alpha C_{p,T} = \underbrace{K \cdot R_{EFF,T} \cdot C_{in,T}}_{\text{Independent of actual transistor widths}} \frac{C_{out}}{C_{in}} + \underbrace{K \cdot R_{EFF,T} \cdot C_{p,T}}_{\text{Independent of actual transistor widths}}$$

Function of transistor widths in template

Should be rough

We begin by deriving an equation for unitless delay in terms of a template

Determine RC for an actual gate relative to the template

$$C_{in} = \alpha \cdot C_{in,T} \quad C_p = \alpha \cdot C_{p,T} \quad R_{EFF} = \frac{R_{EFF,T}}{\alpha}$$

Derive absolute delay in terms of the template

$$d_{abs} = K \cdot R_{EFF}\left(C_{out} + C_p\right) = K \cdot R_{EFF} \cdot C_{out} + K \cdot R_{EFF} \cdot C_p = K \cdot R_{EFF} \cdot C_{in} \frac{C_{out}}{C_{in}} + K \cdot R_{EFF} \cdot C_p$$

$$= K \cdot \frac{R_{EFF,T}}{\alpha} \cdot \alpha C_{in,T} \frac{C_{out}}{C_{in}} + K \cdot \frac{R_{EFF,T}}{\alpha} \cdot \alpha C_{p,T} = K \cdot R_{EFF,T} \cdot C_{in,T} \frac{C_{out}}{C_{in}} + K \cdot R_{EFF,T} \cdot C_{p,T}$$

Function of the actual transistor widths
Also called the gate "fanout"

We begin by deriving an equation for unitless delay in terms of a template

Determine RC for an actual gate relative to the template

$$C_{in} = \alpha \cdot C_{in,T} \quad C_p = \alpha \cdot C_{p,T} \quad R_{EFF} = \frac{R_{EFF,T}}{\alpha}$$

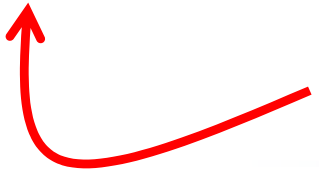
Derive absolute delay in terms of the template

$$d_{abs} = K \cdot R_{EFF}\left(C_{out} + C_p\right) = K \cdot R_{EFF} \cdot C_{out} + K \cdot R_{EFF} \cdot C_p = K \cdot R_{EFF} \cdot C_{in} \frac{C_{out}}{C_{in}} + K \cdot R_{EFF} \cdot C_p$$

$$= K \cdot \frac{R_{EFF,T}}{\alpha} \cdot \alpha C_{in,T} \frac{C_{out}}{C_{in}} + K \cdot \frac{R_{EFF,T}}{\alpha} \cdot \alpha C_{p,T} = K \cdot R_{EFF,T} \cdot C_{in,T} \frac{C_{out}}{C_{in}} + K \cdot R_{EFF,T} \cdot C_{p,T}$$

Normalize this delay to the delay of an min inverter with no parasitics

$$d = \frac{d_{abs}}{\tau} = \frac{K \cdot R_{EFF,T} \cdot C_{in,T}}{K \cdot R_{inv} \cdot C_{inv}} \frac{C_{out}}{C_{in}} + \frac{K \cdot R_{EFF,T} \cdot C_{p,T}}{K \cdot R_{inv} \cdot C_{inv}} = \frac{R_{EFF,T} \cdot C_{in,T}}{R_{inv} \cdot C_{inv}} \times \frac{C_{out}}{C_{in}} + \frac{R_{EFF,T} \cdot C_{p,T}}{R_{inv} \cdot C_{inv}}$$



For our 0.18um technology, $\tau \approx 10$ps

We begin by deriving an equation for unitless delay in terms of a template

$$d = \frac{d_{abs}}{\tau} = \underbrace{\frac{R_{EFF,T} \cdot C_{in,T}}{R_{inv} \cdot C_{inv}}}_{\text{Logical Effort (g)}} \times \underbrace{\frac{C_{out}}{C_{in}}}_{\text{Electrical Effort (h)}} + \underbrace{\frac{R_{EFF,T} \cdot C_{p,T}}{R_{inv} \cdot C_{inv}}}_{\text{Parasitic Delay (p)}}$$

Parasitic Delay is relative to a minimum sized inverter and is roughly independent of actual transistor widths

Electrical Effort is the fanout of the gate and is a function of actual transistor widths

Logical Effort compares characteristic RC time constant of gate to minimum sized inverter and is independent of actual transistor widths

We begin by deriving an equation for unitless delay in terms of a template

$$d = \frac{d_{abs}}{\tau} = \frac{R_{EFF,T} \cdot C_{in,T}}{R_{inv} \cdot C_{inv}} \times \frac{C_{out}}{C_{in}} + \frac{R_{EFF,T} \cdot C_{p,T}}{R_{inv} \cdot C_{inv}}$$

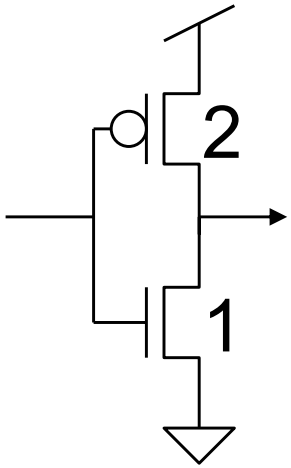
$$d = \frac{d_{abs}}{\tau} = g \times h + p$$

Parasitic Delay is relative to a minimum sized inverter and is roughly independent of actual transistor widths

Electrical Effort is the fanout of the gate and is a function of actual transistor widths

Logical Effort compares characteristic RC time constant of gate to minimum sized inverter and is independent of actual transistor widths

Logical effort is simply ratio of input cap to min inverter with same current drive

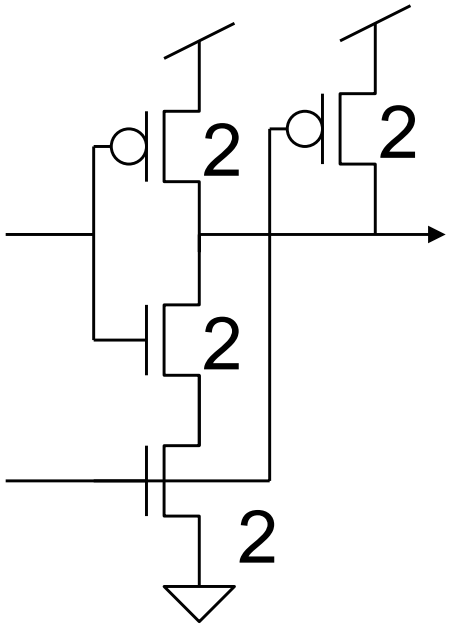


Inverter

Input Cap = 3 units

g = 1 (definition)

p = 1

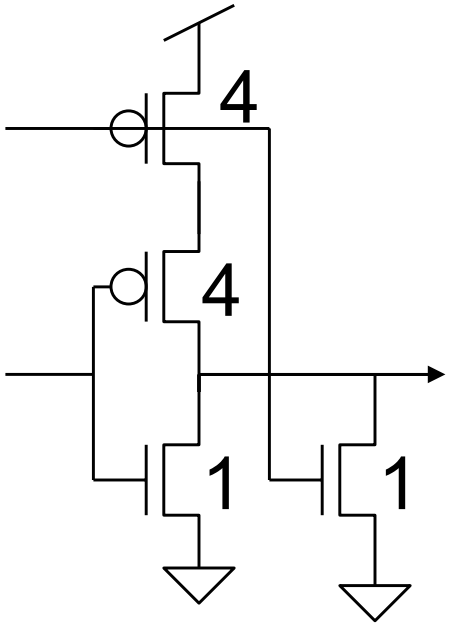


NAND

Input Cap = 4 units

g = 4/3

p = 6/3 = 2



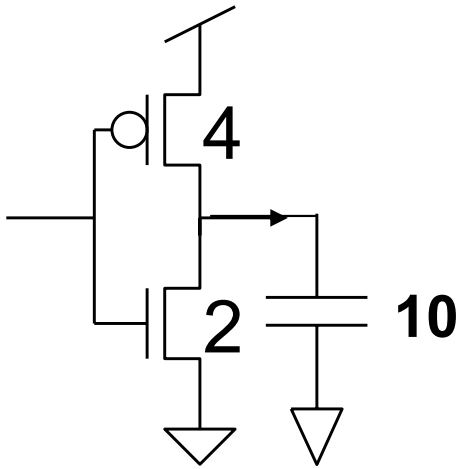
NOR

Input Cap = 5 units

g = 5/3

p = 6/3 = 2

Examples illustrating unit-less delay of gates with equal drive strength



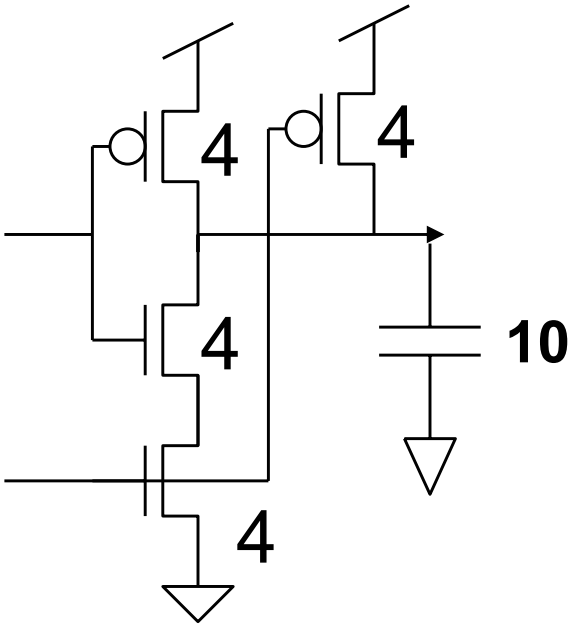
Inverter

g = 1 (definition)

p = 1

h = 10/6 = 1.67

d = gh+p = 2.67



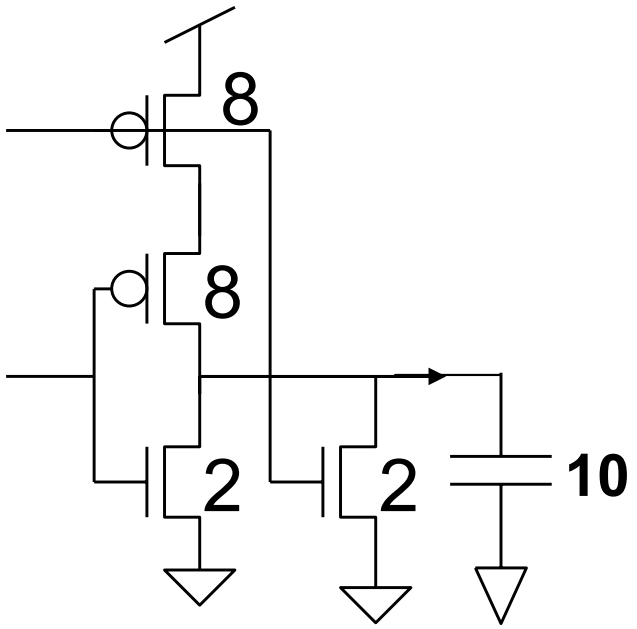
NAND

g = 4/3

p = 2

h = 10/8 = 1.25

d = gh+p = 3.67



NOR

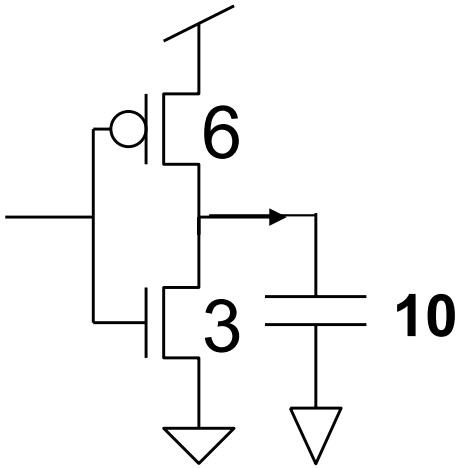
g = 5/3

p = 2

h = 10/10 = 1

d = gh+p = 3.67

Examples illustrating unit-less delay of gates with similar area



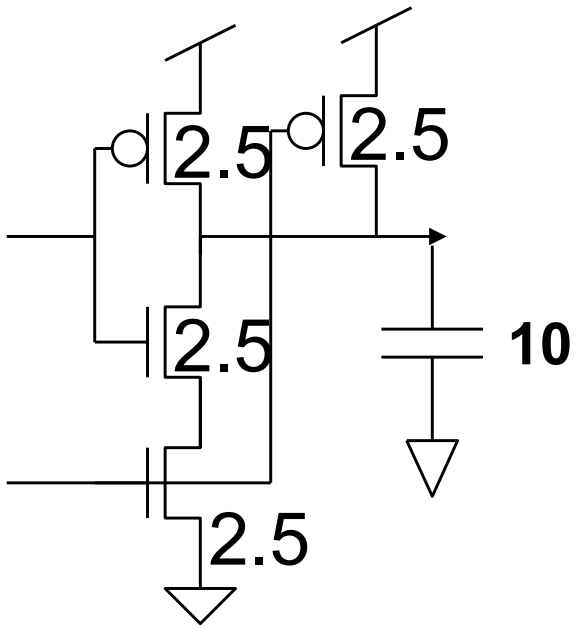
Inverter

g = 1 (definition)

p = 1

h = 10/9 = 1.11

d = gh+p = 2.11



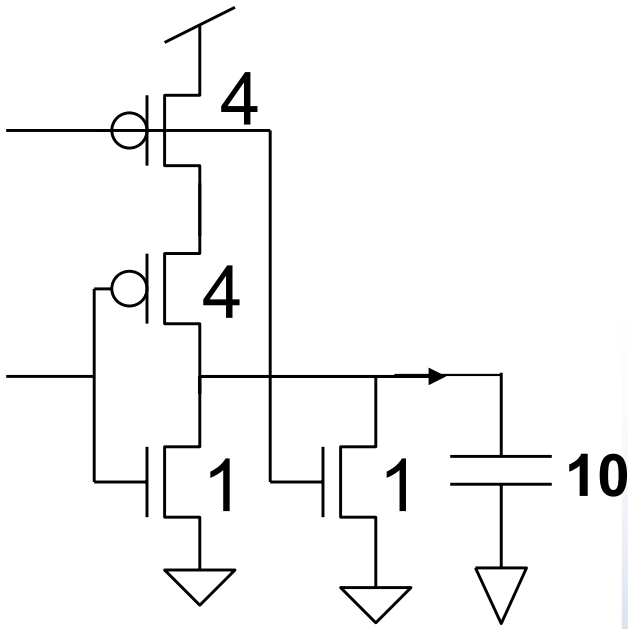
NAND

g = 4/3

p = 2

h = 10/5 = 2

d = gh+p = 4.67



NOR

g = 5/3

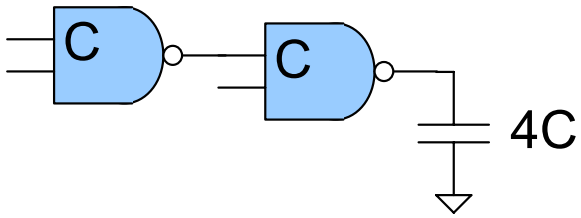
p = 2

h = 10/5 = 2

d = gh+p = 5.33

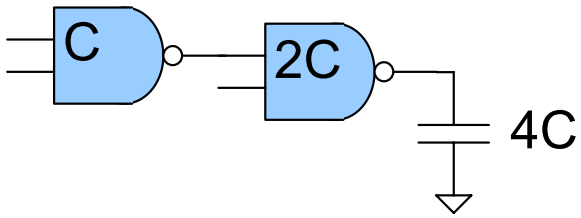
Path delay (D) is just the sum of the stage delays

$$D = \sum_i d_i = \sum_i \left(g_i \times h_i + p_i\right) = \sum_i \left(g_i \times h_i\right) + \sum_i p_i$$



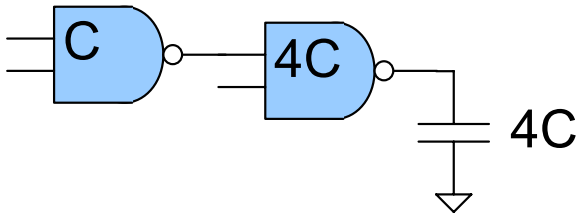
(4/3)x(C/C)
+ (4/3)x(4C/C)
+ 4

= 10.67



(4/3)x(2C/C)
+ (4/3)x(4C/2C)
+ 4

= 9.33



(4/3)x(4C/C)
+ (4/3)x(4C/4C)
+ 4

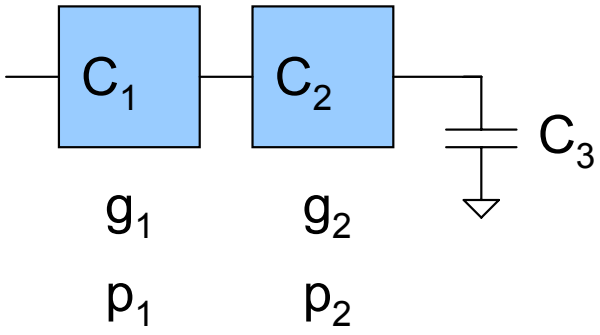
= 10.67

What is the optimal delay for any general two stage topology?

Form unitless delay equation
Only free variable is C_2

$$D = (g_1 h_1 + p_1) + (g_2 h_2 + p_2)$$

$$= \left(g_1 \frac{C_2}{C_1} + p_1\right) + \left(g_2 \frac{C_3}{C_2} + p_2\right)$$



Minimize with respect to C_2

$$\frac{\partial D}{\partial C_2} = \frac{g_1}{C_1} - \frac{g_2 C_3}{(C_2)^2} = 0$$

Minimal delay occurs when stage effort is equal

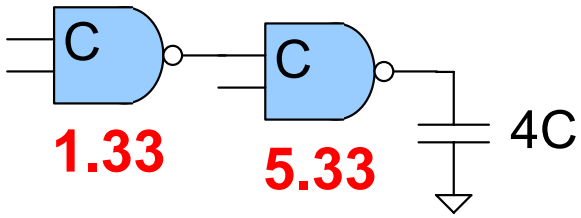
$$\frac{g_1}{C_1} = \frac{g_2 C_3}{(C_2)^2}$$

$$g_1 \frac{C_2}{C_1} = g_2 \frac{C_3}{C_2}$$

$$g_1 h_1 = g_2 h_2$$

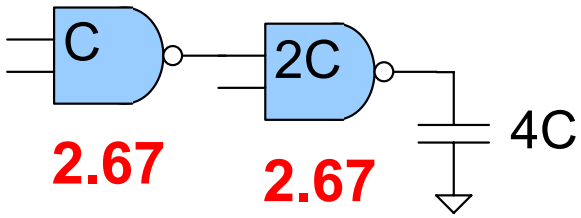
Key Result: Delay is minimized when effort is shared equally among stages

$$D = \sum_i d_i = \sum_i \left(g_i \times h_i + p_i\right) = \sum_i \left(g_i \times h_i\right) + \sum_i p_i$$



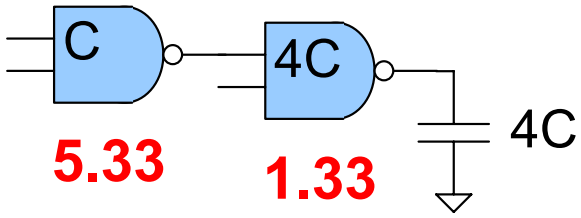
(4/3)x(C/C)
+ (4/3)x(4C/C)
+ 4

= 10.67



(4/3)x(2C/C)
+ (4/3)x(4C/2C)
+ 4

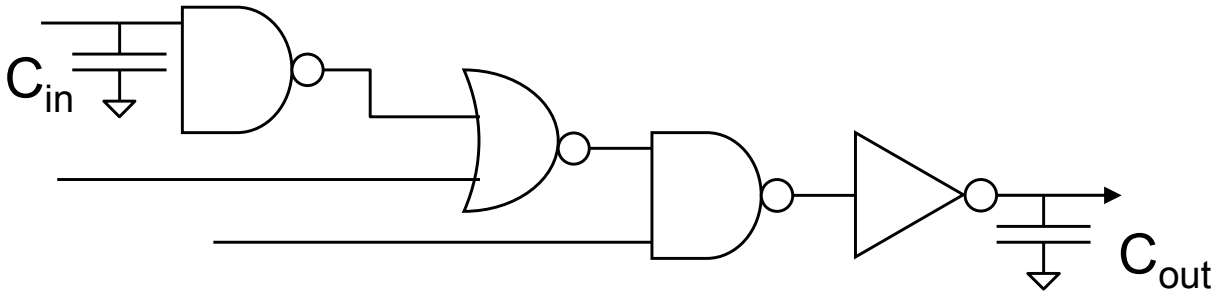
= 9.33



(4/3)x(4C/C)
+ (4/3)x(4C/4C)
+ 4

= 10.67

We now generalize this result with some additional terminology



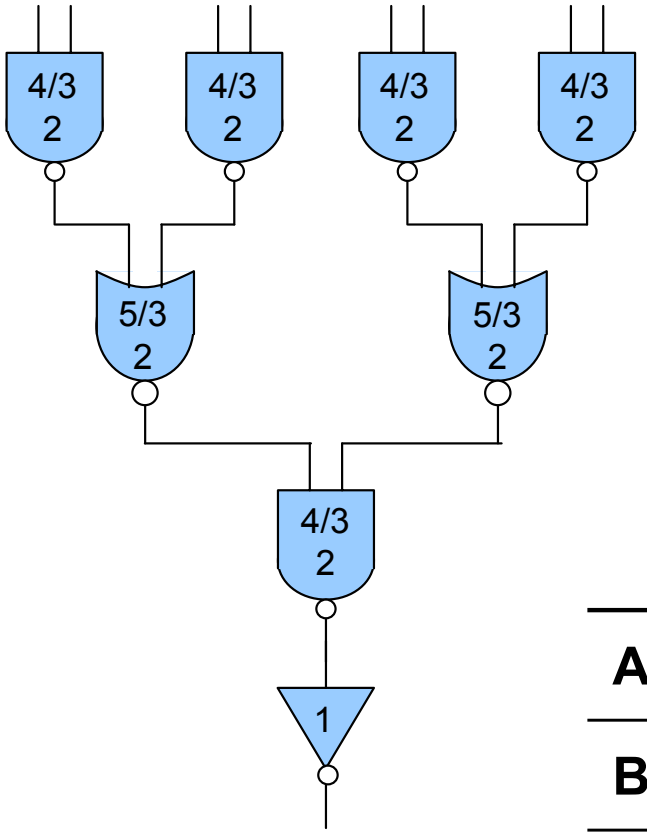
Path delay	$D = \Sigma\, d_i = \Sigma\, g_i h_i + \Sigma\, p_i$	Sum of stage delays
Path logical effort	$G = \Pi\, g_i$	Product of stage LE
Path electrical effort	$H = \Pi\, h_i = C_{out}/C_{in}$	Product of stage EE (Internal C's cancel out)
Path effort	$F = \Pi\, f_i = \Pi\, (g_i h_i) = GH$	Product of stage efforts
Optimal stage effort for N stages	$f_{OPT} = F^{1/N}$	Optimal delay when $g_1h_1 = g_2h_2 = \ldots = g_Nh_N$
Optimal path delay	$D_{OPT} = \Sigma\, f_{OPT} + \Sigma\, p_i$	

Steps for transistor sizing

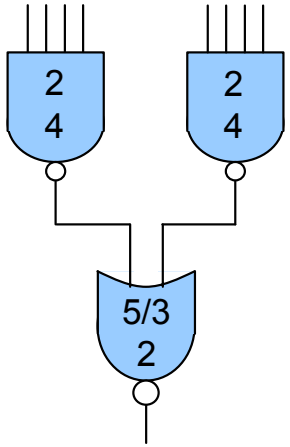
1. Calculate path effort
2. Calculate optimal path delay
3. Assign each stage equal effort
4. Work from C_{out} backwards assigning C_{in} values for each stage
5. Convert C_{in} values into transistors sizes

Finding the path effort and optimal delay (Steps 1 and 2)

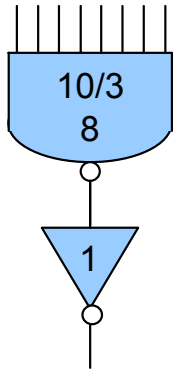
Topology A



Topology B

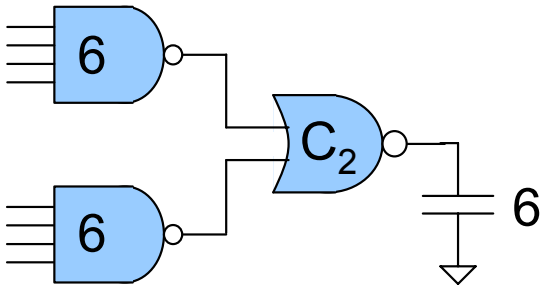


Topology C



	G	N	P	D_{OPT}	H=1	H=12
A	2.96	4	7	$4(2.96H)^{1/4} + 7$	12.25	**16.77**
B	3.33	2	6	$2(3.33H)^{1/2} + 6$	**9.65**	18.64
C	3.33	2	9	$2(3.33H)^{1/2} + 9$	12.65	21.64

Finding actual transistor sizes for H=1 case (Steps 3-5)



Path Effort (F) = GH = (3.33)(1) = 3.33

Divide path effort equally among stages

$$F_{OPT} = F^{1/N} = (3.33)^{1/2} = 1.82$$

C_{out} and C_{in} are given in equivalent gate transistor width cap

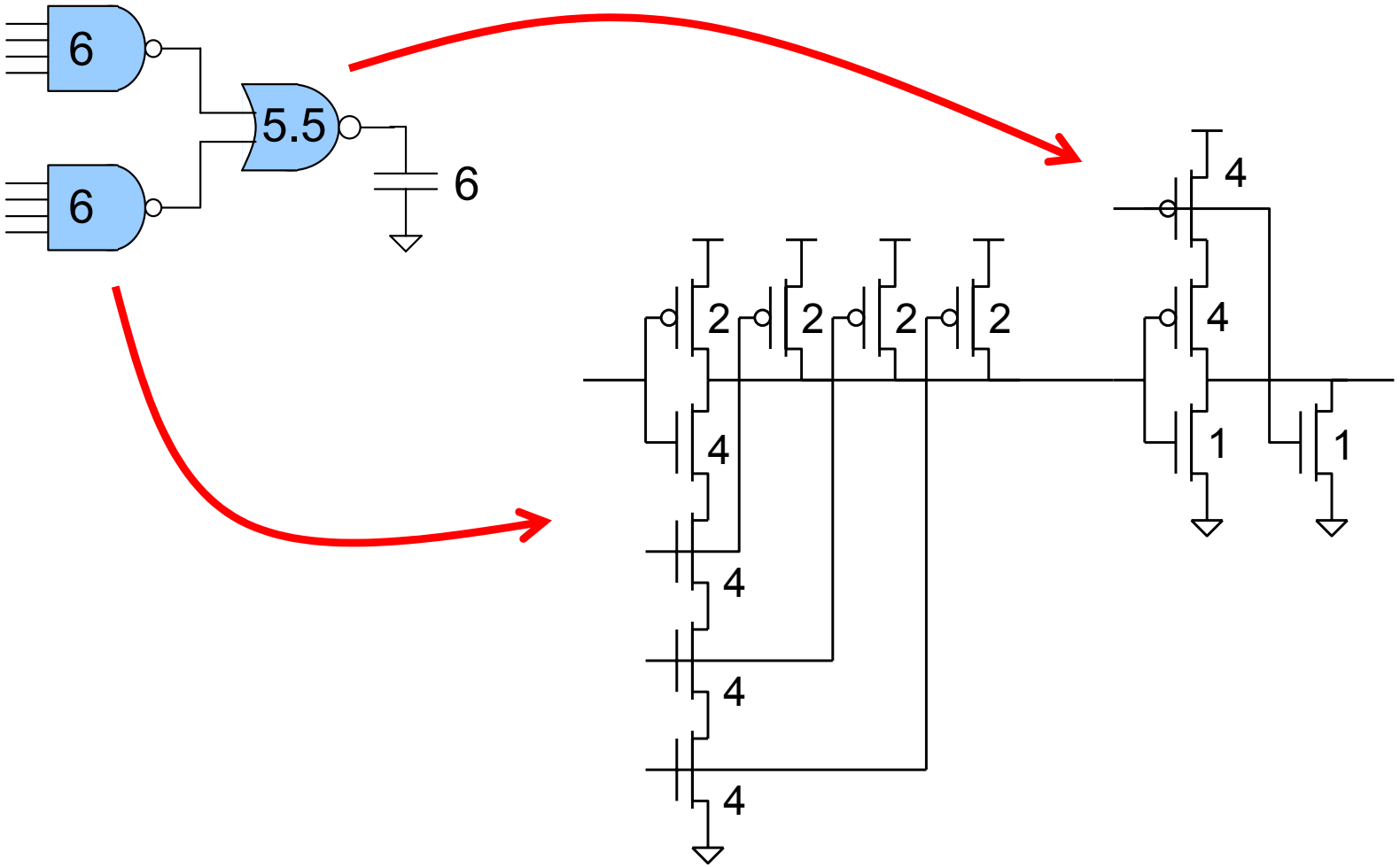
Stage effort of nor gate must equal 1.82
We know logical effort is 5/3, so we can find C_2

$$(5/3)(6/C_2) = 1.82$$
$$C_2 = 5.5$$

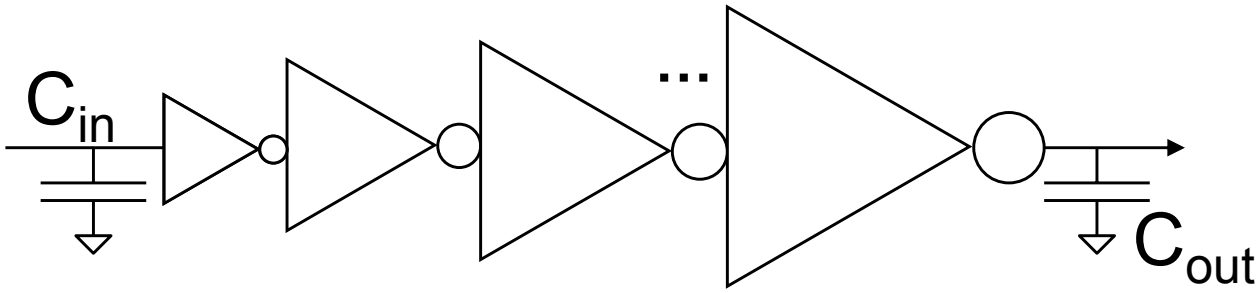
Double check that stage effort of first stage works out

$$(2)(5.5/6) = 1.82$$

Finding actual transistor sizes for H=1 case (Steps 3-5)



How many stages of inverters required if want to drive large load?

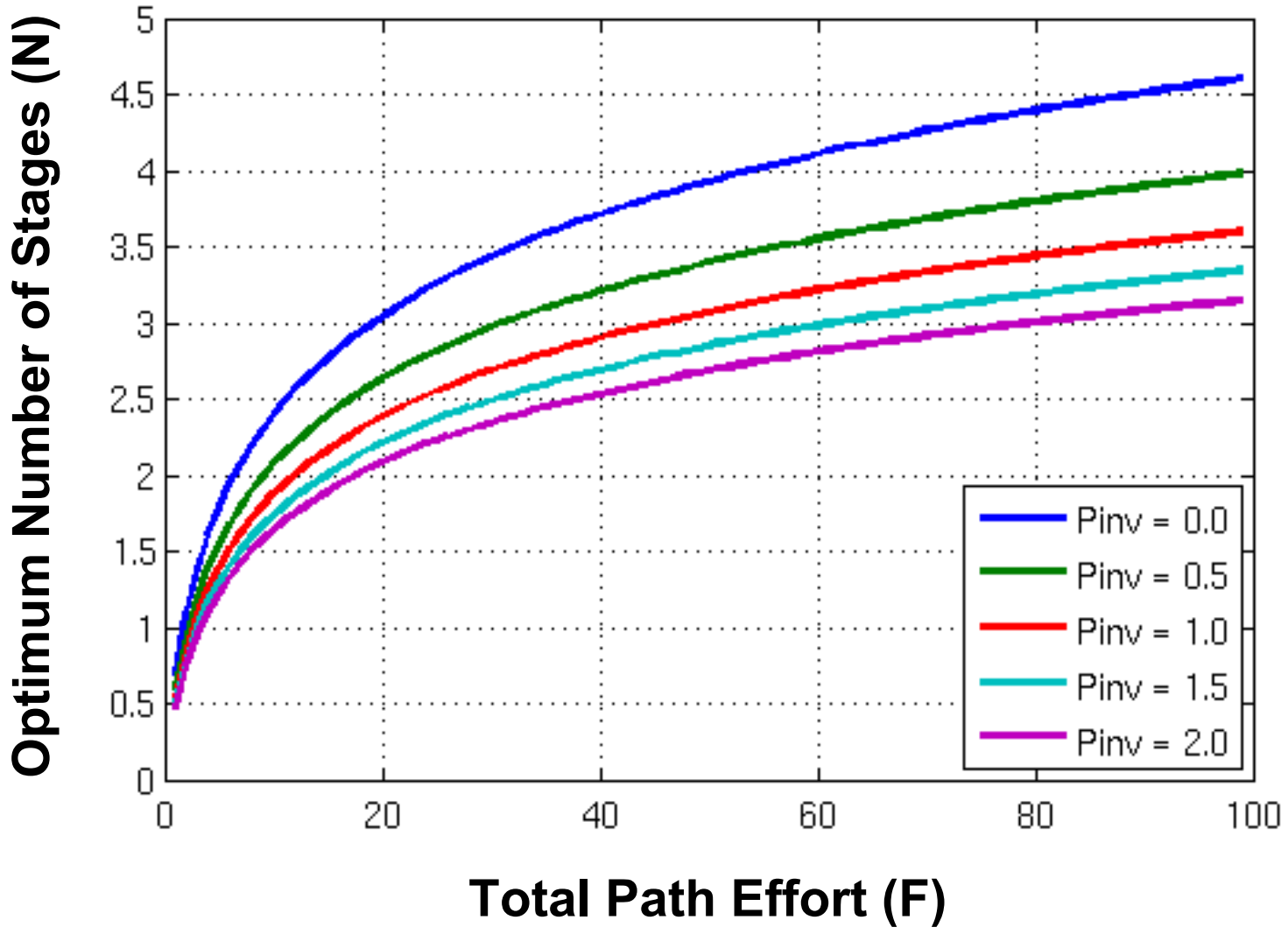


$$D_{OPT} = NF^{1/N} + Np_{inv}$$

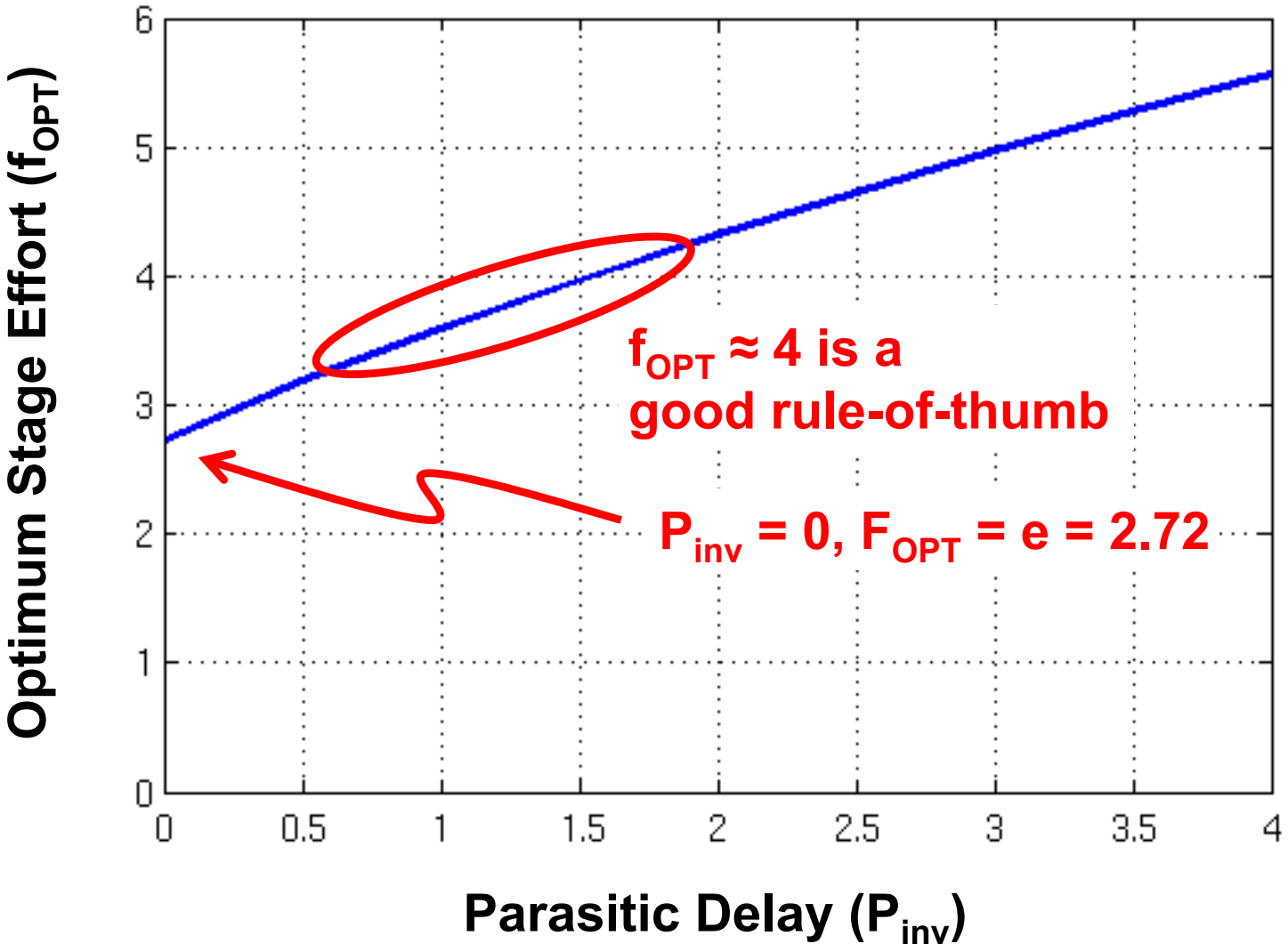
$$\frac{\partial D_{OPT}}{\partial N} = F^{1/N} - F^{1/N}\ln\left(F^{1/N}\right) + p_{inv} = 0$$

No simple closed form solution, but we can examine this function numerically

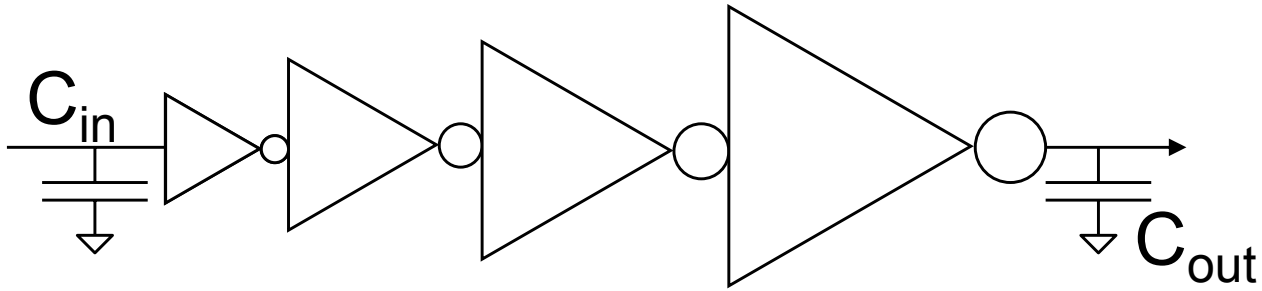
Optimum number of stages for varying parasitic delays and stage effort



Optimum stage effort for varying parasitic delays



A good rule-of-thumb is to target a stage effort around four

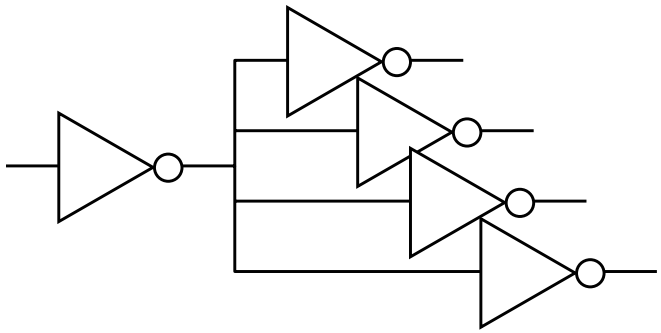


Minimum delay when:

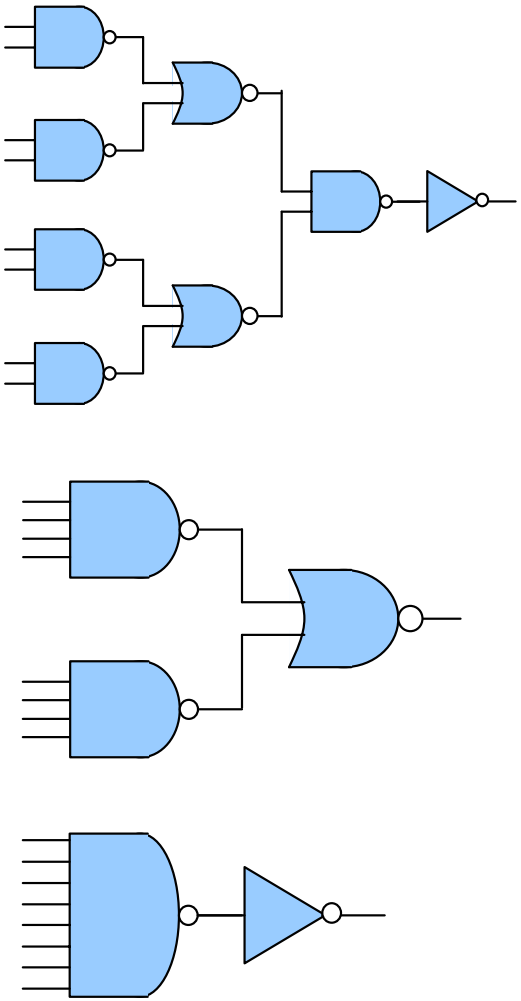
- Stage effort = logical effort x electrical effort ≈ 3.4-3.8
- Some derivations use *e* = 2.718.. – this ignores parasitics
- Broad optimum, stage efforts of 2.4-6.0 within 15-20% of minimum

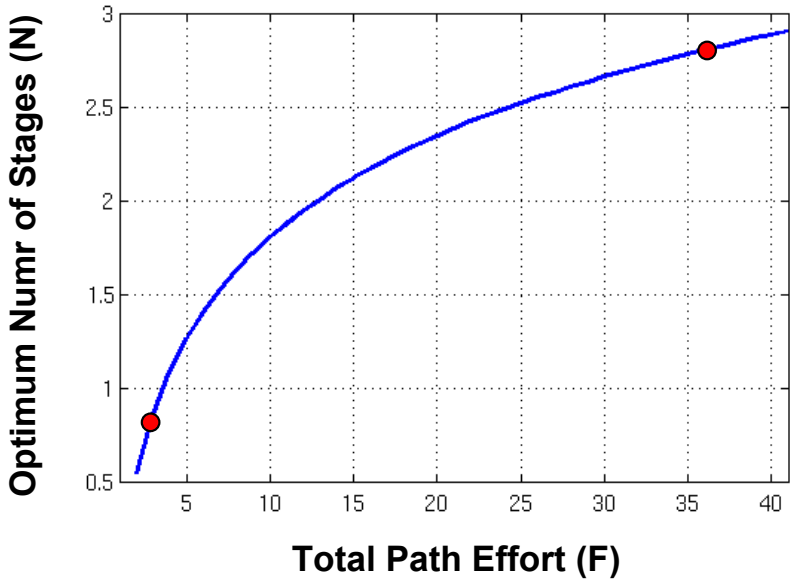
Fan-out-of-four (FO4) is convenient design size *(~5 τ)*

FO4 delay: Delay of inverter driving four copies of itself



Do the topologies in our original example have the optimum number of stages?





	G	N	P	F (H=1)	D (H=1)	F (H=12)	D (H=12)
A	2.96	4	7	2.96	12.25	35.52	**16.77**
B	3.33	2	6	3.33	**9.65**	39.96	18.64
C	3.33	2	9	3.33	12.65	39.96	21.64

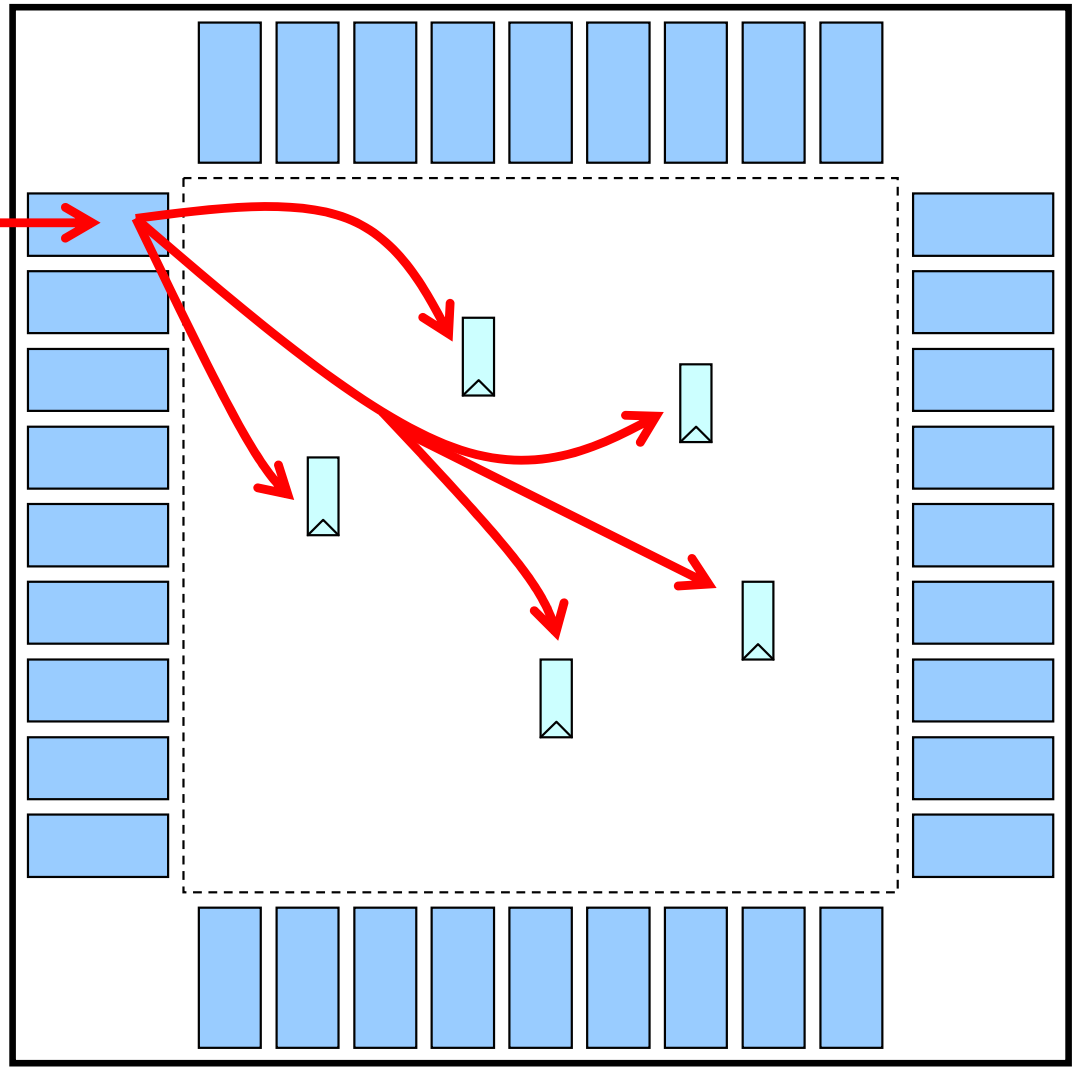
Managing Physical Design Issues in ASIC Toolflows

- Logical Effort
- Physical Design Issues
  - Clock Distribution
  - Power Distribution
  - Wire Delay
  - Power Consumption
  - Capacitive Coupling

1. **What is the issue?**
2. **How do custom designers address the issue?**
3. **How can we approximate these approaches in an ASIC toolflow?**

Clock Distribution: The Issue

Clock propagates across entire chip



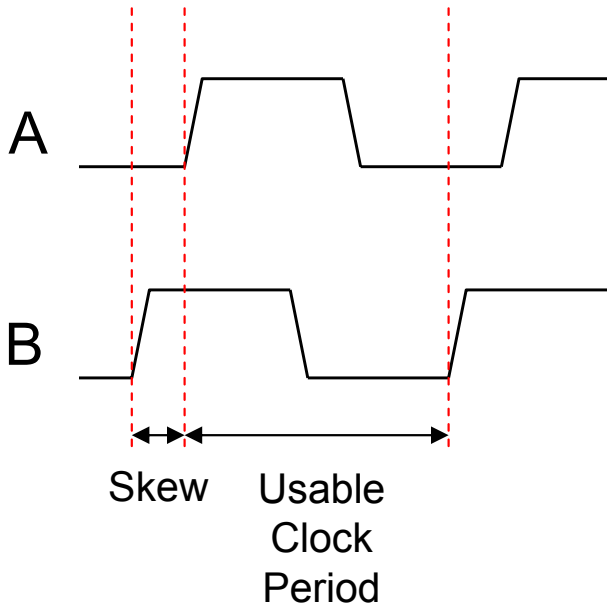
Cannot really distribute clock instantaneously with a perfectly regular period

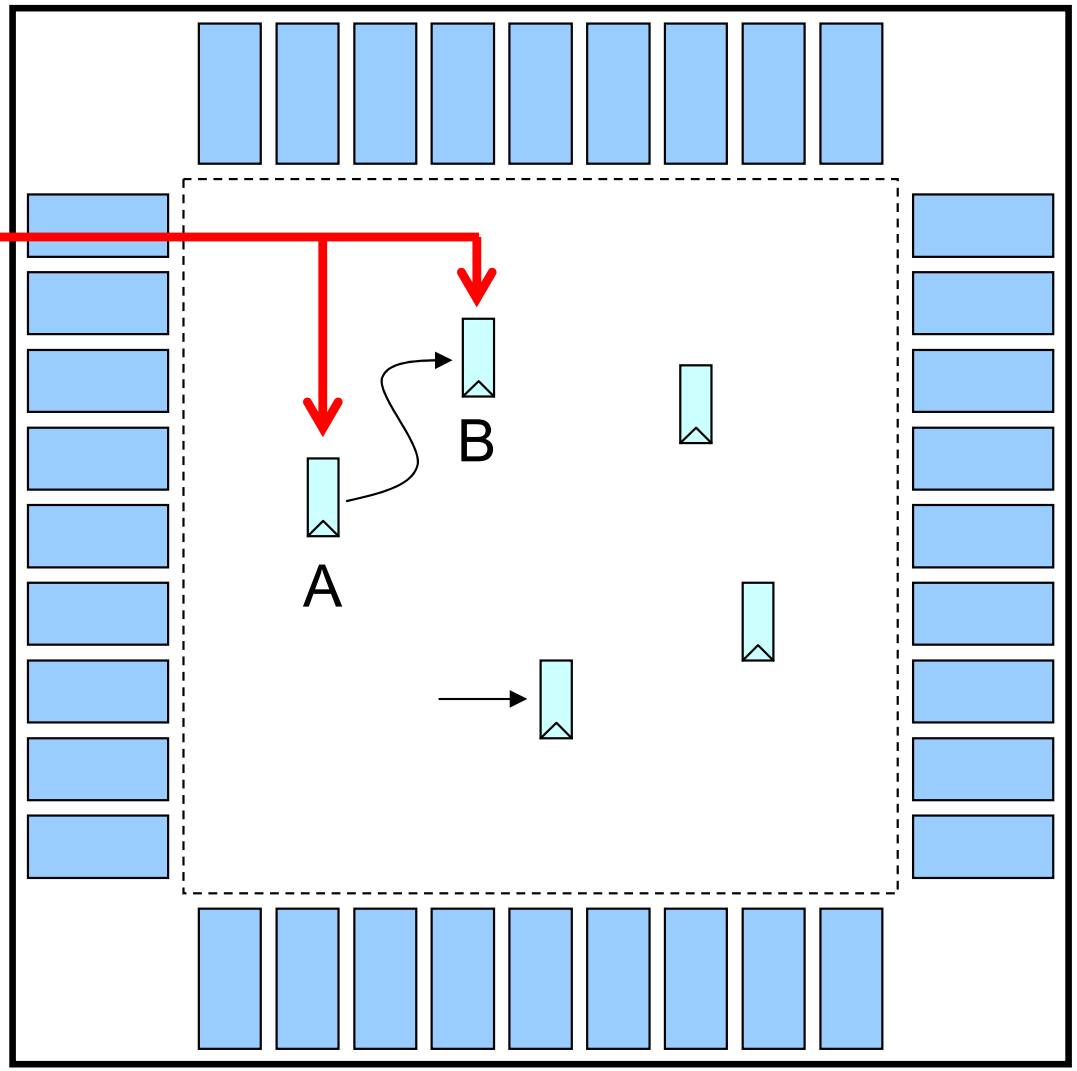
Clock Distribution: The Issue

Two forms of variability

Clock Skew

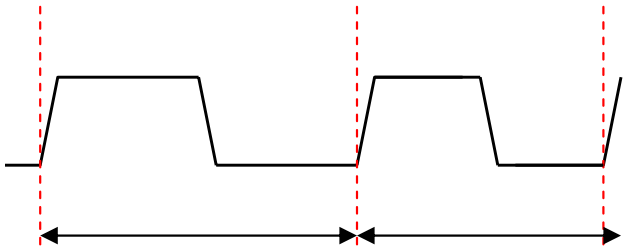
Difference in clock arrival time at two spatially distinct points





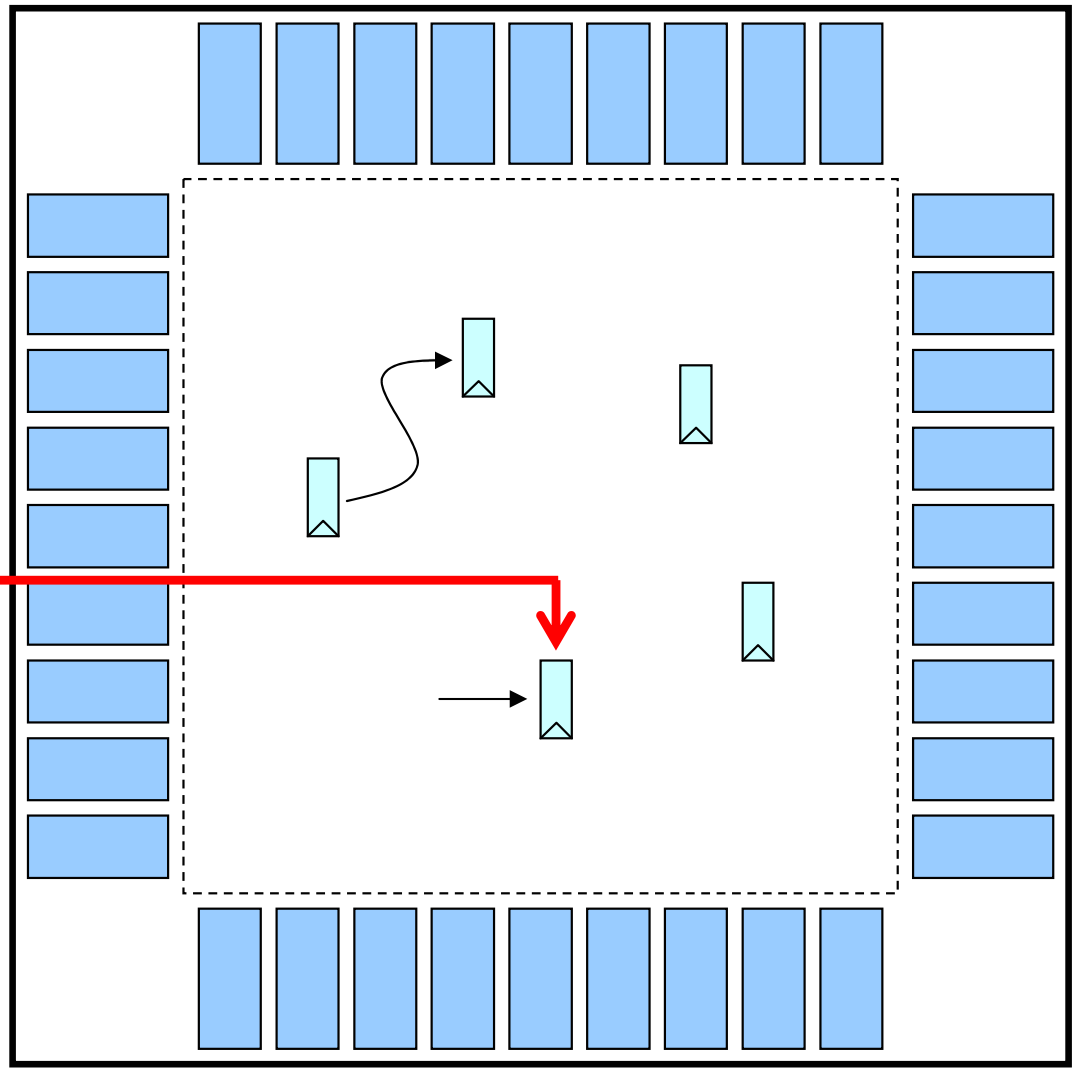
Clock Distribution: The Issue

Two forms of variability



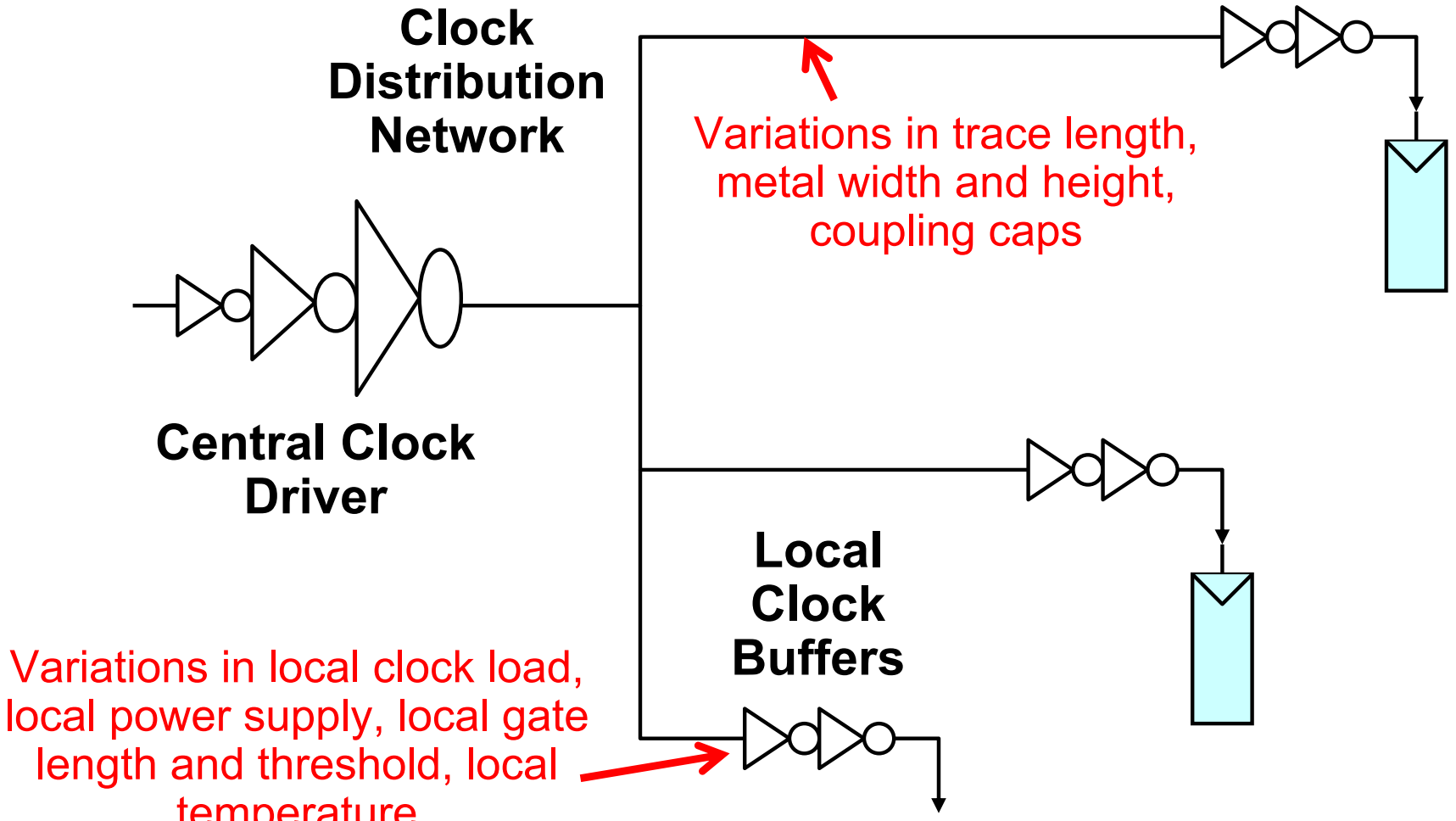
Clock Jitter

Difference in clock period over time



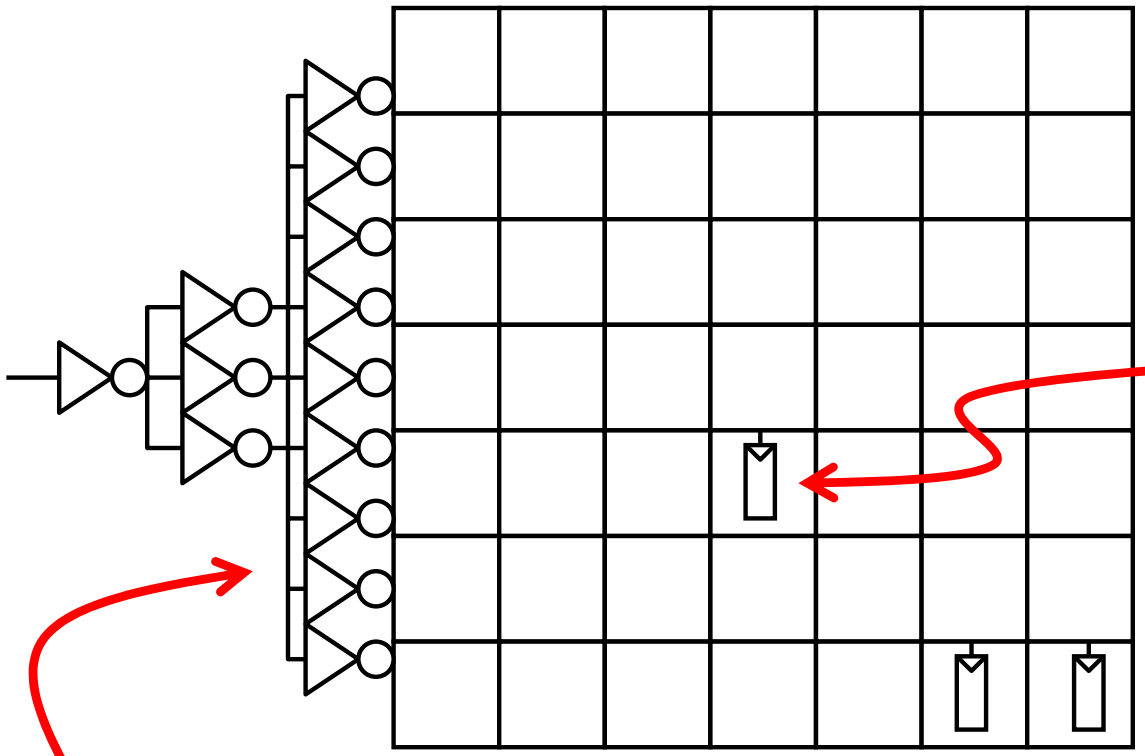
Clock Distribution: The Issue

Why is minimizing skew and jitter hard?



Clock Distribution: Custom Approach

Clock grids lower skew but high power



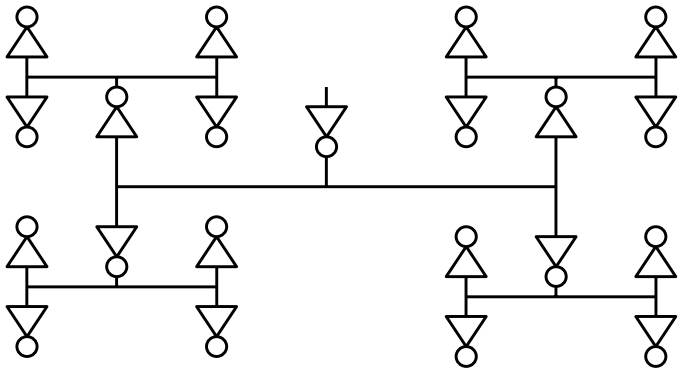
Grid feeds flops directly, no local buffers

Clock driver tree spans height of chip
Internal levels shorted together

Clock Distribution: Custom Approach

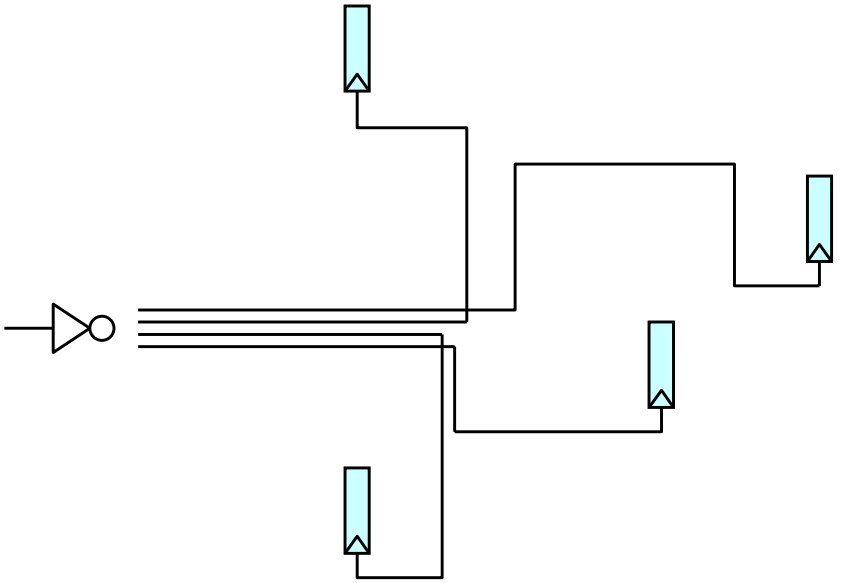
Trees have more skew but less power

H-Tree



Recursive pattern to distribute signals uniformly with equal delay over area

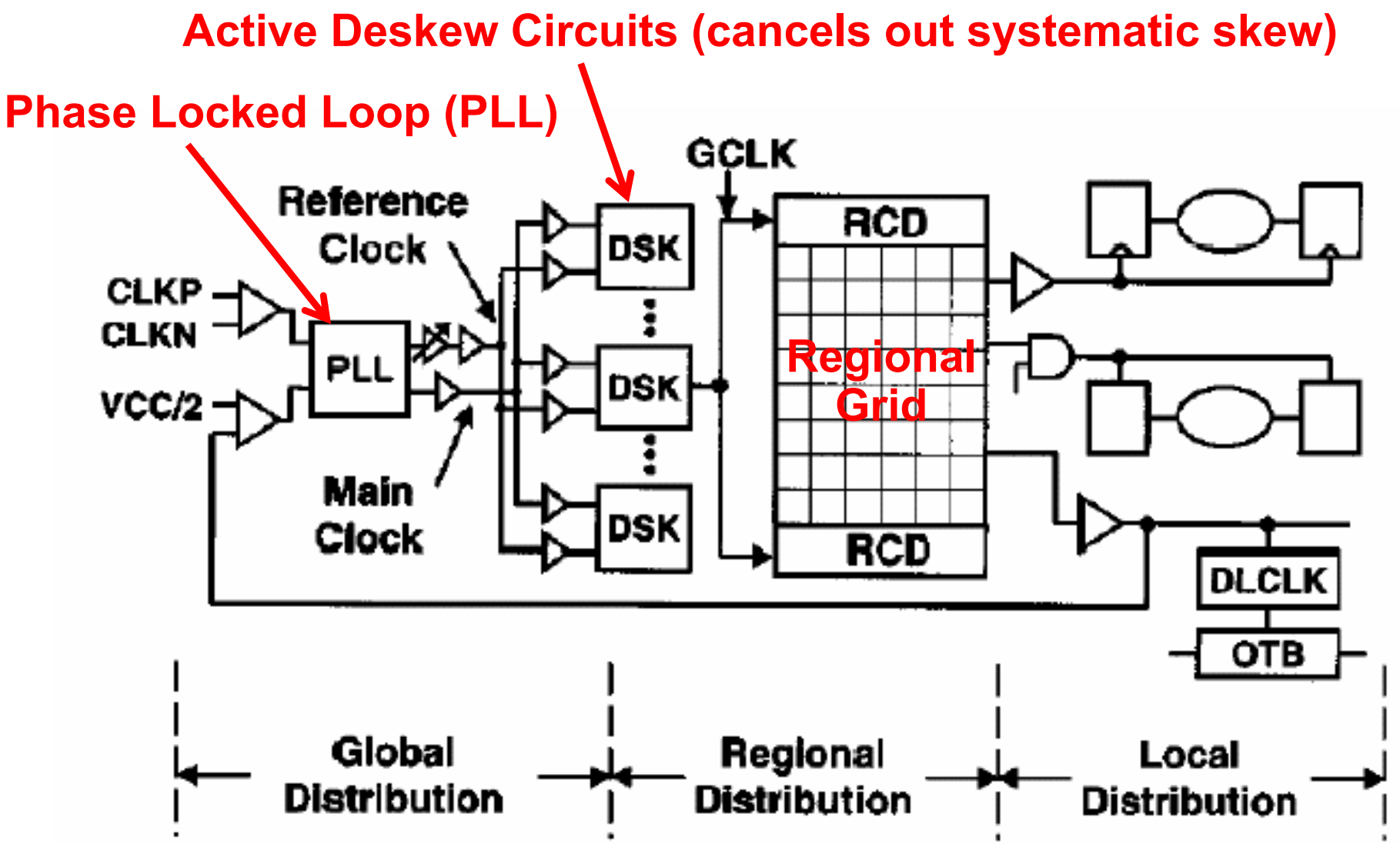
RC-Tree



Each branch is individually routed to balance RC delay

Clock Distribution: Custom Approach

Active deskewing circuits in Intel Itanium



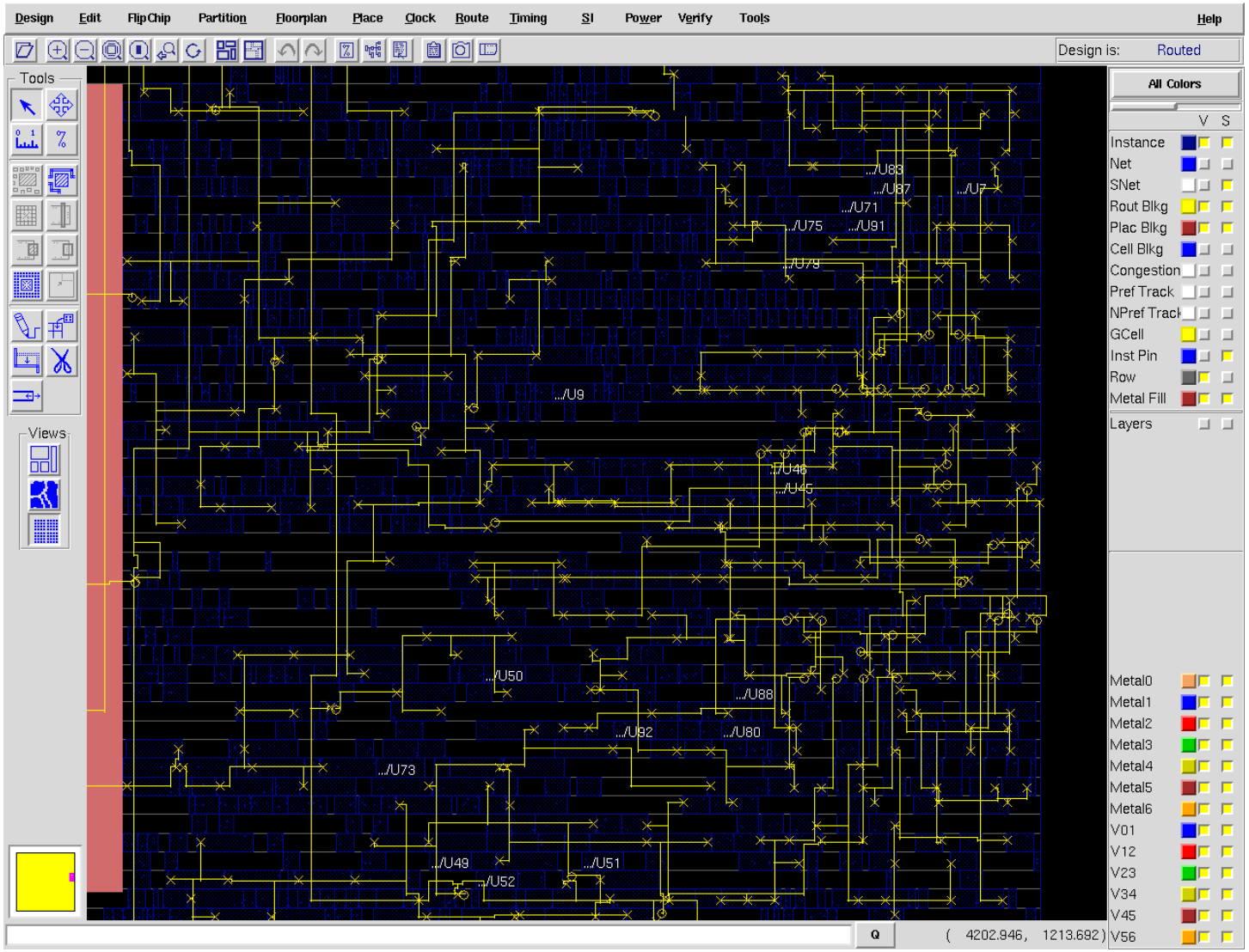
Clock Distribution: Custom Approach
Other techniques

- Use latch-based design
  - Time borrowing helps reduce impact of clock uncertainty
  - Timing analysis can be more difficult
- Make logical partitioning match physical partitioning
  - Limits global communication where skew is usually the worst
  - Helps break distribution problem into smaller subproblems
- Use globally asynchronous, locally synchronous design
  - Divides design into synchronous regions which communicate through asynchronous channels
  - Requires overhead for inter-domain communication
- Use asynchronous design
  - Avoids clocks all together
  - Incurs its own forms of control overhead

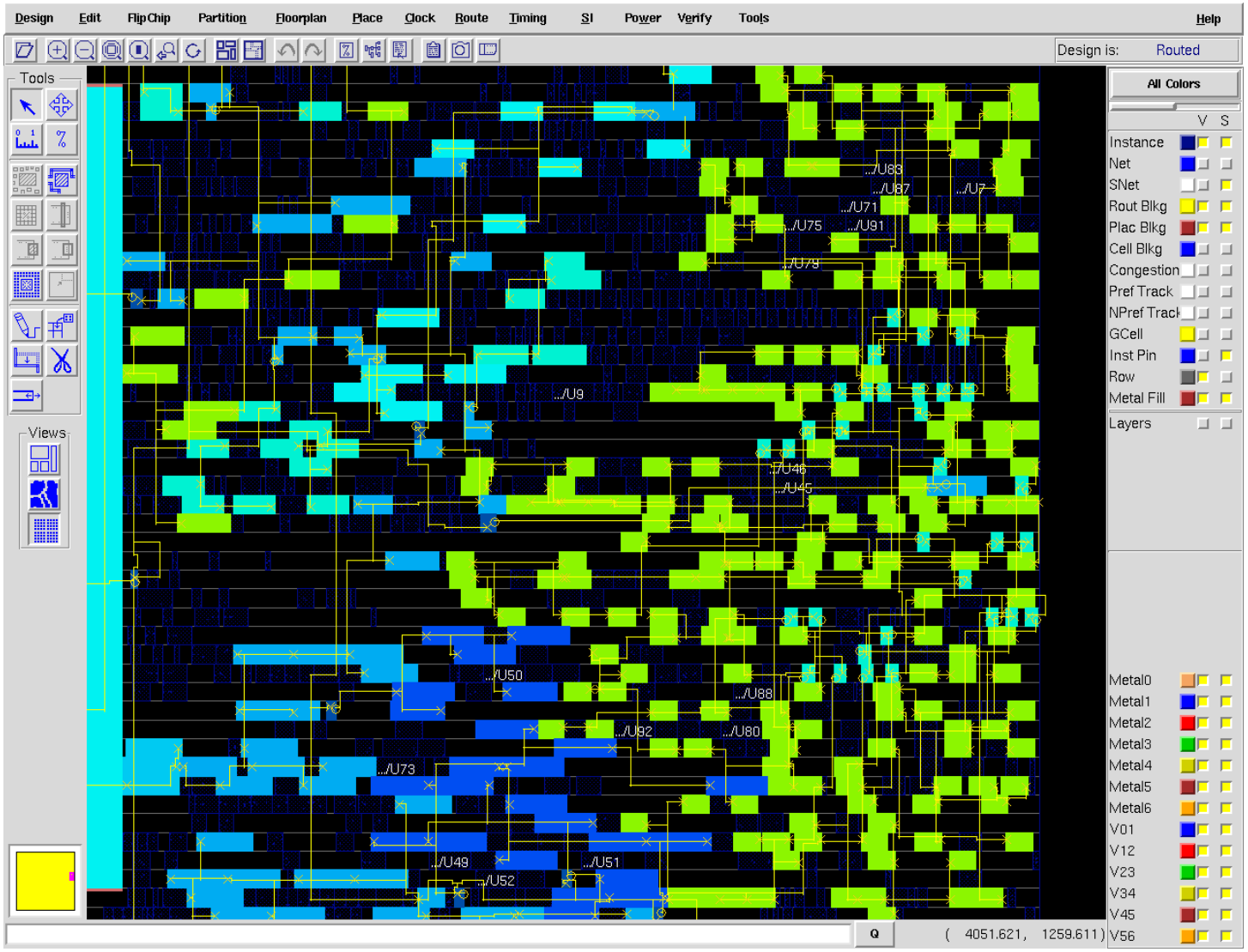
Clock Distribution: ASIC Approach
Clock Tree Synthesis

- Modern back-end tools include clock tree synthesis
  - Creates balanced RC-trees
  - Uses special clock buffer standard cells
  - Can add clock shielding
  - Can exploit useful clock skew
- Automatic clock tree generation still results in significantly worse clock uncertainties as compare to custom clock trees

Example of clock tree synthesis using commercial ASIC back-end tools

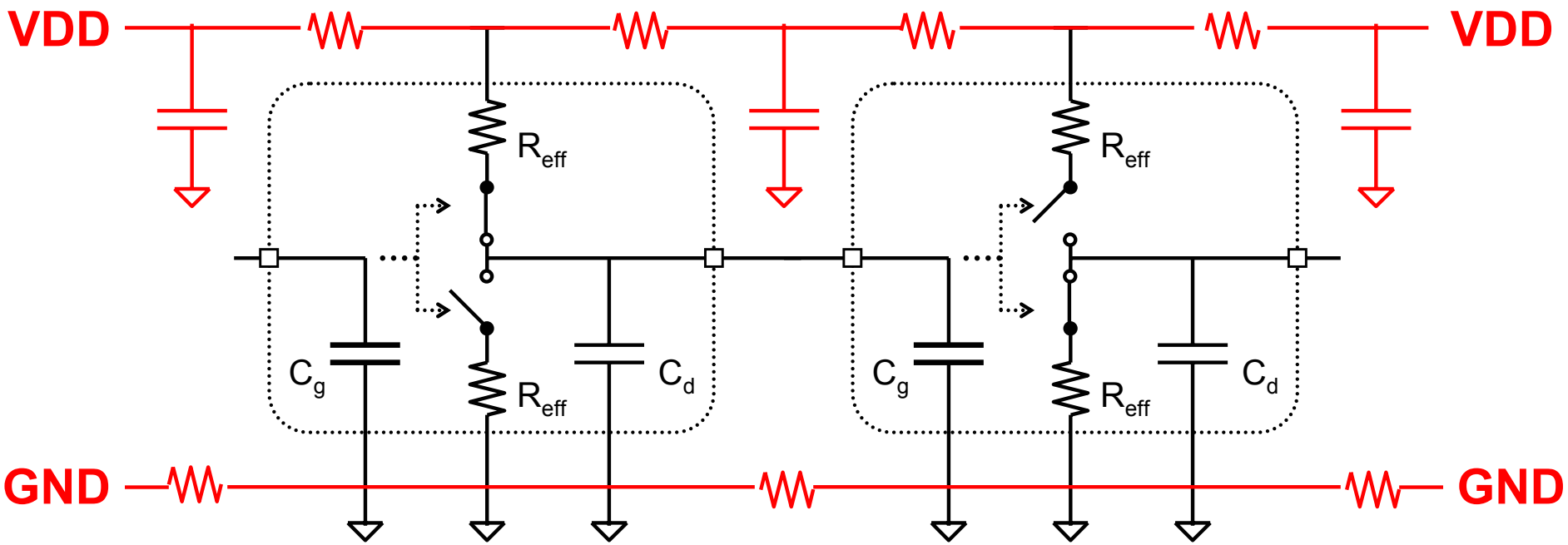


Example of clock tree synthesis using commercial ASIC back-end tools



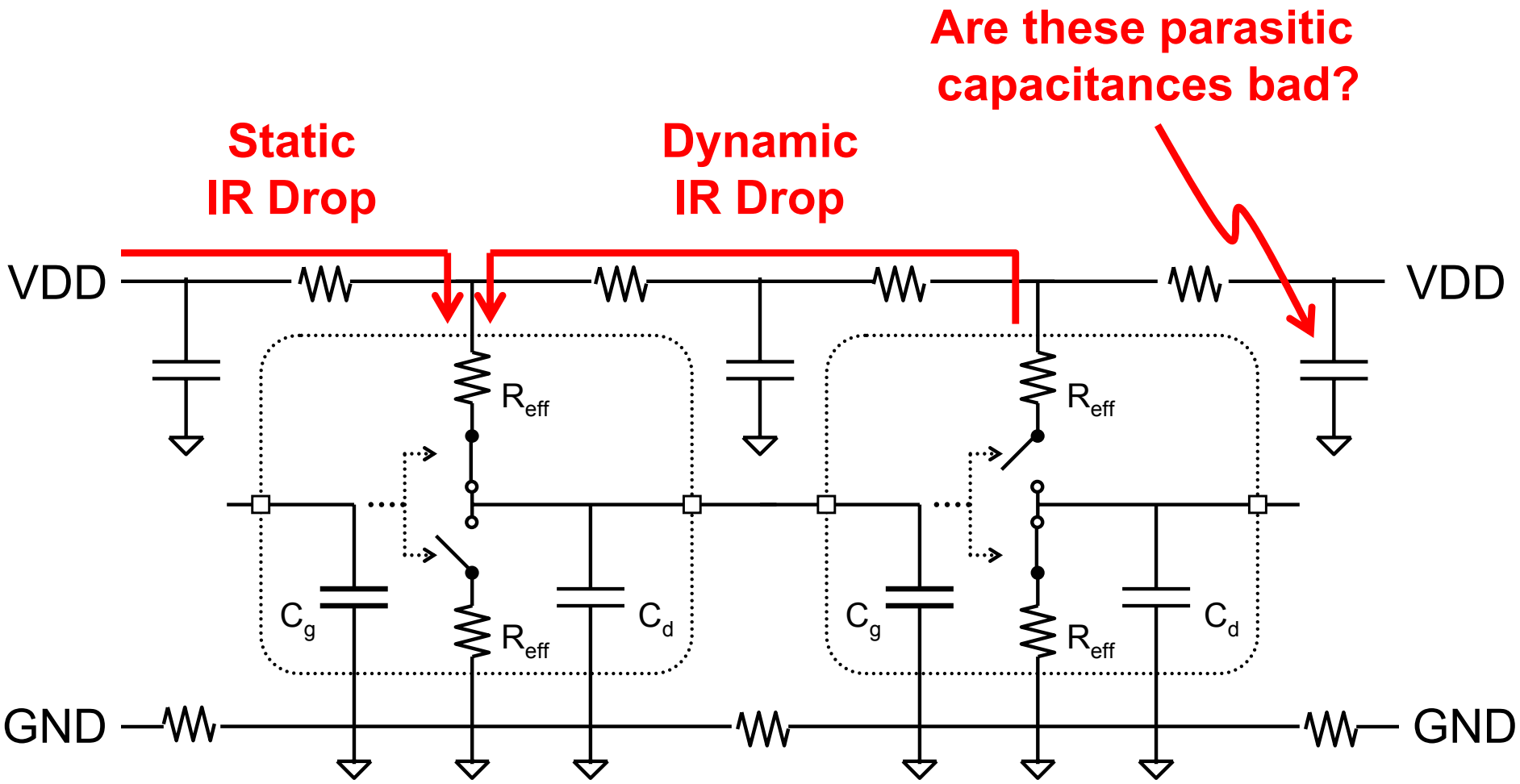
Power Distribution: The Issue

Possible IR drop across power network



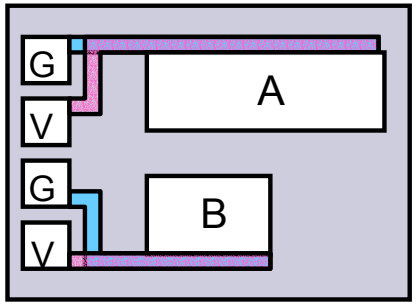
Power Distribution: The Issue

IR drop can be static or dynamic

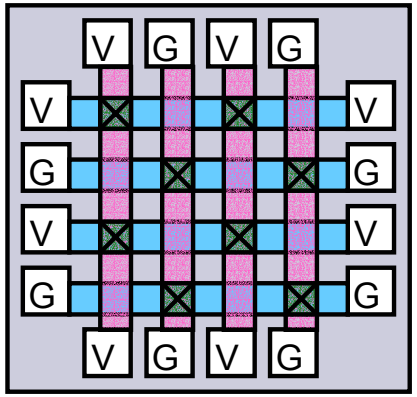


Power Distribution: Custom Approach

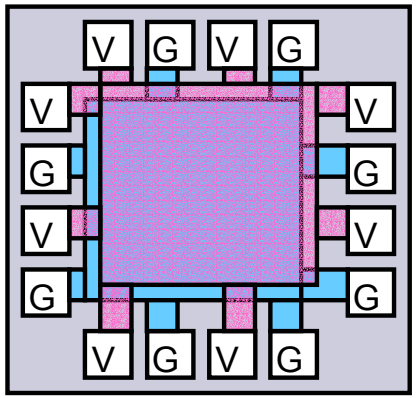
Carefully tailor power network



Routed power distribution on two stacked layers of metal (one for VDD, one for GND). OK for low-cost, low-power designs with few layers of metal.



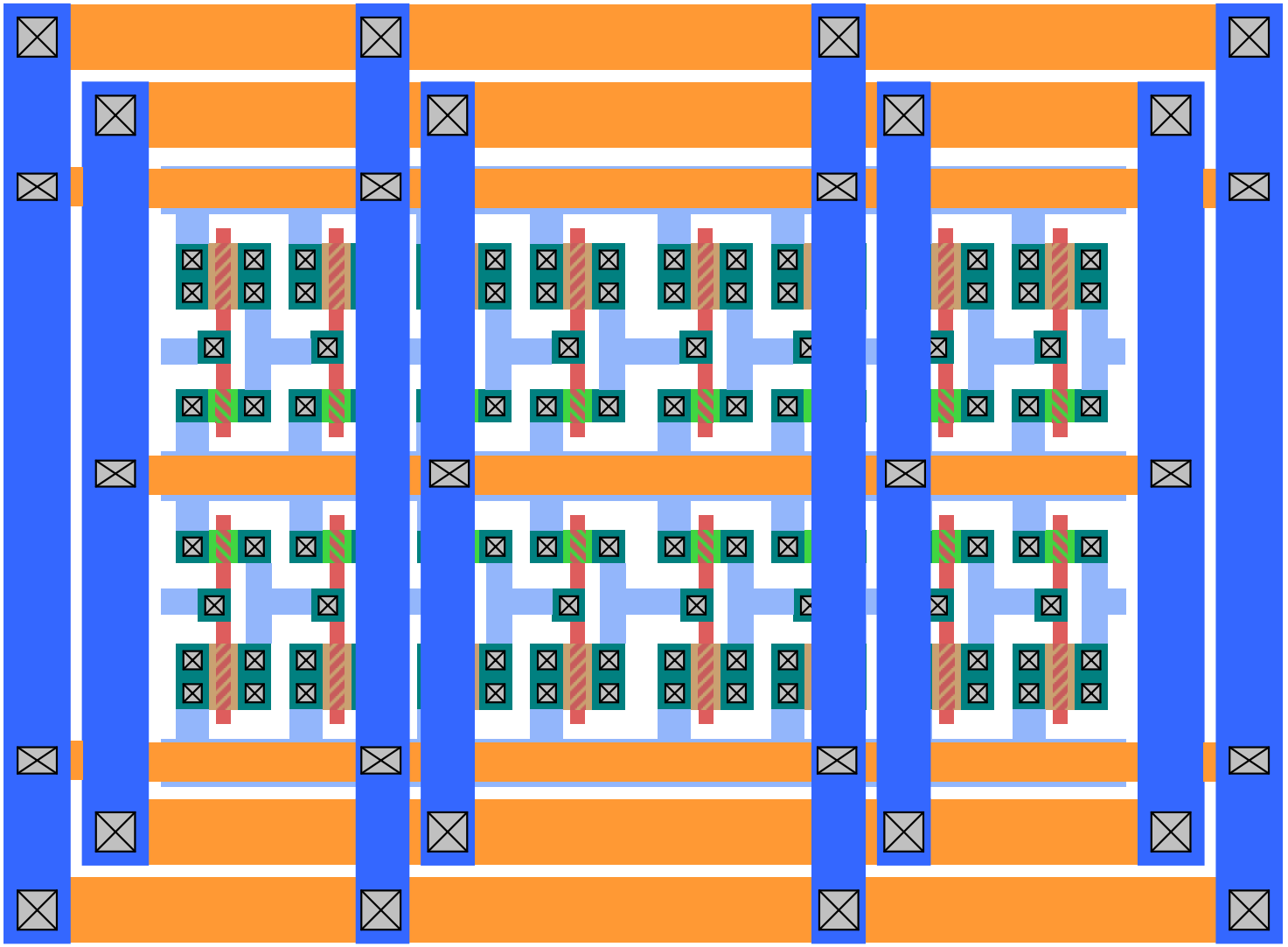
Power Grid. Interconnected vertical and horizontal power bars. Common on most high-performance designs. Often well over half of total metal on upper thicker layers used for VDD/GND.



Dedicated VDD/GND planes. Very expensive. Only used on Alpha 21264. Simplified circuit analysis. Dropped on subsequent Alphas.

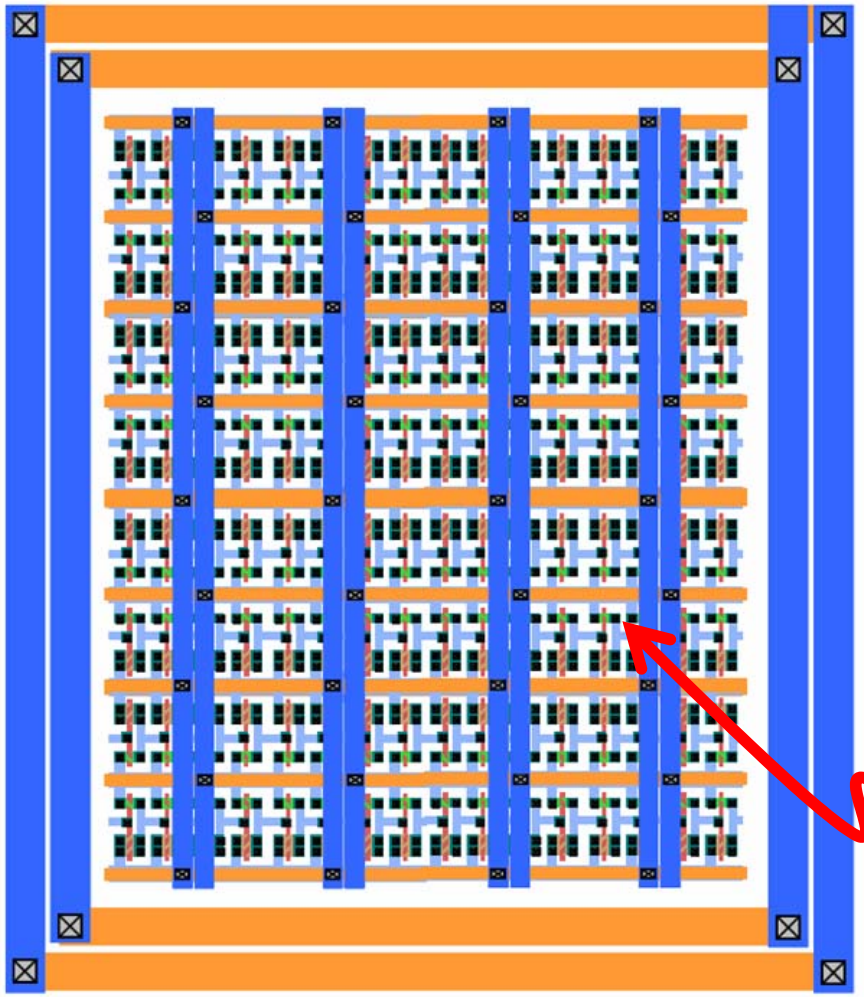
Power Distribution: ASIC Approach

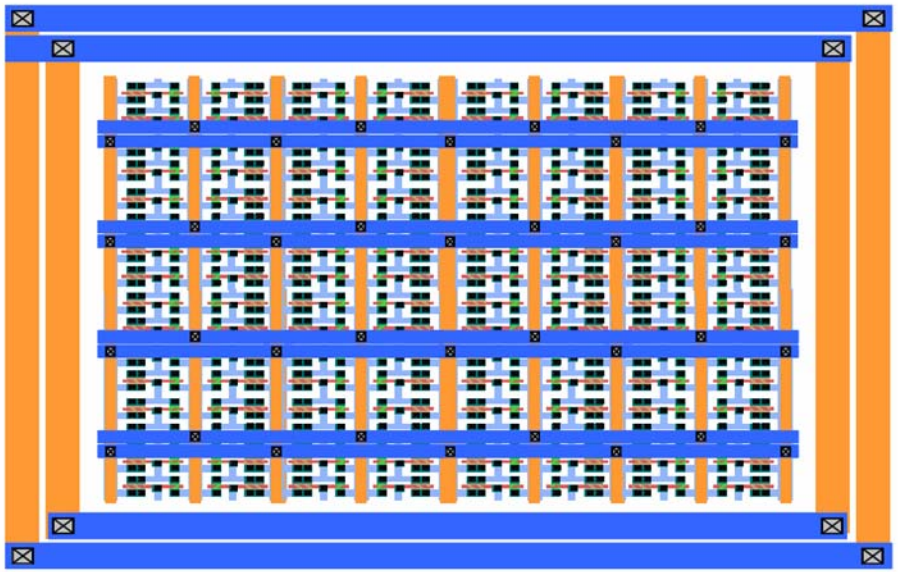
Strapping and rings for standard cells



Power Distribution: ASIC Approach

Power rings partition the power problem

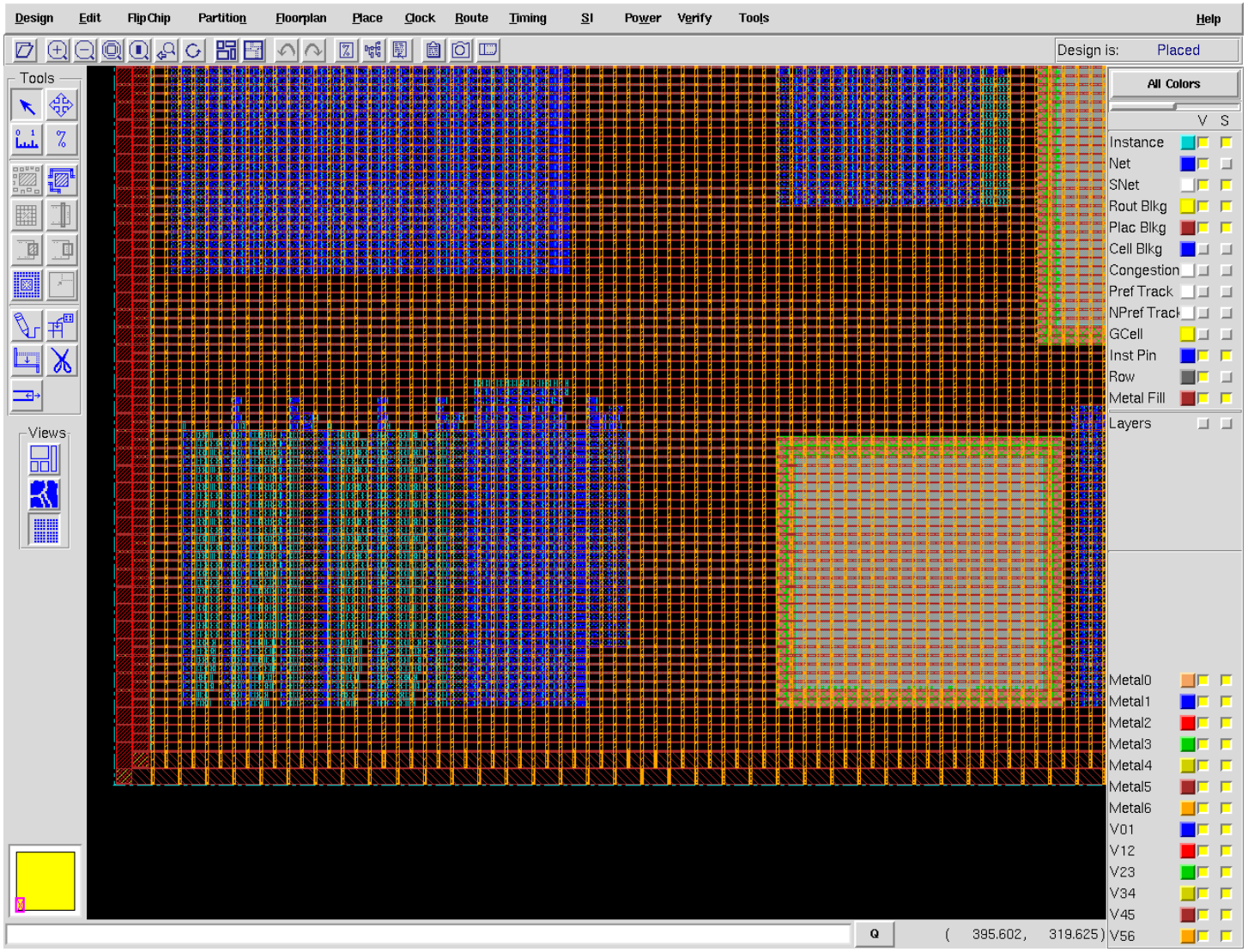




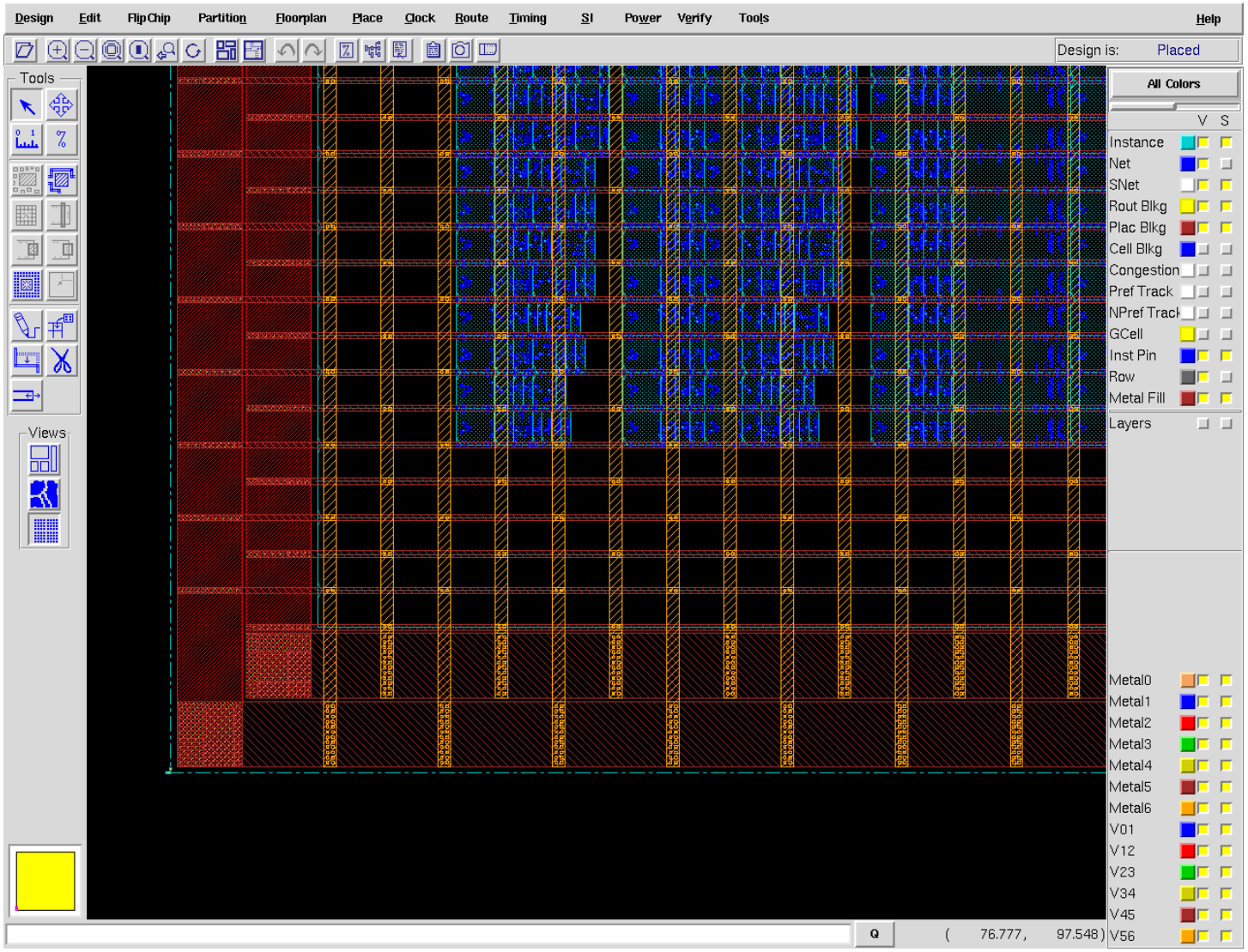
Early physical partitioning and prototyping is essential

Can use special filler cells to help add decoupling cap

Example of power distribution network using commercial ASIC back-end tools

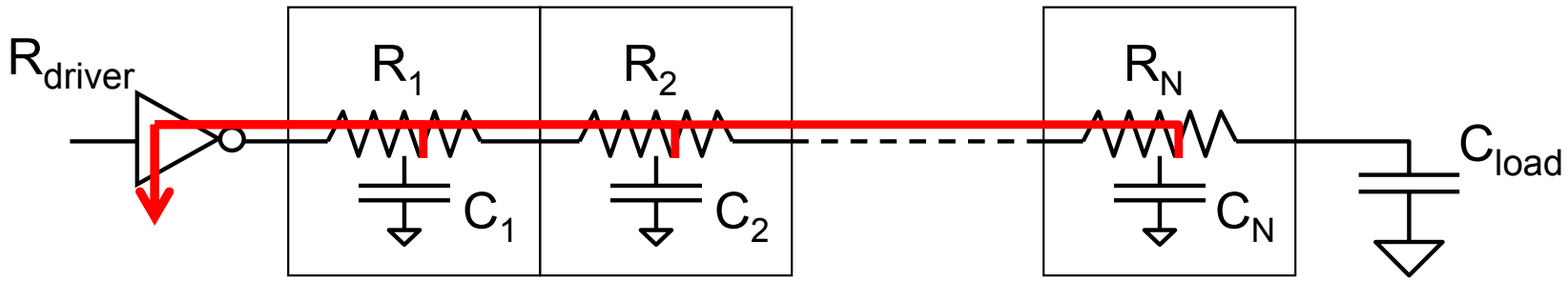


Example of power distribution network using commercial ASIC back-end tools



Wire Delay: The Issue

Large RC makes long wires slow

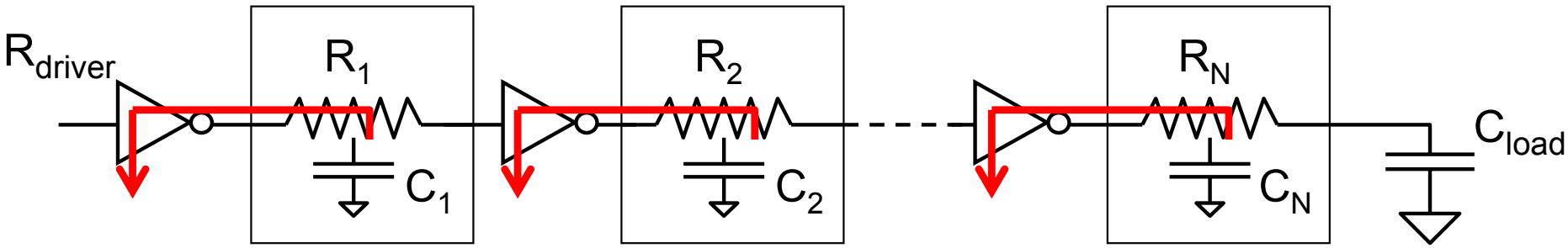


$$\text{Delay} \propto \sum_{i}^{N} \left(\sum_{j=1}^{j=i} R_j \right) \cdot C_i$$

Wire delay increases quadratically

Wire Delay: Custom Approach

Manual insertion of repeaters

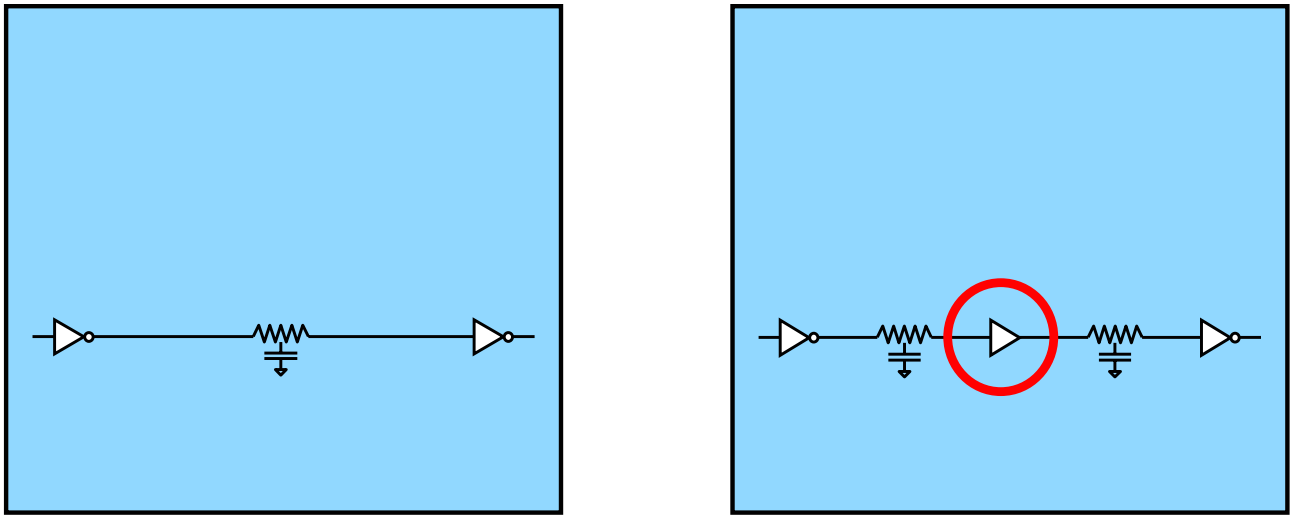


$$\text{Delay} \propto \sum_{i}^{N} R_i C_i$$

Wire delay increases linearly

Wire Delay: Custom Approach

Several issues with repeater insertion



- Repeater must connect to transistor layers
- Blocks other routes with vias that connect down
- Requires space on active layers for buffer transistors
- Repeaters often grouped in preallocated repeater boxes spread around chip, and thus repeater location might not give ideal spacing

Wire Delay: Impact on RTL

- Make logical, physical partitioning match
  - Limits global communication
  - Helps simplify automatic buffer insertion
- Add extra pipeline stages for wire delay
  - P4 included stages just for driving signals
  - Requires very early physical prototyping
- Use latency insensitive methodology
  - Create macroblocks with registered interfaces
  - Enables pipelining wires late in design cycle

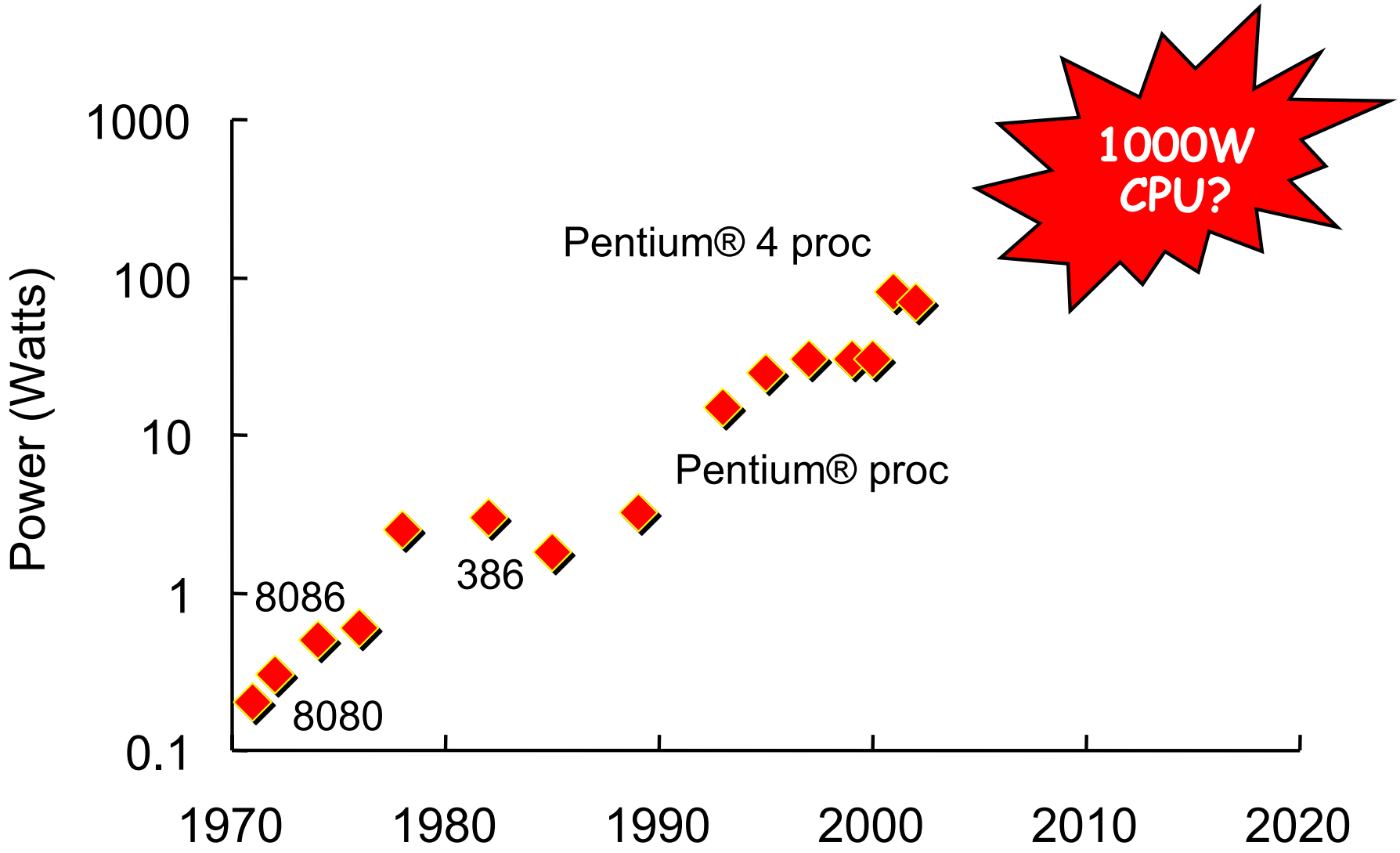
1	TC Next IP
2	
3	TC Fetch
4	
5	Drive
6	Alloc
7	Rename
8	
9	Queue
10	Schedule 1
11	Schedule 2
12	Schedule 3
13	Dispatch 1
14	Dispatch 2
15	Register File 1
16	Register File 2
17	Execute
18	Flags
19	Branch Check
20	Drive

Wire Delay: ASIC Approach

- Front-end tools include rough wire-load models
  - Usually statistical in nature
  - Helps synthesis tool with technology mapping
- Back-end tools include better wire-load models
  - After trial placement can use Manhattan distance
  - Tool will automatically insert buffers where necessary

Power Consumption: The Issue

Power has been increasing rapidly



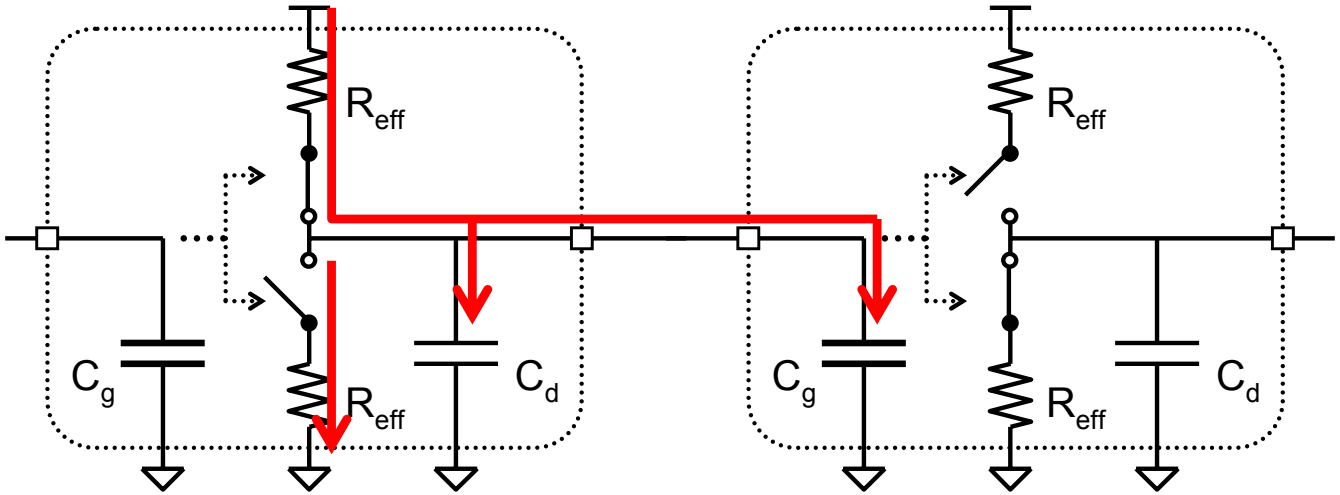
[Source: Intel]

Power Consumption: The Issue
Why is it a problem?

- Power dissipation is limiting factor in many systems
  - Battery weight and life for portable devices
  - Packaging and cooling costs for tethered systems
  - Case temperature for laptop/wearable computers
  - Fan noise for media hubs
- Example 1: Cellphone
  - 3 Watt hard power limit – any more and customers complain
  - Battery life is a strong product differentiator
- Example 2: Internet data center
  - ~8,000 servers, ~2 MegaWatts
  - 25% of operational costs are in electricity bill for supplying power and running air-conditioning to remove heat

Power Consumption: The Issue

Main forms are dynamic and static power



Dynamic Power

Switching power used to charge up load capacitance

$$P_{dynamic} = \alpha \; F \; (1/2) \; C \; V_{DD}^2$$

Static Power

Subthreshold leakage power when transistor is "off"

$$P_{static} = V_{DD} \; I_{off}$$

Power Consumption: Custom Approach

$$P_{dynamic} = \alpha \; F \; (1/2) \; C \; V_{DD}^2$$

Reduce Activity

- Clock gating so clock node of inactive logic doesn't switch
- Data gating so data nodes of inactive logic doesn't switch
- Bus encodings to minimize transitions
- Balance logic paths to avoid glitches during settling

Reduce Frequency

- Doesn't save energy, just reduces rate at which it is consumed
- Lower power means less heat dissipation but must run longer

Power Consumption: Custom Approach

$$P_{dynamic} = \alpha \; F \; (1/2) \; C \; V_{DD}^2$$

Reduce Switched Capacitance

- Careful transistor sizing (small transistors off critical path)
- Tighter layout (good floorplanning)
- Segmented tri-state bus structures

Reduce Supply Voltage

- Need to lower frequency as well – quadratic+ power savings
- Can lower statically for cells off critical path
- Can lower dynamically for just-in-time computation

Power Consumption: Custom Approach

$$P_{static} = V_{DD} I_{OFF}$$

Reduce Supply Voltage

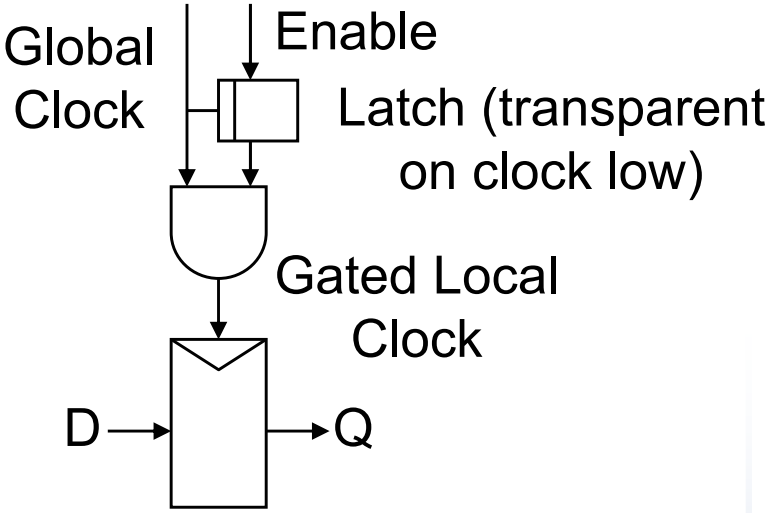
- In addition to dynamic power reduction, reducing Vdd can help reduce static power

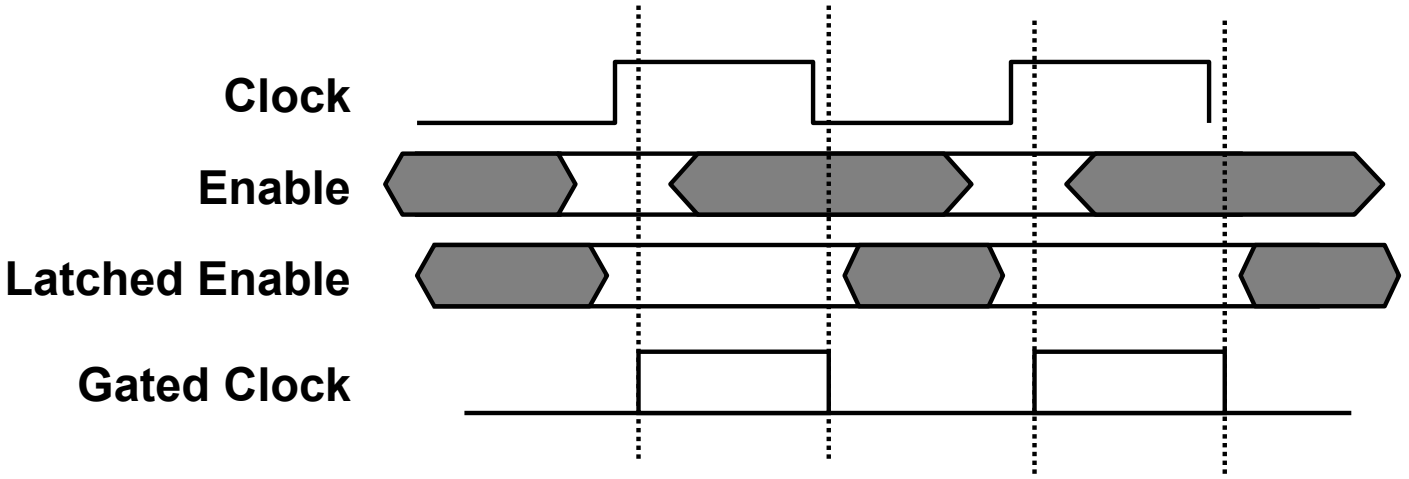
Reduce Off Current

- Increase length of transistors off critical path
- Use high-Vt cells off critical path (extra Vt increases fab costs)
- Use stacked devices
- Use power gating (ie switch off the power supply with a large transistor)

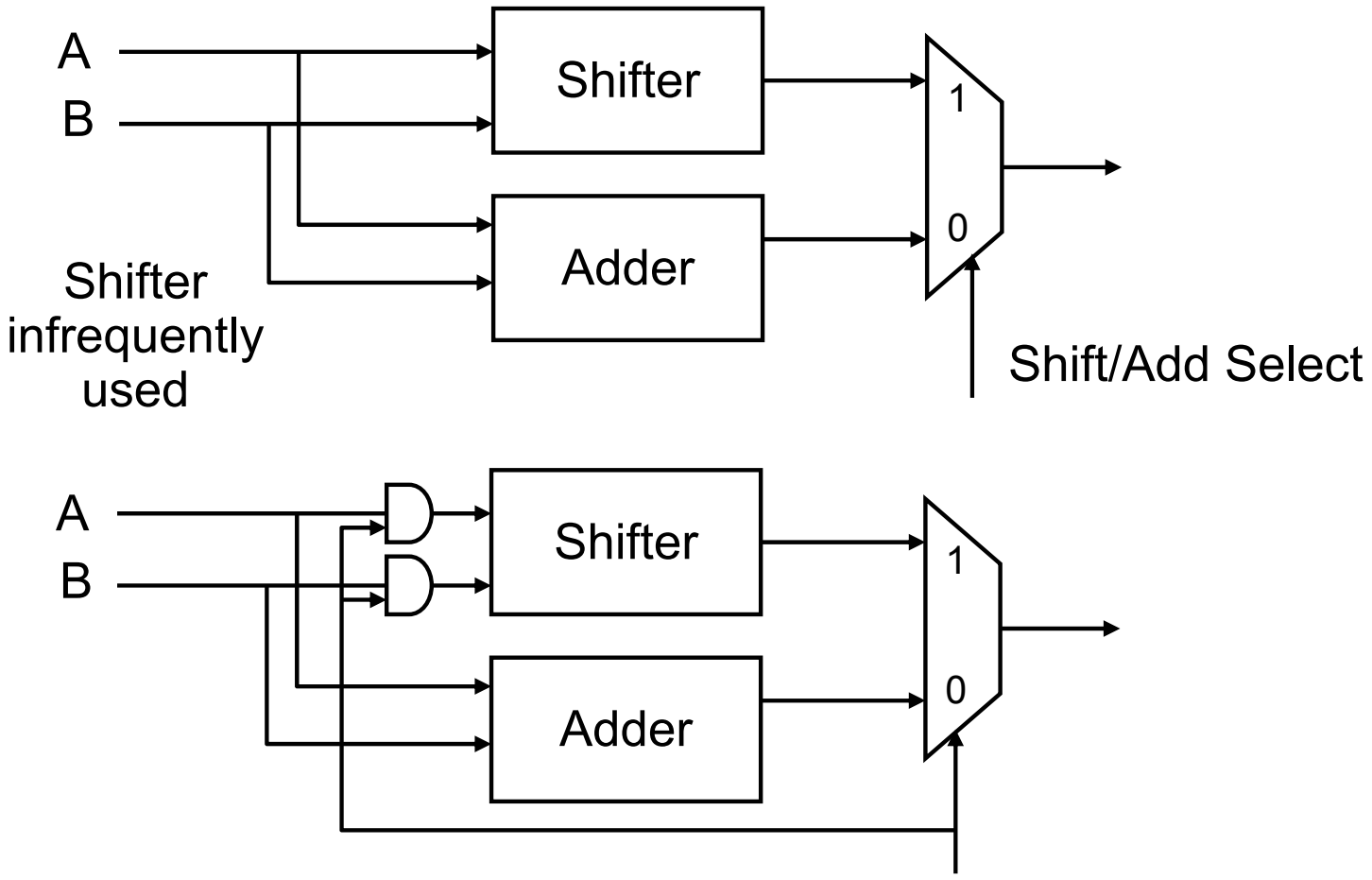
Power Consumption
Reducing activity with clock gating

- Don't clock flip-flop if not needed
- Avoids transitioning downstream logic
- Enable adds control logic complexity
- P4 has hundreds of gated clock domains



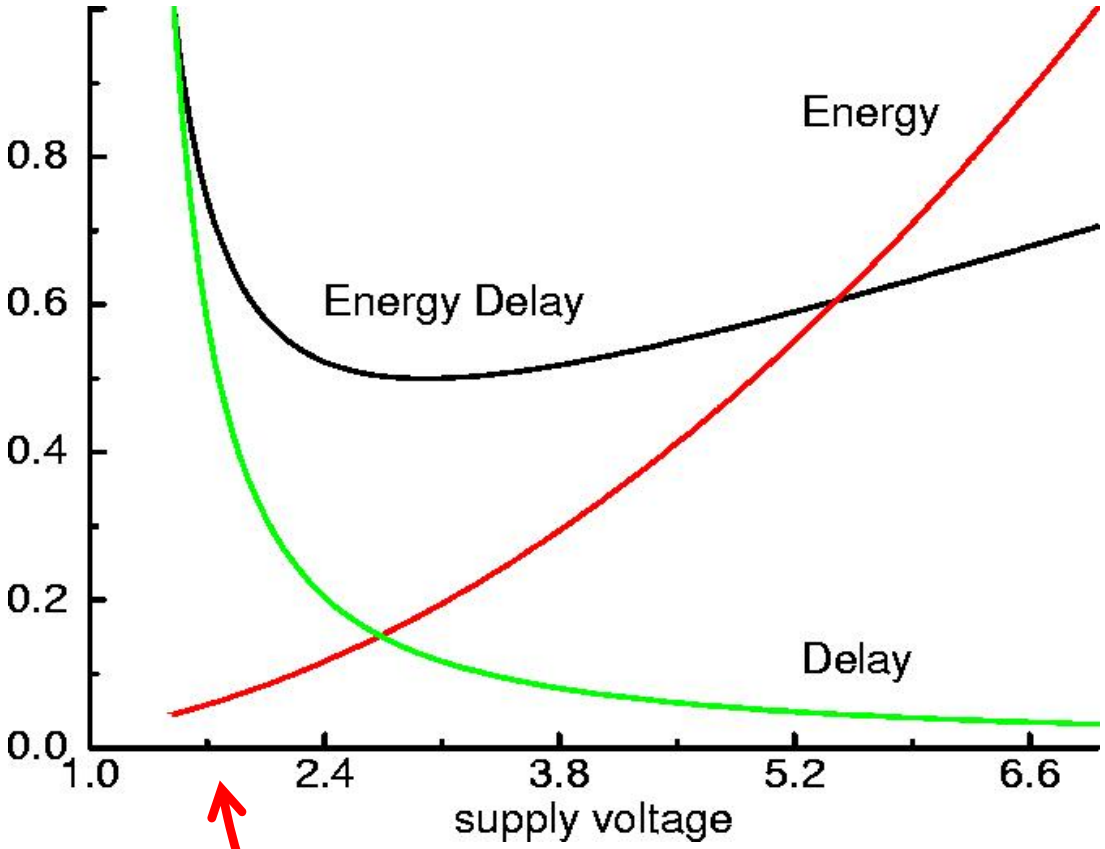


Power Consumption
Reducing activity with data gating



Could use transparent latch instead of AND gate to reduce number of transitions, but would be bigger and slower.

Power Consumption
Reducing supply voltage



Both static and dynamic voltage scaling is possible

Delay rises sharply as supply voltage approaches Vt

[Source: Horowitz]

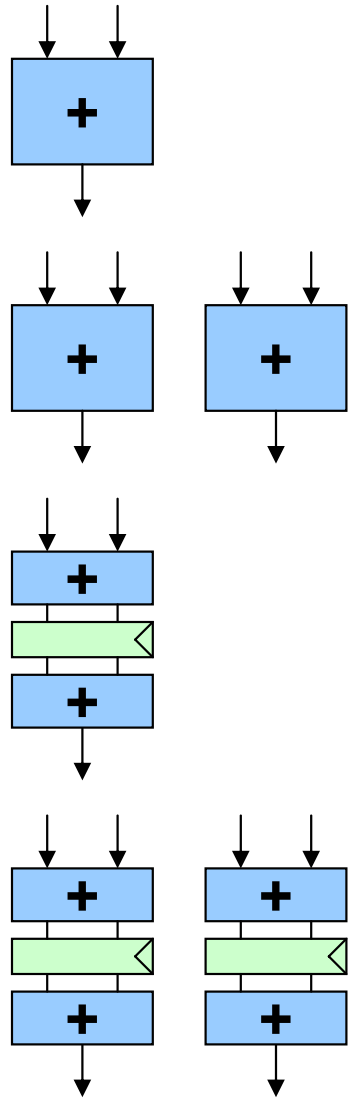
Power Consumption
Parallel architecture to reduce energy

8-bit adder/cmp
- 40MHz at 5V, area = 530 $k\mu^2$
- Base power P_{ref}

Two parallel interleaved adder/cmp units
- 20MHz at 2.9V, area = 1,800 $k\mu^2$ (3.4x)
- Power = 0.36 P_{ref}

One pipelined adder/cmp unit
- 40MHz at 2.9V, area = 690 $k\mu^2$ (1.3x)
- Power = 0.39 P_{ref}

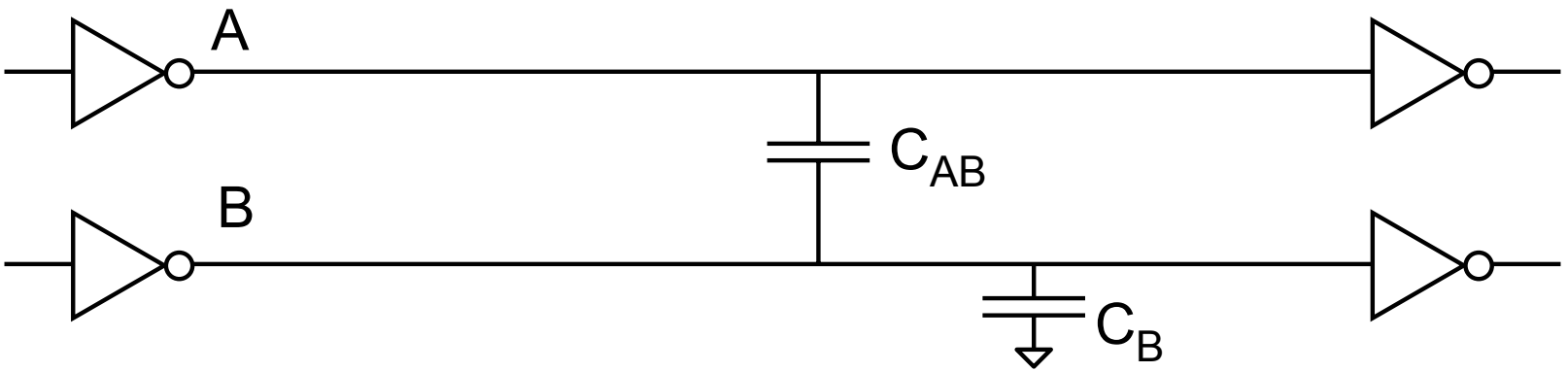
Pipelined and parallel
- 20MHz at 2.0V, area = 1,961 $k\mu^2$ (3.7x)
- Power = 0.2 P_{ref}



Power Consumption: ASIC Approach

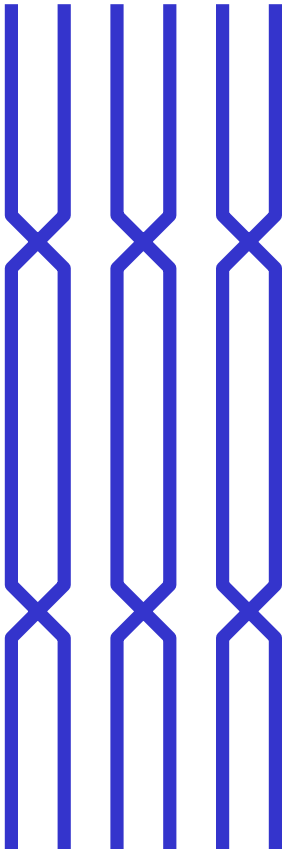
- Minimize activity
  - Automatic clock gating is possible if we write Verilog so tools can infer gating
  - Partition designs so minimal number of components activated to perform each operation
  - Floorplan units to reduce length of power-hungry global wires
- Use lowest voltage and slowest frequency necessary to reach target performance
  - Use pipelined and parallel architectures if possible
- Modern standard cell libraries include low-power cells, high-VT cells, and low-VT cells – tools can automatically replace non-critical cells to optimize for power

Capacitive Coupling: The Issue
Delay is a function of switching on neighbors



- Most of the wire capacitance is to neighboring wires
- If A switches then it injects voltage noise on where the magnitude depends on capacitive divider formed [$C_{AB}/(C_{AB}+C_B)$]
  - If A switches in opposite direction while B switches, coupling capacitance effectively doubles (Miller effect)
  - If A switches in same direction while B switches, coupling capacitance disappears
- These effects can lead to large variance in possible delay of B driver, possibly factor of 5 or 6 between best and worst case

Capacitive Coupling
Custom vs ASIC Approach



Custom Approach

- Avoid placing simultaneously switching signals next to each other for long parallel runs (use swizzling)
- Reroute signals which will be quiet during switching in between simultaneous switching signals
- Route signals close to power rails for capacitance ballast
- Extensive dynamic signal simulation

ASIC Approach

- Automatic routers can specifically avoid long straight routes, sometimes this causes the router to avoid the “most direct” route
- Critical nets (such as the clock) can use automatic shielding
- Static timing tools help focus dynamic signal simulation
- Fixing a coupling problem can require a point change which itself might cause new problems

Take away points

- **Logical effort** is a useful tool for quickly determining transistor sizing and number of stages
- It is essential to consider physical design issues early and often in ASIC design
  - **Physical prototyping** enables designers to evaluate impact of physical design issues early in the design process with
  - Making **logical partitioning match physical partitioning** helps expose physical design tradeoffs at the RTL level

Next Lecture: Arvind will introduce using guarded atomic actions to describe hardware