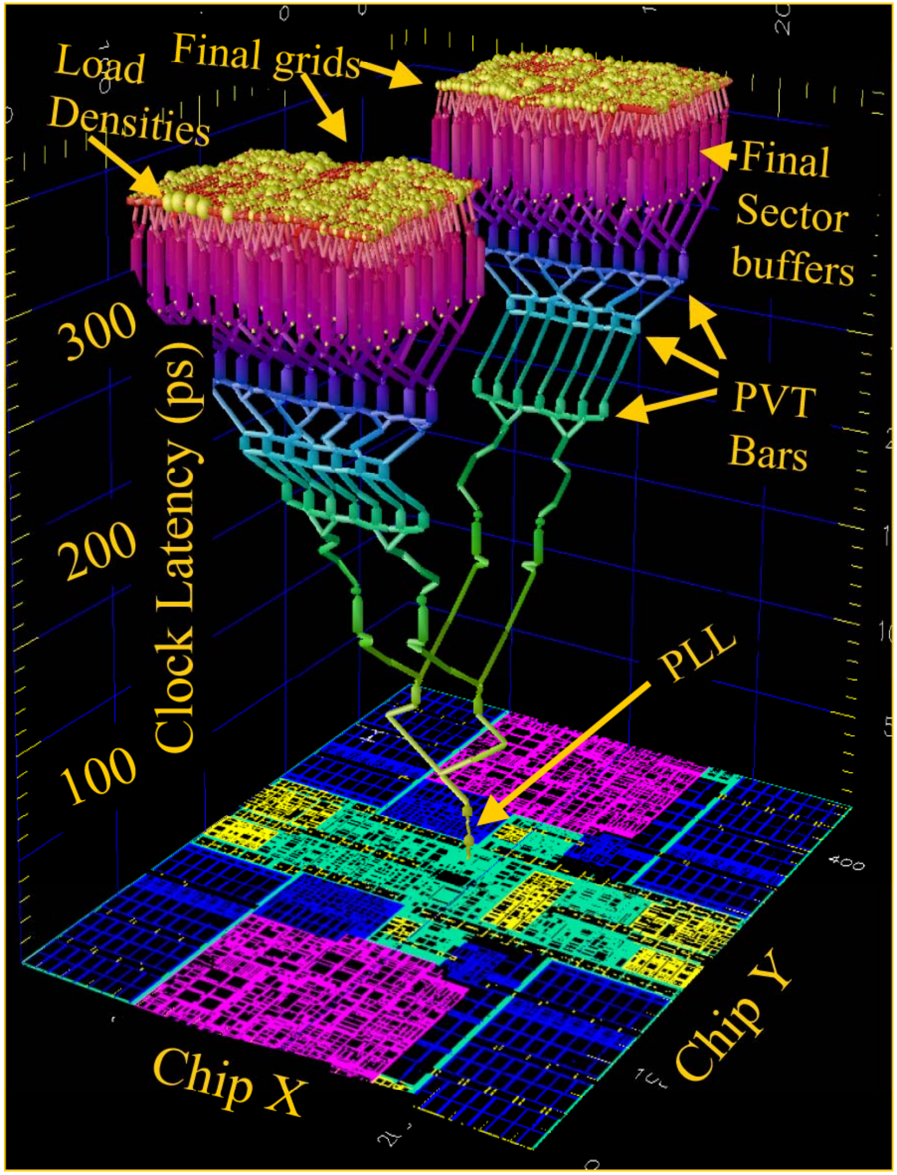
MODULE 6

TIMING DESIGN



Outline

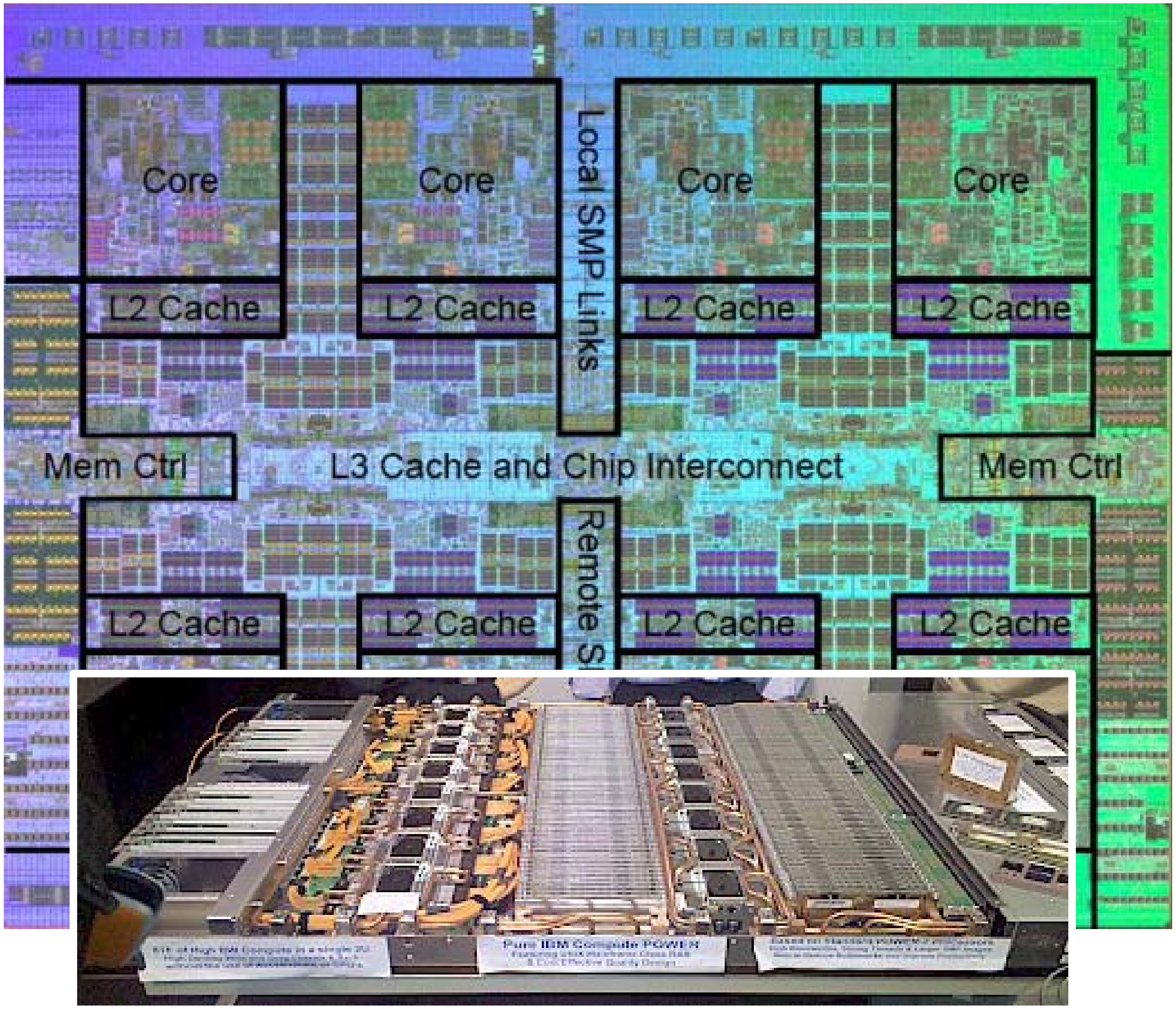
- Timing Design Background and Motivation
  - Delay variations, impact
  - Sequential circuits, synchronous design
  - Pipelining, metrics reminder
- The Clock Skew Problem
- Controlling Clock Skew
- Case Study

Get basic appreciation of some system level design issues

Design of LARGE Integrated Circuits

- Correct signal
  - Logic value
  - Right level (restoring logic, …)
- At right place
  - Interconnect (R, C, L)
  - Busses
  - Off-chip drivers, and receivers
- At right time
  - How to cope with (uncertain) delay

For Reference:
IBM Power 7 Chip, 775 supercomputer



2010
1.2 B transistors
45 nm
567 mm^2
Max 4.25 GHz
4 threads / core
8 cores / chip
4 chips / module
8 modules / drawer
12 drawers / rack
170 racks / system

524,288 cores / sys
2,097,152 threads/ sys
☺

Uncertain Delay

- Data-dependent Delay
- Short and long combinational paths
- Device parameters variations (§3.4)

  Batch to batch

  Wafer to wafer

  Die to die

  V_t threshold voltage

  k’ transconductance

  W, L dimensions
- Supply Variations

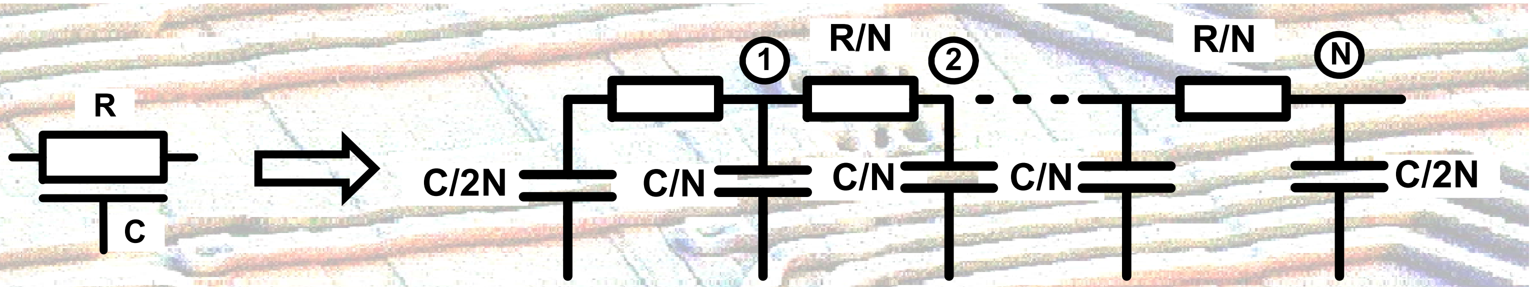
  IR drop, dI/dt drop, ringing,
- Interconnect Delay

  Don’t know length of line during logic design

  Delay at begin of line smaller than at end

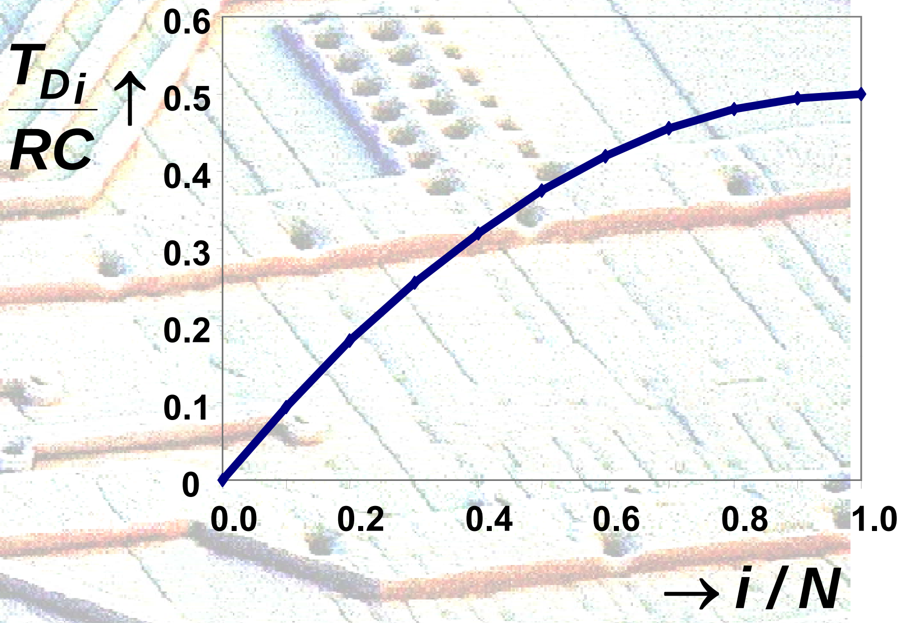
  Interconnect parameter variability

Delay Along a Wire



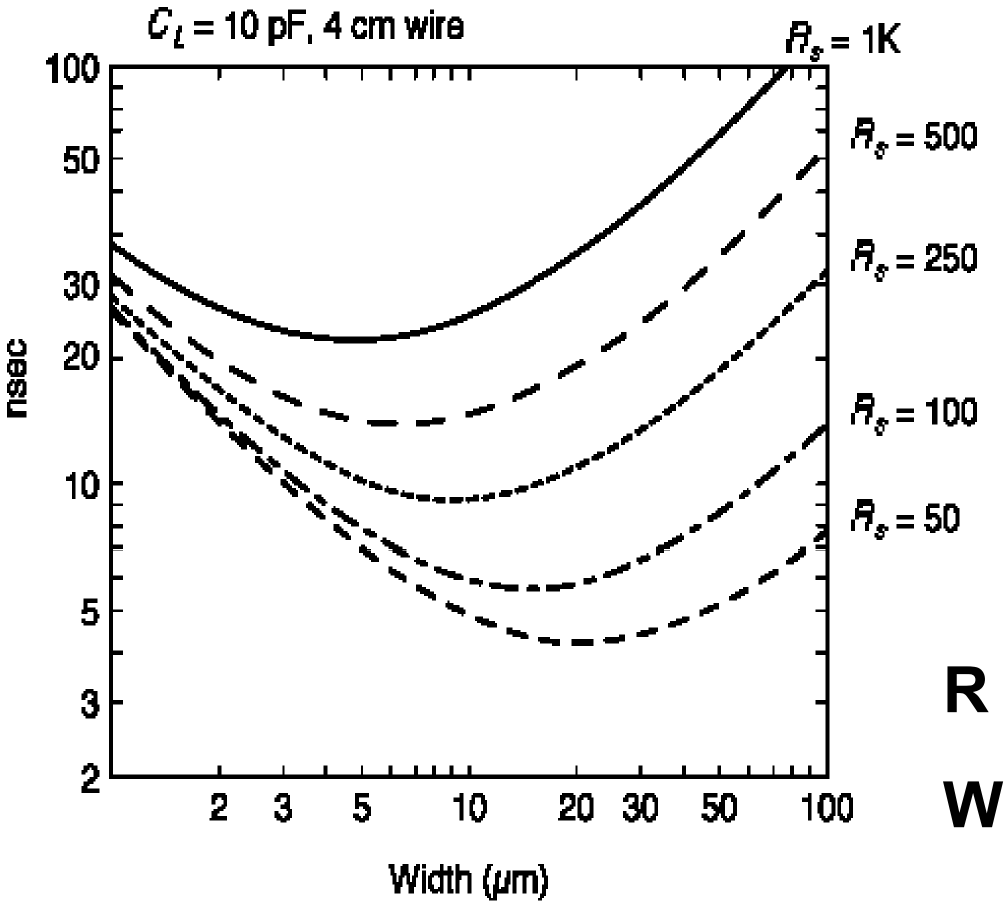
- Signal on RC line exhibits **finite propagation** speed
- **Delay** at beginning of line smaller than at end
- Follows from solving **diffusion equation**
- **Elmore delay model** also predicts propagation behavior (only approximately)

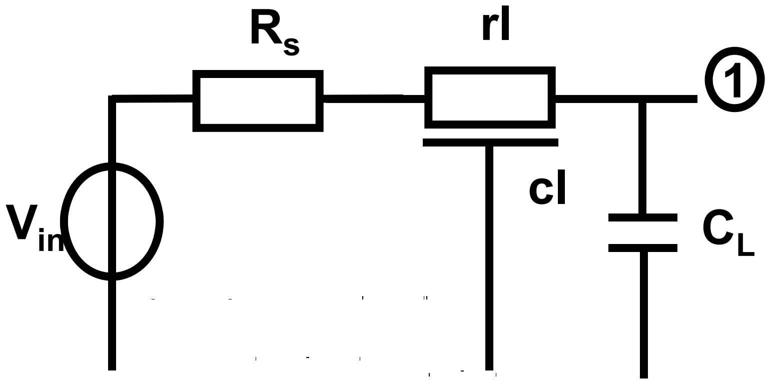
$$T_{Di} = \sum_{k=1}^{N} R_{ik} C_k$$



- **Exercise:** draw this graph for N = 3, N = 6, or use integration to solve limit for N →∞

Delay of Clock Wire





R = 0.07 Ω/□, C = 0.04 fF/μm^2

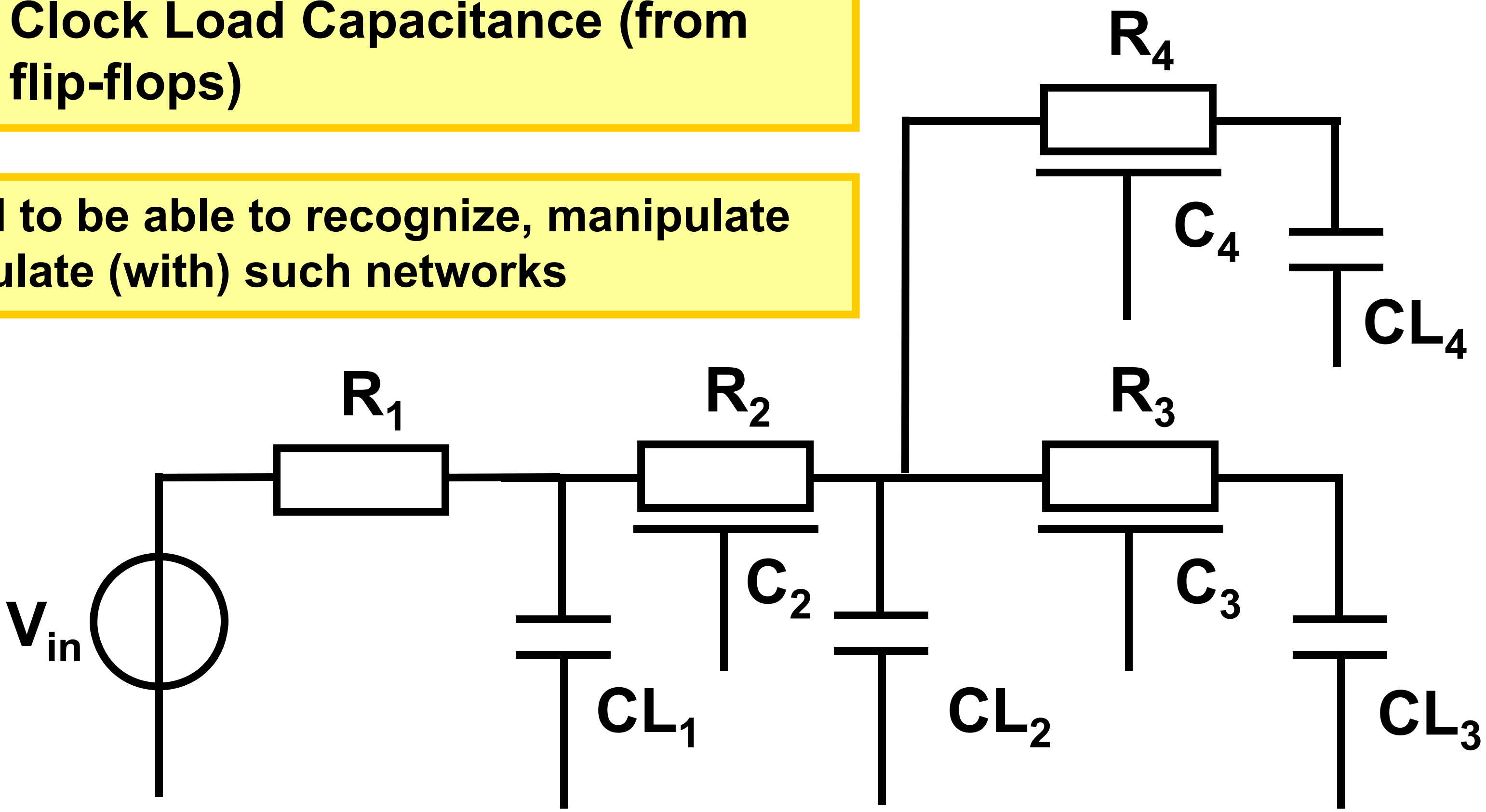
W = Width, r = R/W, c = C*W

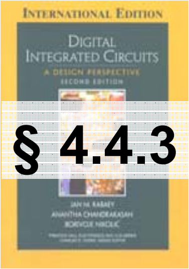
5ns compares with 200 MHz

Canonical Clock Tree Network

R_1	Clock Driver Output Resistance
R_2-R_4	Clock Line Segment Resistance
C_2-C_4	Clock Line Segment Capacitance
CL_1-CL_4	Clock Load Capacitance (from flip-flops)

You need to be able to recognize, manipulate and calculate (with) such networks





§ 4.4.3

Impact of Uncertain Delay

- Combinational circuits will eventually settle at correct output values when inputs are stable
- Sequential circuits
  - Have state
  - Must guarantee storing of correct signals at correct time
  - Require ordered computations

Sequential Circuits

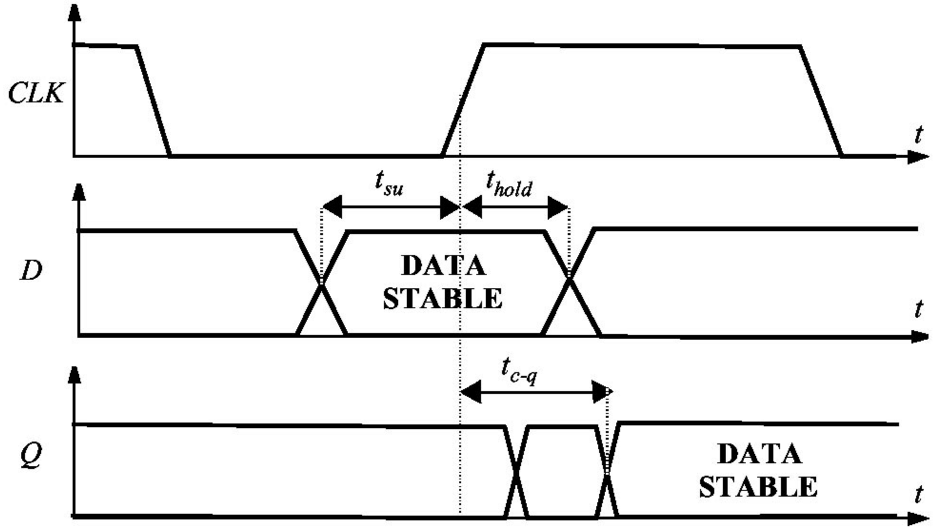
- Sequential circuits require **ordered computation**
- Several ways for **imposing ordering**
  - ✓ **Synchronous** (clock)
  - ✗ **Asynchronous** (unstructured)
  - ✗ **Self-timed** (negotiation)

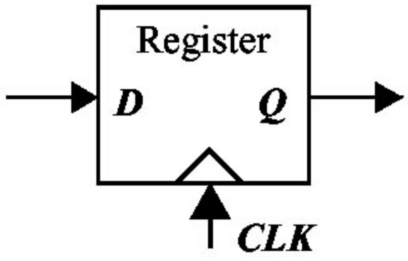
- Clock works like an **orchestra conductor**

Synchronous Design

- Global Clock Signal
- Synchronicity may be defeated by
  - Delay uncertainty in clock signal
  - Relative timing errors: clock skew
  - Slow logic paths
  - Fast logic paths

Timing Metrics Reminder





§ 7.1.1

§10.3.1

$t_{c\text{-}q}$: delay from clock (edge) to Q

t_{su} : setup time

t_{hold} : hold time

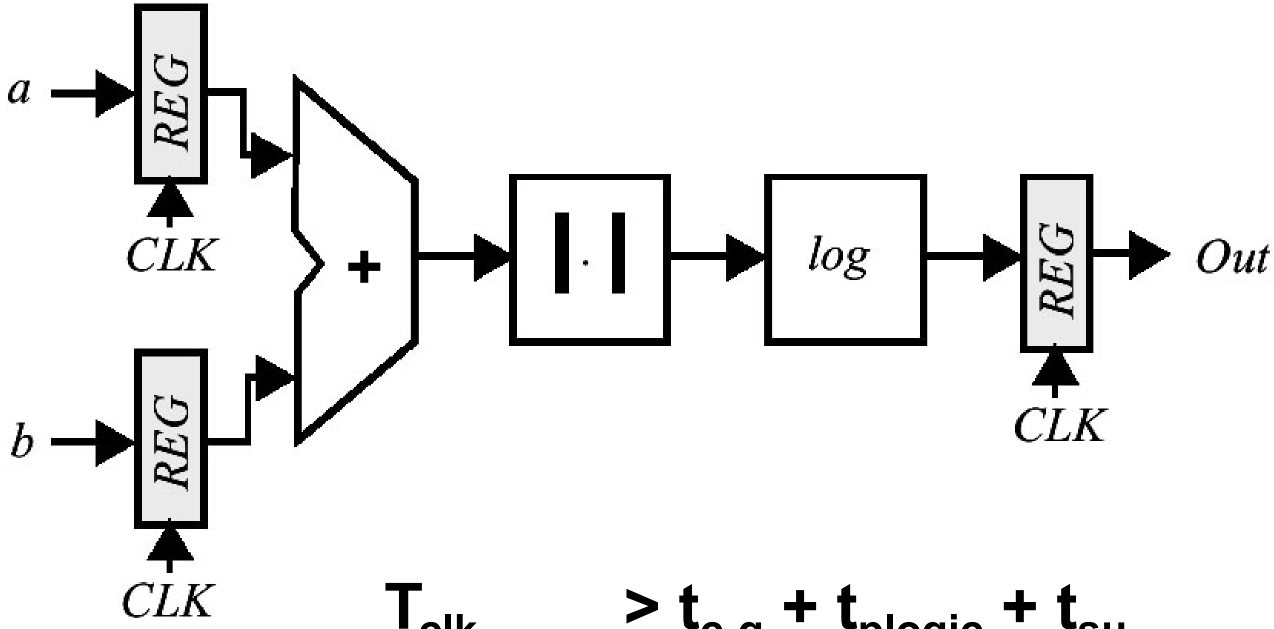
t_{plogic} : worst case propagation delay of logic

t_{cd} : best case propagation delay (contamination delay)

T : clock period

$$T \geq t_{c\text{-}q} + t_{plogic} + t_{su}$$
$$t_{cdregister} + t_{cdlogic} \geq t_{hold}$$

Sequential Circuit Timing.

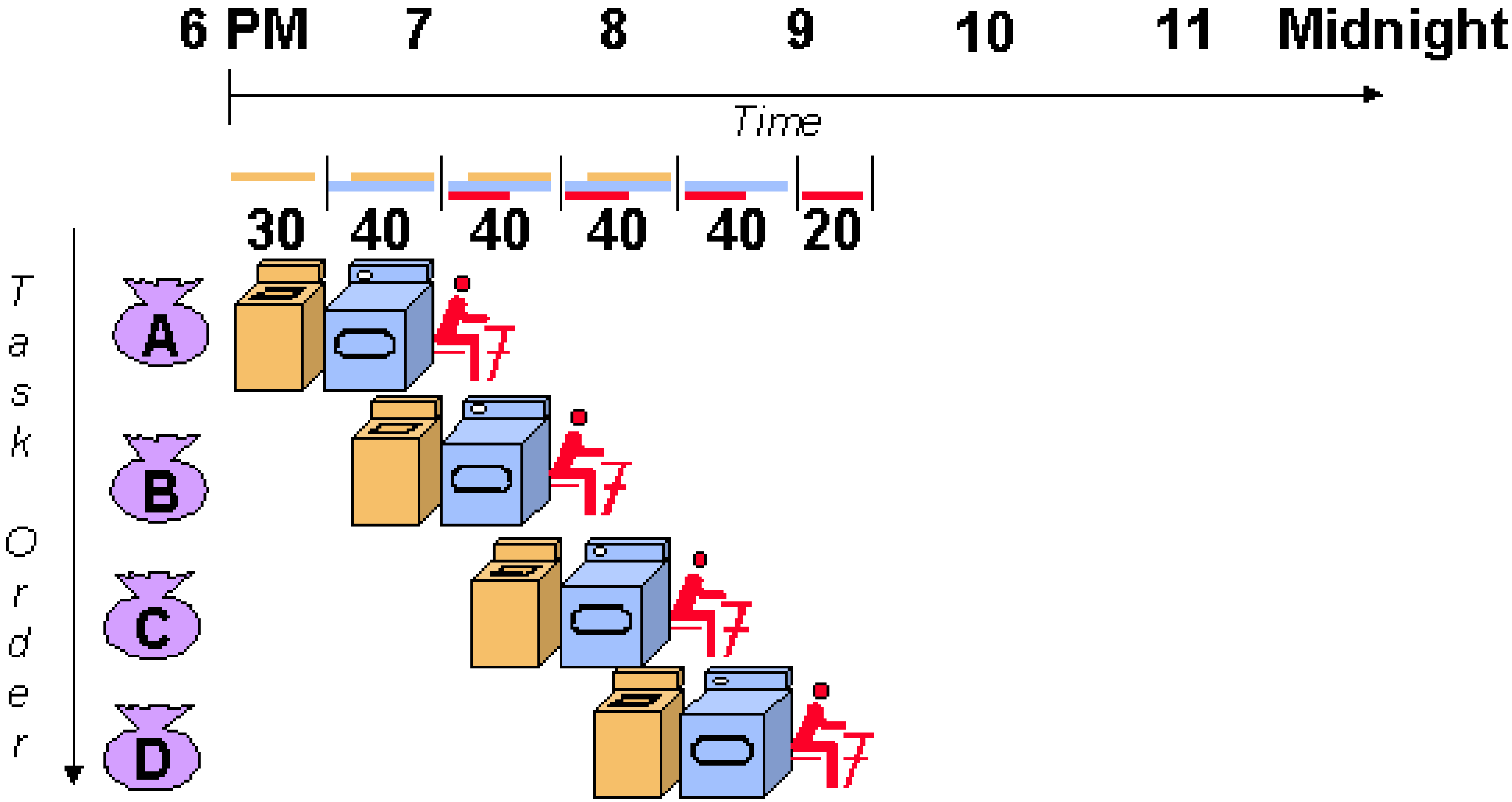


$$T_{clk} > t_{c\text{-}q} + t_{plogic} + t_{su}$$

$$t_{plogic} = t_{p,add} + t_{p,abs} + t_{p,log}$$

How to reduce T_{clk}?

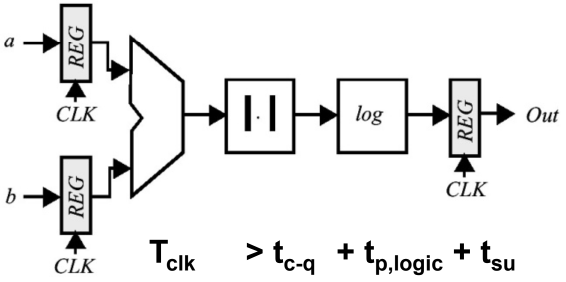
Pipelined Laundry System



Also: http://en.wikipedia.org/wiki/Pipelining

From http://cse.stanford.edu/class/sophomore-college/projects-00/risc/pipelining/index.html
which credited http://www.ece.arizona.edu/~ece462/Lec03-pipe/

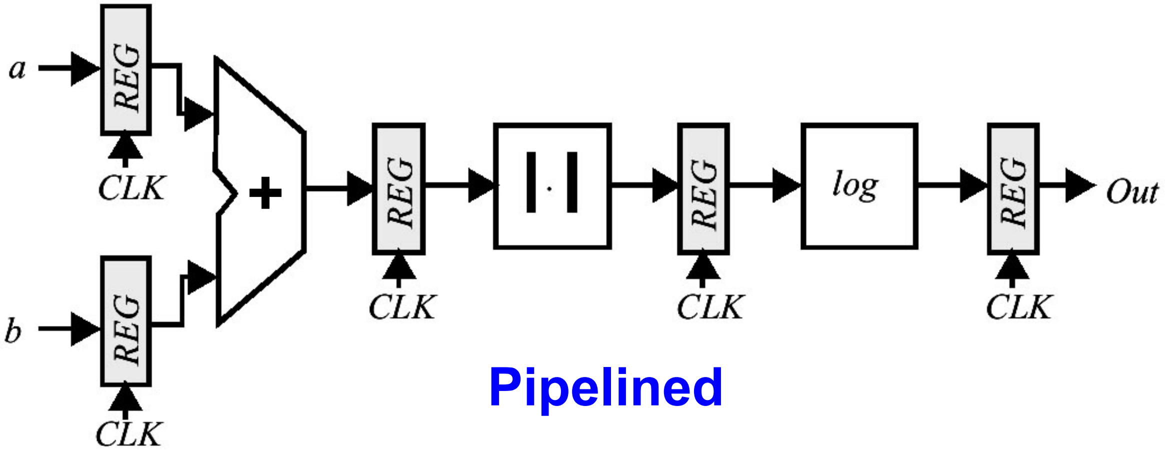
Pipelining



$T_{clk} > t_{c\text{-}q} + t_{p,logic} + t_{su}$

$T_{p,logic} = t_{p,add} + t_{p,abs} + t_{p,log}$

Non-pipelined



Pipelined

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

§ 7.5

$T_{clk} > t_{c\text{-}q} + \max(t_{p,add}, t_{p,abs}, t_{p,log}) + t_{su}$

- Improve resource utilization
- Increase functional throughput

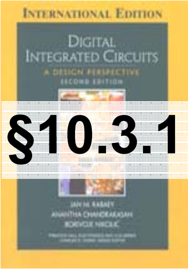
Pipelining Observations

- Very popular/effective measure to increase functional throughput and resource utilization
- At the cost of increased *latency*
- All high performance microprocessors excessively use pipelining in instruction fetch-decode-execute sequence
- Pipelining efficiency may fall dramatically because of branches in program flow
  - Requires emptying of pipeline and restarting
  - Partially remedied by advanced branch prediction techniques
- There was an era when all was dictated by GHz marketing drive
  - All a customer asked was: “How many GHz?”
  - Or said: “Mine is … GHz!”

Bottom line: more flip-flops, greater timing design problems

The Clock Skew Problem

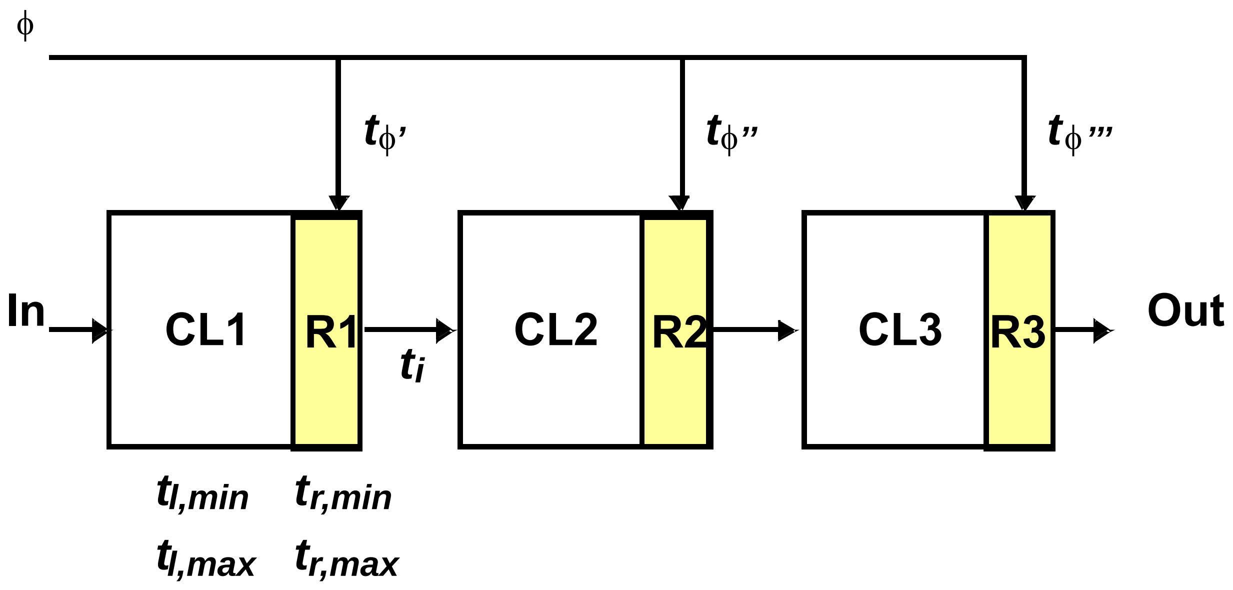
- In Single Phase Edge Triggered Clocking
- In Two Phase Master-Slave Clocking



§10.3.1

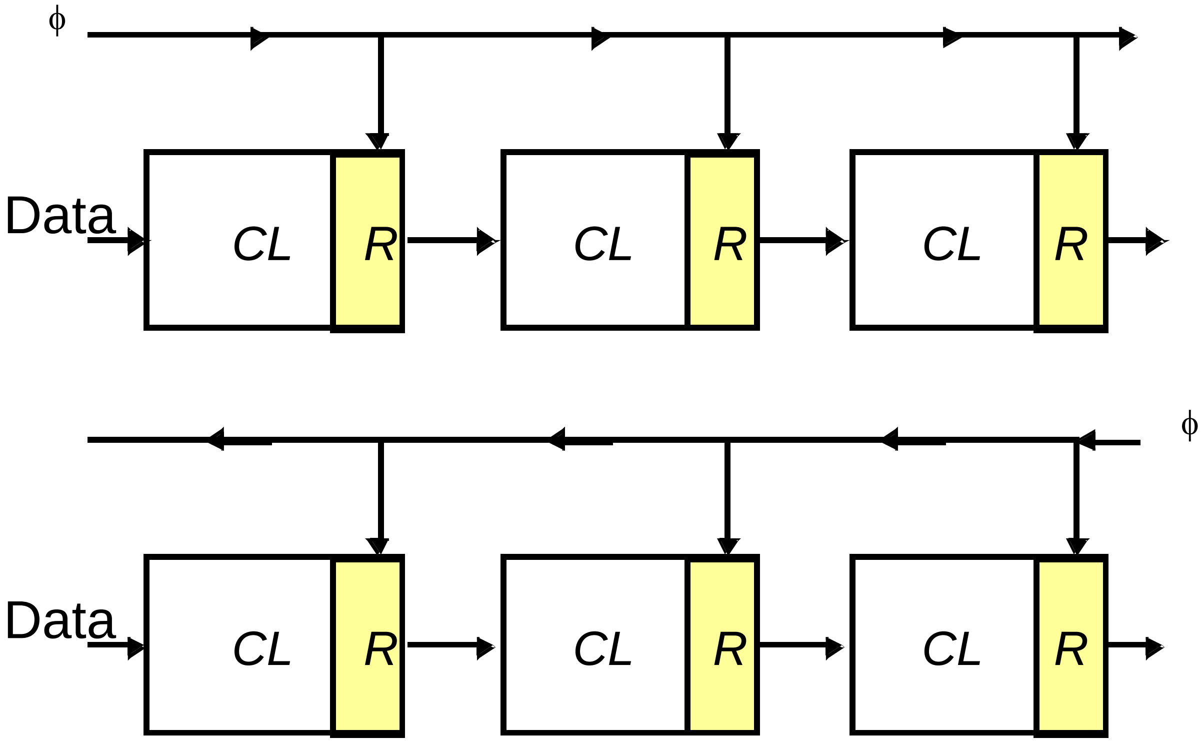
The Clock Skew Problem

Clock Rates >> 1 Ghz in CMOS



- **Clock Edge Timing Depends upon Position**
  - **Because clock network forms distributed RC line with lumped load capacitances at multiple sites (see earlier slide)**
- **(Relative) Clock Skew $\delta = t_{\phi''} - t_{\phi'}$**
- **Clock skew can take significant portion of T_{clk}**

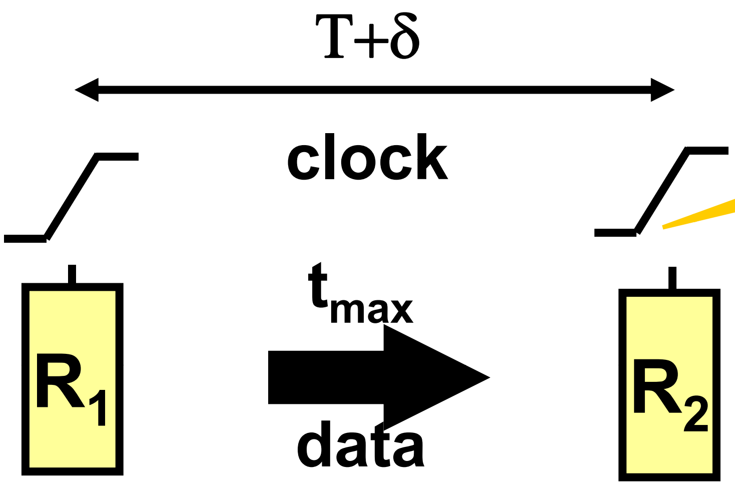
Positive and Negative Skew



(a) Positive skew

(b) Negative skew

Edge-Triggered Slow Path Skew Constraint



One clock tick later

Data at R2 should be stable before clock pulse is applied

Timing constraint (t_i = interconnect delay)

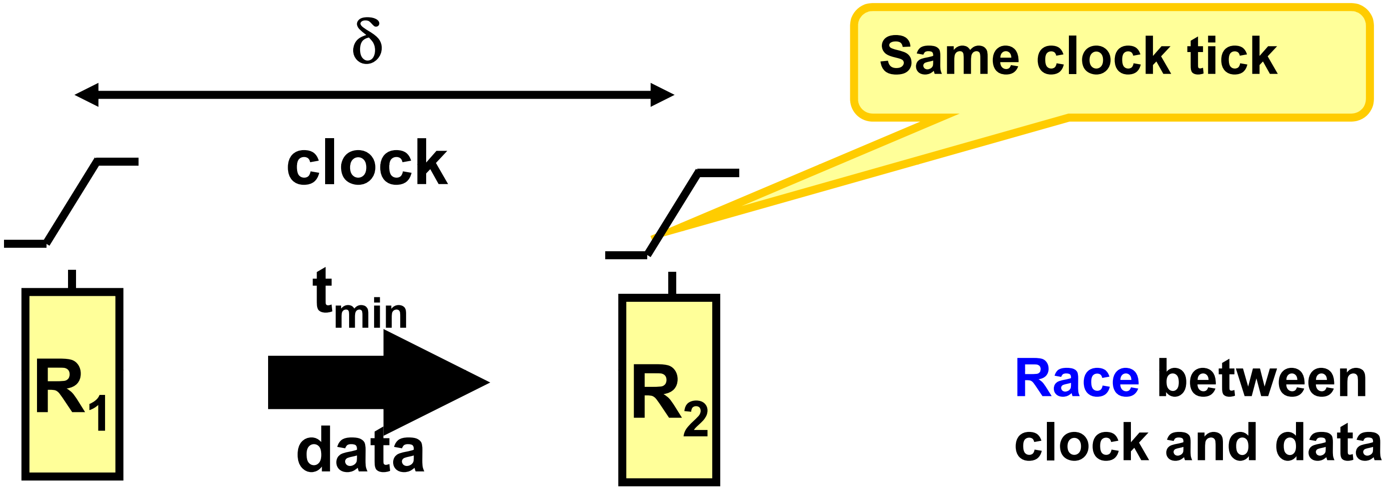
$$T + \delta \geq t_{max} \;\; = t_{p,logic} + t_i + \boxed{t_{c\text{-}q,max} + t_{su}}$$

Internal delay of flip-flop

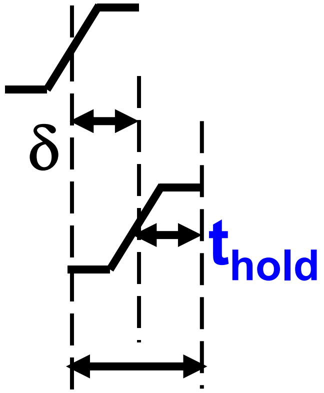
$$T \geq t_{max} - \delta$$

- **Minimum Clock Period Determined by Maximum Delay between Latches minus skew**

Edge-Triggered Fast Path Skew Constraint



Race between clock and data



$$\delta + t_{hold} < t_{cq,min} + t_{cd,logic} + t_i$$

$$\delta + t_{hold} < t_{cq,min} + t_{cd,logic} + t_i$$

$$\delta < t_{min} = t_{cq,min} + t_{cd,logic} + t_i - t_{hold}$$

- **Maximum Clock Skew** Determined by Minimum Delay between Latches

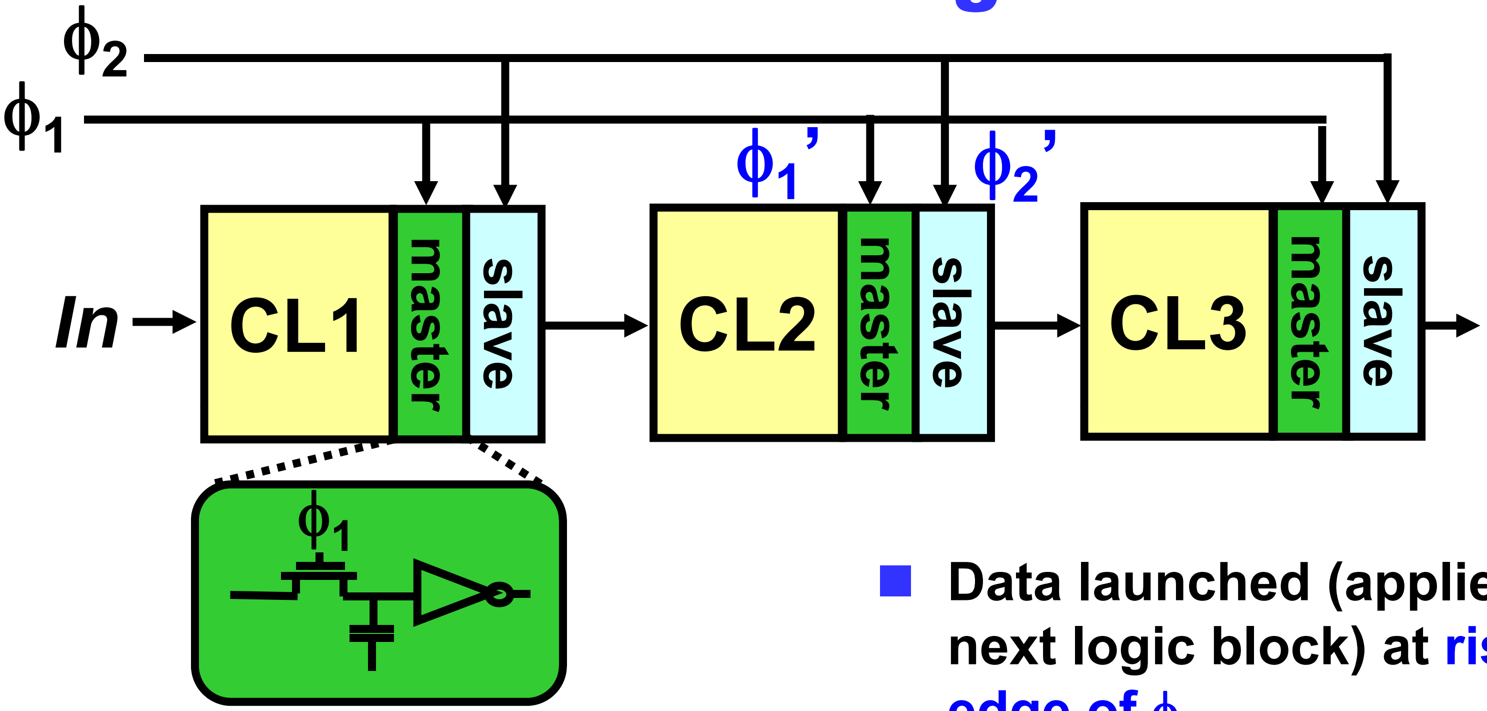
Clock Constraints in Edge-Triggered Logic

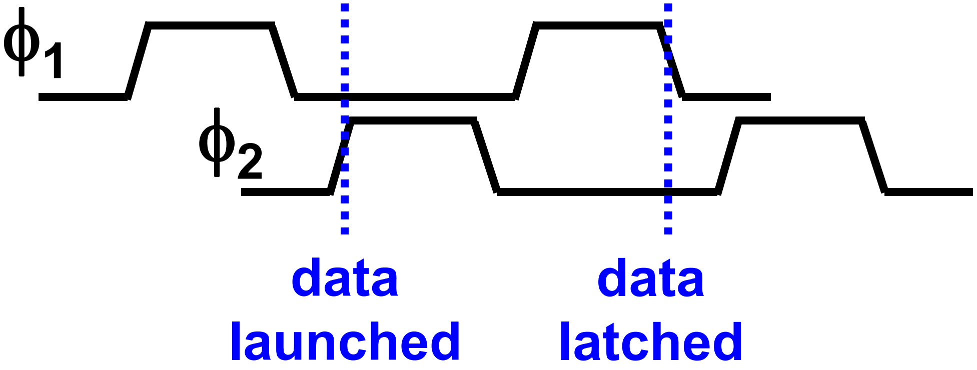
$$T \geq t_{max} - \delta$$

$$\delta \leq t_{min}$$

- Observe:
  - Minimum Clock Period Determined by Maximum Delay between Registers minus clock skew
  - Maximum Clock Skew Determined by Minimum Delay between Registers
- Conclude:
  - Positive skew must be bounded
  - Negative skew reduces maximum performance

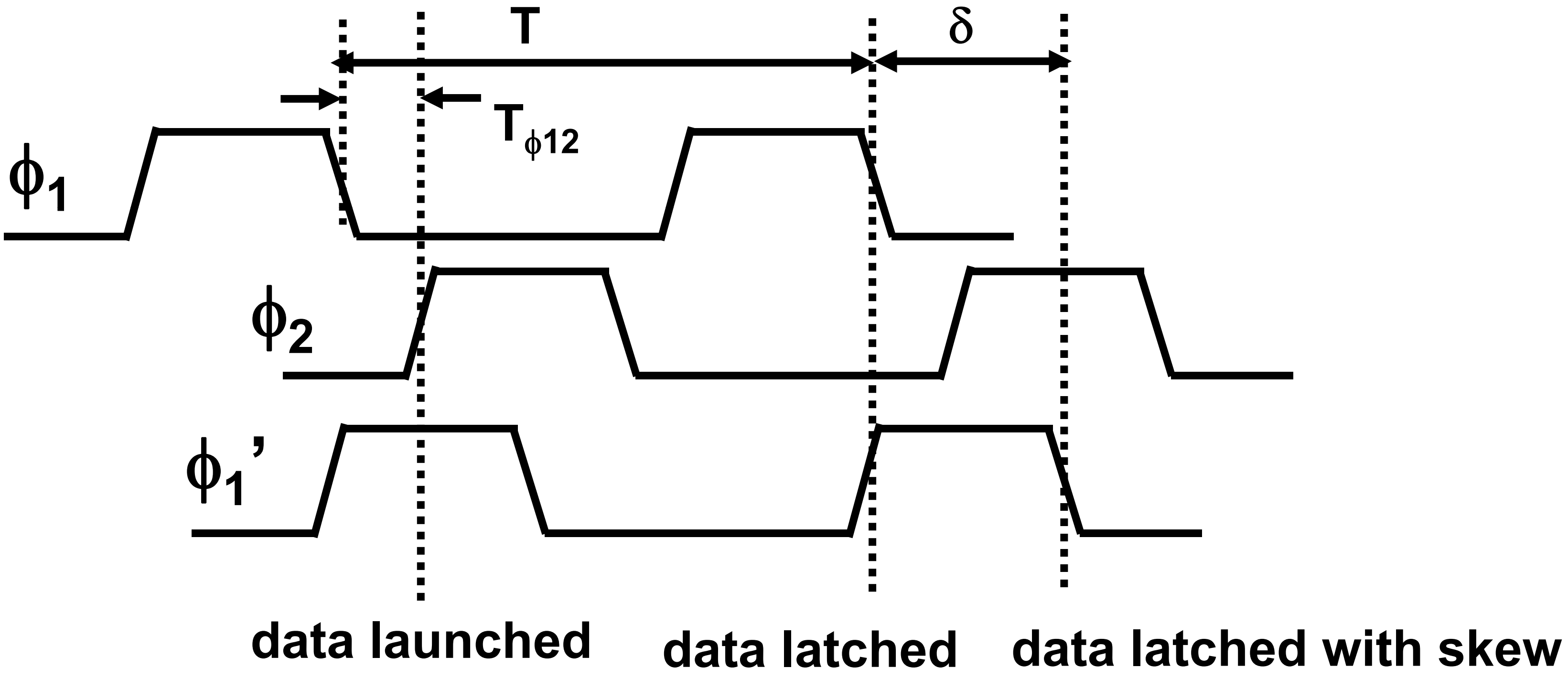
Clock Skew in Master-Slave 2-Phase Design





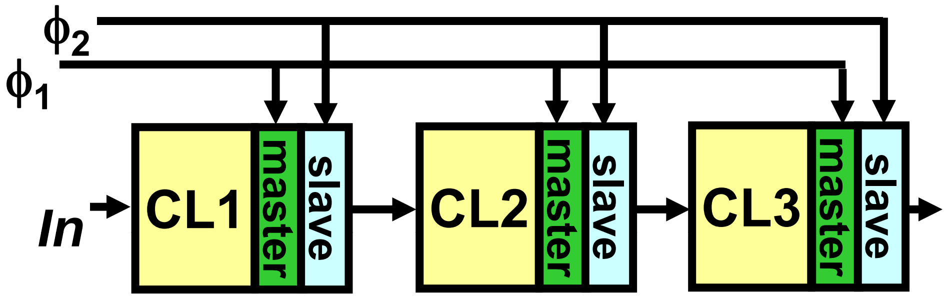
- Data launched (applied to next logic block) at **rising edge of ϕ_2**
- Data latched at **falling edge of ϕ_1**
- Clock skew modifies **timing rules**

2-Phase M/S Slow Path Skew Constraint

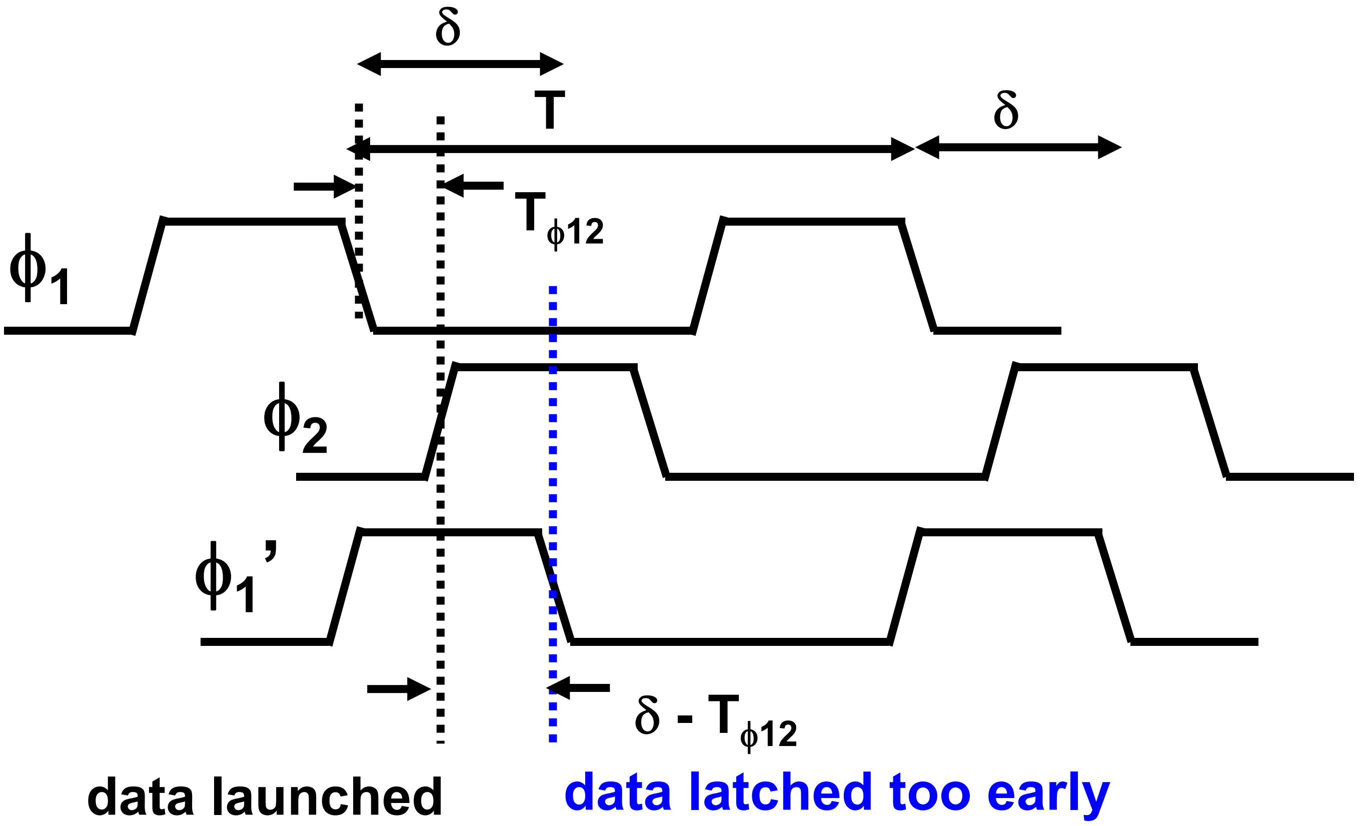


Timing constraint

$$t_{max} \le T - T_{\phi12} + \delta$$

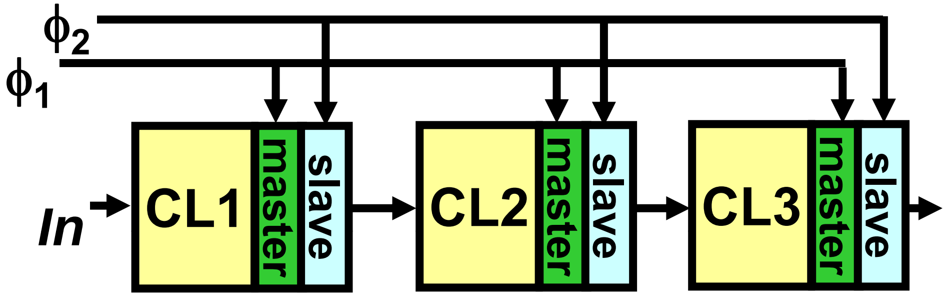


2-Phase M/S Fast Path Skew Constraint



Timing constraint

$t_{min} \geq \delta - T_{\phi12}$



Clock Constraints in 2-Phase Design

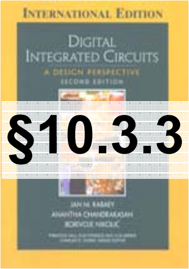
$T \geq t_{max} - \delta + T_{\phi 12}$

$\delta \leq t_{min} + T_{\phi 12}$

- Observe:
  - Minimum Clock Period Determined by Maximum Logic Delay t_{max} minus clock skew δ
  - plus $T_{\phi 12}$ (0-0 Overlap Time)
  - Maximum Clock Skew Determined by Minimum Logic Delay plus $T_{\phi 12}$
- Conclude again:
  - Negative skew reduces maximum performance
- However:
  - Positive skew can be made harmless by increasing $T_{\phi 12}$
  - $T_{\phi 12}$ reduces maximum performance

Controlling Clock Skew

Case Study

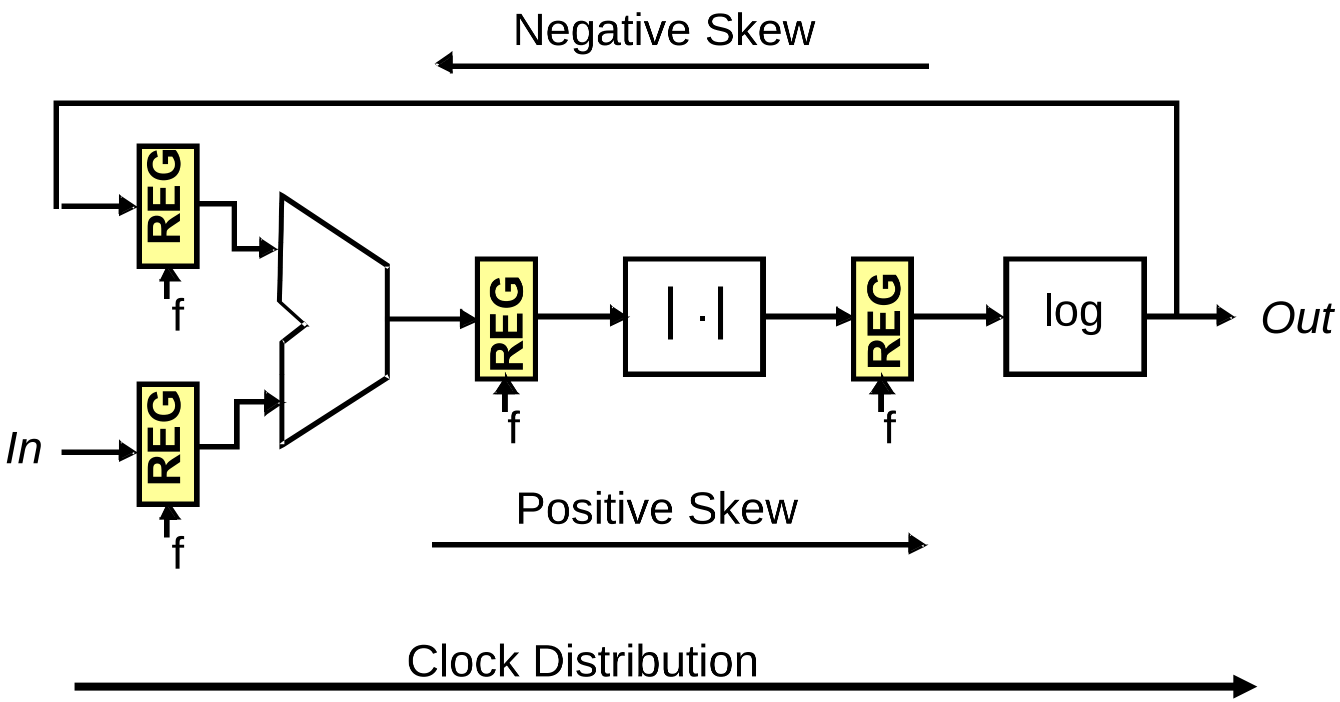


§10.3.3

Countering Clock Skew Problems

- Routing the clock in opposite direction of data (negative skew)
  - Hampers performance
  - Dataflow not always uni-directional
  - Maybe at sub circuit (e.g. datapath) level
  - Other approaches needed at global chip-level
  - Useful skew (or beneficial skew) is serious concept
- Enlarging non-overlap periods of clock [only with two-phase clocking]
  - Hampers performance
  - Can theoretically always be made to work
  - Delay in clock network may require impractical/excessively large scheduled $T_{\phi 12}$ to guarantee minimum $T_{\phi 12}$ everywhere across chip
  - Is becoming less popular for large high performance chips

Dataflow not unidirectional



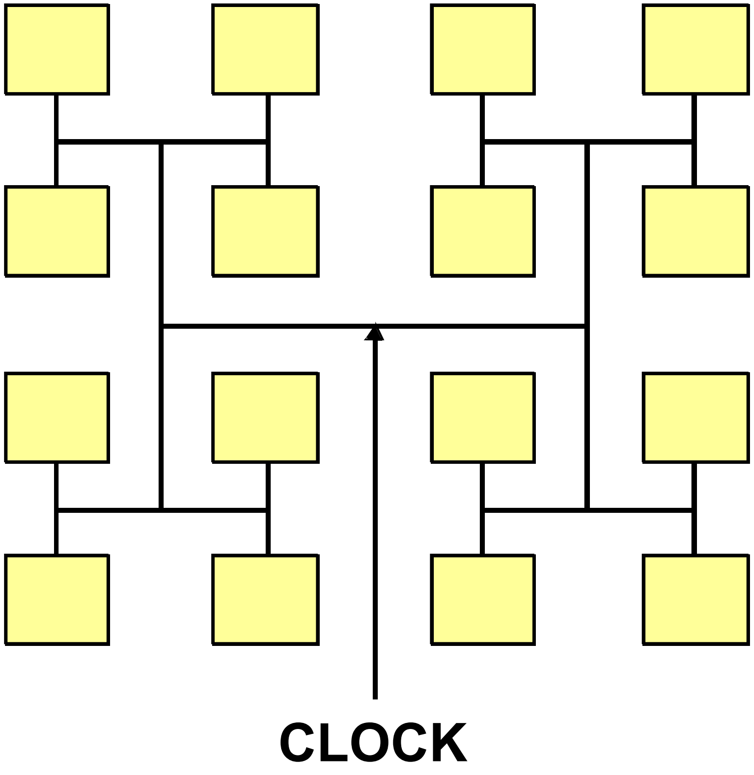
- **Data and Clock Routing**
- **Cannot unambiguously route clock in opposite direction of data**
- **Need bounded skew**

Need bounded Skew

- Bounded skew most practical measure to guarantee functional correctness without reducing performance

- Clock Network Design
  - Interconnect material
  - Shape of clock-distribution network
  - Clock driver, buffers
  - Clock-line load
  - Clock signal rise and fall times
  - …

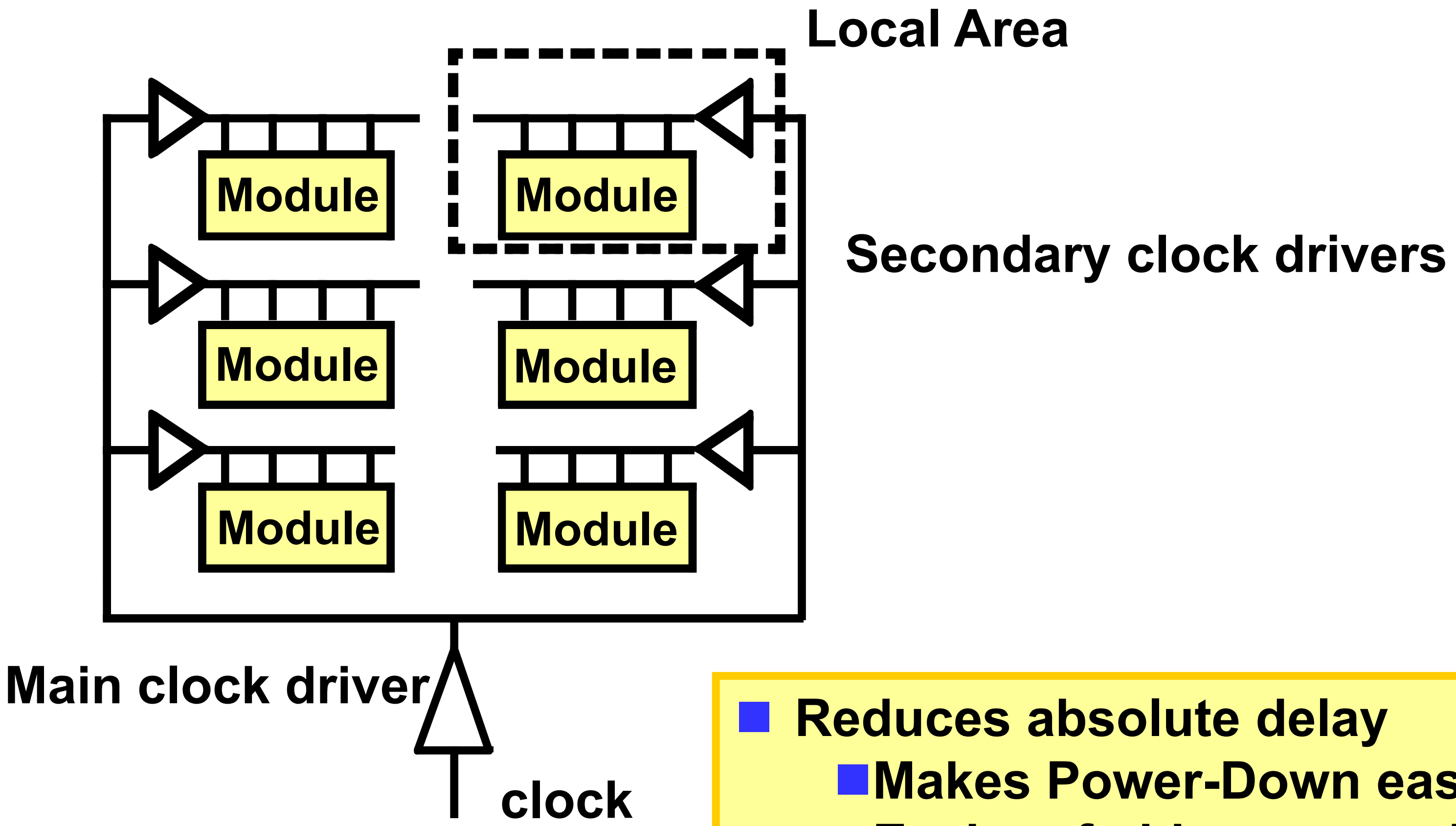
H-tree Clock Network



- All blocks equidistant from clock source ⇒ zero (relative) skew
- Sub blocks should be small enough to ignore intra-block skew
- In practice perfect H-tree shape not realizable

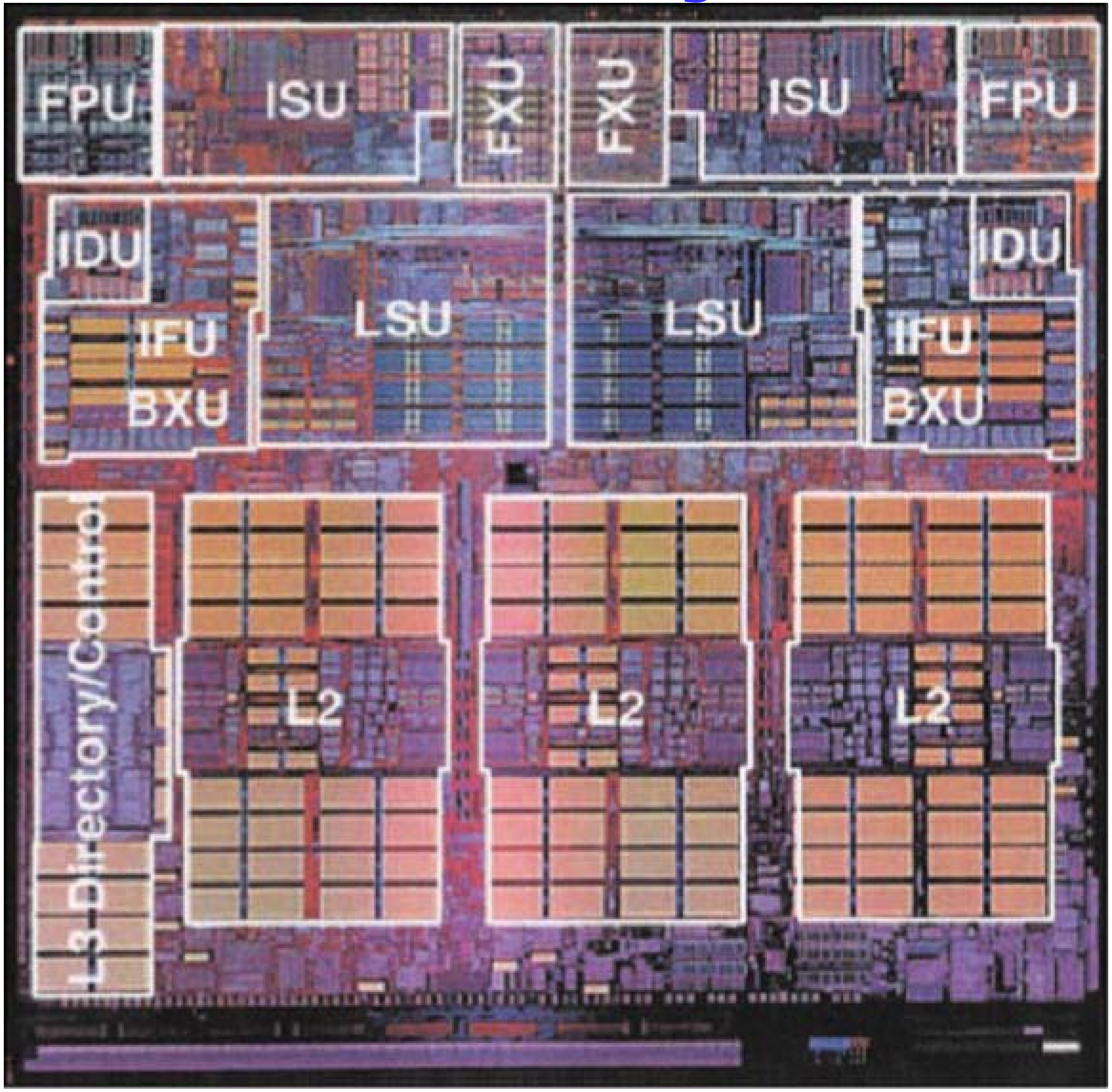
Observe: Only Relative Skew Is Important

Clock Network with Distributed Buffering



- Reduces absolute delay
  - Makes Power-Down easier
  - Easier of-chip communication
- Sensitive to variations in Buffer Delay

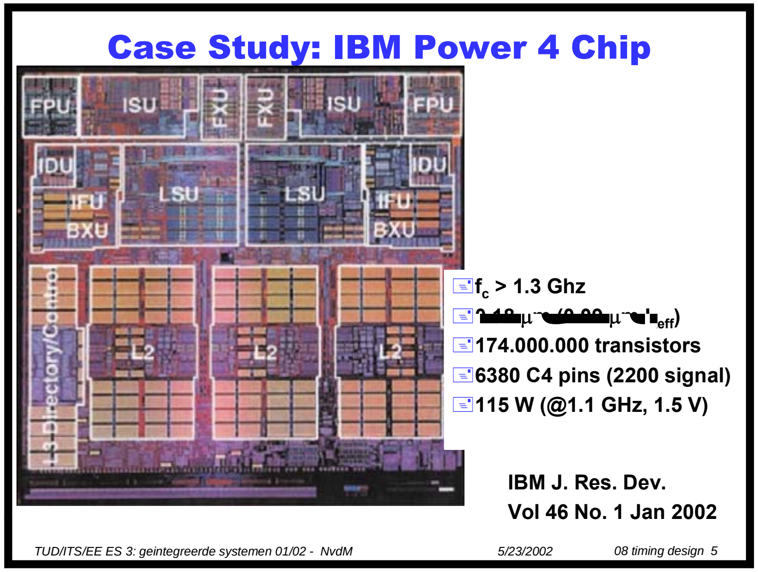
Case Study: IBM Power 4 Chip



- f_c > 1.3 Ghz
- 0.18 μm (0.09 μm L_{eff})
- 174.000.000 transistors
- 6380 C4 pins (2200 signal)
- 115 W (@1.1 GHz, 1.5 V)

IBM J. Res. Dev.
Vol 46 No. 1 Jan 2002

Clock Distribution

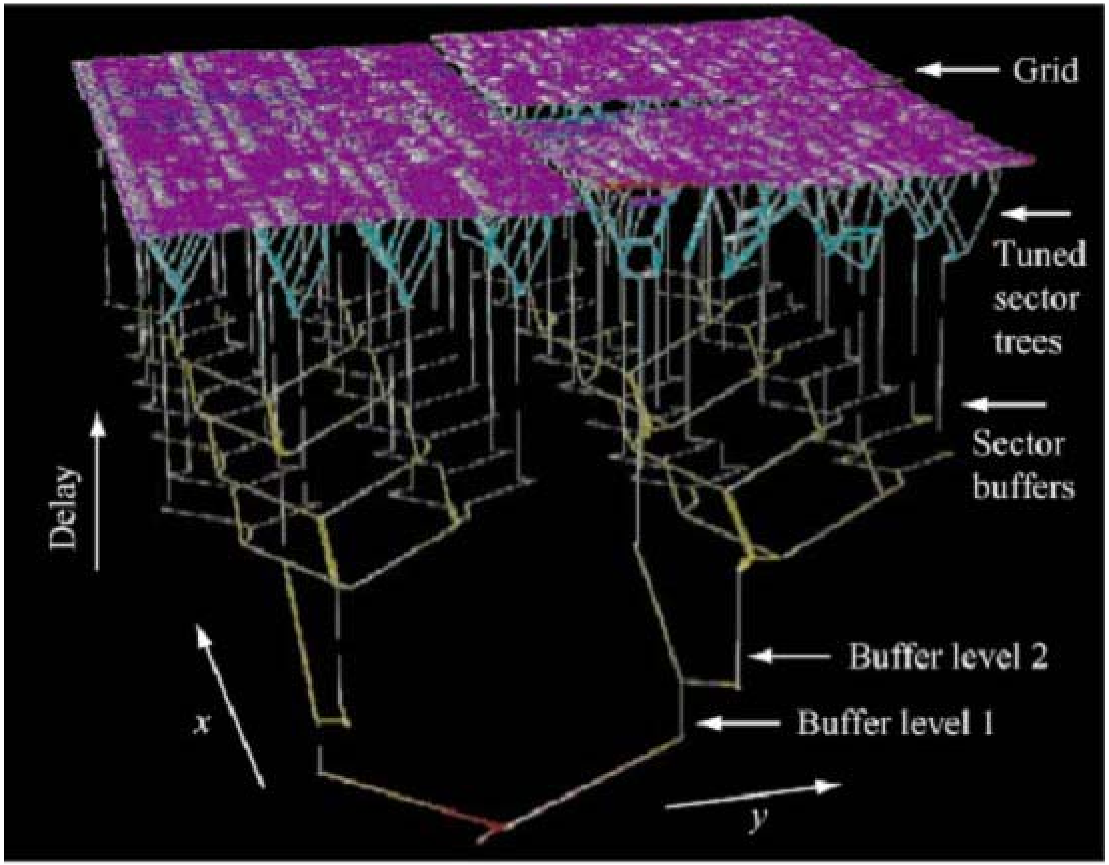

FPU
ISU
FXU
FXU
ISU
FPU
IDU
IFU
BXU
LSU
LSU
IDU
IFU
BXU
L3 Directory/Control
L2
L2
L2

Case Study: IBM Power 4 Chip

- f_c > 1.3 Ghz
- 0.18 µm (0.09 µm L_{eff})
- 174.000.000 transistors
- 6380 C4 pins (2200 signal)
- 115 W (@1.1 GHz, 1.5 V)

IBM J. Res. Dev.
Vol 46 No. 1 Jan 2002

TUD/ITS/EE ES 3: geintegreerde systemen 01/02 - NvdM 5/23/2002 08 timing design 5



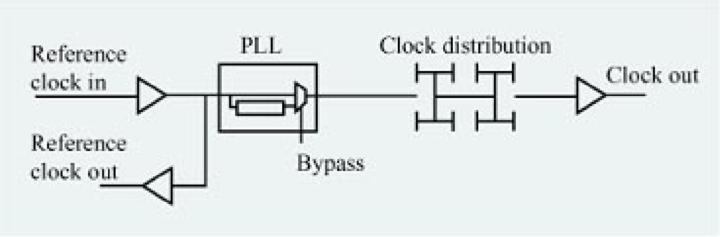


Figure 6

Schematic diagram of global clock generation and distribution.

Figure 7

3D visualization of the entire global clock network. The *x* and *y* coordinates are chip *x*, *y*, while the *z* axis is used to represent delay, so the lowest point corresponds to the beginning of the clock distribution and the final clock grid is at the top. Widths are proportional to tuned wire width, and the three levels of buffers appear as vertical lines.

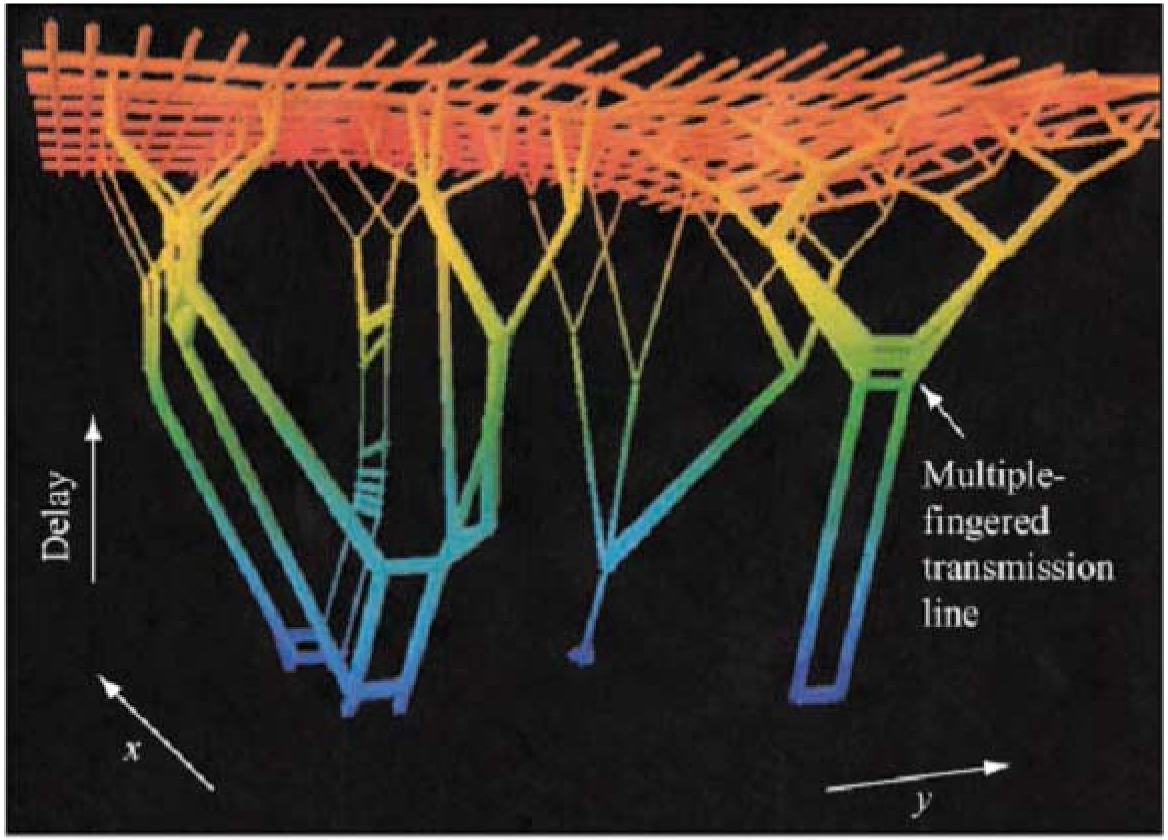
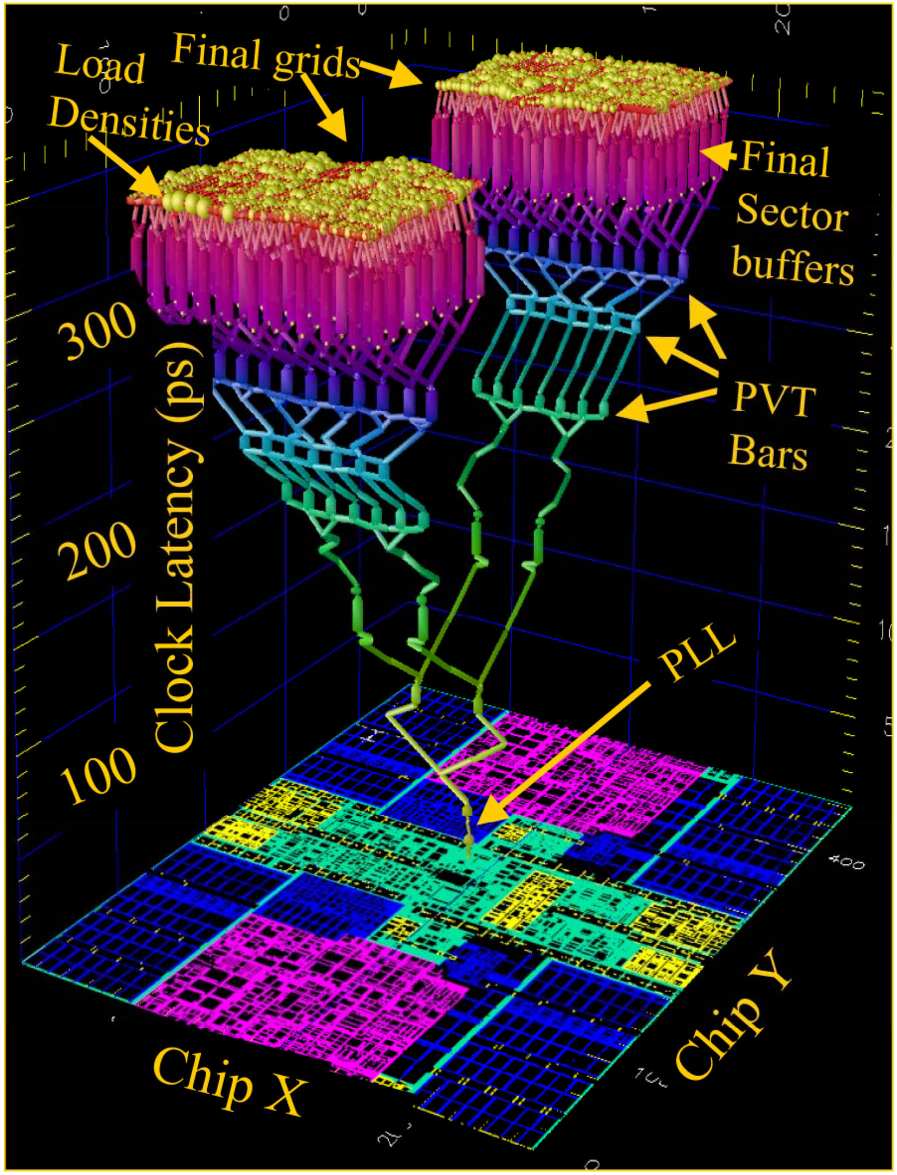
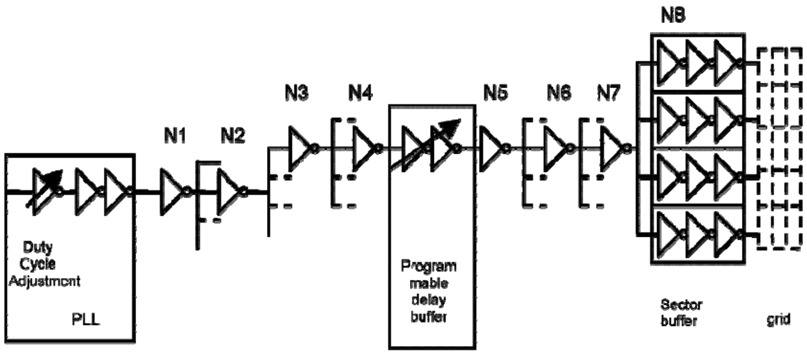


Figure 8

Visualization of four of the 64 sector trees driving the clock grid, using the same representation as Figure 7. The complex sector trees and multiple-fingered transmission lines used for inductance control are visible at this scale.

Power6 Clock Distribution

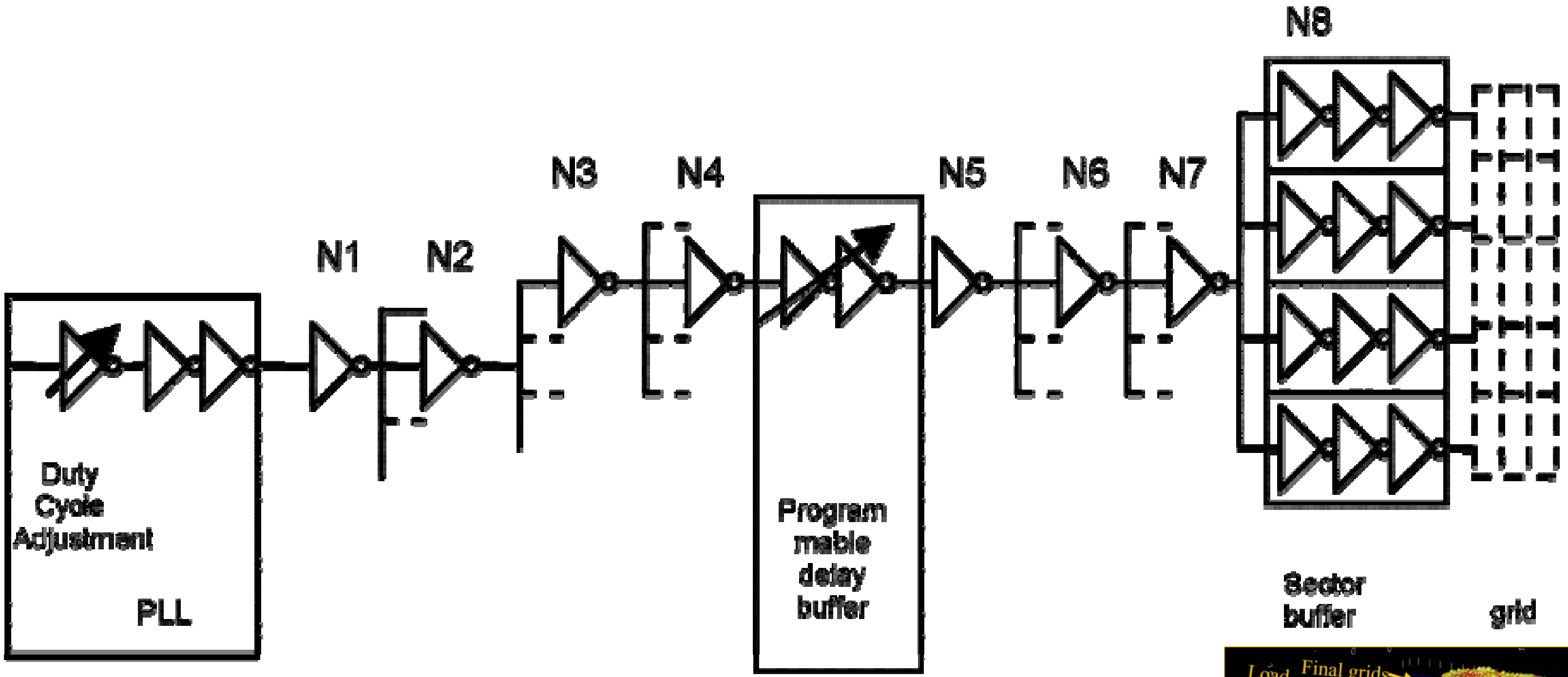


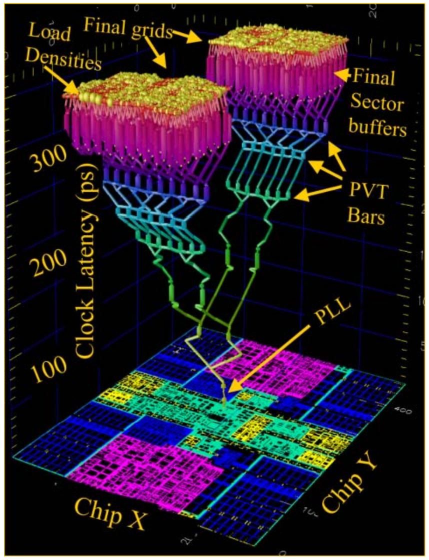


- Latency ~ cycle time

friedrich, isscc 2007

Power6 Clock Distribution

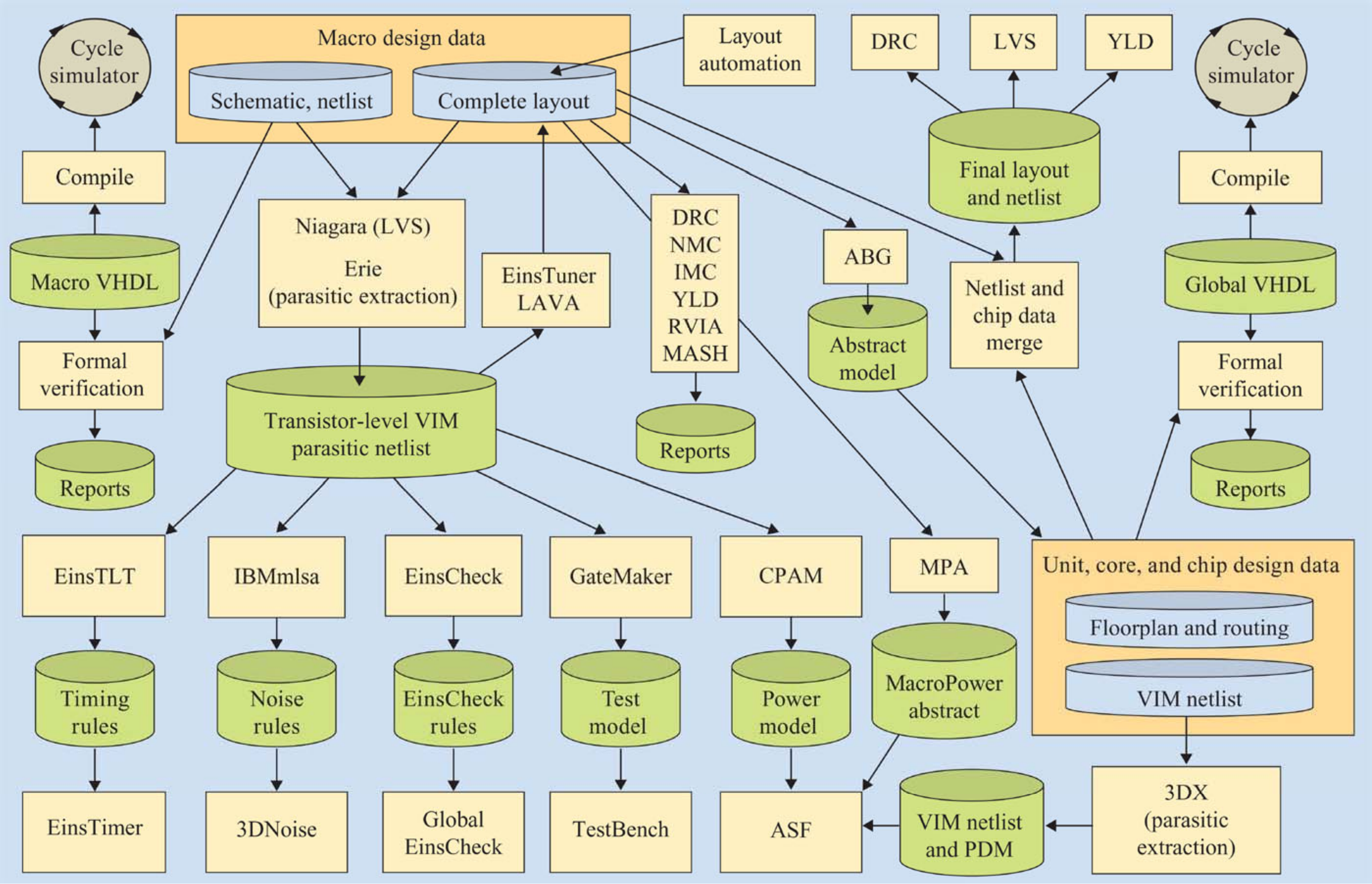




stolt, jssc 2008

Philip Restle animations

IBM Power6 Physical Design Flow



berridge, ibmjrd 2007

Timing Design

- Clocking Scheme is important design decision
- Influences
  - Power
  - Robustness
  - Ease of design, design time
  - Performance
  - Area, shape of floor plan
- Needs to be planned early in design phase
- But is becoming design bottle neck nevertheless
  - Clock frequencies increase
  - Die sizes increase
  - Clock skew significant fraction of T_{clk}
- Alternatives
  - Asynchronous *or* self-timed

Not in this course ☺

More Clocking Issues

- Clock power (~ 30 % of total chip power)
- Asynchronous design
  for reasons of power, and variability
- Resonant clocking
  Use principles of (buck) convertors
  to recycle energy
- Multiple clock domains, power domains, sleep states
  - for standby modes
  - for integrating IP

Summary

- Timing Design Background and Motivation
  - Delay variations, impact
  - Sequential circuits, synchronous design
  - Pipelining, metrics reminder
- The Clock Skew Problem
  - In single Phase Edge Triggered Clocking
  - In Two Phase Master-Slave Clocking
- Controlling Clock Skew
- Case Study

Got basic appreciation of some system level design issues?