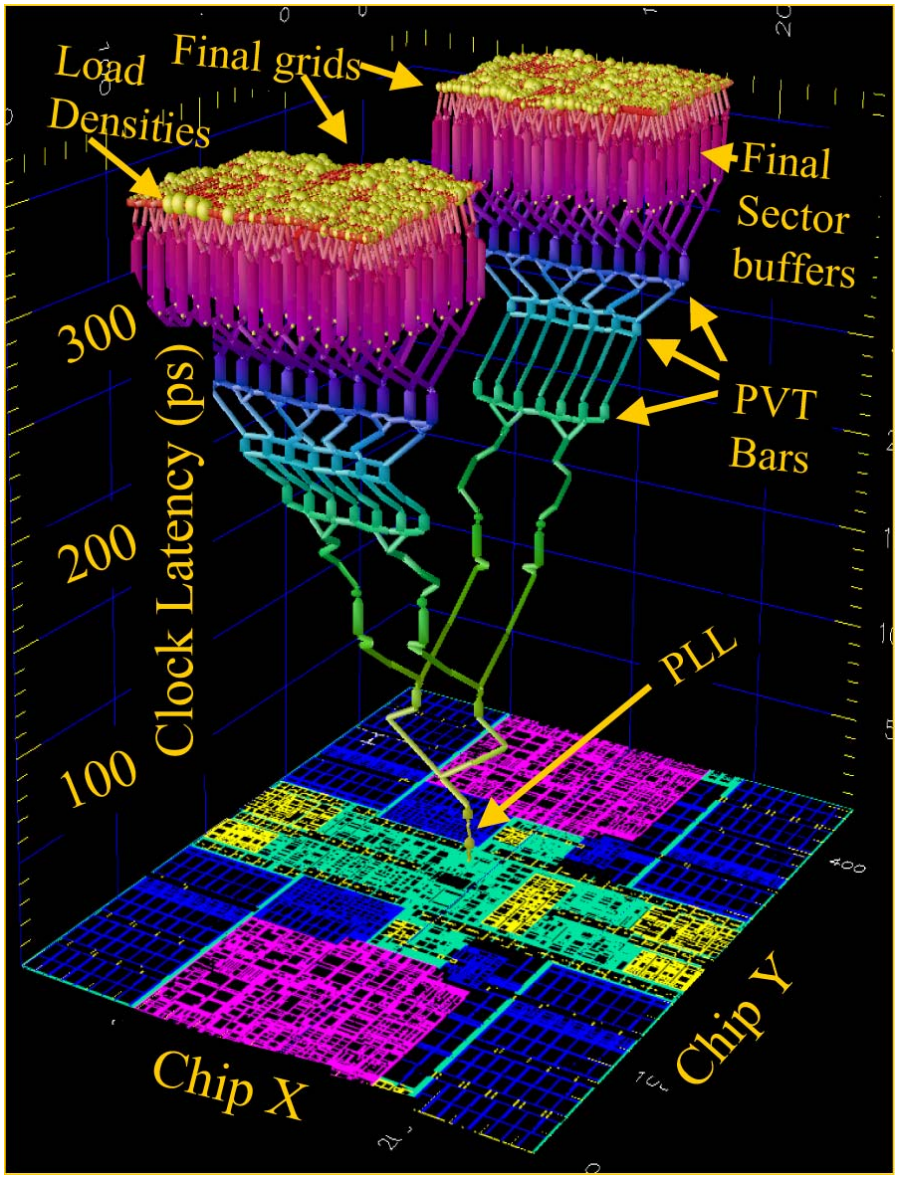
# MODULE 7

# TIMING DESIGN



# Course Material for Timing Design

| | | | |
|---|---|---|---|
| P | 10.1 | Introduction | 492 |
| I | 10.2 | Timing Classification | 492 – 495 |
| P | 10.3.1 | Synchronous Timing Basics | 495 – 500 |
| I | | Clock Jitter | 500 |
| I | | The combined impact of Skew and Jitter | 501 – 502 |
| I | 10.3.2 | Sources of Skew and Jitter | |
| I | 10.3.3 | Clock Distribution Techniques | |
| O | 10.3.4 | Latch-Based Clocking | 516 – 518 |
| O | 10.4-10.7 | Self-Timed Circuit Design – Future Directions | 519 – 551 |
| P | *7.5 – 7.5.1 (!)* | Pipelining | 358-361 |

***Material from Chapter 10, one section from Chapter 7***

# Outline

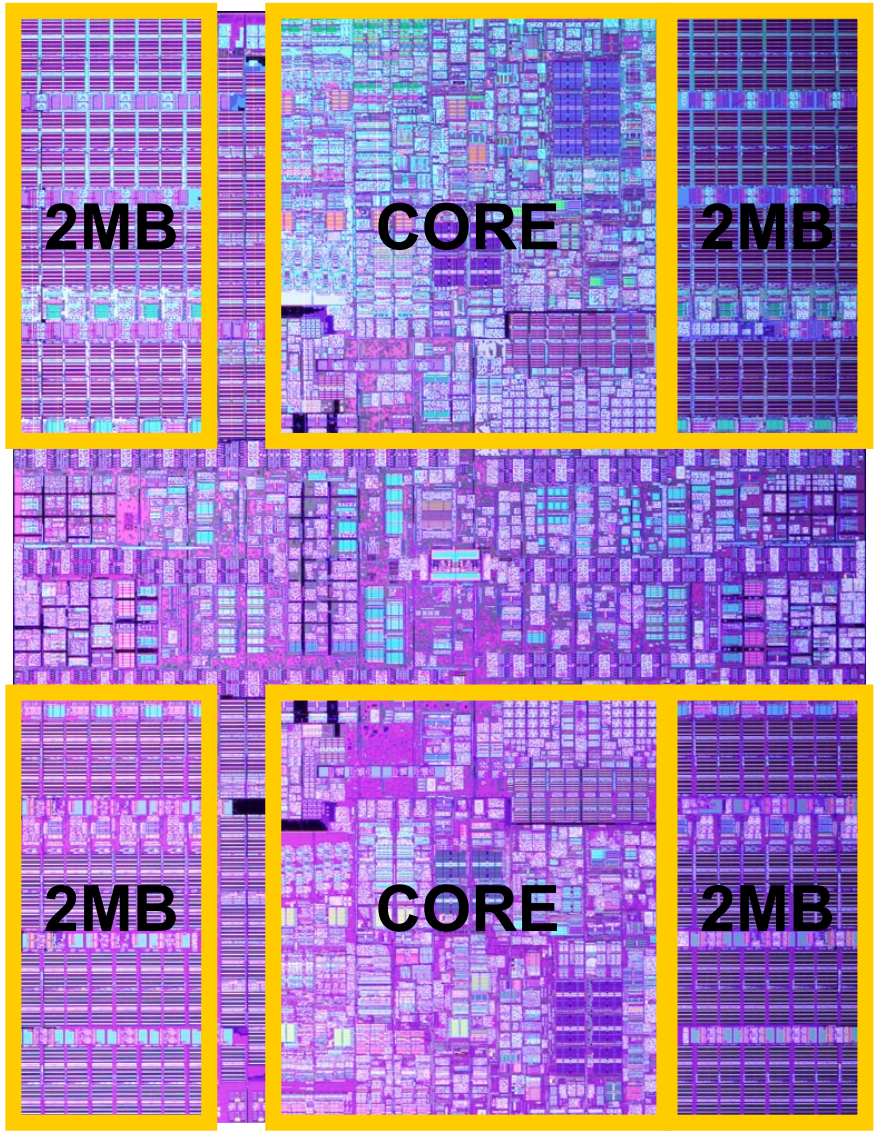
- Timing Design Background and Motivation
  - Delay variations, impact
  - Sequential circuits, synchronous design
  - Pipelining, metrics reminder
- The Clock Skew Problem
- Controlling Clock Skew
- Case Study

**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**

# Case Study: IBM Power6 CPU

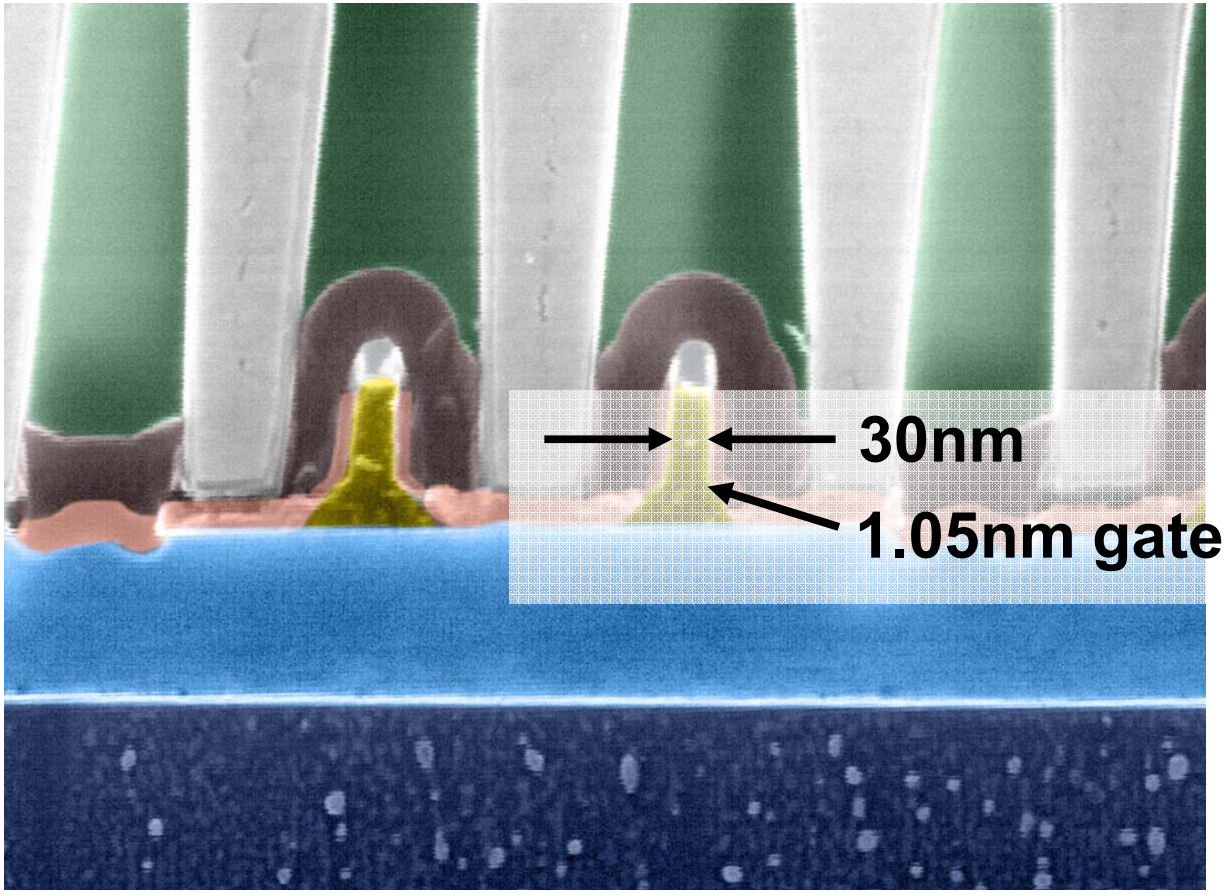



- introduced 21 may 2007
- 64 bit, dual core
- 790 million transistors
- 4.7 ($5^{+}$) GHz
- 65nm SOI, 10 Cu levels interconnect
- 2 Cores
- 8 MB on-chip level2 cache
- processor bandwidth: 300GB/sec
- 1953 signal I/O, 5449 power I/O

http://en.wikipedia.org/wiki/POWER6

http://www-03.ibm.com/press/us/en/pressrelease/21580.wss

# IBM 65nm SOI Technology




**Gate oxide: 1.05 nm ~ 5 atom layers in Si (!!)**

# Uncertain Delay

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

  **Batch to batch** — **$V_t$ threshold voltage**

  **Wafer to wafer** — **k' transconductance**

  **Die to die** — **W, L dimensions**
- **Supply Variations**

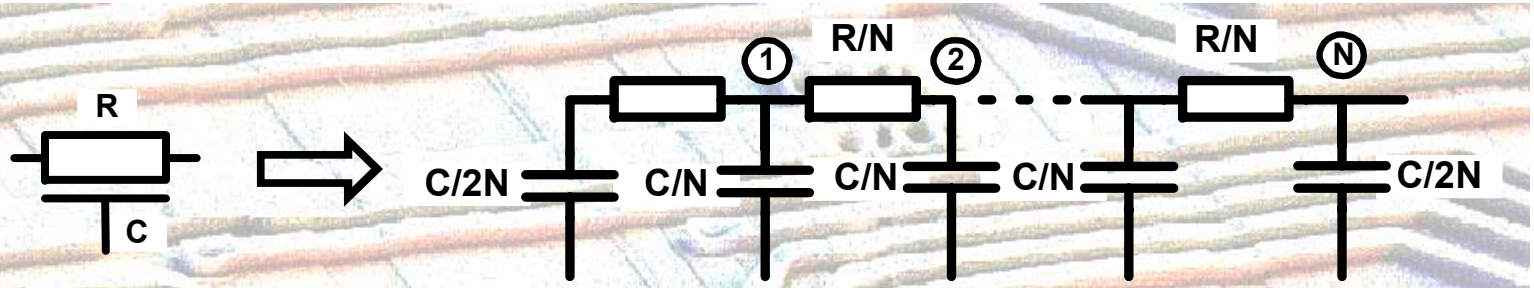
  **IR drop, dI/dt drop, ringing,**
- **Interconnect Delay**

  **Don't know length of line during logic design**

  **Delay at begin of line smaller than at end**

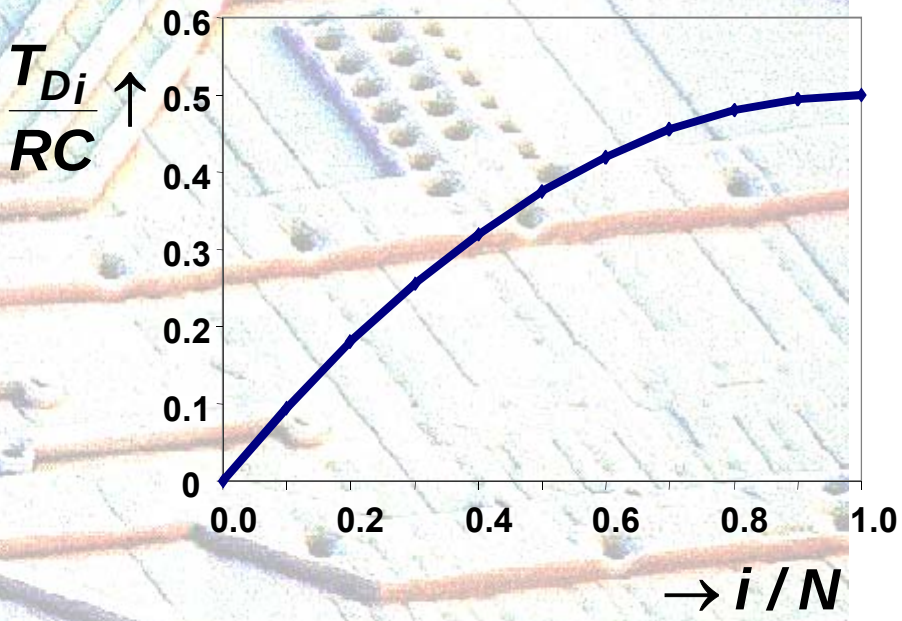
  **Interconnect parameter variability**

# Delay Along a Wire (Module 3)



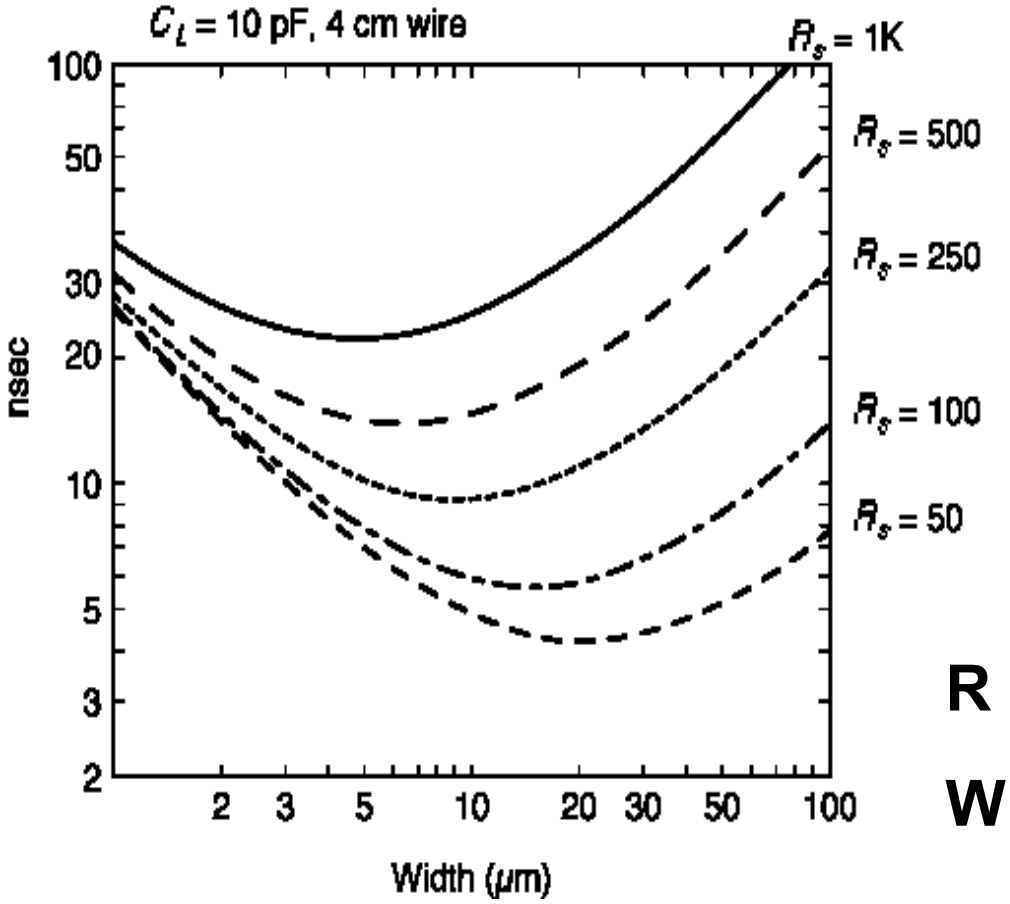

- Signal on RC line exhibits finite propagation speed
- Delay at beginning of line smaller than at end
- Follows from solving diffusion equation
- Elmore delay model also predicts propagation behavior (only approximately)

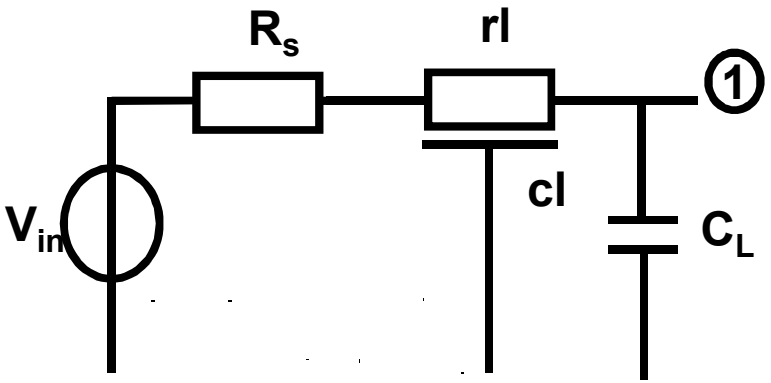
$$T_{Di} = \sum_{k=1}^{N} R_{ik} C_k$$




- **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/μm²**

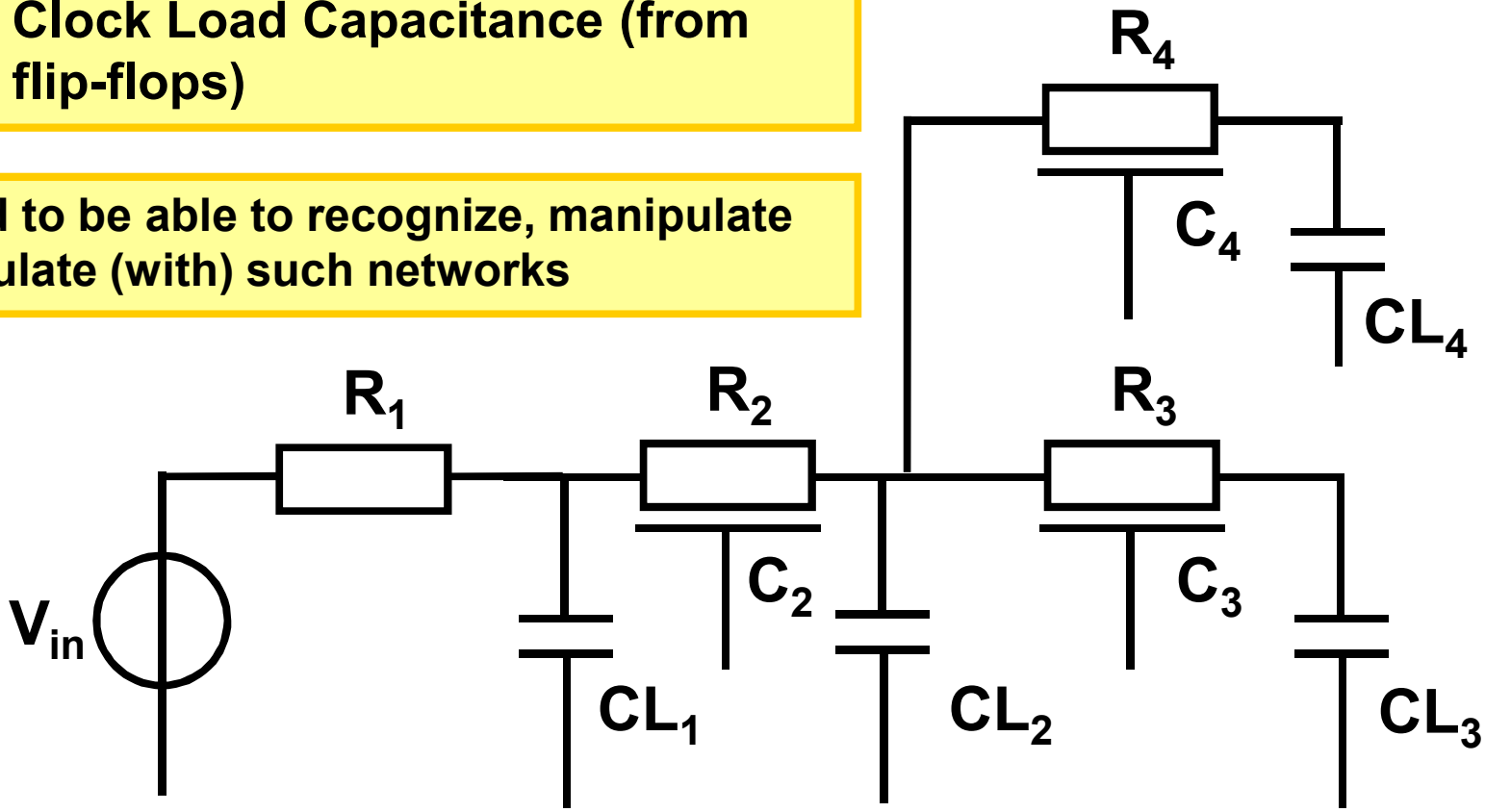
**W = Width, r = R/W, c = C*W**

**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**




# 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

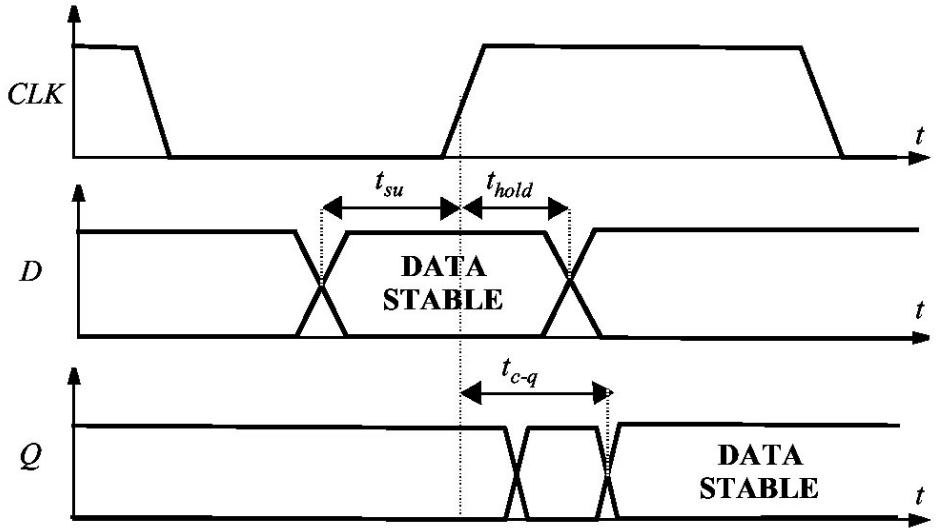
- Sequential circuits require ordered computation
- Several ways for imposing ordering
  - ✓ Synchronous (clock)
  - ✗ Asynchronous (unstructured)
  - ✗ Self-timed (negotiation)

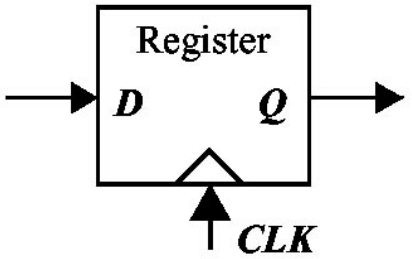
- Clock works like an orchestra conductor

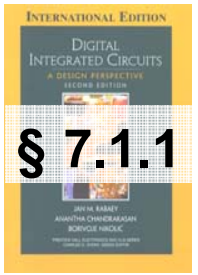
# Synchronous Design

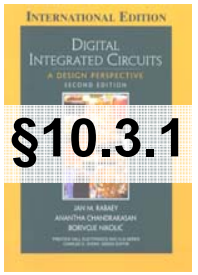
- Global Clock Signal
- Synchronicity may be defeated by
  - Delay uncertainty in clock signal
  - Relative timing errors: clock skew
  - Slow logic paths
  - Fast logic paths

# Timing Metrics Reminder




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

**$t_{su}$ : setup time**

**$t_{hold}$ : hold time**

**$t_{plogic}$ : worst case propagation delay of logic**

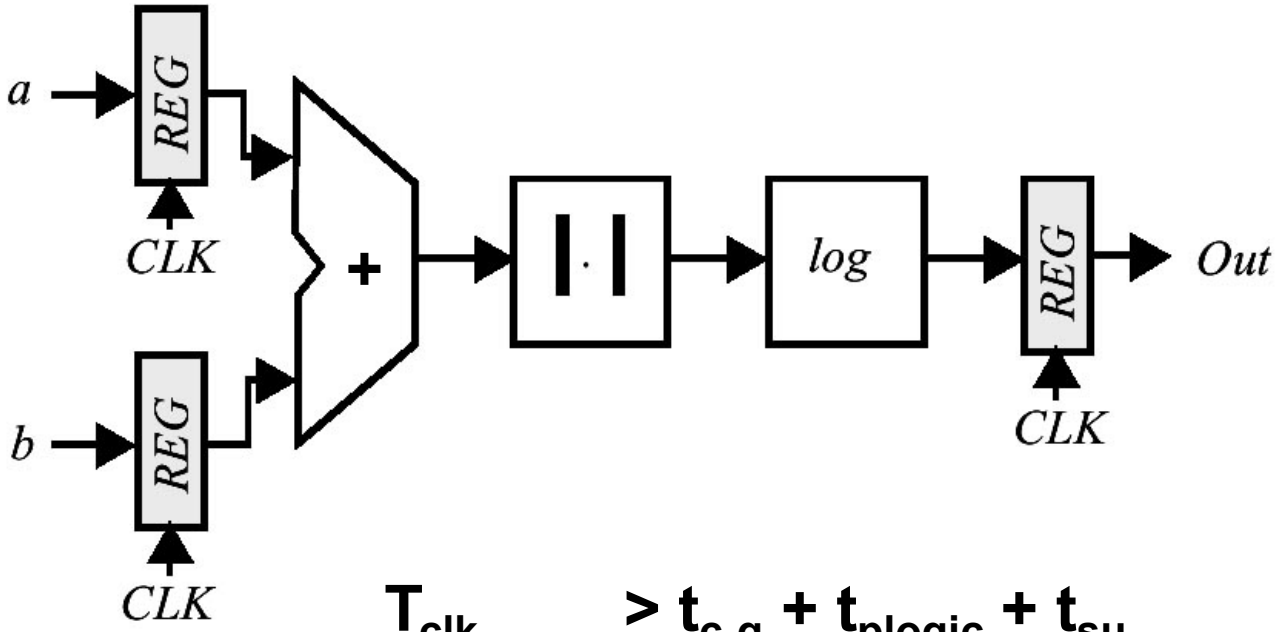
**$t_{cd}$ : best case propagation delay (contamination delay)**

**T : clock period**

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

$$t_{cdregister} + t_{cdlogic} \geq t_{hold}$$

# Sequential Circuit Timing.

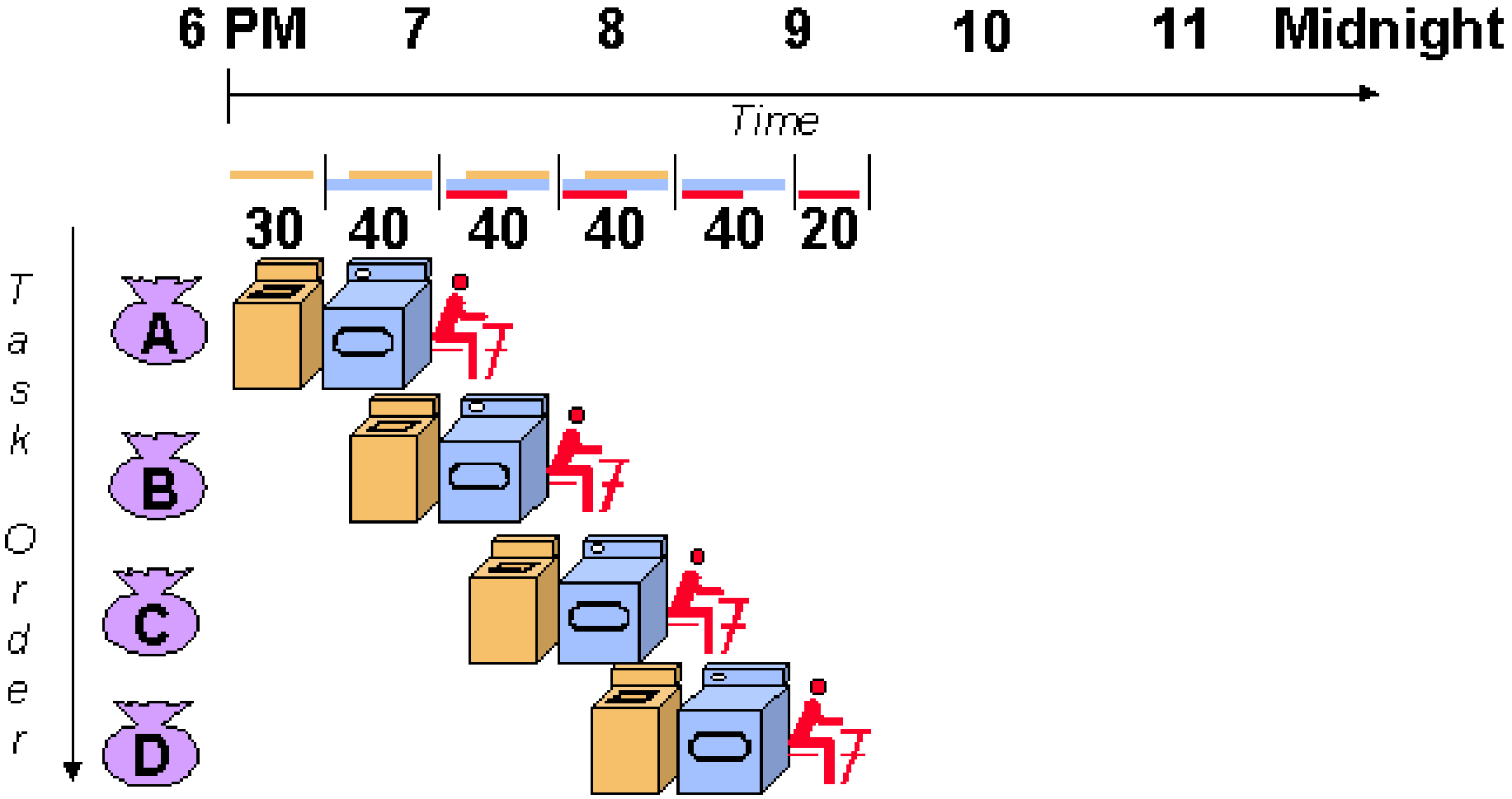



$$T_{clk} > t_{c\text{-}q} + t_{plogic} + t_{su}$$

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

**How to reduce $T_{clk}$?**

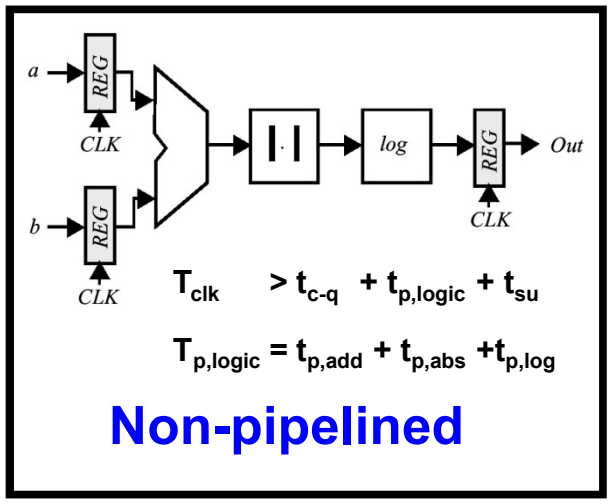
# Pipelined Laundry System

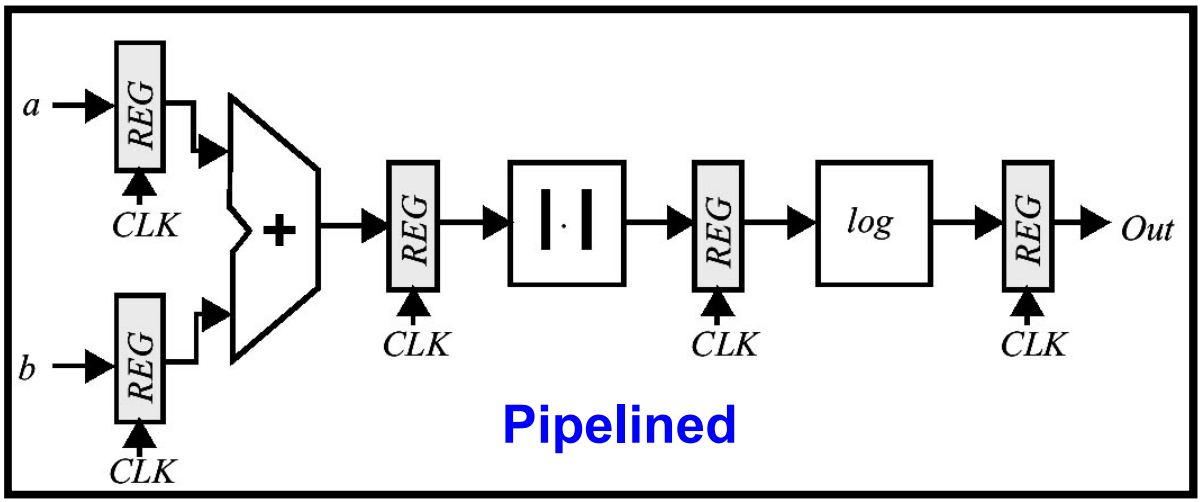



**Also: http://en.wikipedia.org/wiki/Pipelining**

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




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


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

- Improve resource utilization
- Increase functional throughput

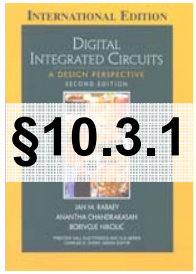
# Pipelining Observations.

- 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
- But all is dictated by GHz marketing drive
  - All a customer asks is: "How many GHz?"
  - Or says: "Mine is … GHz!"

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

# The Clock Skew Problem

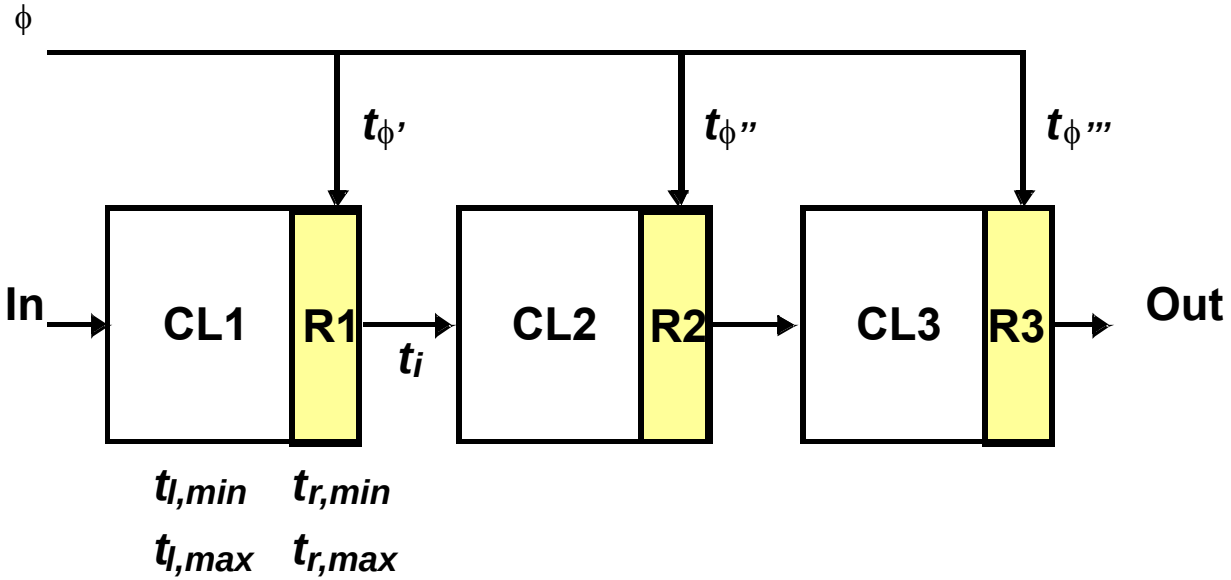
- In Single Phase Edge Triggered Clocking




**Brown**
**§7.15**
**§10.3.1**

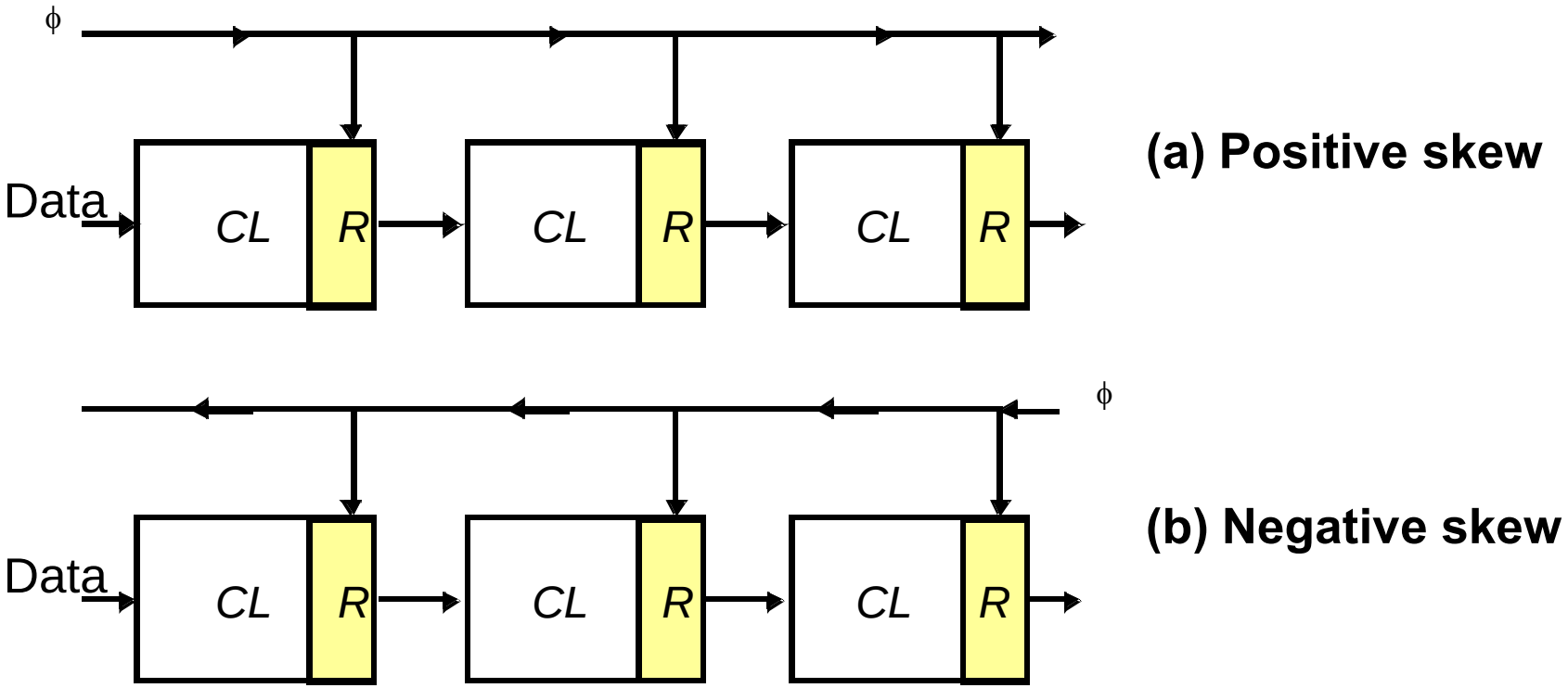
# The Clock Skew Problem

**Clock Rates >> 1 Ghz in CMOS**

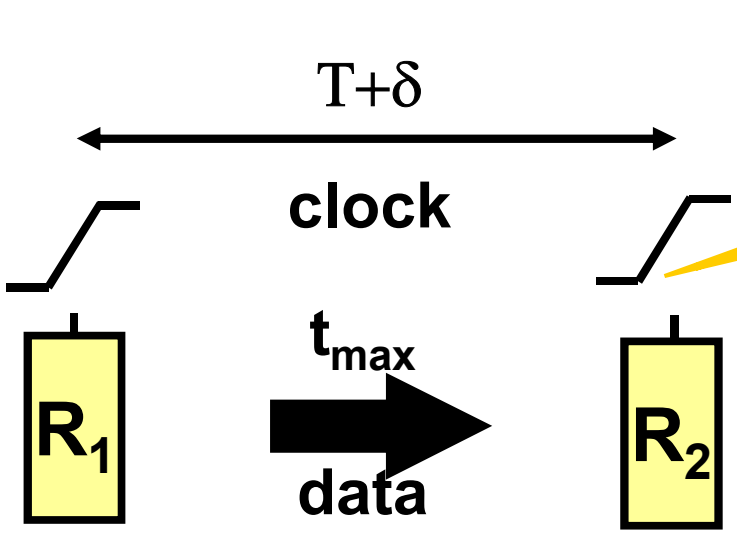



- **Clock Edge Timing Depends upon Position**
  - **Because clock network forms distributed RC line with lumped load capacitances at multiple sites (see earlier slide)**
- **(Relative) Clock Skew $\delta = t_{\phi''} - t_{\phi'}$**
- **Clock skew can take significant portion of $T_{clk}$**

# Positive and Negative Skew



# Edge-Triggered Slow Path Skew Constraint




One clock tick later

Data at $R_2$ should be stable before clock pulse is applied

**Timing constraint** ($t_i$ = interconnect delay)

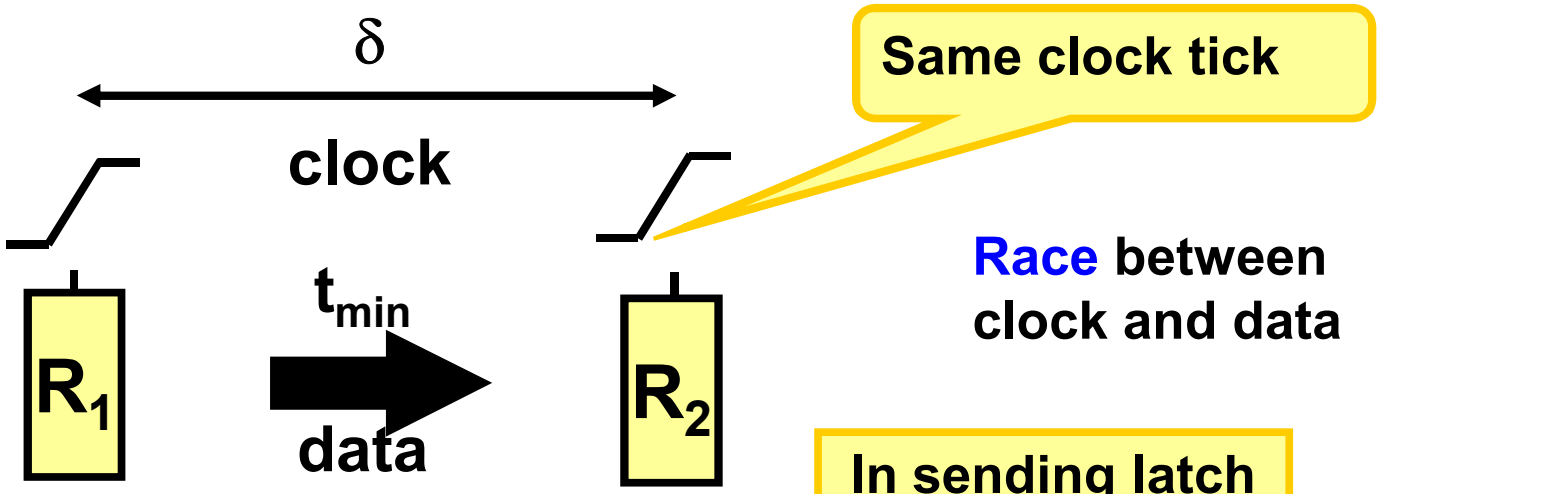
$$T + \delta \geq t_{max} = t_{p,logic} + t_i + t_{c\text{-}q,max} + t_{su}$$

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




**Timing constraint**

$$\delta \leq t_{min} = t_{cd,logic} + t_i + t_{c\text{-}q,min} - t_{hold}$$

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

# Clock Constraints in Edge-Triggered Logic.

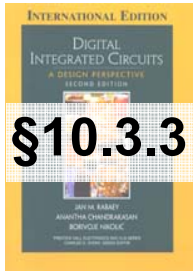
$$T \geq t_{max} - \delta$$

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

- **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**

# Controlling Clock Skew

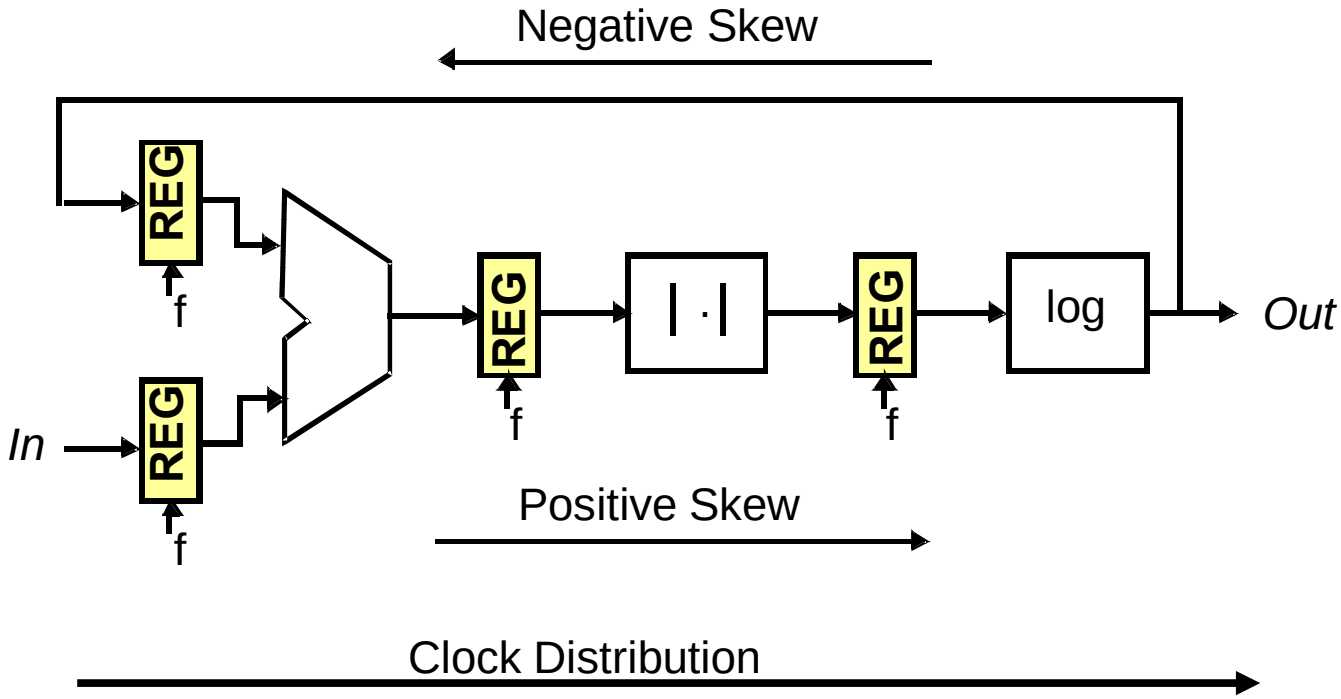
# Case Study



# 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



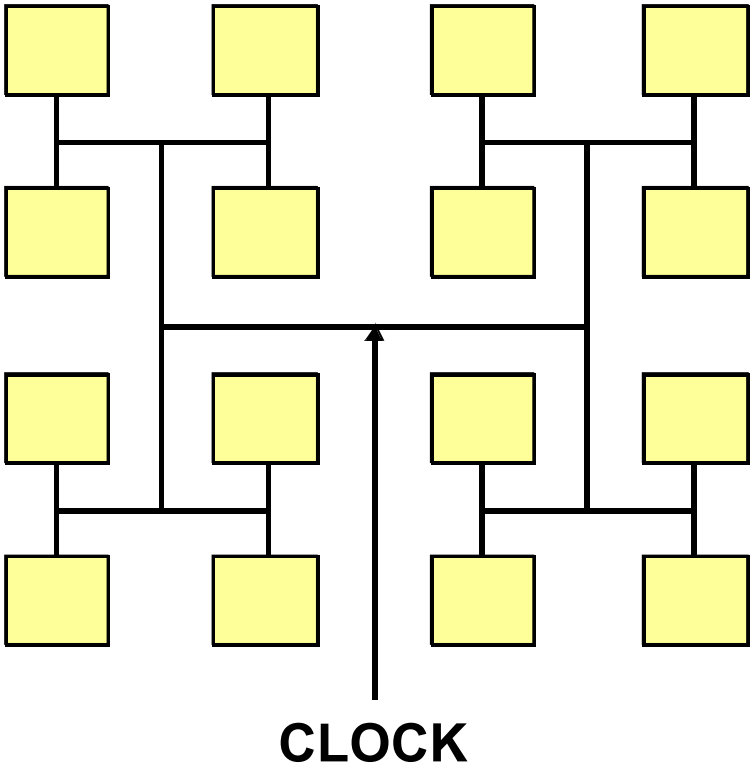

- **Data and Clock Routing**
- **Cannot unambiguously route clock in opposite direction of data**
- **Need bounded skew**

# 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**
  - **…**

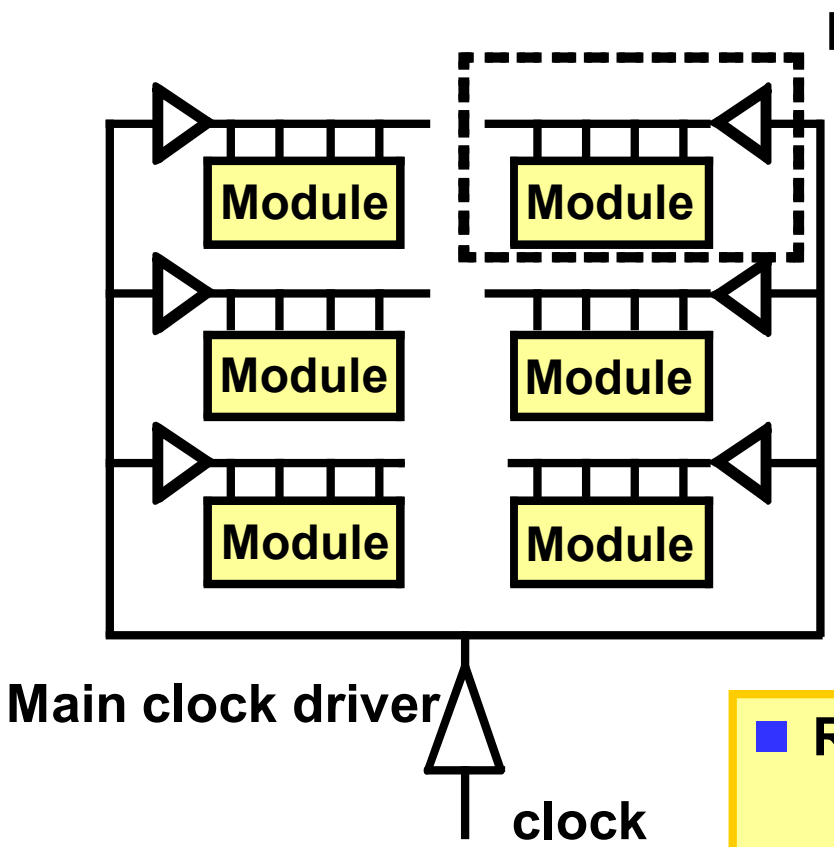
# H-tree Clock Network




- 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

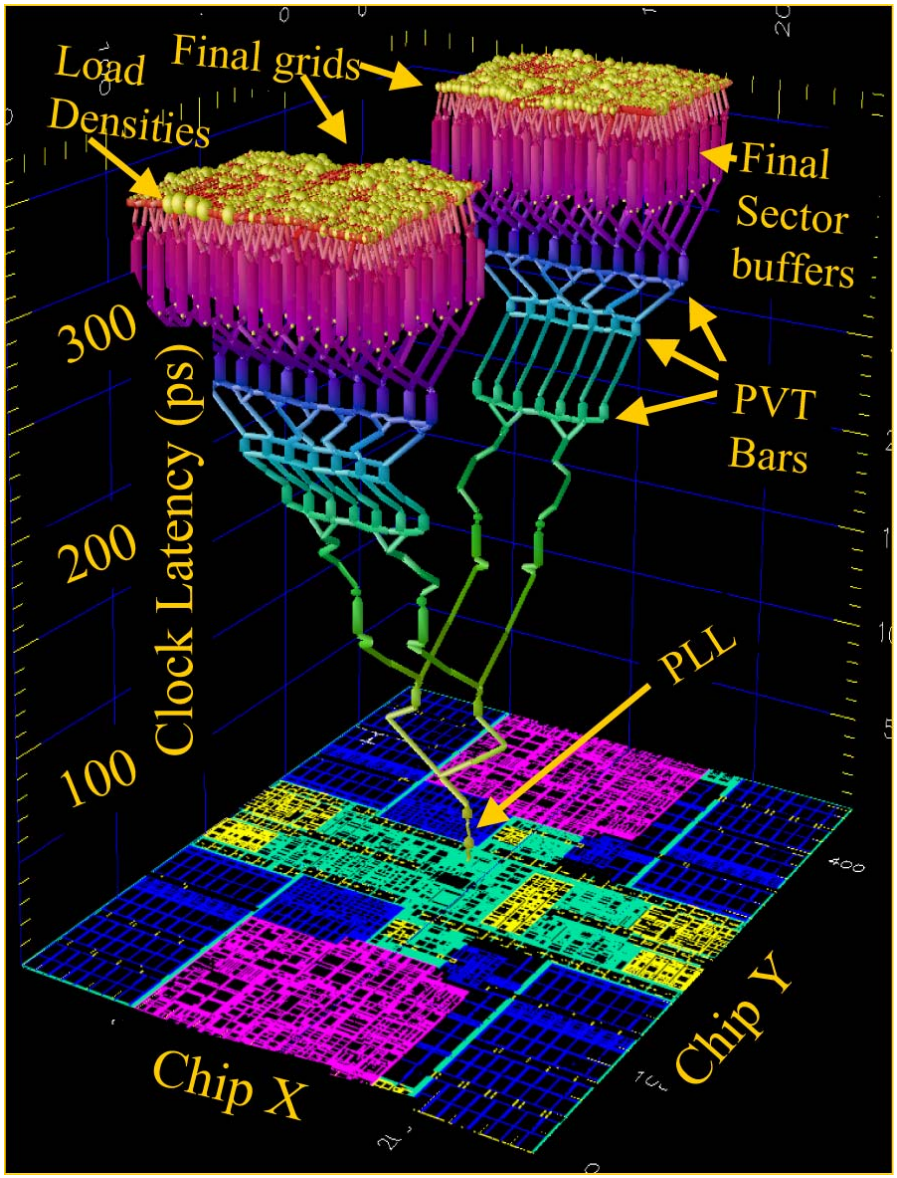
**Observe: Only Relative Skew Is Important**

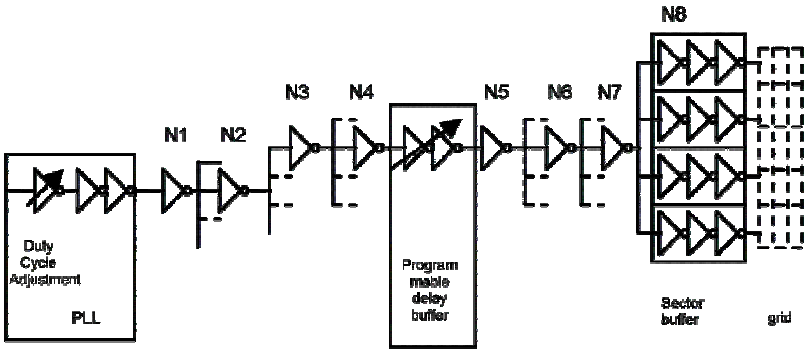
# Clock Network with Distributed Buffering




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

# Power6 Clock Distribution

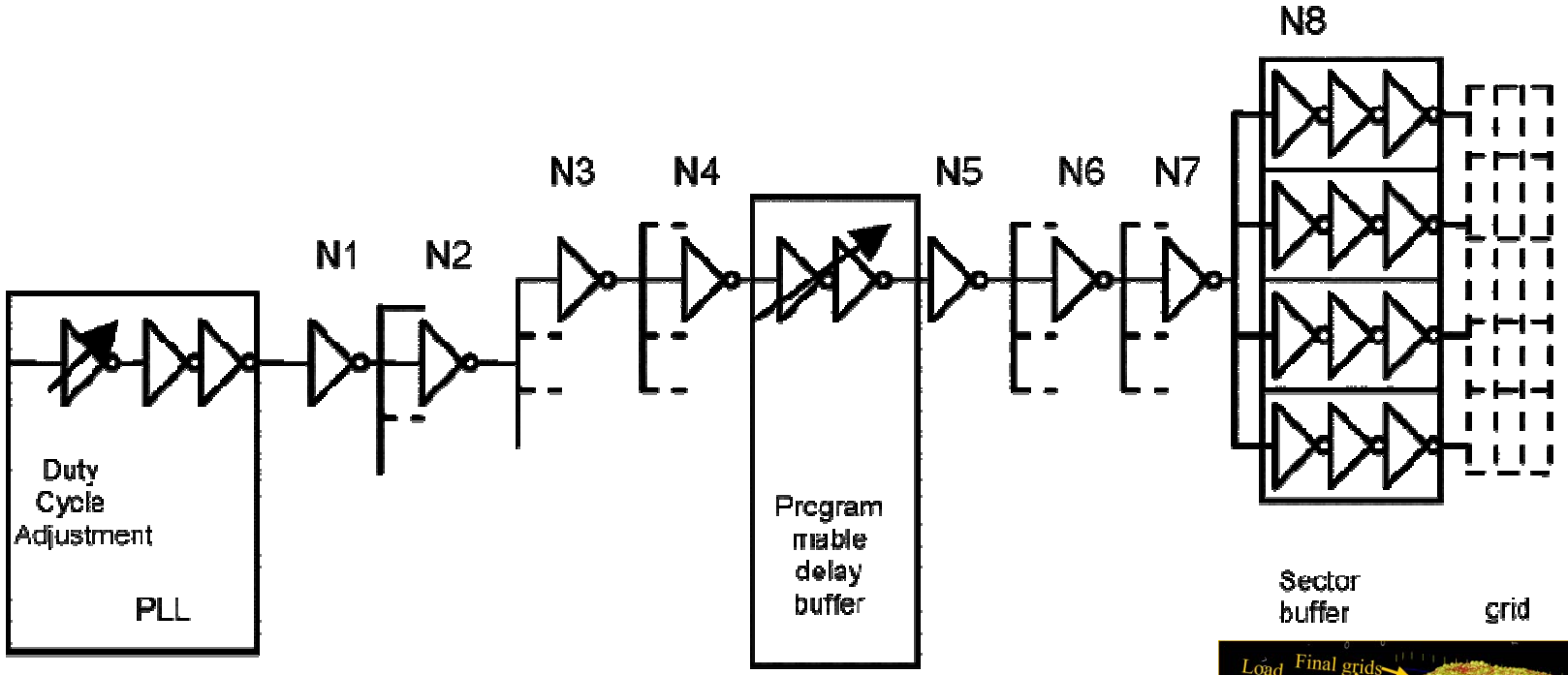


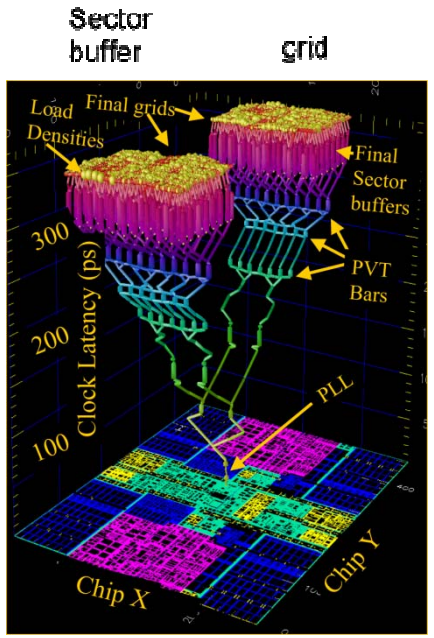




- **Latency ~ cycle time**

**friedrich, isscc 2007**

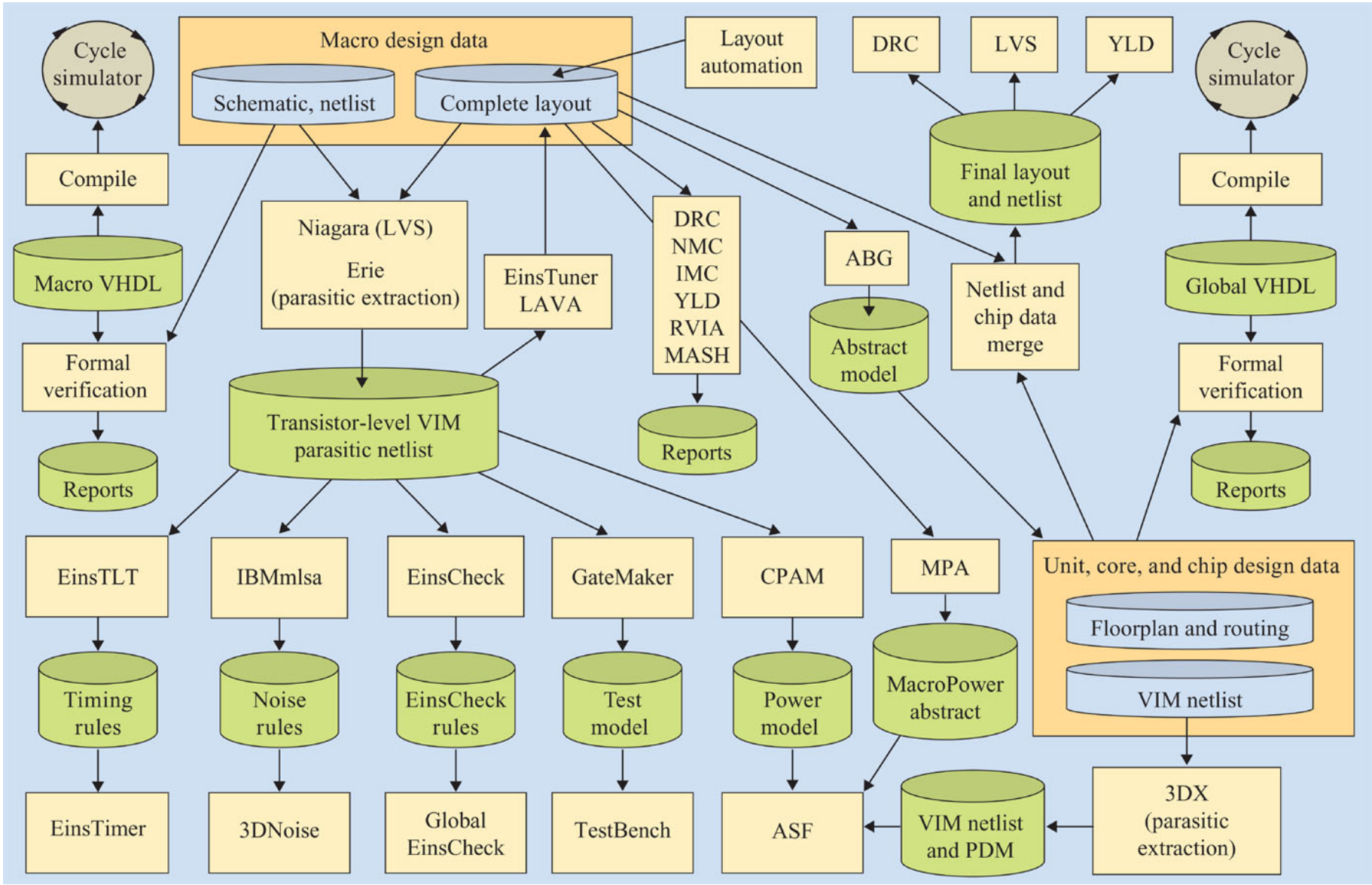
# Power6 Clock Distribution




**stolt, jssc 2008**

# 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**

# Summary

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

**Got basic appreciation of some system level design issues?**