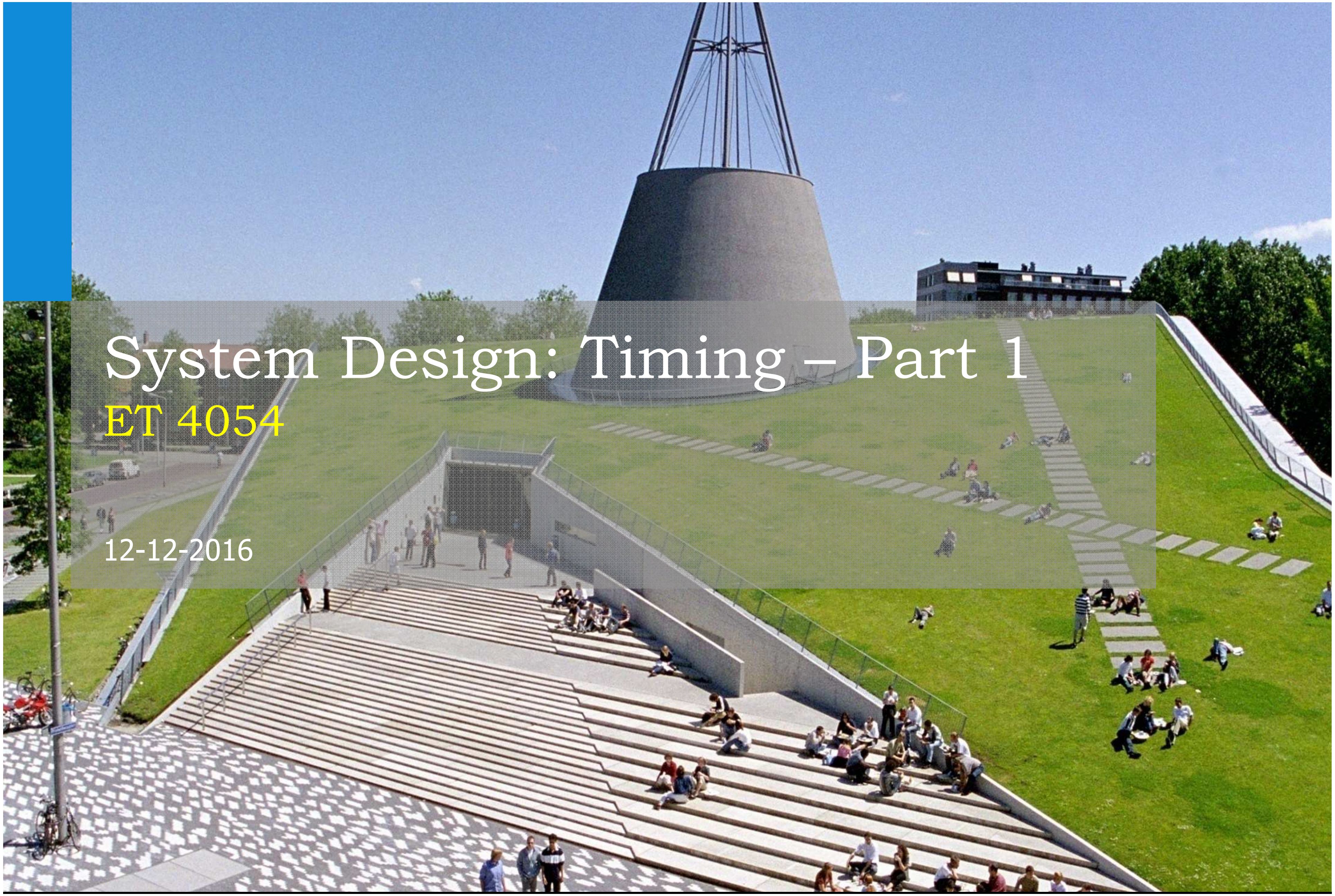


# System Design: Timing – Part 1

ET 4054

12-12-2016

TU Delft Delft University of Technology

Challenge the future

# Amir Zjajo, PhD

Research focus:

- **Biomedical Interfaces/SoC**
  - From sensors to sense-making: Signal acquisition, conditioning, quantization, detection, classification
- **Neuromorphic Cognitive Systems** (Hardware)
  - Brain-like systems with adaptation, self-organization, and learning
- Currently (co-)supervising 12 MSc students

# Amir Zjajo, PhD

Contact:

- http://cas.et.tudelft.nl/~zjajo
- **Office:** Circuits and Systems Group
  EWI 17.270
- **E-mail:** a.zjajo@tudelft.nl

# Acknowledgement

- Michel Berkelaar

# Overview

- Design Constraints
    - Power, Area, Frequency, CMOS Scaling
- Timing
    - Timing Metrics, Paths, Variability and Delay
- Deterministic Timing Analysis (Static Timing Analysis)
    - Models, Interconnect, Networks-on-Chip, Clock Distribution
- Statistical Timing Analysis
    - Probability, Spatial Correlations, MAX function
- Design Flow
    - Synthesis, Transformation, Definitions, Constrains

# Design Constraints

# After all of the high level synthesis, what do we have?

- a set of operation units to implement (*, +, <, etc)
- some memory to implement (registers, ...)
- a controller to implement
- connections between all of these, with multiplexers, branch logic, etc.
- a vague idea of time (cycles)

# Questions

- What objectives does a designer have?
- What causes delay on an IC?

# Where do we need to go?

- Design of a real chip / block of a chip, ready for fabrication
- ... optimized as best as you can
  - Power (= battery life and heat)
  - Area (= cost and yield)
  - Clock frequency (constraint? higher is better?)
- ... functioning with the rest of the IC / PCB
- ... with all the nasty details sorted out (reset, test, power distribution, clock distribution, EMC, IO standards, etc)

# How do we get there?

- There is good software to help us
- But we always need to help it by specifying what we want, especially by providing timing constraints!
- Unless we know what we are doing, the design
  - may (will) not work
  - may (will) use more power / area than necessary

# CMOS Power

- Probably your most important design parameter!

# CMOS Power

- CMOS has relatively low static dissipation
- Power dissipation was the reason that CMOS technology won over bipolar and NMOS technology for digital IC's
- (Extremely) high clock frequencies increase dynamic dissipation
- Low $V_T$ increases leakage
- Advanced IC design is a continuous struggle to contain the power requirements!

# CMOS Power

## Estimate

| | | |
|---|---|---|
| ■ Furnace: | 2000 W, r=10 cm | ➔ P ≈ 6 W/cm$^2$ |
| ■ Processor chip: | 100 W, 3 cm$^2$ | ➔ P ≈ 33 W/cm$^2$ |
| ■ Human brain: | 20 W, ~1.3 dm$^3$ | ➔ P ≈ 0.015 W/cm$^3$ |

# CMOS Power

- Dynamic Power Consumption
  Charging and discharging capacitors
- Short Circuit Currents
  Short circuit path between supply rails during switching (NMOS and PMOS on together)
- Leakage
  Leaking diodes and transistors
  Important for battery-operated equipment

# CMOS Power

$$P \sim \alpha \cdot (C_L + C_{CS}) \cdot V_{swing} \cdot V_{DD} \cdot f + (I_{DC} + I_{Leak}) \cdot V_{DD}$$

α – switching activity

- $C_L$ – load capacitance
- $C_{CS}$ – short-circuit capacitance
- $V_{swing}$ – voltage swing
- $f$ – frequency

- $I_{DC}$ – static current
- $I_{leak}$ – leakage current

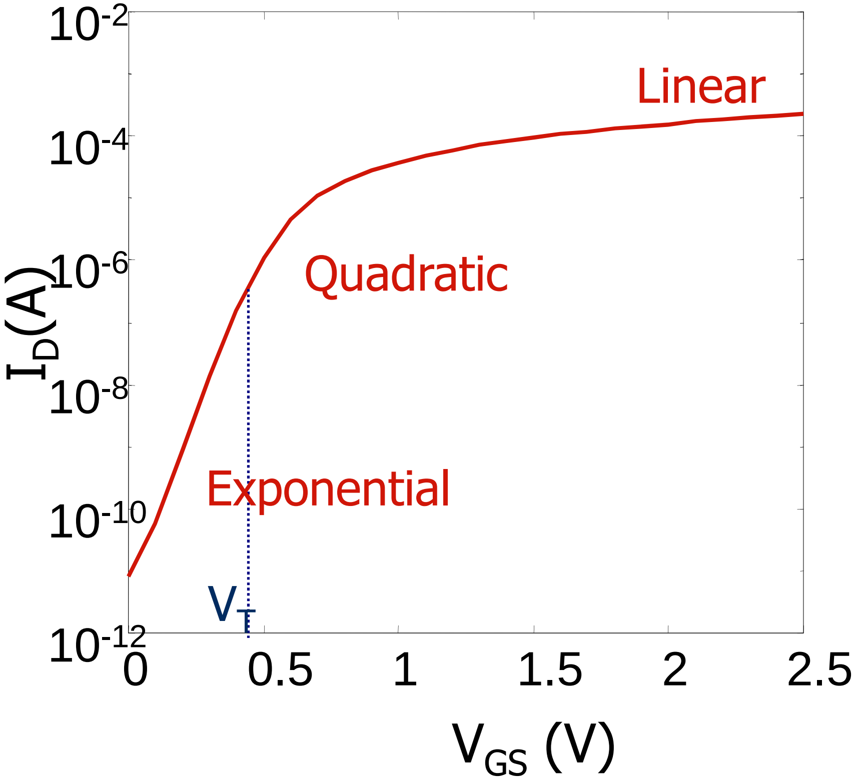
$$P = \frac{energy}{operation} \times rate + static\ \ power$$

# CMOS Leakage Power

- Sub-threshold current of MOS devices

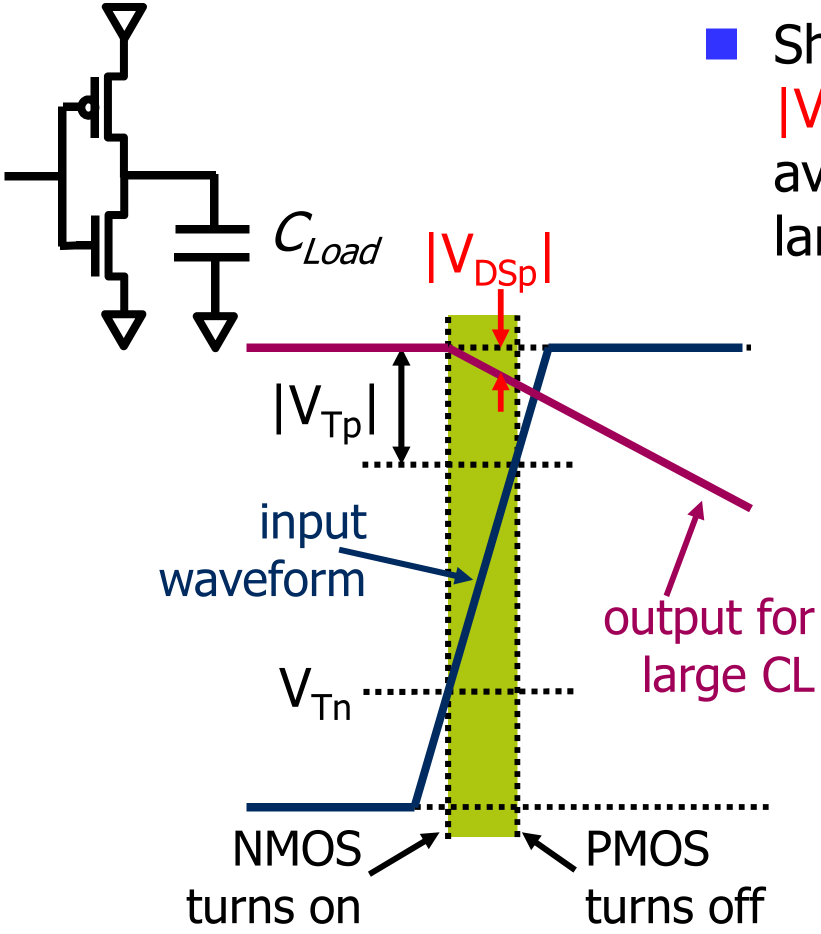
$$I_{D(sub-threshold)} \approx K_1 W e^{-V_T/nV_{thermal}} (1 - e^{-V_{DD}/V_{thermal}})$$

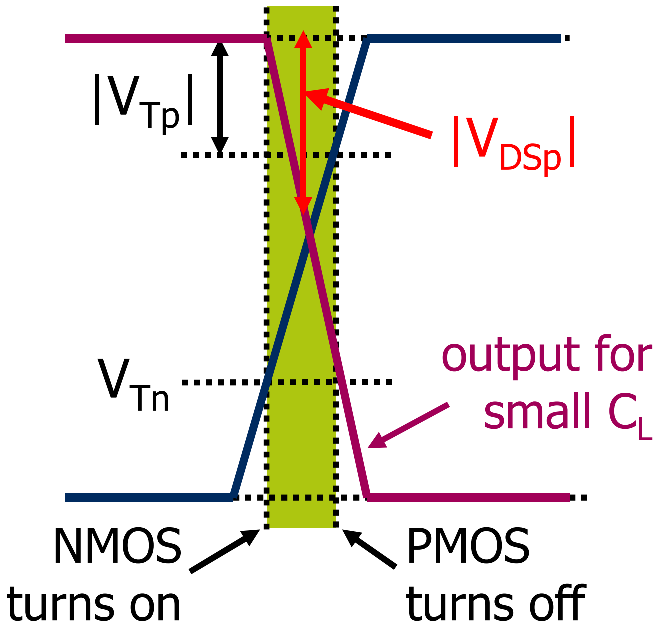
- No channel ➔ parasitic bipolar device: n+ (source) – p (bulk) – n+ (drain)



# CMOS Short-Circuit Power

- Short-circuit current increases with $|V_{DSp}|$. This is clearly much larger on average for small $C_L$ compared to large $C_L$.





Best to maintain approximately equal input/output slopes

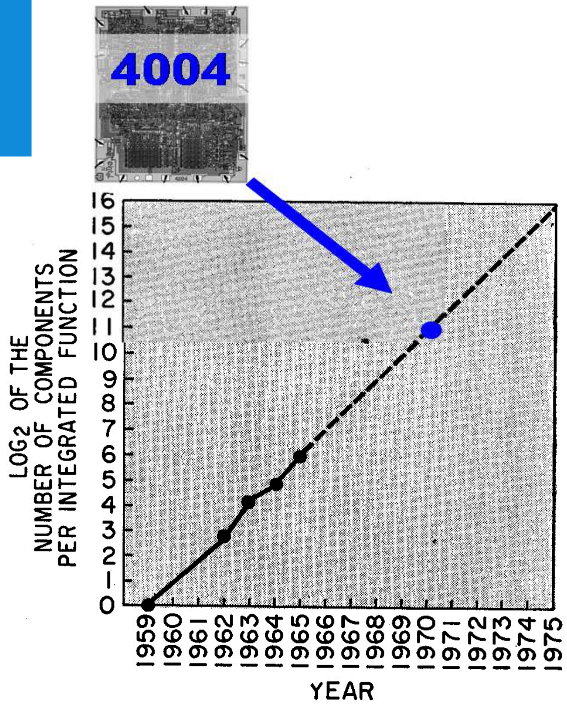
# Area

- Bigger means more expensive, more chance of defects during production
- $$$

# Clock frequency

- Delay in a switch/wire dictated by physics on the chip
- Big transistors: small R, high power
- Big transistors: large C
- More modern technology -> better transistors, more resistive wires (but shorter distances)!

# CMOS Scaling



## Moore’s Law

The number of transistors that can be integrated on a single chip will double every 12 months (later adjusted to 18 months)

Gordon Moore, co-founder of Intel [Electronics, vol. 38, no. 8, 1965]

# CMOS Scaling

- Reduce price per function:
  - Want to sell more functions (transistors) per chip for the same money ➔ better products
  - Build same products cheaper, sell the same part for less money ➔ larger market
  - Price of a transistor has to be reduced
- But also want to be faster, smaller, lower power

# CMOS Scaling

- Fixed Voltage Scaling
- most common model until 1990’s
- only dimensions scale, voltages remain constant

- Full Scaling (Constant Electrical Field)
- ideal model - dimensions and voltage scale together by the same factor S

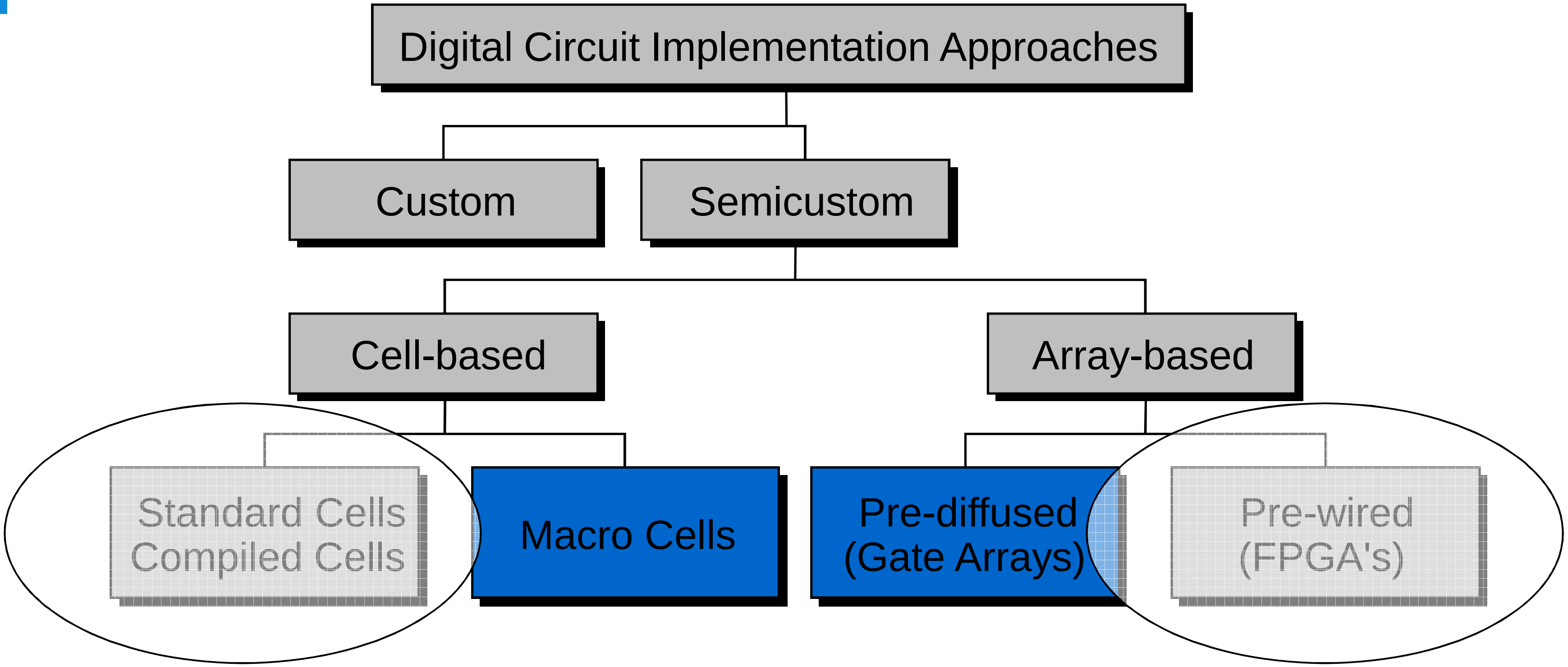
- General Scaling
- most realistic for today’s situation - voltages and dimensions scale with different factors

# CMOS Scaling

Constant Field Scaling: S = U

| Parameter | Relation | General Scaling |
|---|---|---|
| $W, L, t_{ox}$ | | $1/S$ |
| $V_{DD}, V_T$ | | $1/U$ |
| *Area / Device* | $WL$ | $1/S^2$ |
| $C_{ox}$ | $1/t_{ox}$ | $S$ |
| $C_{gate}$ | $C_{ox}\ W\ L$ | $1/S$ |
| $I_{sat}$ | $C_{ox}\ W\ V$ | $1/U$ |
| *Current Density* | $I_{sat}$ / *Area* | $S^2/U$ |
| $R_{on}$ | $V / I_{sat}$ | $1$ |
| *Intrinsic Delay* | $R_{on}\ C_{gate}$ | $1/S$ |
| *Power / Device* | $I_{sat}\ V$ | $1/U^2$ |
| *Power Density* | *P / Area* | $S^2/U^2$ |

# Design Style Choice



# ASIC Design Advantages

- Cost: lower unit costs
- Speed: ASICs are faster than FPGAs
- Power: ASICs consume less power
- Complexity: bigger designs can fit
- Can add analog / mixed circuit blocks

# ASIC Disadvantages

- Time-to-market: some large ASICs can take a year or more to design
- Design Issues: all the dirty details (Floorplan, Signal Integrity, power/clock distribution, EMC, DFT, etc)
- Expensive Tools: ASIC design tools are very expensive
- A design bug means re-fabrication...

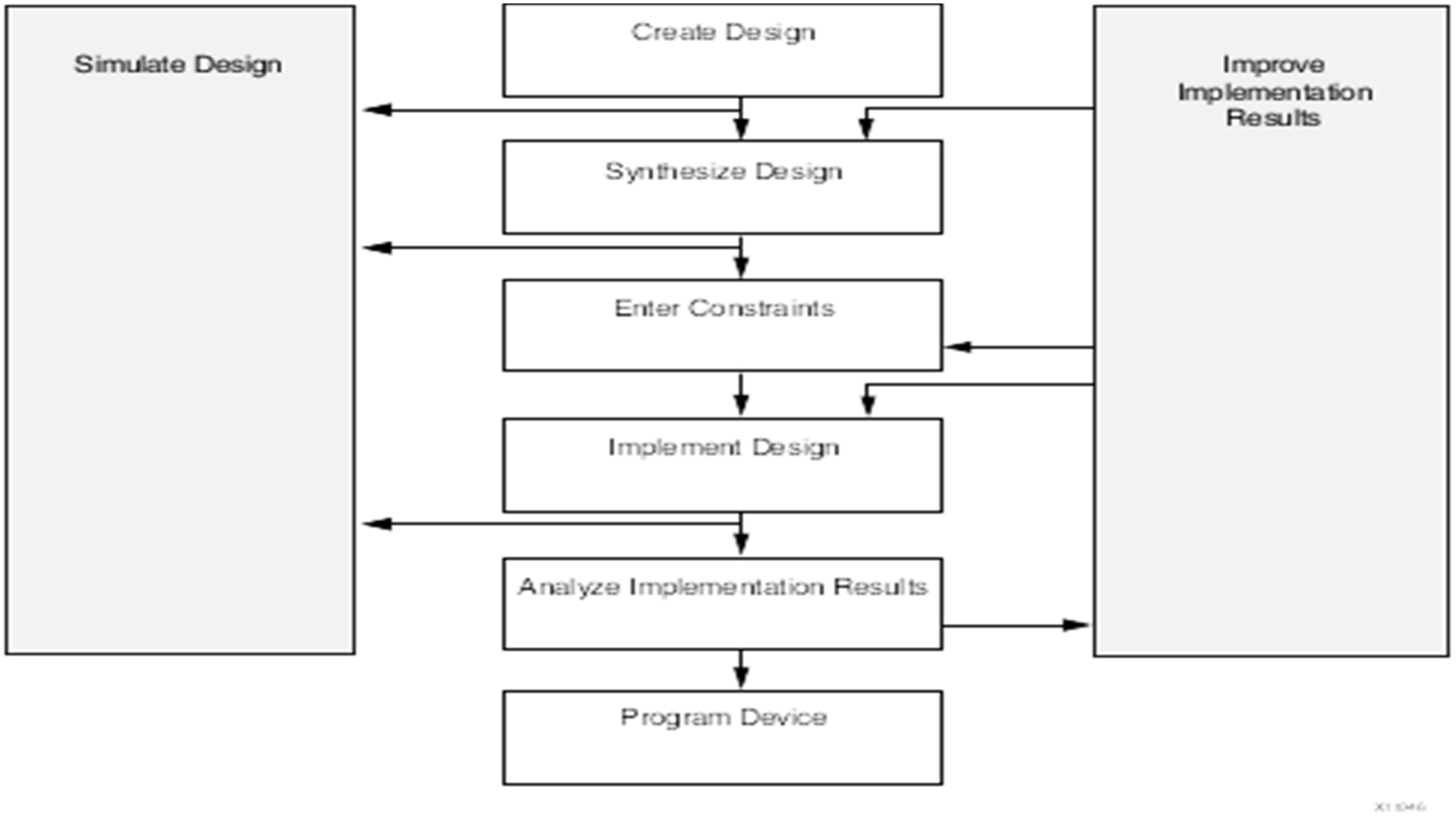
# FPGA Design Advantages

- Faster time-to-market: No layout, masks or other manufacturing steps are needed for FPGA design.
- No NRE (Non Recurring Expenses)
- Simpler design cycle
- Field Reprogrammable (bug fixes...)
- Reusable for other design
- FPGAs are good for prototyping and limited production
- Generally FPGAs are used for lower speed, lower complexity and lower volume designs

# FPGA Disadvantages

- Lower performance (10x)
- Higher power
- High cost / unit

# Xilinx FPGA Tool Flow



# Tool flow details

- Much of the tool flow is automatic
- But: timing constraints need to be specified!
- To be able to do this, you need to understand timing!
  - sources of delay
  - how it is measured / estimated
  - how your constraints impact the tool outcome

# Questions

- How do you know your RTL design is correct?

# Questions

- RTL design is used in the **logic** design phase
- RTL description (usually) converted to a gate-level description by a logic synthesis tool
- The synthesis results are then used by placement and routing tools to create a physical layout

- Logic simulation tools may use a design's RTL description to verify its correctness

# Estimating timing: simulation

- Before synthesis, VHDL simulation is free from timing issues
- After synthesis, delay can be added to the simulation
- Simulation is never a proof: it only shows what happens for the few vectors you are simulating
- Even if simulation shows no problems, the chip may not work for other inputs!
- Simulating all transitions with N input bits requires $2^{2N}$ vectors!
- Internal memory (state) makes this (much) worse.

# Static Timing Analysis

- We need a method that guarantees that the chip will always work.
- We may need to allow some level of inaccuracy (pessimistic!) to make it computationally efficient.

# Overview

- Design Constraints
  - Power, Area, Frequency, CMOS Scaling

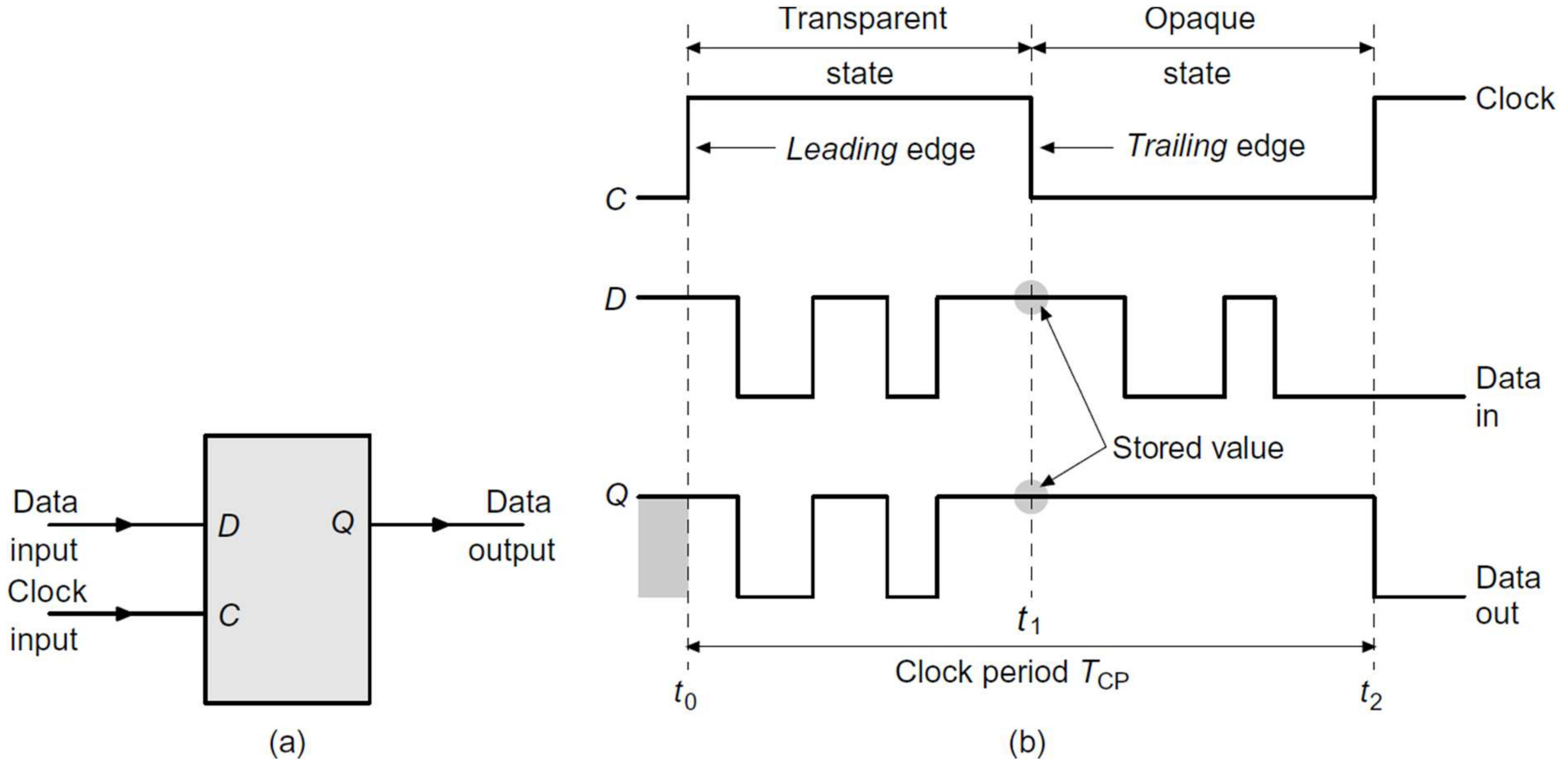
- Timing
  - Timing Metrics, Paths, Variability and Delay

- Deterministic Timing Analysis (Static Timing Analysis)
  - Models, Interconnect, Networks, Clock Distribution

- Statistical Timing Analysis
  - Probability, Spatial Correlations, MAX function

- Design Flow
  - Synthesis, Transformation, Definitions, Constrains

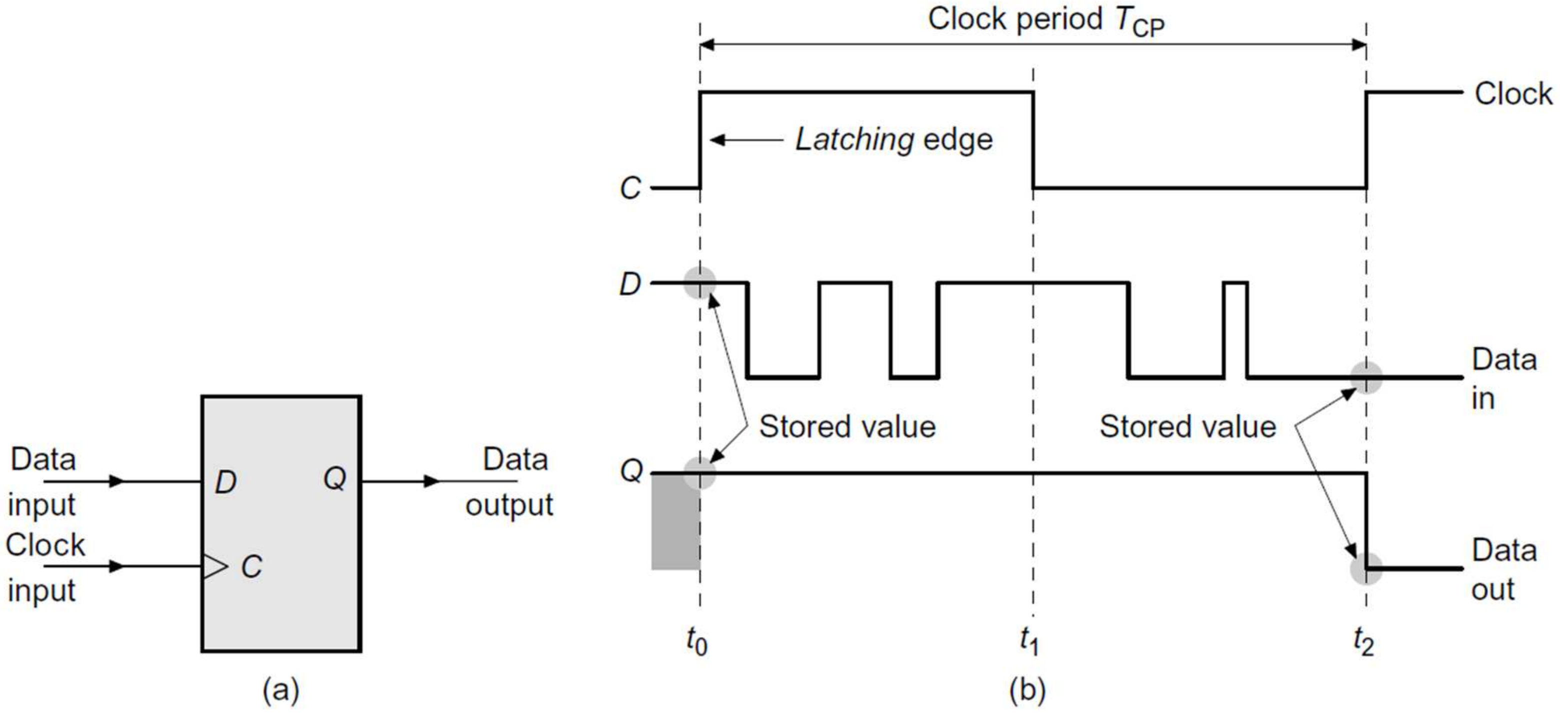
# D-Latch



[Taskin, Kourtev & Friedman, The VLSI Handbook]

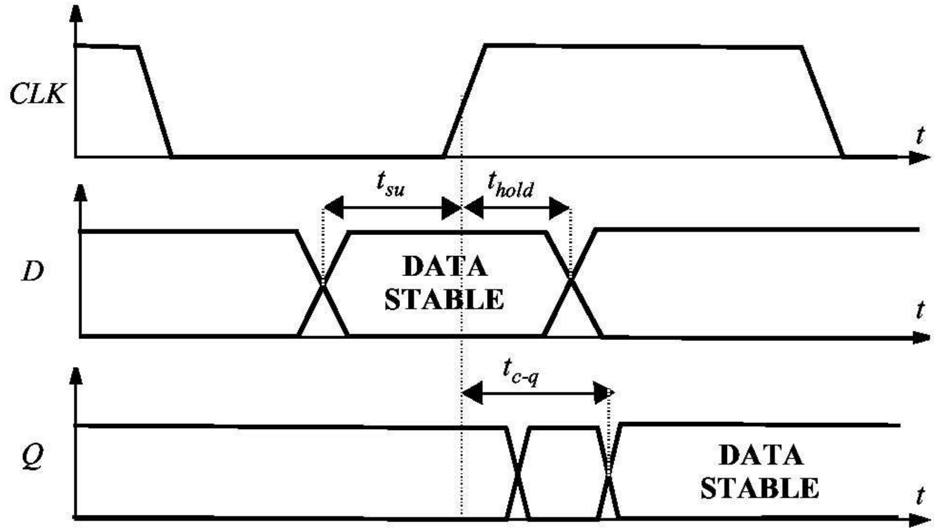
# D-Register

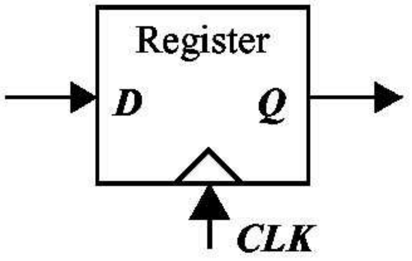
(more commonly known as D-Flip-Flop)



[Taskin, Kourtev & Friedman, The VLSI Handbook]

# Timing Metrics Reminder





$t_{su}$ : setup time

$t_{hold}$ : hold time

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

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

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

T : clock period

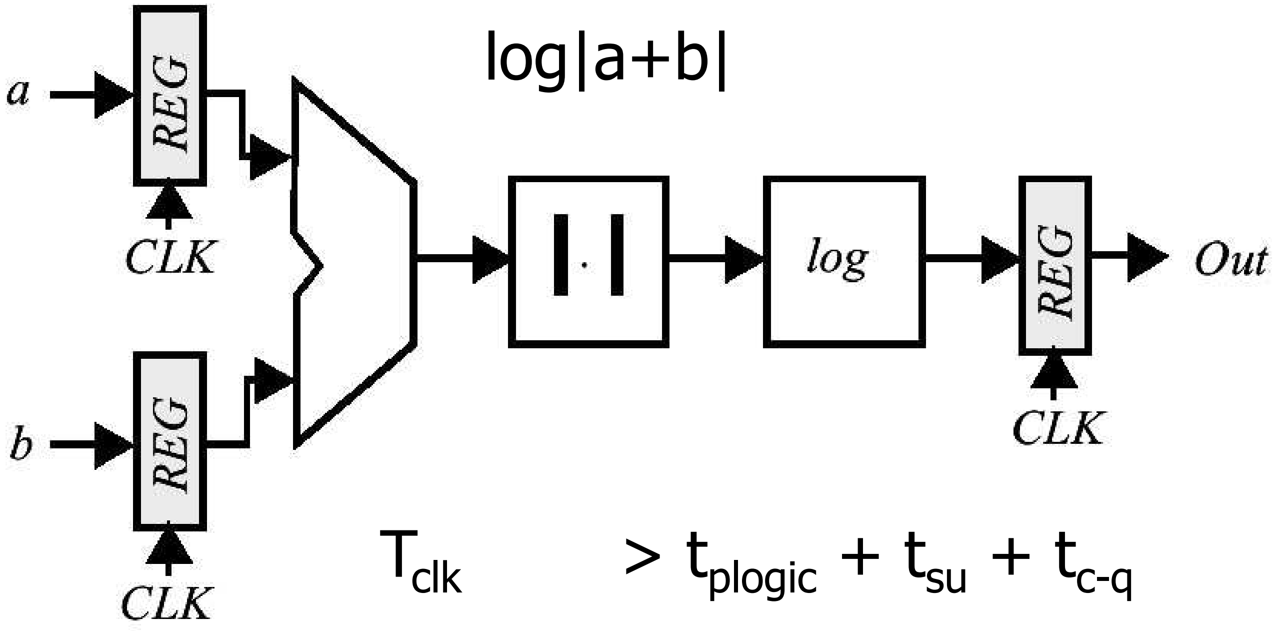
$$T \geq t_{plogic} + t_{su} + t_{c\text{-}q}$$

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

TUDelft

# Example: How to reduce clock period

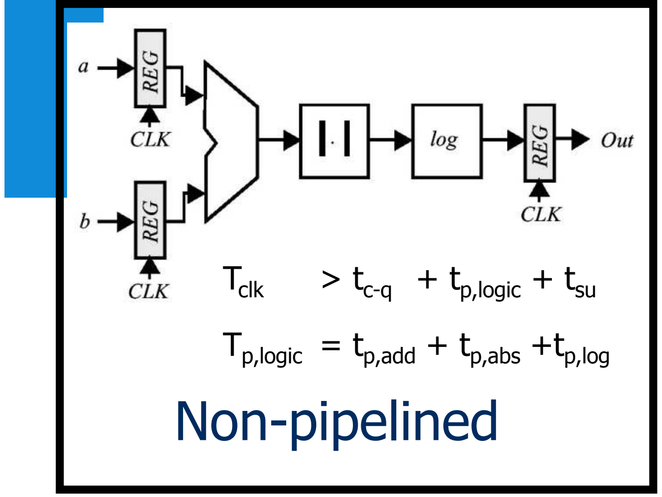
log|a+b|



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

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

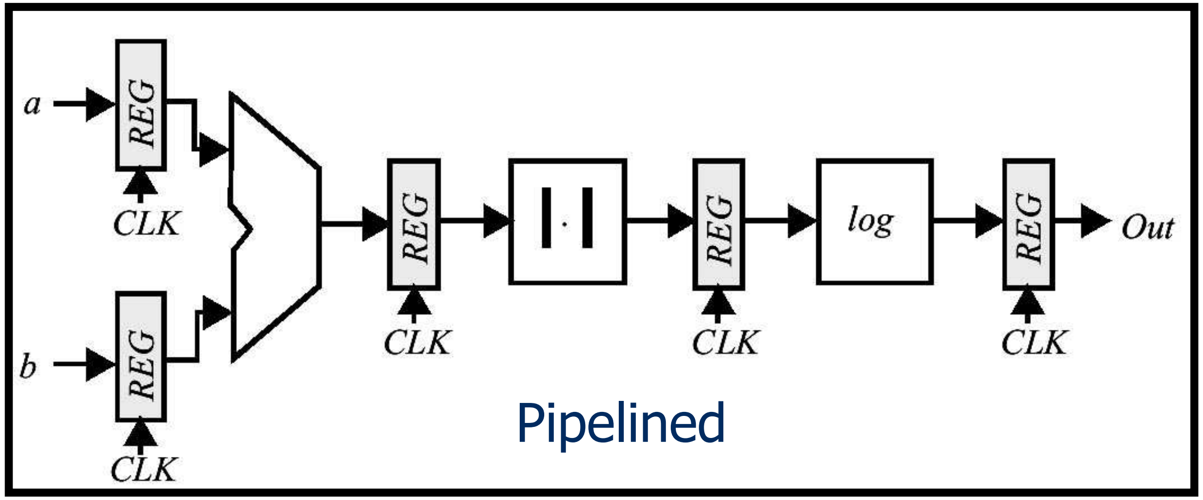
# Example: How to reduce clock period



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

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

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 | $a_5 + b_5$ | $\lvert a_4 + b_4 \rvert$ | $\log(\lvert a_3 + b_3 \rvert)$ |

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

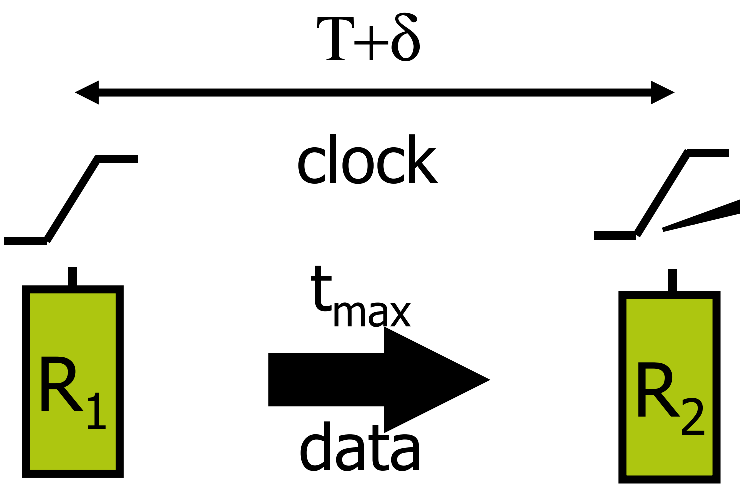
Increase functional throughput

# Example: How to reduce clock period

- Pipelining: 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

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

# 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)

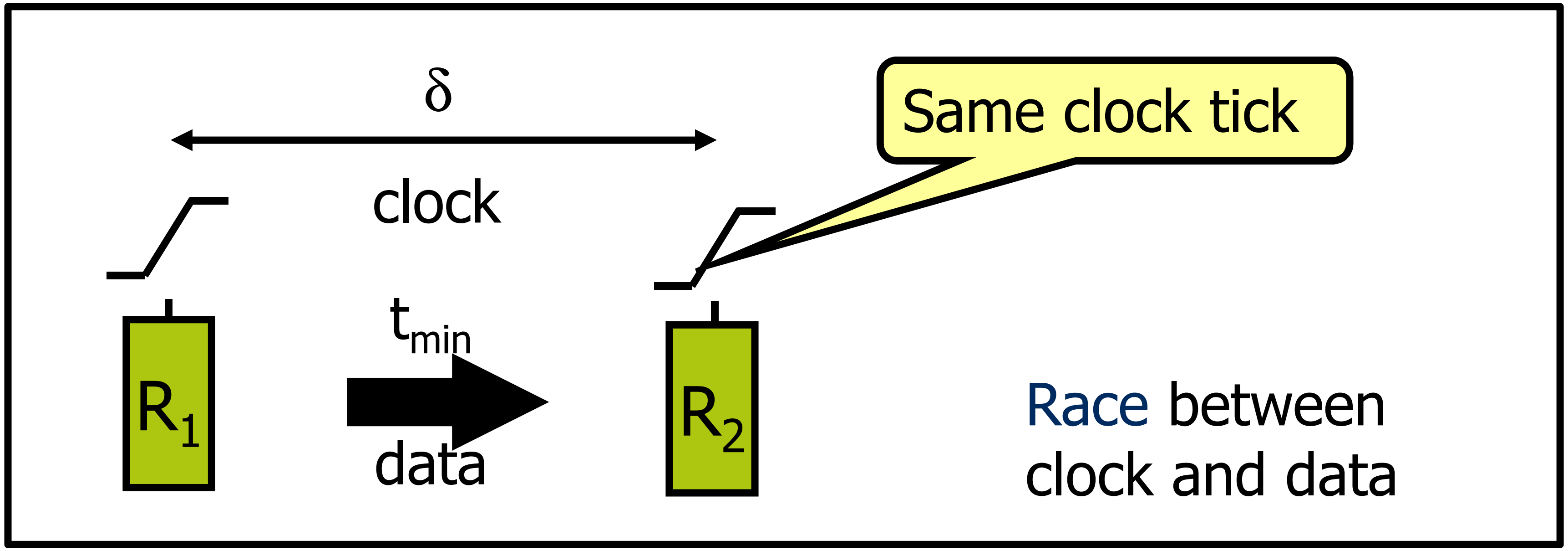
$T + \delta \geq t_{max} = t_{p,logic} + t_i + \boxed{t_{c\text{-}q,max} + t_{su}}$

Internal delay of flip-flop

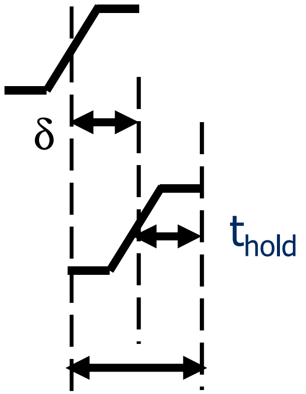
$T \geq t_{max} - \delta$

Minimum Clock Period Determined by Maximum Delay between Latches minus skew δ

# Fast Path Skew Constraint



Race between clock and data



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

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

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

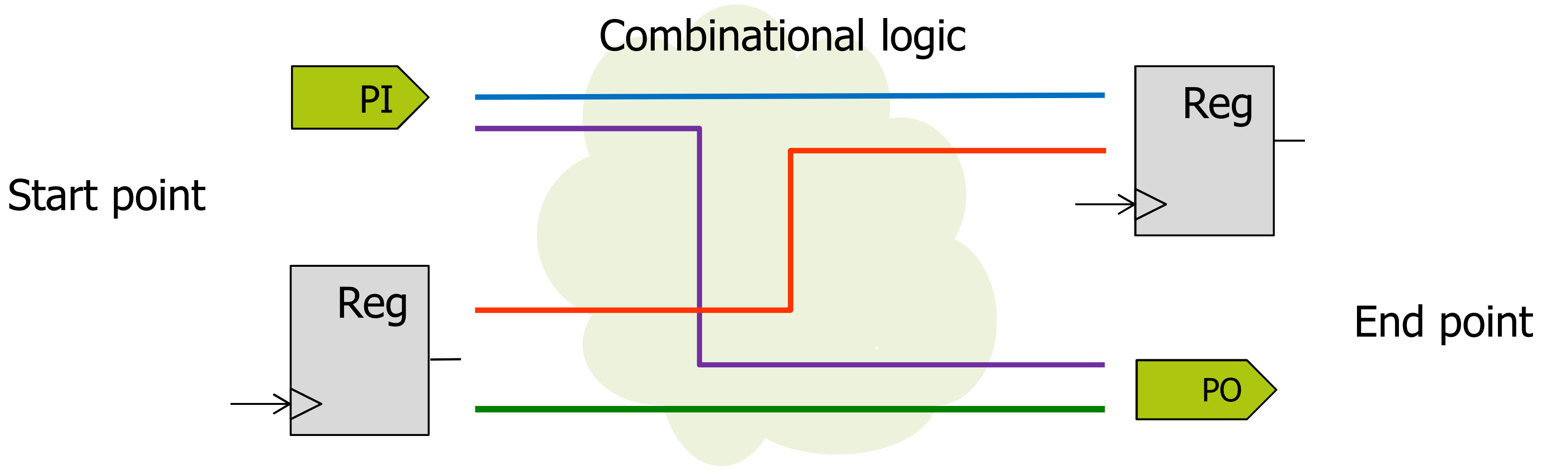
**Maximum Clock Skew**
**Determined by Minimum Delay between Latches**

# Setup and Hold

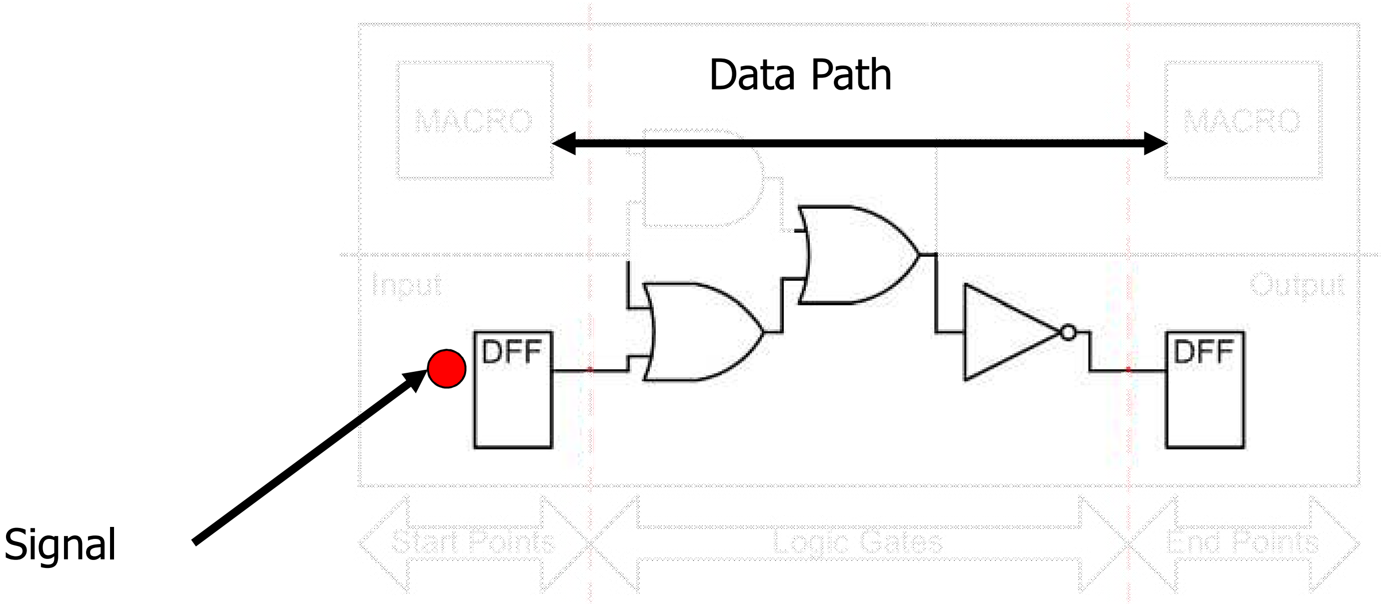
- If your design does not meet the setup timing constraints, it will work at a lower clock frequency
- If your design does not meet hold timing constraints, it will not work at any clock frequency!

# Timing Paths

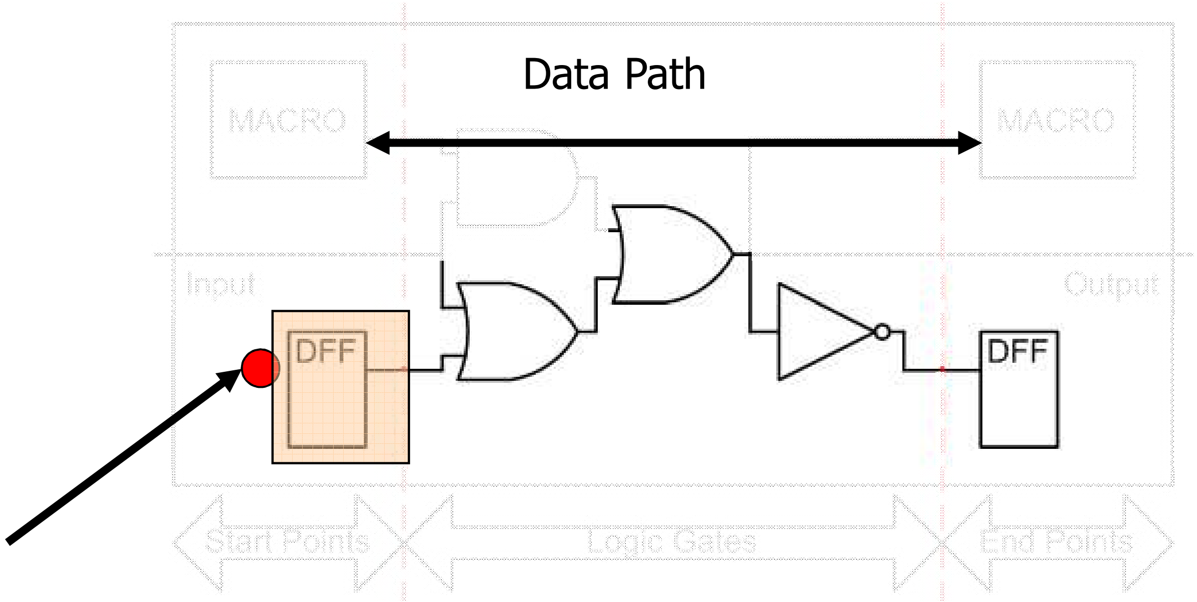
- Four types of paths:



# Digital Circuit – Sequential Path



# Static Timing Analysis (STA)



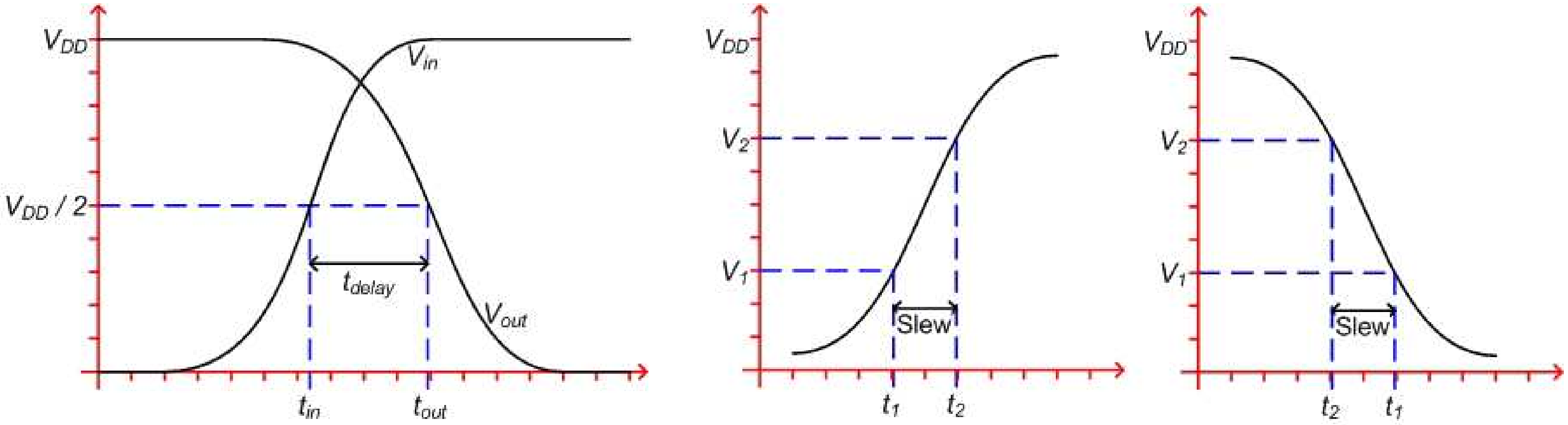
Data Path

Signal

Stage by stage delay calculation

Limited signal information is stored after each stage

# From Signal to

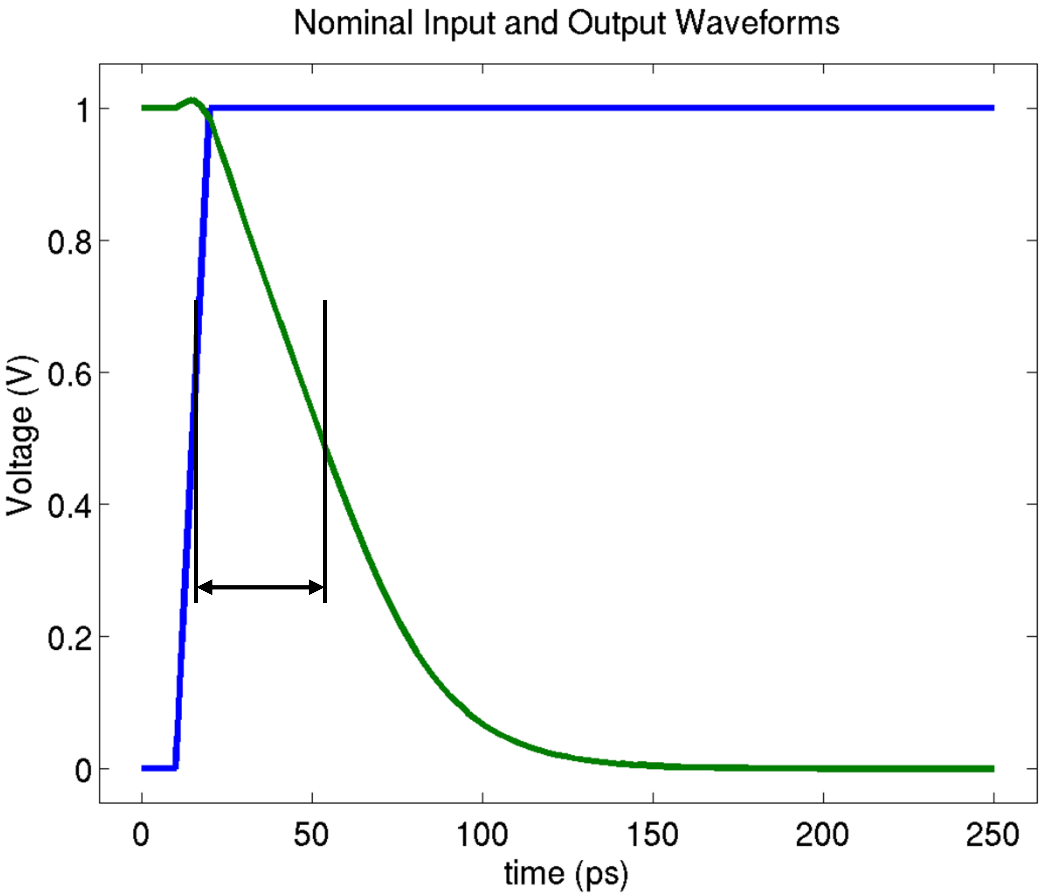


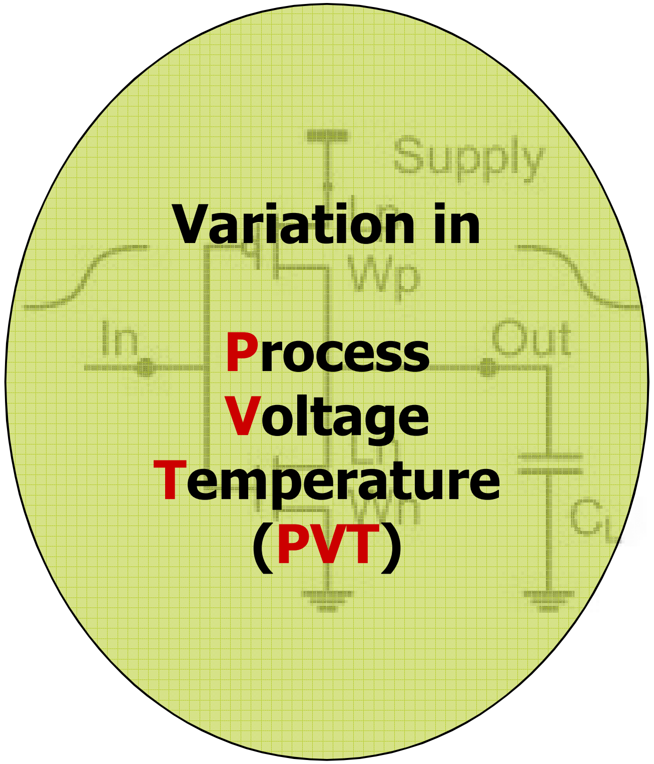
$t_{delay} = t_{out} - t_{in}$

Slew = $|t_2 - t_1|$

Limited signal information is stored after each stage

# Digital Circuit – Delay

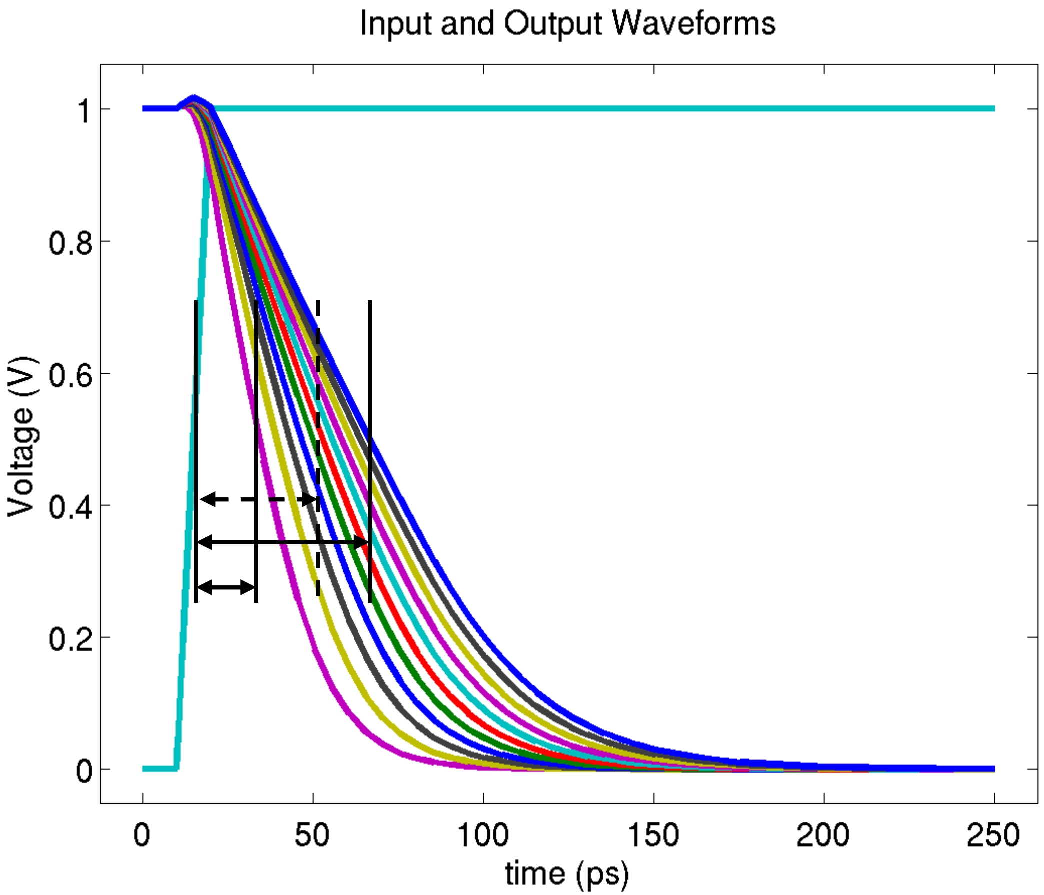




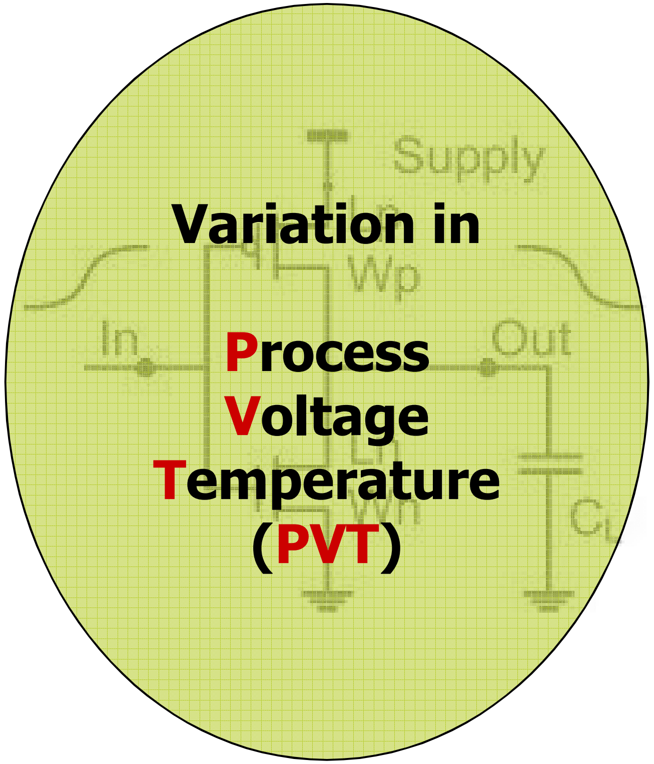
**Variation in**

**Process**
**Voltage**
**Temperature**
**(PVT)**

# Digital Circuit – Delay Variation



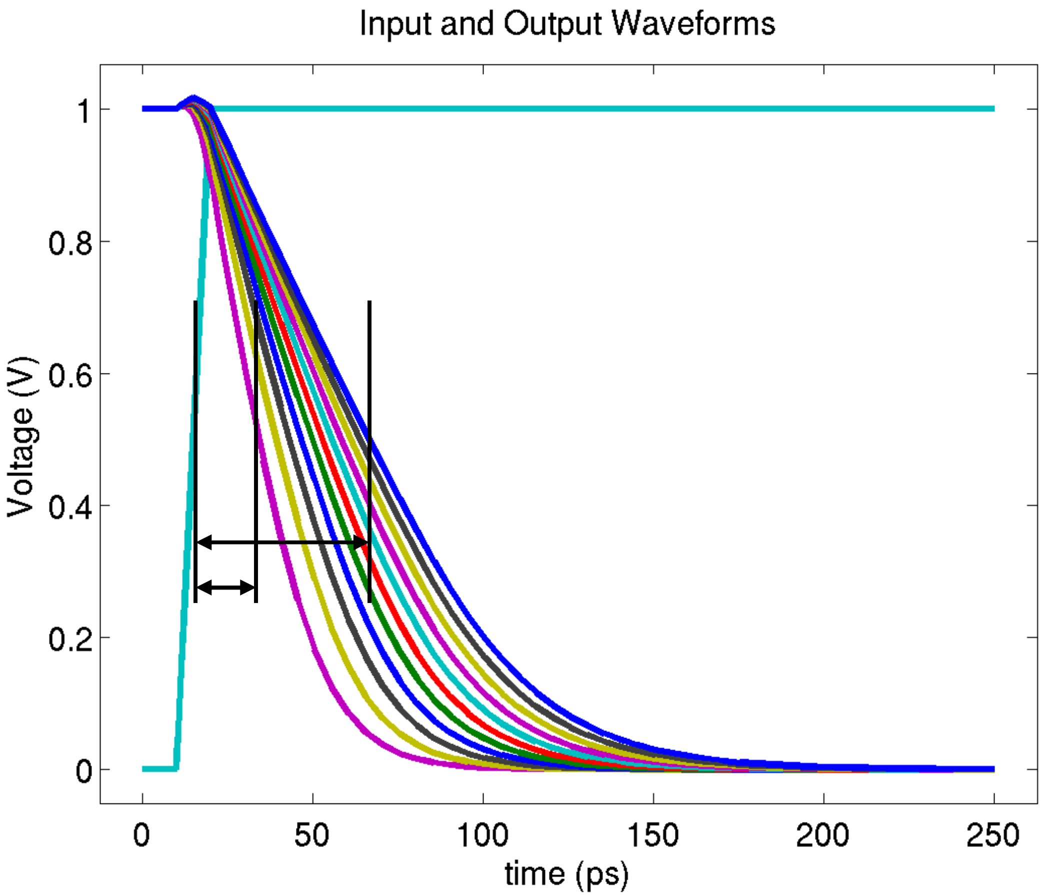
Input and Output Waveforms



**Variation in**

**Process**
**Voltage**
**Temperature**
**(PVT)**

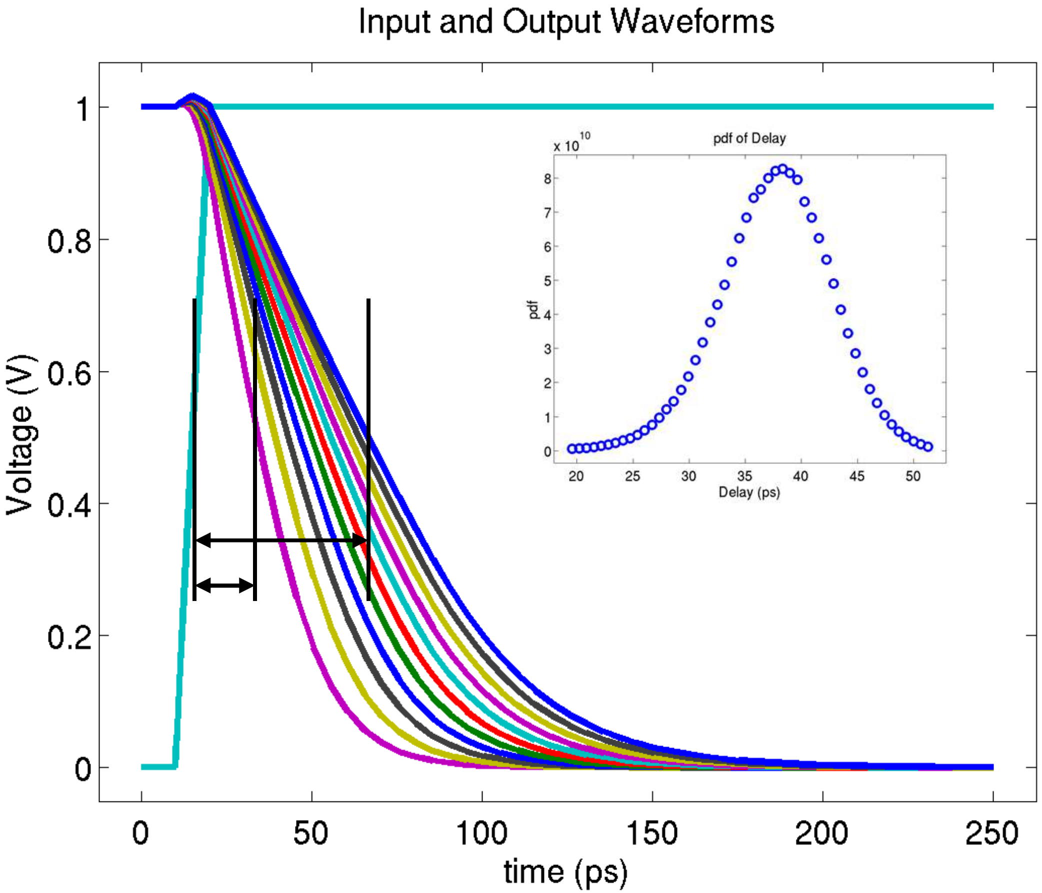
# Delay Variation

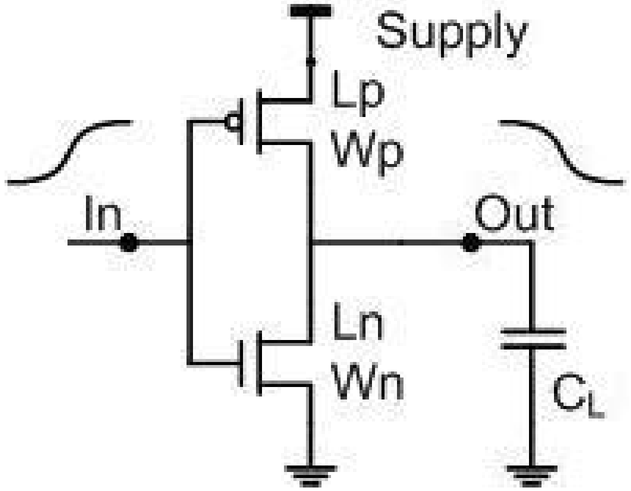


- Delay is not deterministic
- Its distributions is PVT dependent

- So, there is a need of to handle this. Methods:
  - Statistical STA
  - Corner-based

# Digital Circuit – Delay PDF

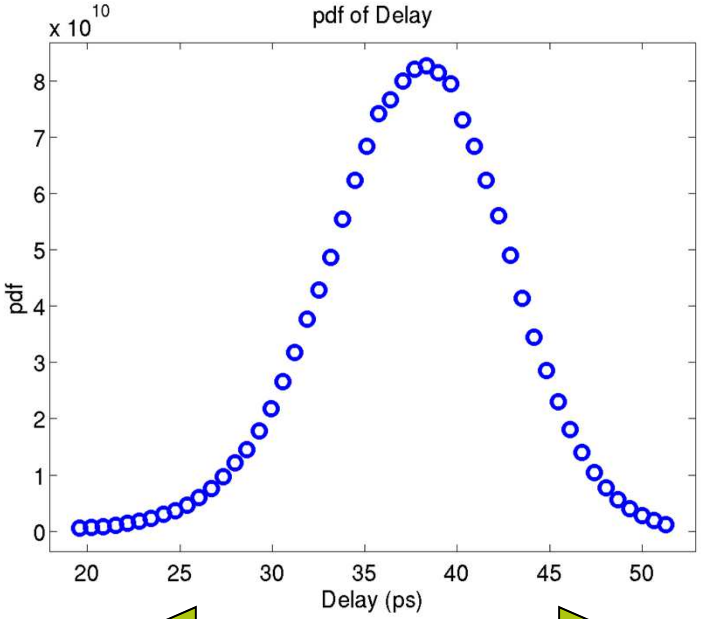




# Statistical Moments

- Measure the appearance of a distribution

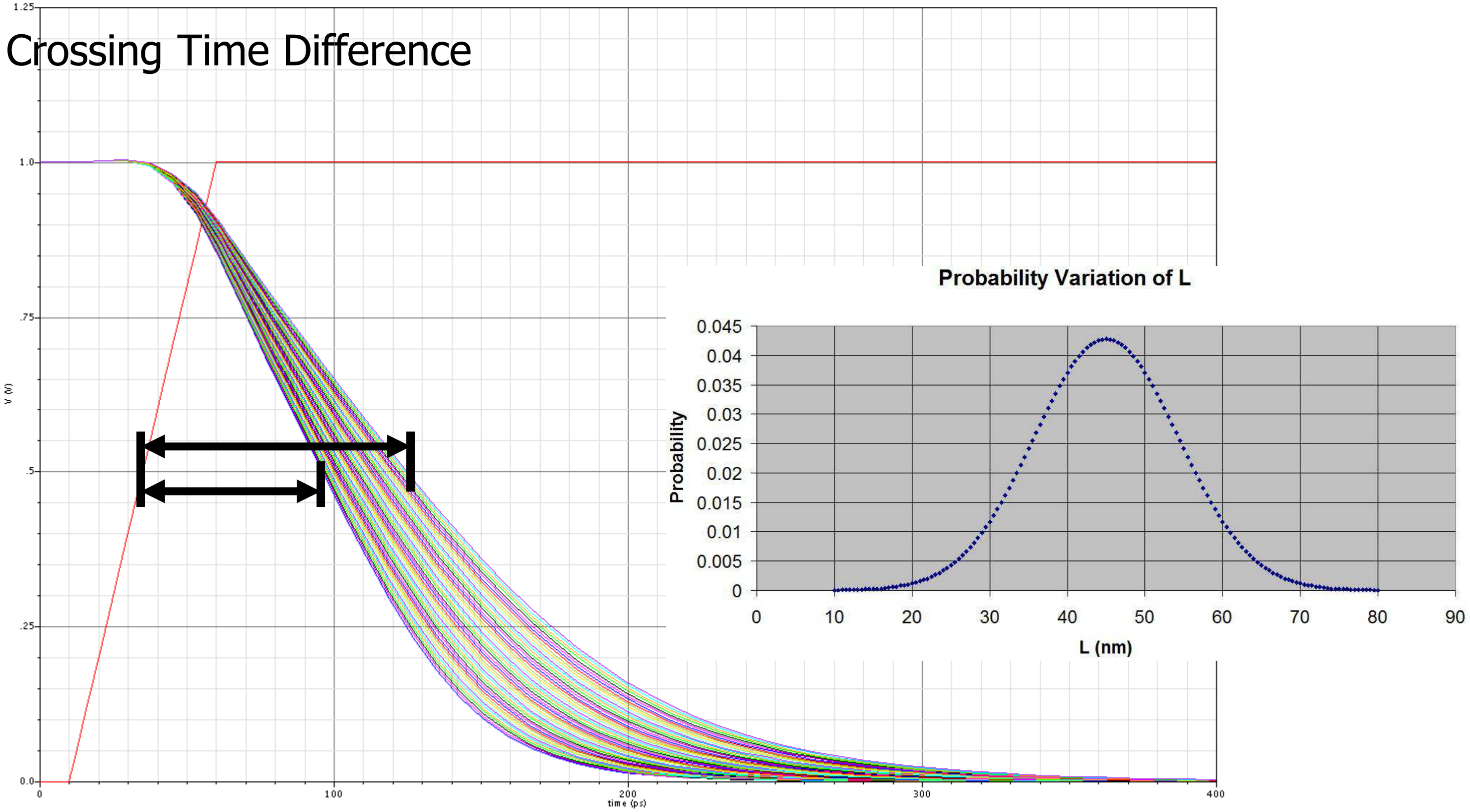
1. Mean (μ) – Location
2. Variance ($\sigma^2$) – Spread
3. Skewness (γ) – Symmetry
4. Kurtosis (κ) – Flatness



Circuit Simulation Results

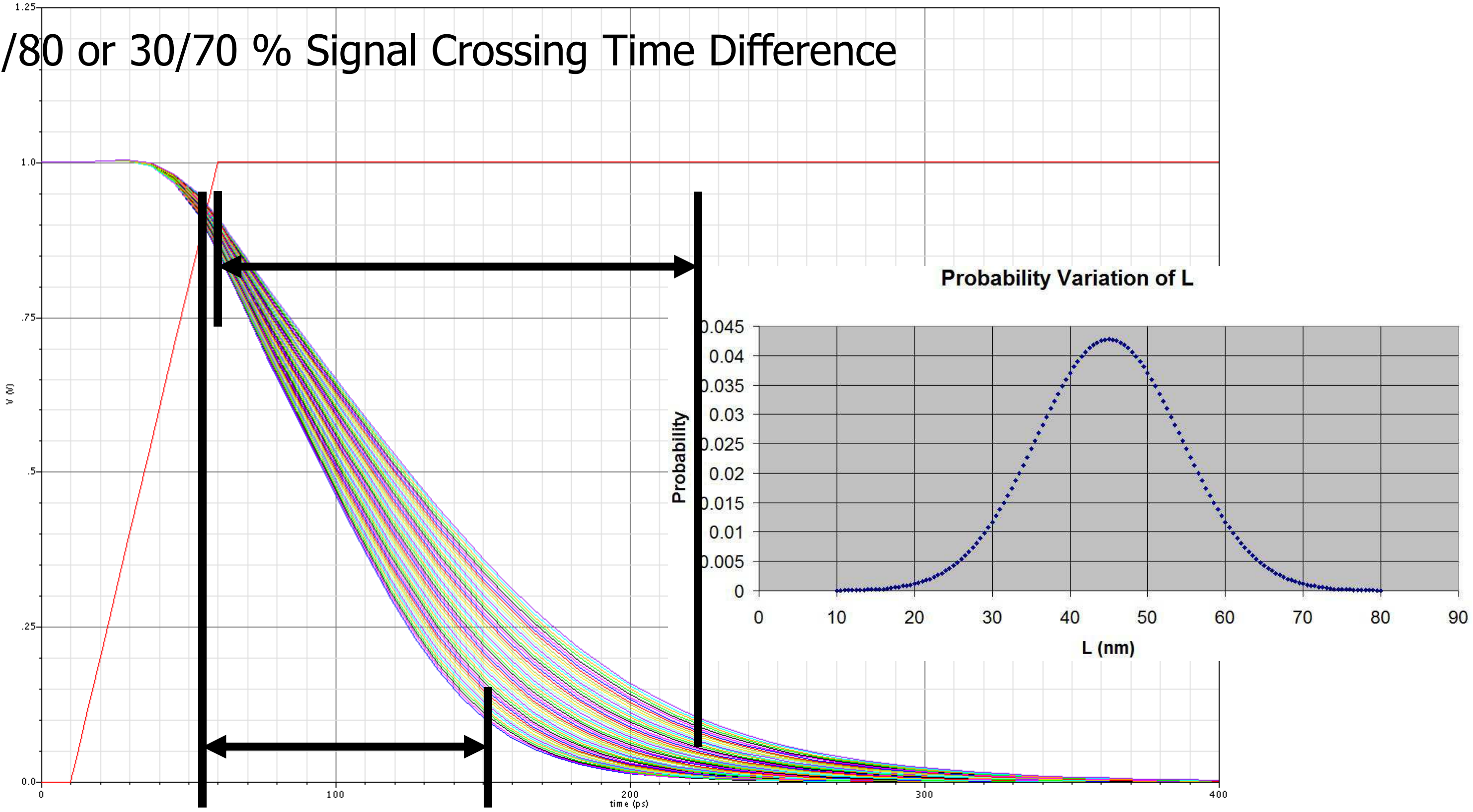
# Statistical Delay

50% Signal Crossing Time Difference

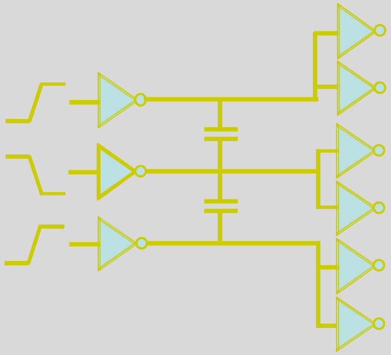
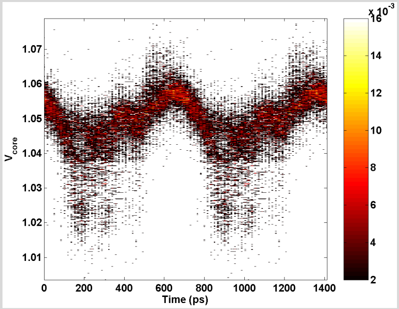
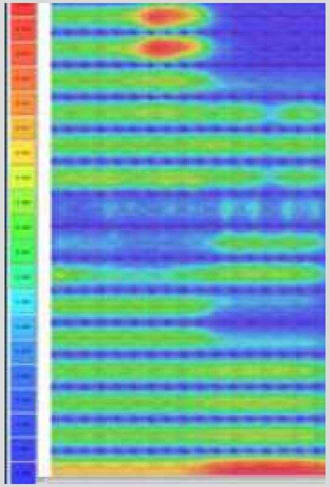
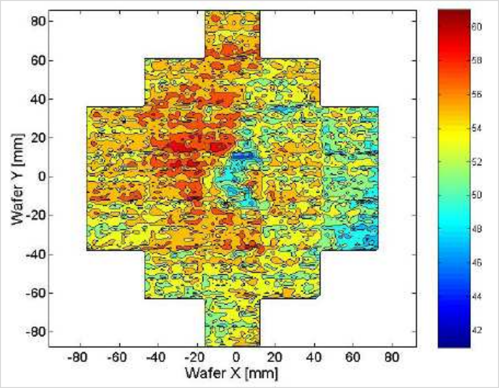


# Statistical Slew

10/90 or 20/80 or 30/70 % Signal Crossing Time Difference

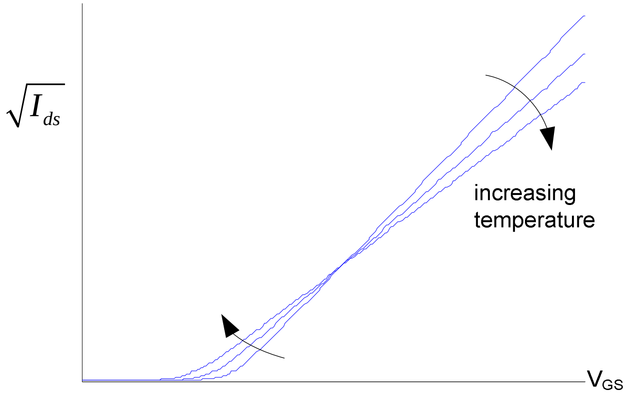


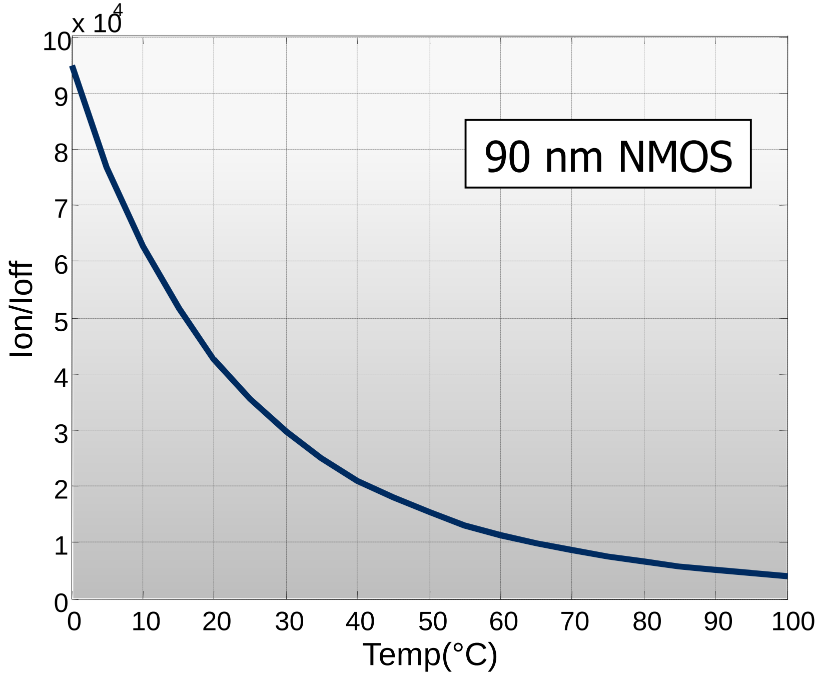
# Variability Sources and their Time Scales

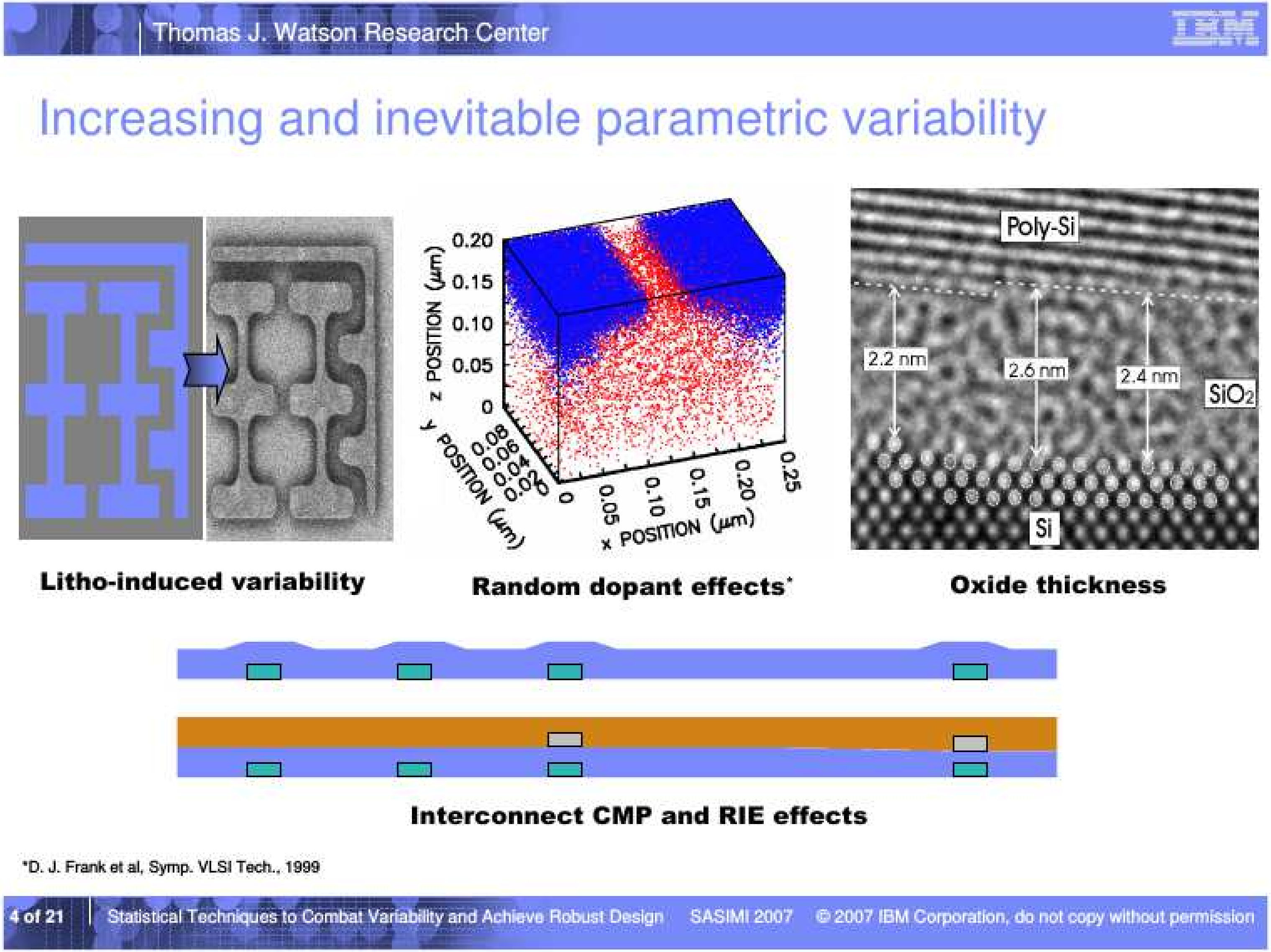
| Signal Coupling | Supply/Package Noise | Temperature – Modal Operation | Manufacturing – Wear-out |
|---|---|---|---|
| $10^{-10}$-$10^{-8}$ | $10^{-7}$-$10^{-5}$ | $10^{-4}$-$10^{-2}$ | $10^{5}$-$10^{7}$ |
|  |  |  |  |

# Temperature Sensitivity

- Increasing temperature
  - Reduces mobility
  - Reduces $V_{TH}$
- $I_{ON}$ decreases with temperature
- $I_{OFF}$ increases with temperature







Thomas J. Watson Research Center

# Increasing and inevitable parametric variability

**Litho-induced variability**

**Random dopant effects***

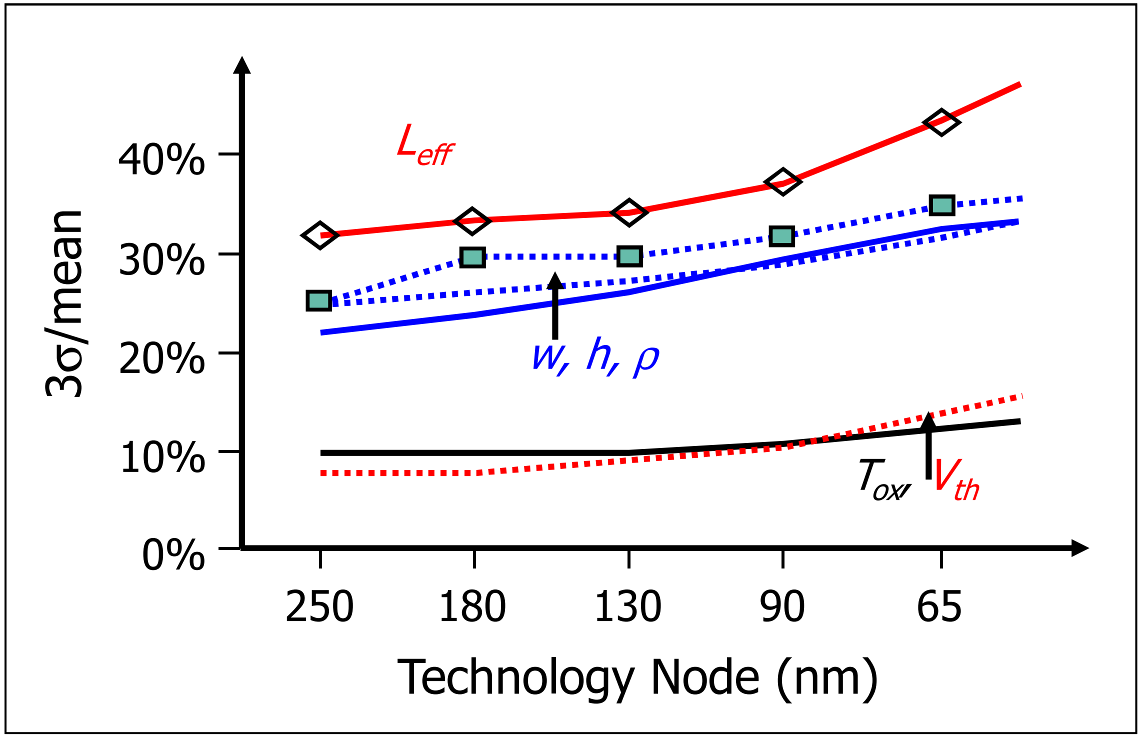
**Oxide thickness**

**Interconnect CMP and RIE effects**

*D. J. Frank et al, Symp. VLSI Tech., 1999

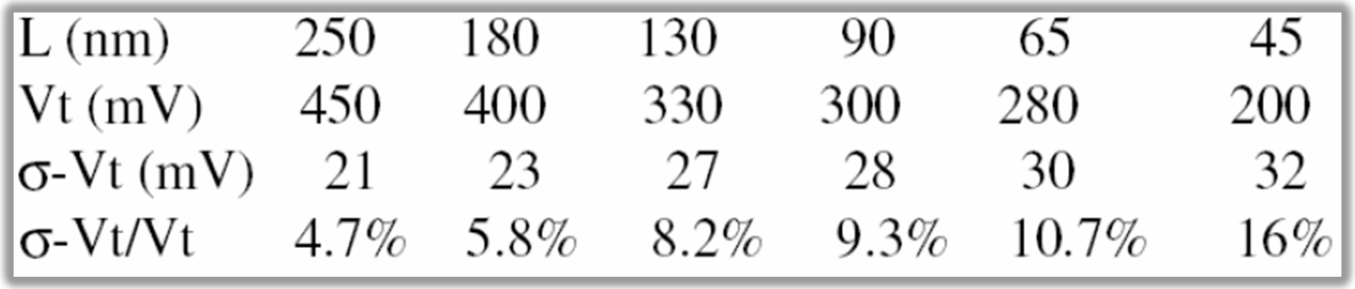
4 of 21 | Statistical Techniques to Combat Variability and Achieve Robust Design SASIMI 2007

# Process Variations



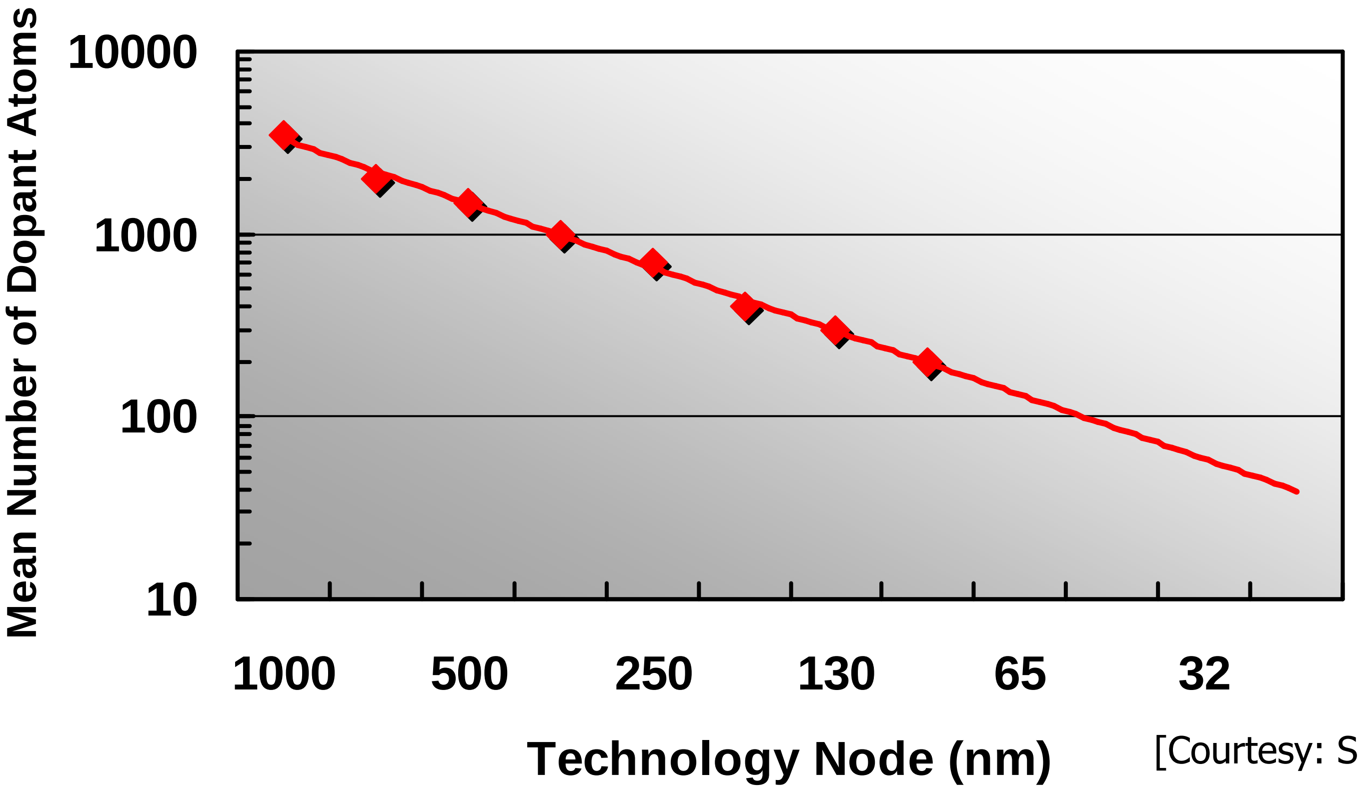
Percentage of total variation accounted for by within-die variation (device and interconnect)

[Courtesy: S. Nassif, IBM]



| L (nm) | 250 | 180 | 130 | 90 | 65 | 45 |
|---|---|---|---|---|---|---|
| Vt (mV) | 450 | 400 | 330 | 300 | 280 | 200 |
| σ-Vt (mV) | 21 | 23 | 27 | 28 | 30 | 32 |
| σ-Vt/Vt | 4.7% | 5.8% | 8.2% | 9.3% | 10.7% | 16% |

# Threshold Variations Most Important for Power



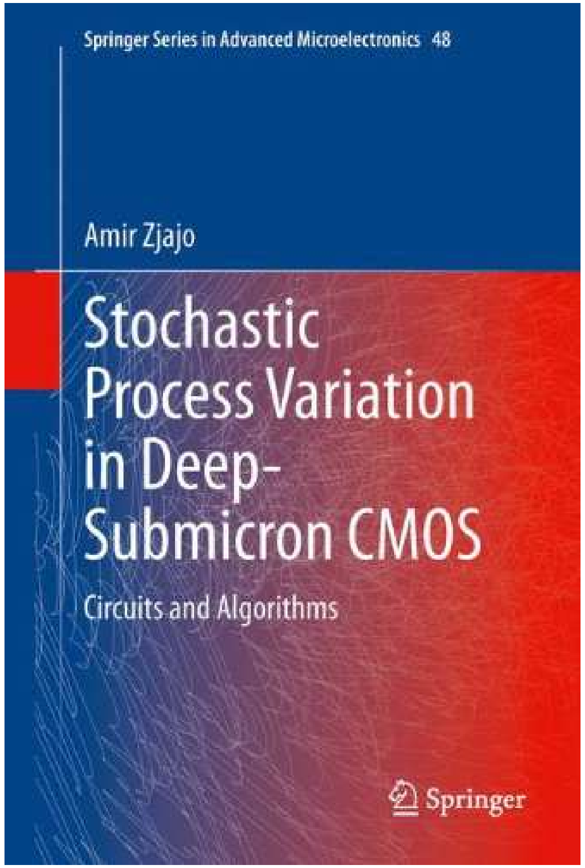
[Courtesy: S. Borkar, Intel]

Decrease of random dopants in channel increases impact of variation on threshold voltage

# Device and Technology Innovations

- Power challenges introduced by nanometer MOS transistors can be partially addressed by new device structures and better materials
  - Higher mobility
  - Reduced leakage
  - Better control
- However …
  - Most of these techniques provide only a one (two) technology generation boost
  - Need to accompanied by **circuit and system** level methodologies

# Stochastic Process Variation in Deep-Submicron CMOS: Circuits and Algorithms



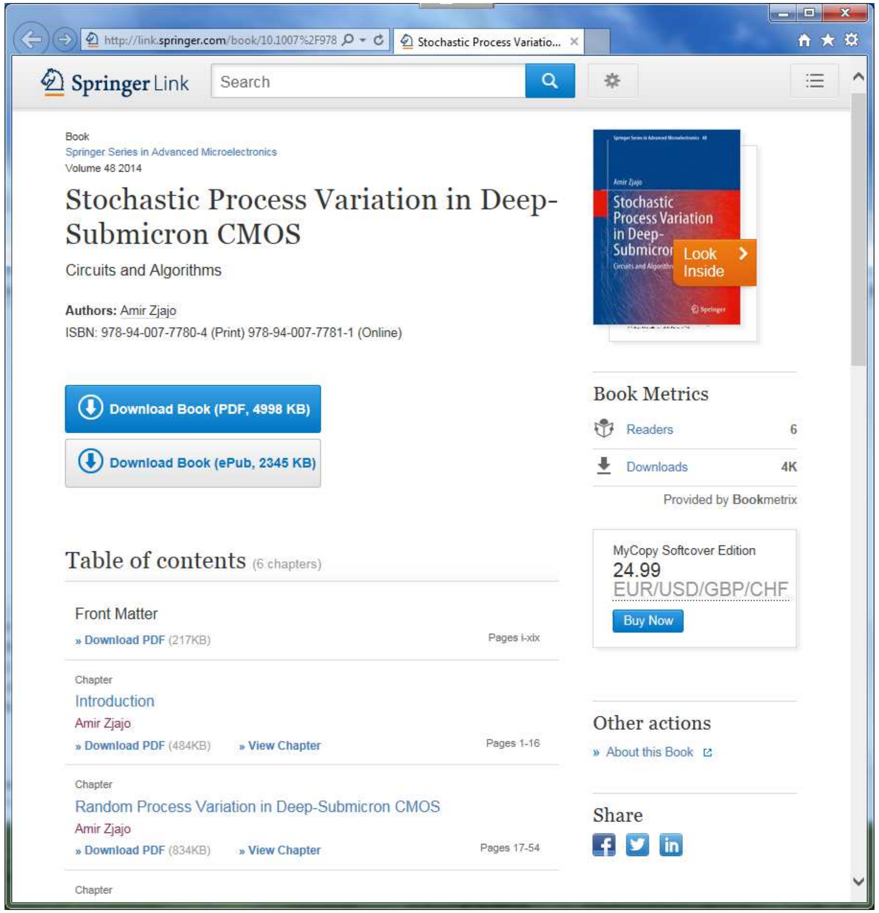
Springer Series in Advanced Microelectronics 48

Amir Zjajo

Stochastic Process Variation in Deep-Submicron CMOS

Circuits and Algorithms

Springer



http://link.springer.com/book/10.1007%2F978
Stochastic Process Variatio...
Springer Link
Search

Book
Springer Series in Advanced Microelectronics
Volume 48 2014

Stochastic Process Variation in Deep-Submicron CMOS

Circuits and Algorithms

**Authors:** Amir Zjajo
ISBN: 978-94-007-7780-4 (Print) 978-94-007-7781-1 (Online)

Download Book (PDF, 4998 KB)

Download Book (ePub, 2345 KB)

Book Metrics

| | |
|---|---|
| Readers | 6 |
| Downloads | 4K |

Provided by Bookmetrix

MyCopy Softcover Edition
24.99
EUR/USD/GBP/CHF
Buy Now

Table of contents (6 chapters)

Front Matter
» Download PDF (217KB) Pages i-xix

Chapter
Introduction
Amir Zjajo
» Download PDF (484KB) » View Chapter Pages 1-16

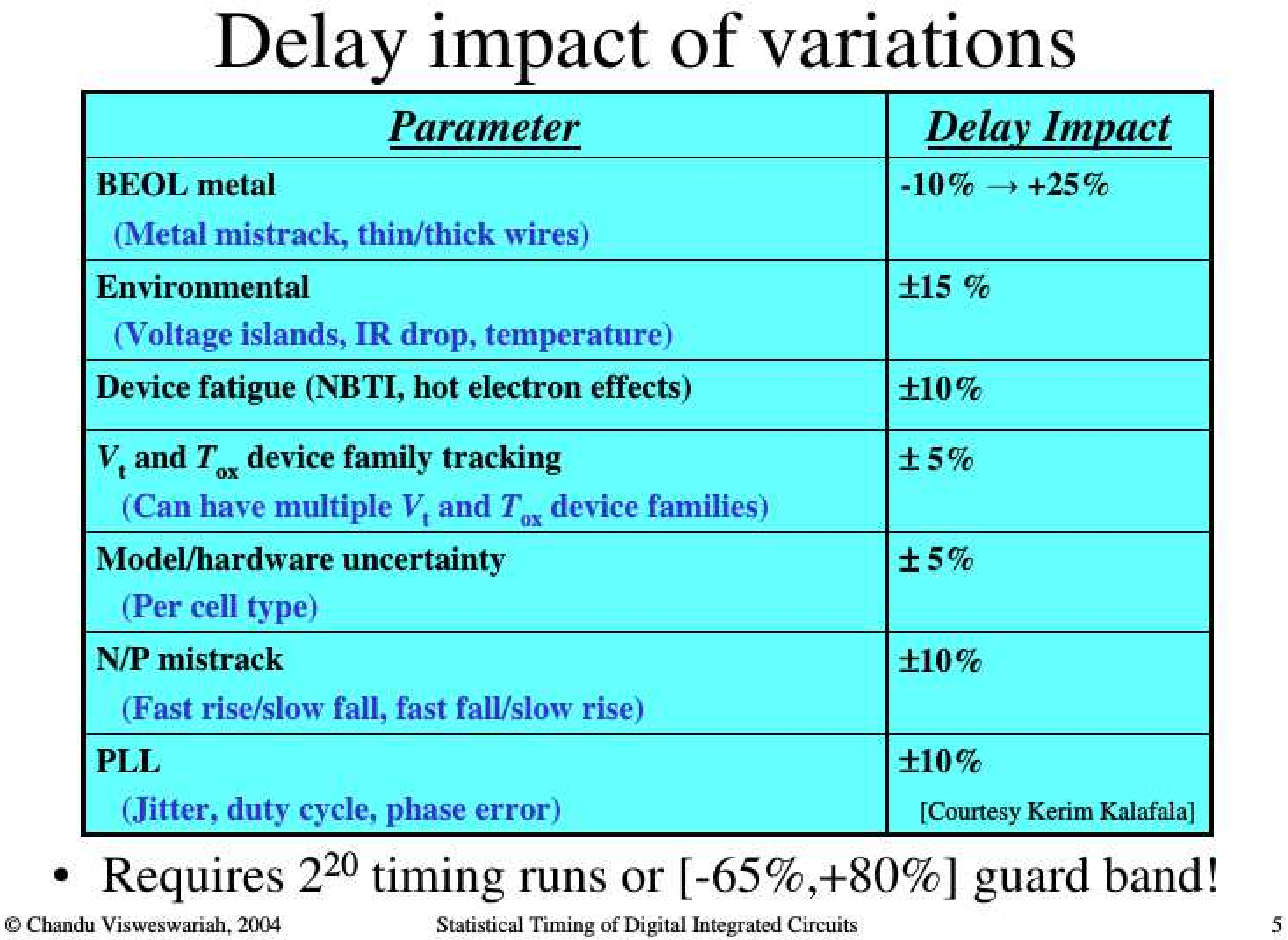
Chapter
Random Process Variation in Deep-Submicron CMOS
Amir Zjajo
» Download PDF (834KB) » View Chapter Pages 17-54

Chapter

Other actions
» About this Book

Share

- Recommended reading
  (**free** download via SpringerLink book site – through University subscription)



# Delay impact of variations

| ***Parameter*** | ***Delay Impact*** |
|---|---|
| **BEOL metal**<br>**(Metal mistrack, thin/thick wires)** | **-10% → +25%** |
| **Environmental**<br>**(Voltage islands, IR drop, temperature)** | **±15 %** |
| **Device fatigue (NBTI, hot electron effects)** | **±10%** |
| **$V_t$ and $T_{ox}$ device family tracking**<br>**(Can have multiple $V_t$ and $T_{ox}$ device families)** | **± 5%** |
| **Model/hardware uncertainty**<br>**(Per cell type)** | **± 5%** |
| **N/P mistrack**<br>**(Fast rise/slow fall, fast fall/slow rise)** | **±10%** |
| **PLL**<br>**(Jitter, duty cycle, phase error)** | **±10%**<br>[Courtesy Kerim Kalafala] |

- Requires $2^{20}$ timing runs or [-65%,+80%] guard band!

© Chandu Visweswariah, 2004 Statistical Timing of Digital Integrated Circuits 5

# Handling Variations

- Variability is huge! [-65%, +80%] guard band.
- Corners: provide best $(\mu - 3\sigma)$, typical, worst $(\mu + 3\sigma)$ case values
  - With *n* varying parameters $3^n$ corners (!)
  - Simple calculations
  - Pessimistic
- Statistical Analysis:
  - Complex calculations (correlations!)
  - Result hard to interpret

# Overview

- Design Constraints
  - Power, Area, Frequency, CMOS Scaling
- Timing
  - Timing Metrics, Paths, Variability and Delay
- Deterministic Timing Analysis (Static Timing Analysis)
  - Models, Interconnect, Networks, Clock Distribution
- Statistical Timing Analysis
  - Probability, Spatial Correlations, MAX function
- Design Flow
  - Synthesis, Transformation, Definitions, Constrains