Life on the treadmill

Krisztián Flautner
VP R&D, ARM

krisztian.flautner@arm.com

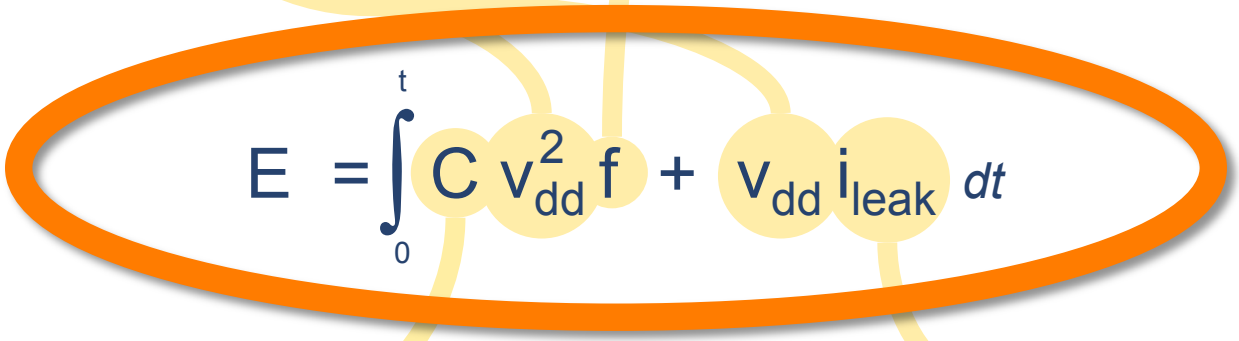
May 1, 2009

ARM

The problem in a nutshell …

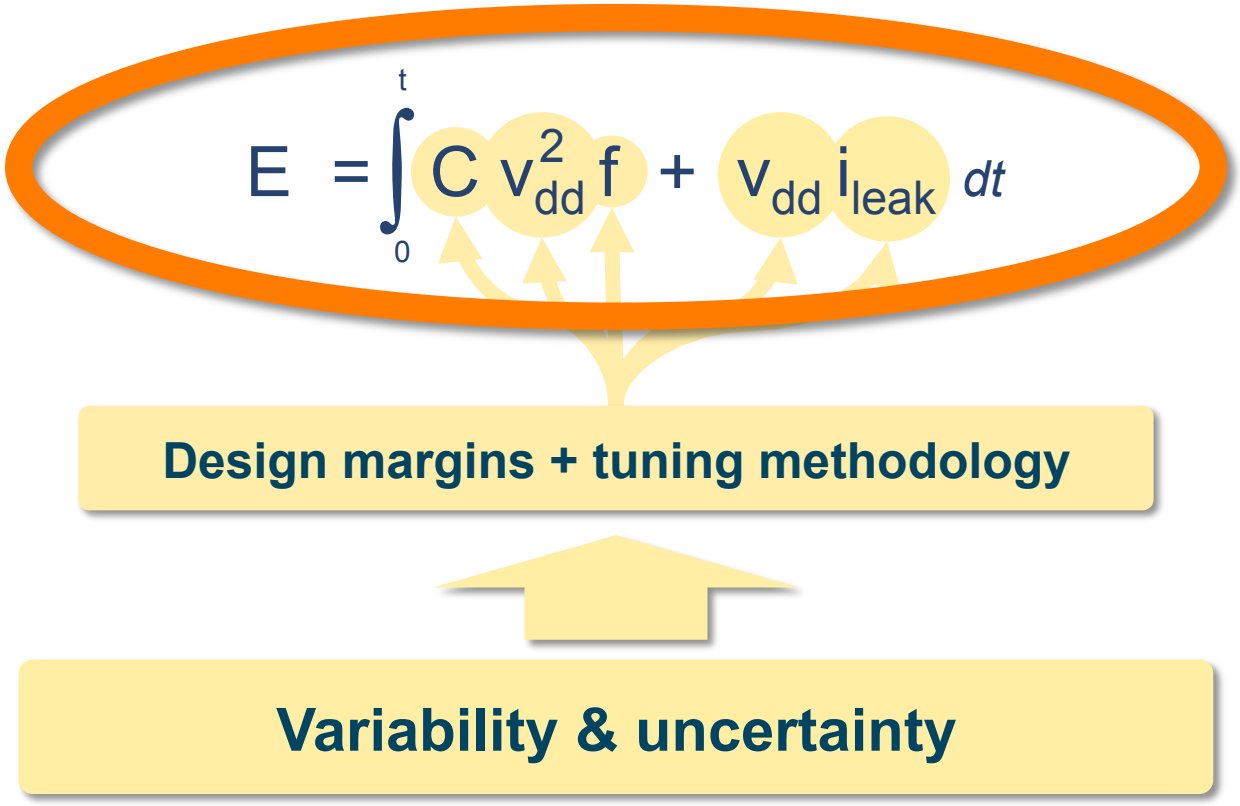
Nominal voltage only reduced occasionally to maintain robustness
Voltage ➔

Market demand for more
Frequency ↗



C xtor reduces, C wire + gate is up
Capacitance ↗

Lots of attention, lots of solutions: materials & modality
No silver bullet: **Leakage ➔↗**


$E = \int_0^t C\, v_{dd}^2\, f + v_{dd}\, i_{leak}\, dt$
Design margins + tuning methodology
Variability & uncertainty

Silicon validation of libraries

Basic idea

- Measure silicon, compare with model prediction

Things to measure

- Delay
- Power
  - Leakage
  - Dynamic

Challenges

- Where does silicon fit in “corners”
- Measurement accuracy
- Single point versus table
- Model versus SPICE
- SPICE versus silicon
- Parametric variation
- Presence of “soft” defects

Overcoming challenges

Where does silicon fit in “corners”?

- Oscillator data, test structure data

Measurement accuracy

- Understand equipment, measure deltas
- Big challenge for power

Single point versus table

- Carefully select design point
- Shmoo across voltage, temperature

Model versus SPICE

- Understand characterization issues

SPICE versus silicon

- Work with foundries to understand

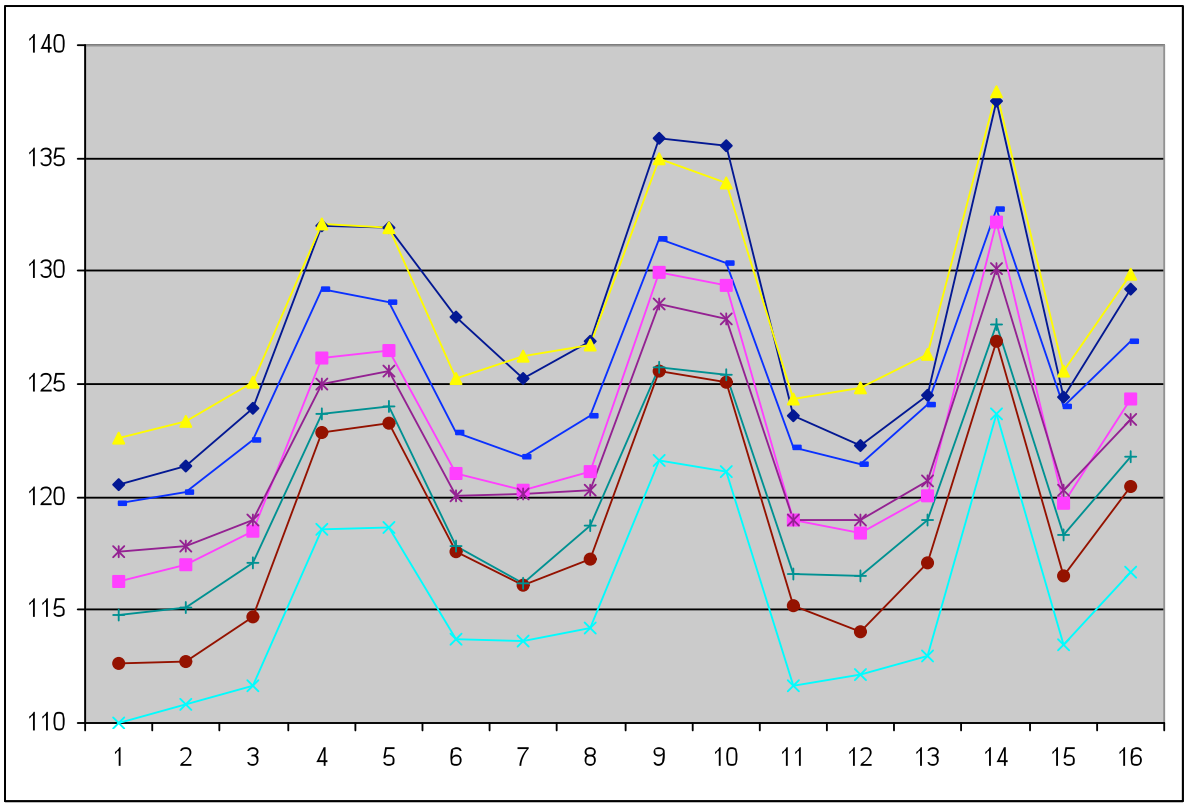
Parametric variation

- Design around local variation

Presence of “soft” defects

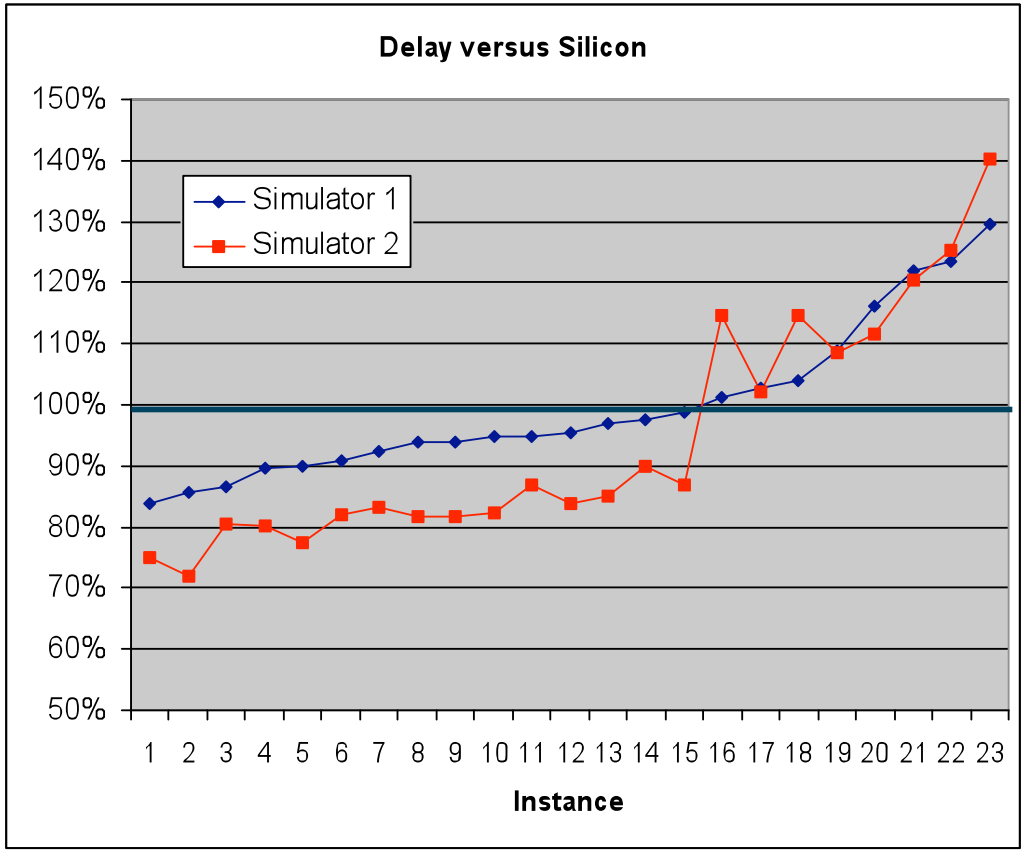
- Look for trends across chips

Validation in practice



- Variability observed for similar objects across chips
- “Correct” value somewhere in the middle
- Does this validate it?

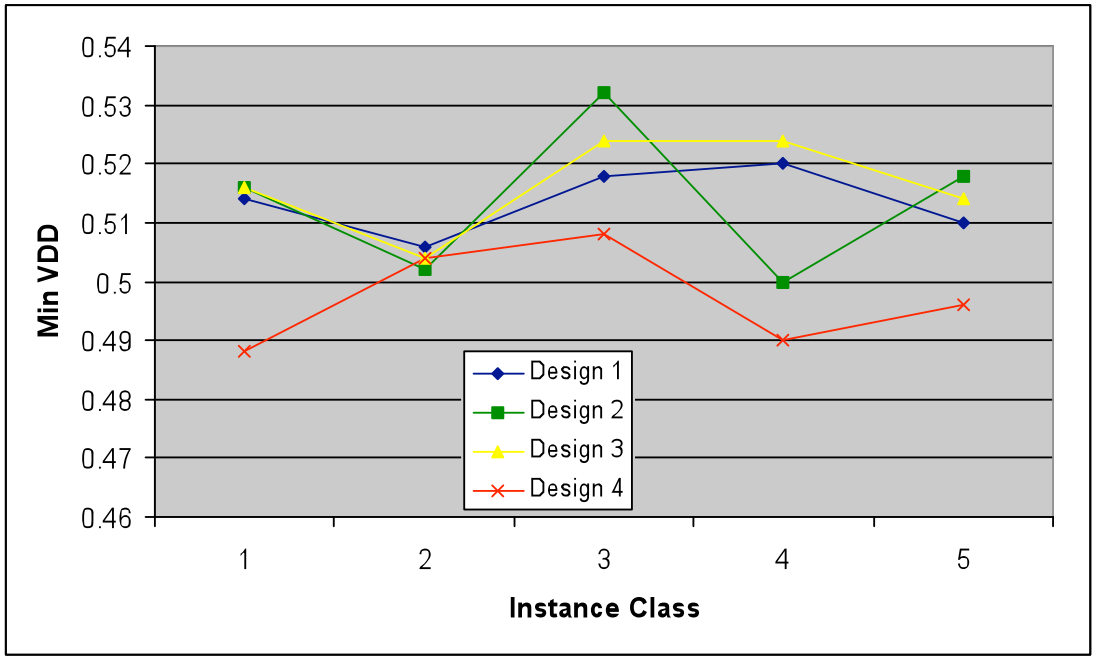
How close is it to SPICE?



- Significant difference between simulators observed
- Variety of issues represented
  - Model file interpretation
  - Performance options
  - Extraction issues
  - Silicon variability

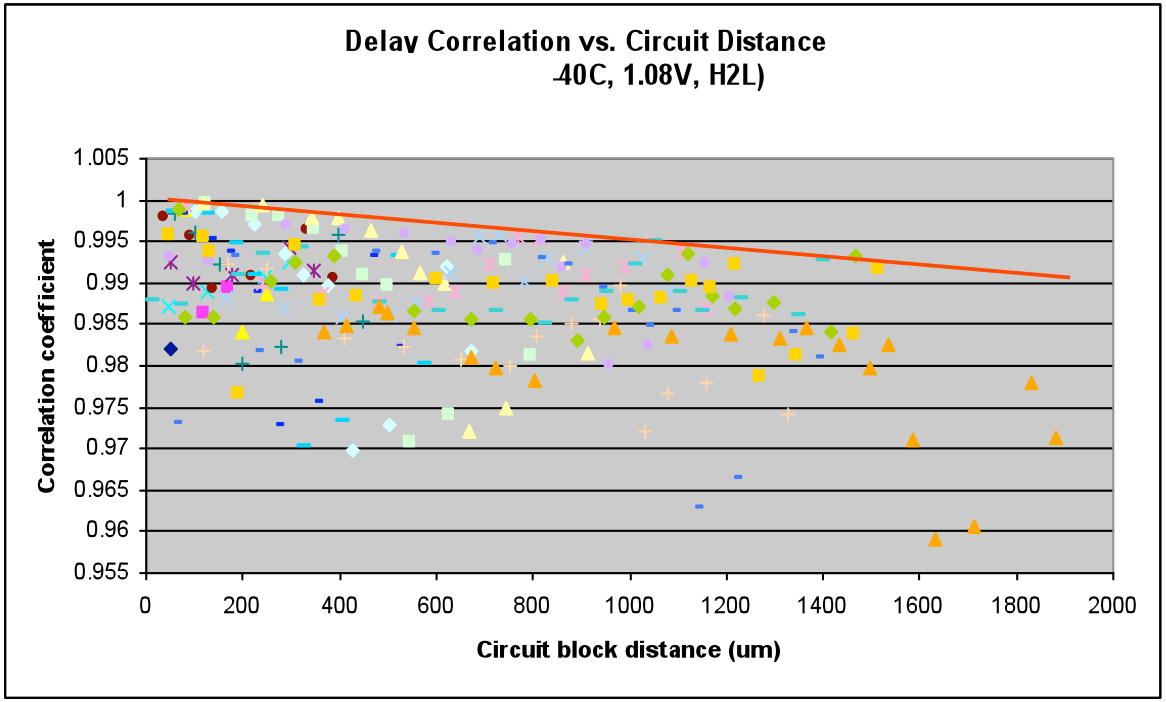
Remember this the next time some tool claims to be within X% of SPICE

Min V_{DD} for different design styles



- Question: Are any of these designs better?
- Statistically: No
- But: More data might give the edge to design 4

Variability and Validation



- Look for correlation
- 65nm data
- Result: small, but measurable, distance-based effect observed

Sources of Variability

Lithography

- Line edge roughness
- CD variation
- Influence of neighbors

Device

- Well boundary effects
- Variation between N and P
- Stress/strain effects

Interconnect

- Dielectric variation
- Via/contact quality
- Metal width/height variation

Deterministic versus random

Effects of Variability

$$I_d \approx \mu \cdot C_{ox} \left(\frac{W}{L} \right) (V_{gs} - V_t)^{\alpha}$$

Leakage

- Variation in L, Vt, μ, t_{ox}

Performance

- Changes in L, W, R, C, Vt, μ,

Min V_{DD}

- Changes in Vt, L, W
- SRAM bit cell main limiter

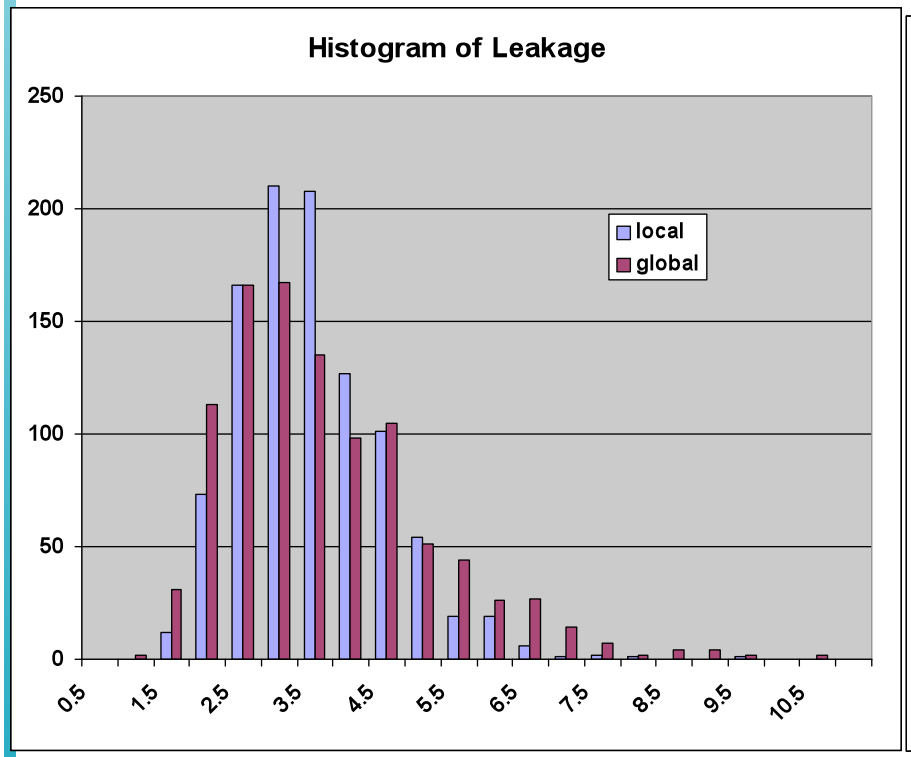
Dynamic power

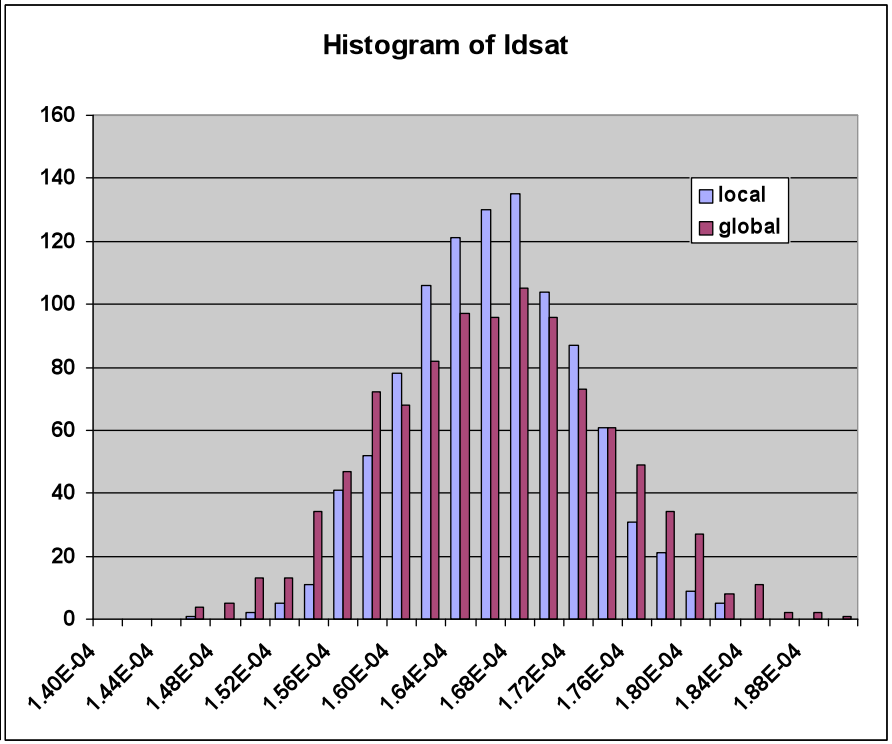
- Changes in C
- Side effect of changes in performance, leakage

Yield

- Indirect result of others
- Parameter goes beyond spec + tolerance

Local Variation Dominates in VDSM





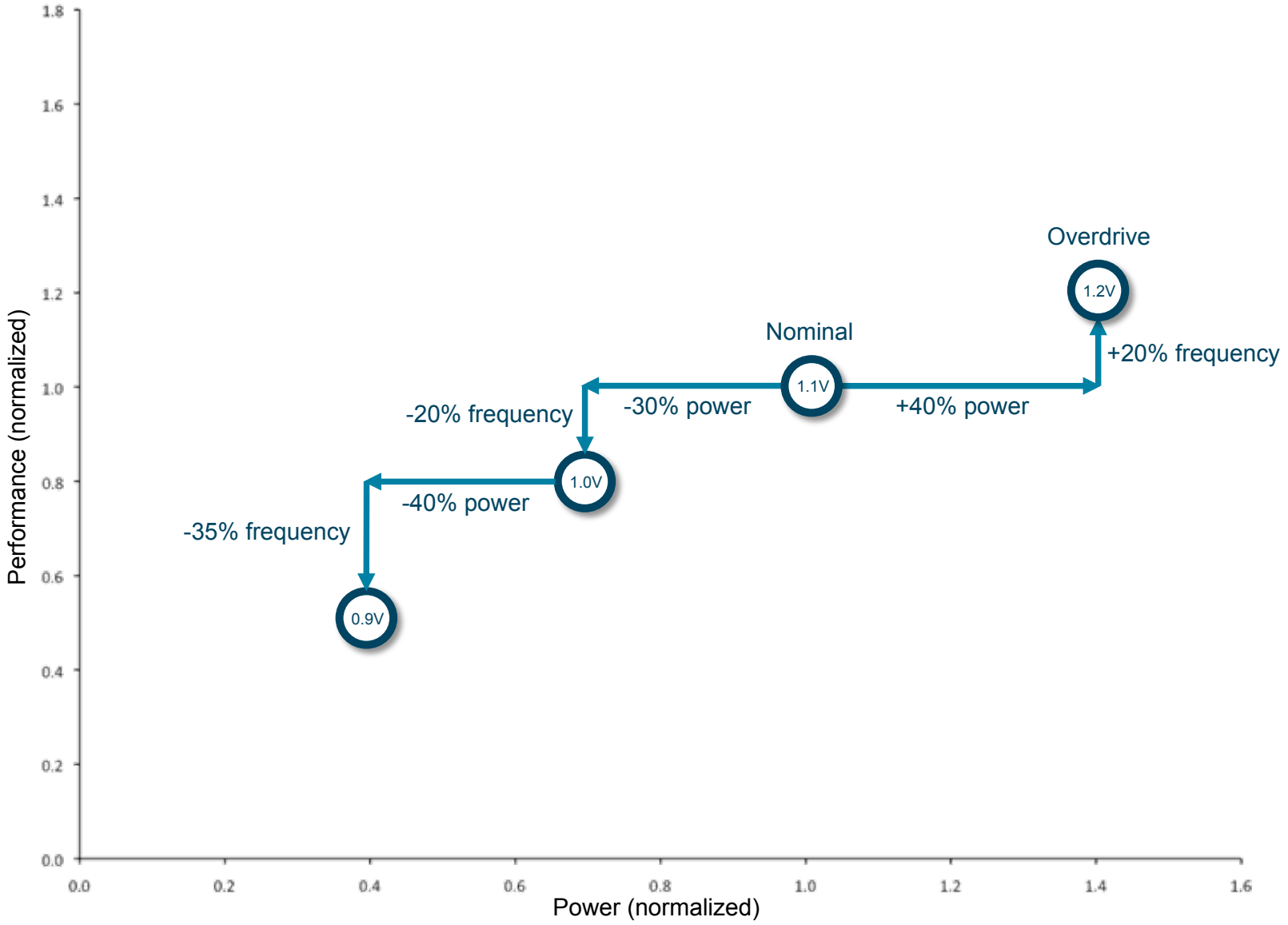
1000 Monte Carlo samples, 45nm technology

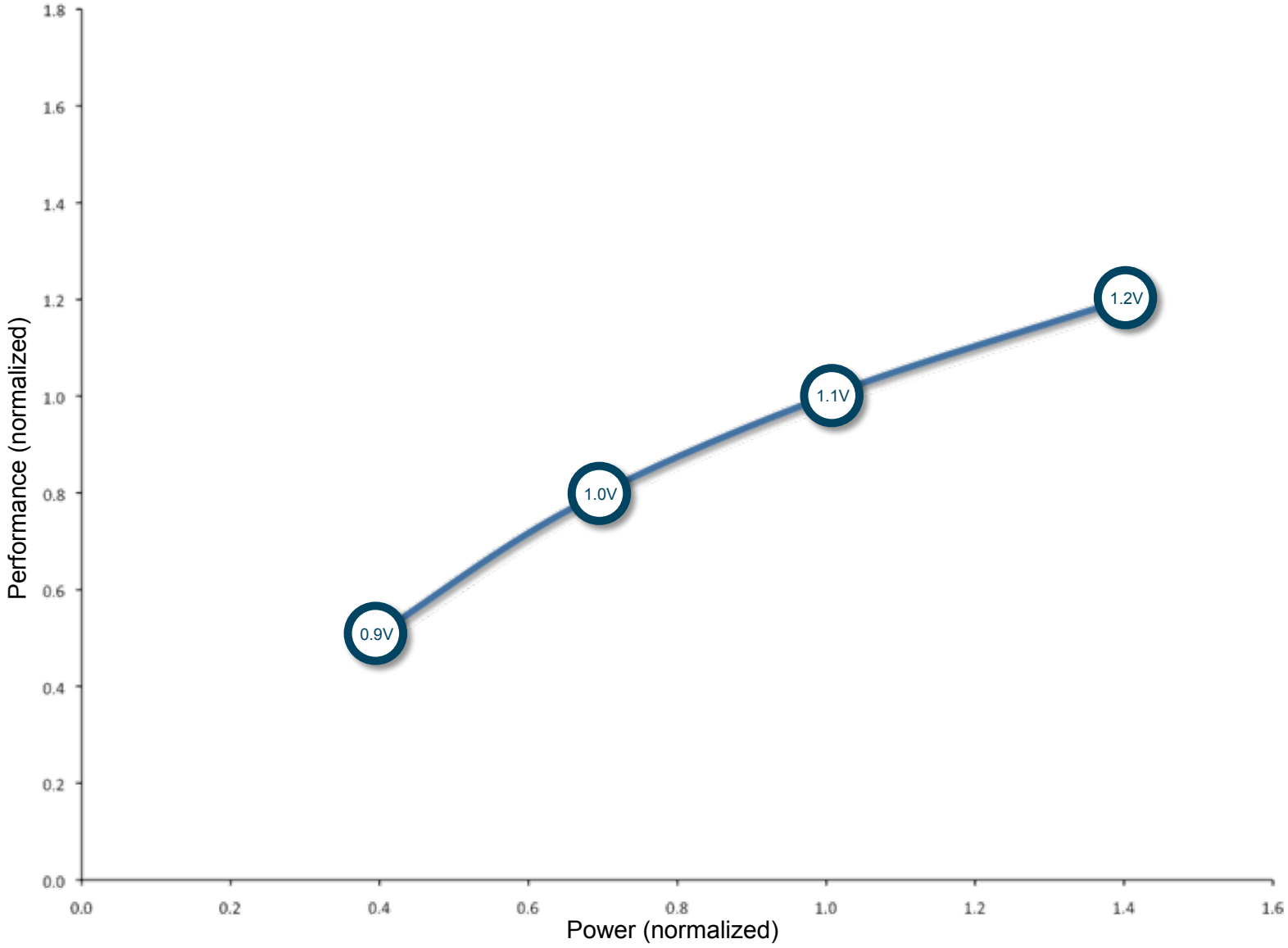
- Local variation (within chip) is nearly as much as global variation (between chips) at 45nm

Critical Variability

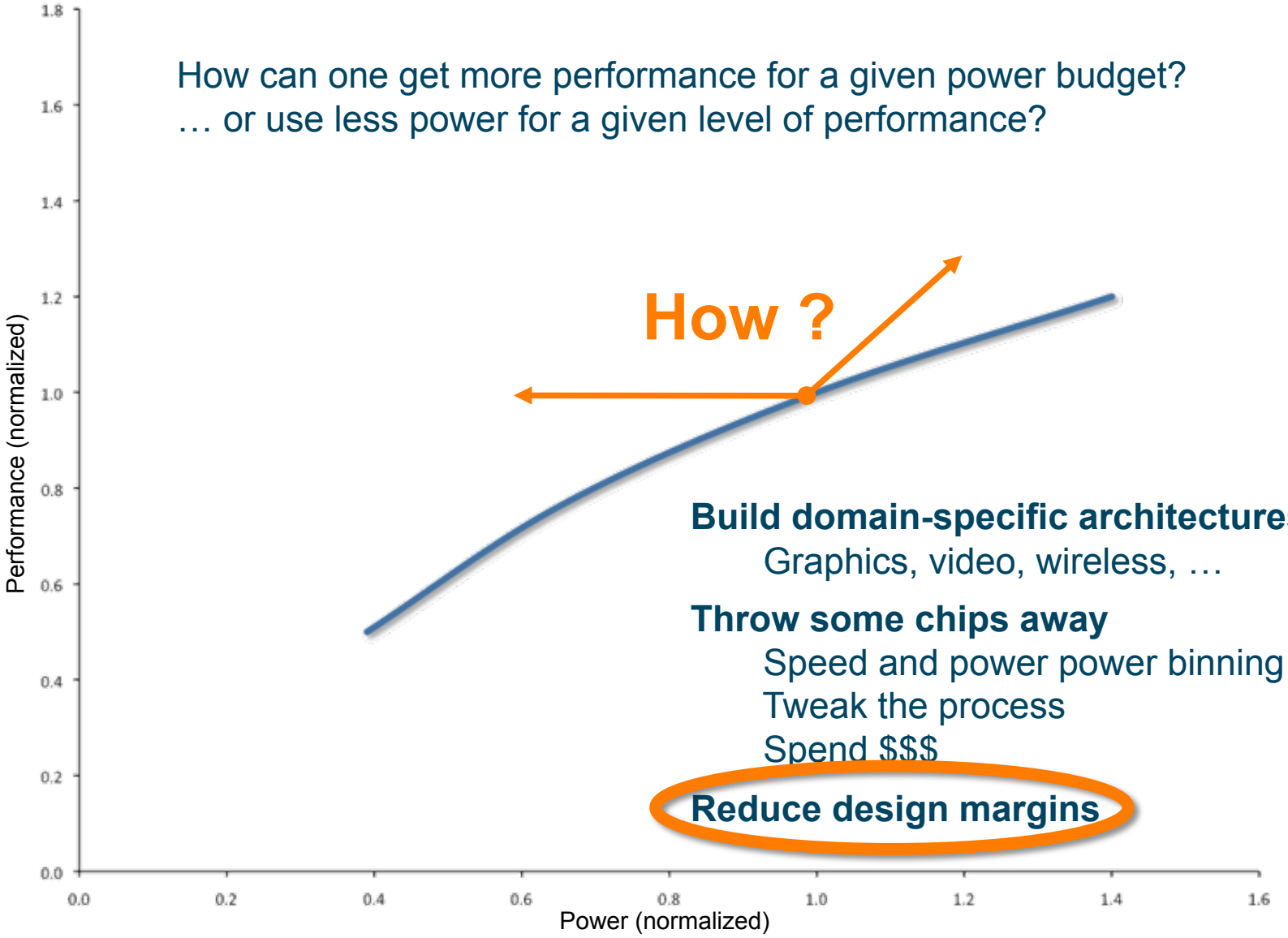
- Designed for a predetermined operating point, plus margin
- Example: SS Corner, 0.9V, 125C, 0 slack
  - Case 1: TT silicon, 0.9V, 85C
    - >2X nominal delay will still function correctly
  - Case 2: SS silicon, 0.9V, 85C
    - ~20% extra delay will cause failure
  - Influential factors:
    - Process, voltage, temperature, slack, noise
  - Expected silicon distribution: SS < 1%, TT(+/-) > 50%
    - Might expect a 1-2% yield hit if SS corner just misses timing
  - Guaranteed silicon distribution: None (usually)
    - Could wind up with no yield 1-2% of the time (1 week per year)
    - Or worse…

Voltage is the main knob which allows one to make trade-offs between performance and power consumption


Performance (normalized)
Power (normalized)
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
Overdrive
1.2V
+20% frequency
Nominal
1.1V
-30% power
+40% power
-20% frequency
1.0V
-40% power
-35% frequency
0.9V


Performance (normalized)
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
Power (normalized)
0.9V
1.0V
1.1V
1.2V

How can one get more performance for a given power budget?
… or use less power for a given level of performance?



Build domain-specific architectures:
Graphics, video, wireless, …

Throw some chips away
Speed and power power binning
Tweak the process
Spend $$$

Reduce design margins

But aren't margins there for a reason?!

YES they are!

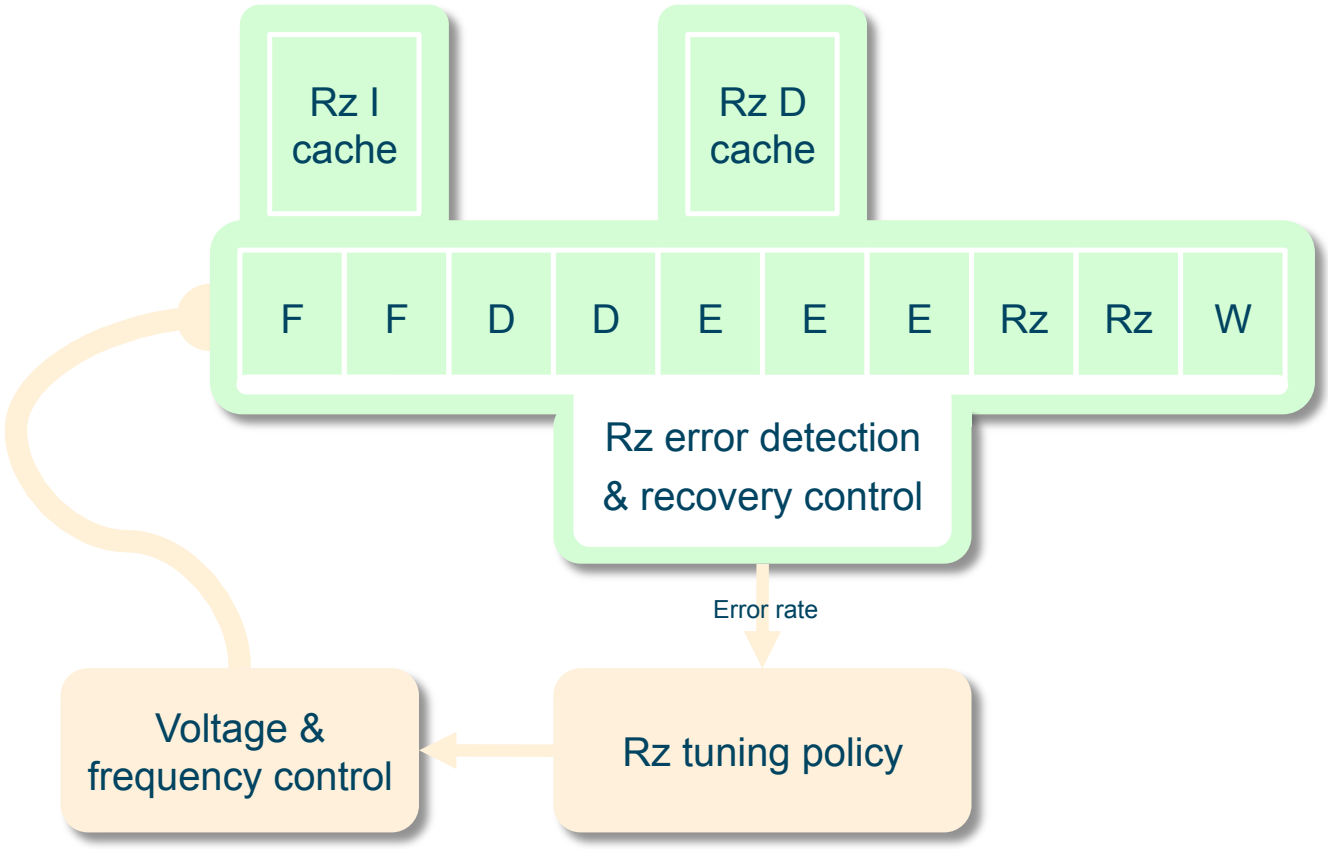
Margins are usually determined empirically
… not scientifically (how does one reason about changing them?)
… over many product generations

Margins are useful as they enable separation of concerns
… between engineering groups
… and legal entities

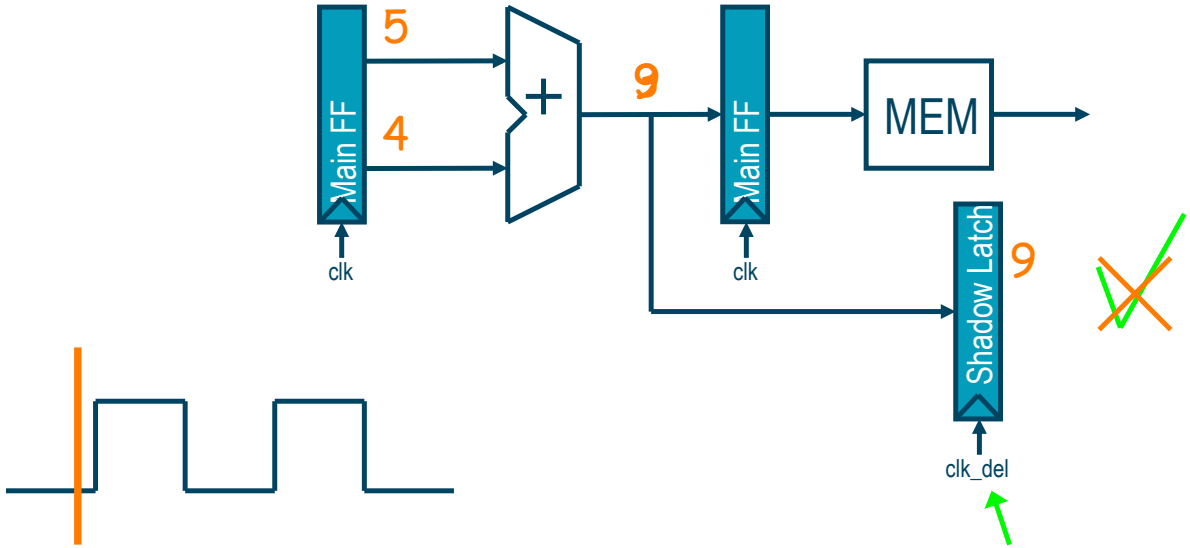
So let's not eliminate margins but be smarter about when and how much is applied

Invoke margins on demand!

High-level view of a Razor system



Razor timing-fault tolerance



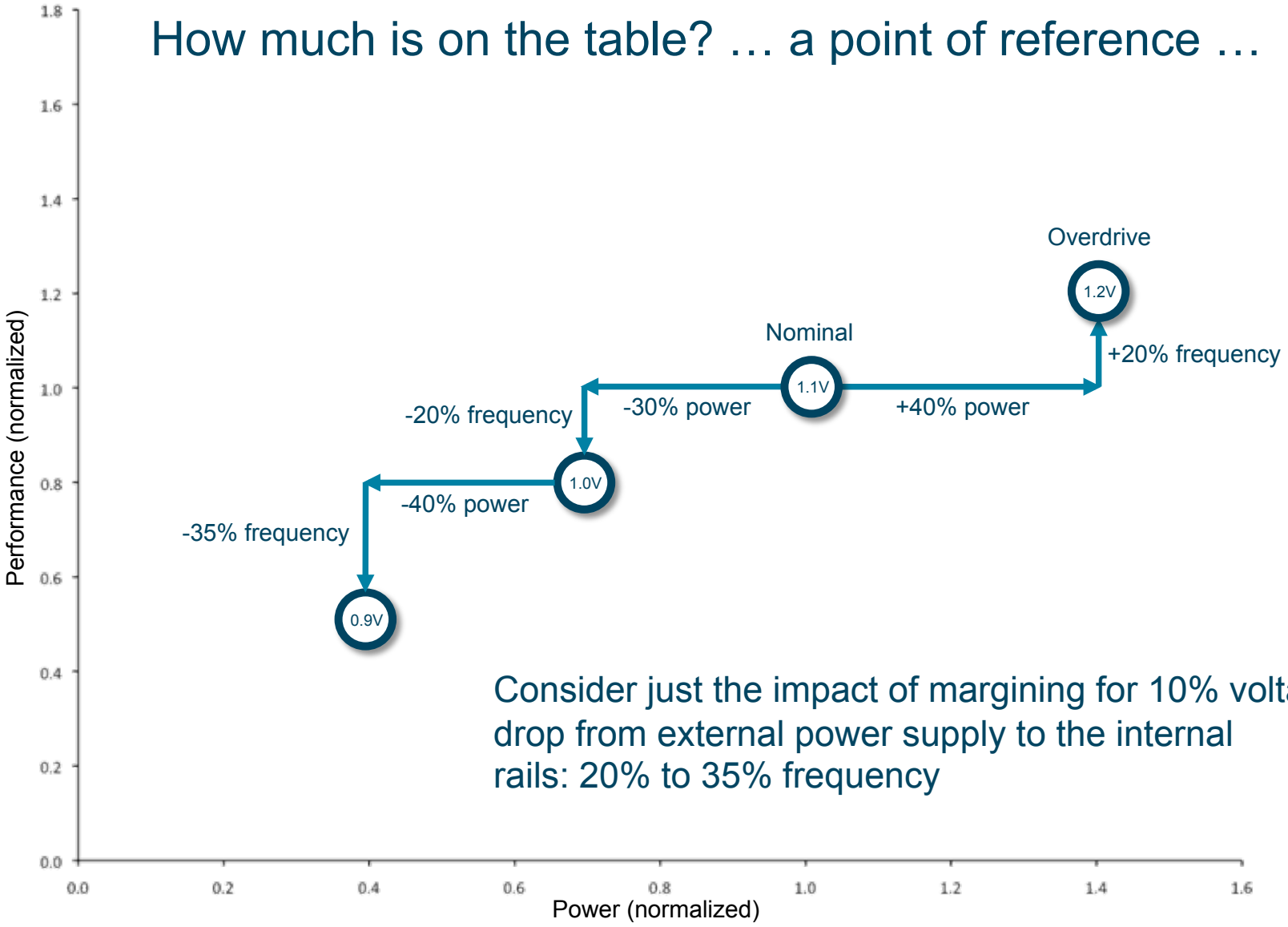
Double-sampling metastability tolerant latches detect timing errors

- Second sample is correct-by-design

Microarchitectural support restores state

- Timing errors treated like branch mispredictions

How much is on the table? … a point of reference …

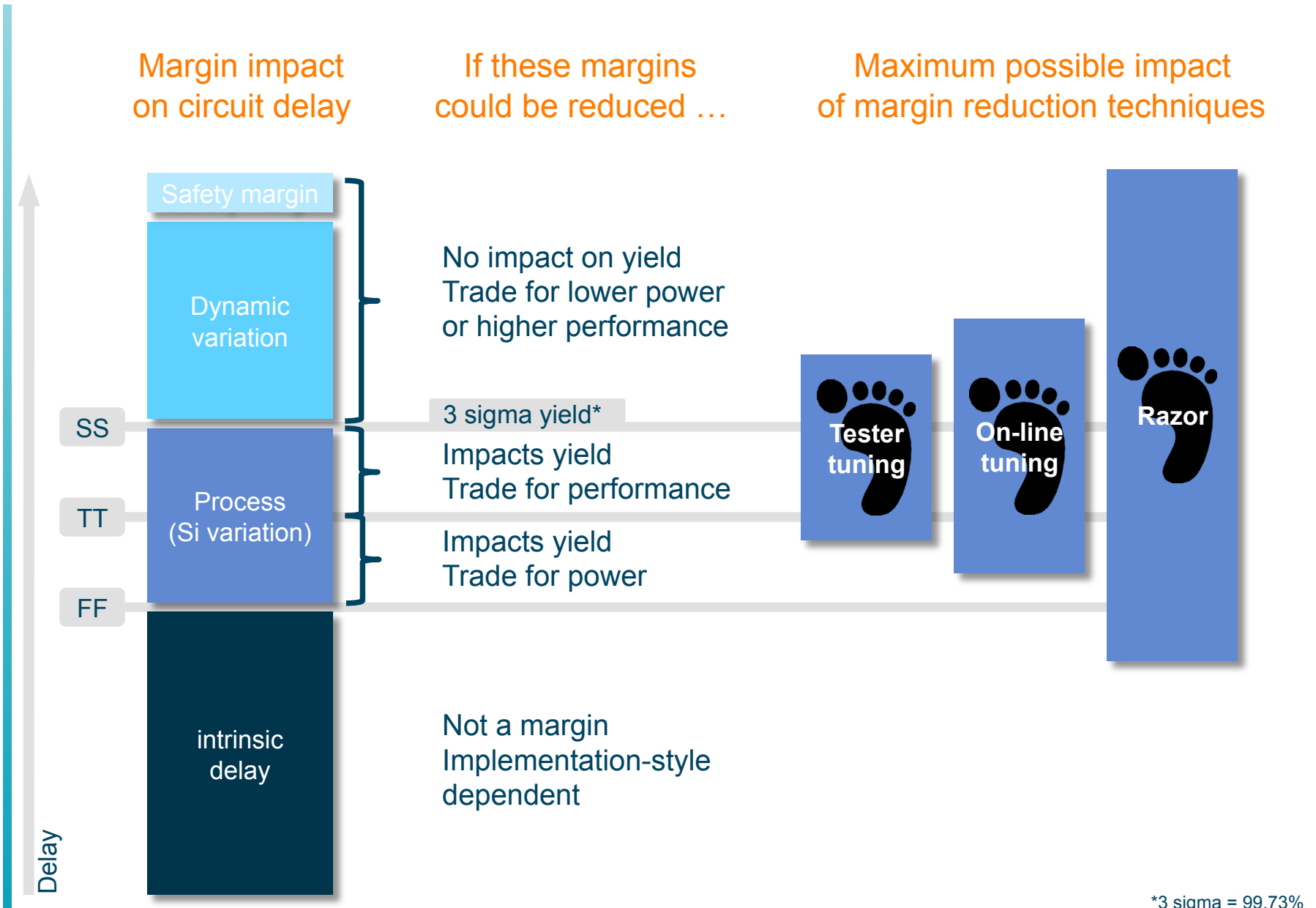


Consider just the impact of margining for 10% voltage drop from external power supply to the internal rails: 20% to 35% frequency

How paranoid should one be?

"There are known knowns. There are things we know that we know. There are known unknowns. That is to say, there are things that we now know we don't know. But there are also unknown unknowns. **There are things we do not know we don't know**."

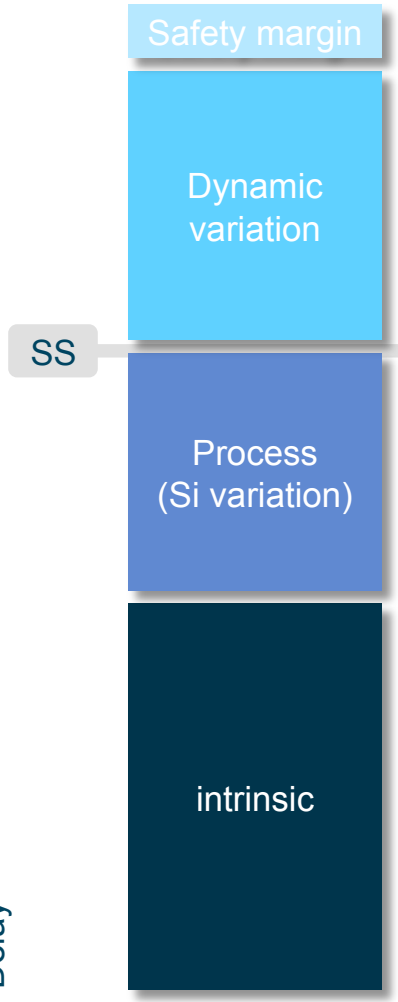
-- Donald Rumsfeld


Margin impact on circuit delay
If these margins could be reduced …
Maximum possible impact of margin reduction techniques
Safety margin
Dynamic variation
SS
Process (Si variation)
TT
FF
intrinsic delay
Delay
No impact on yield
Trade for lower power
or higher performance
3 sigma yield*
Impacts yield
Trade for performance
Impacts yield
Trade for power
Not a margin
Implementation-style dependent
Tester tuning
On-line tuning
Razor

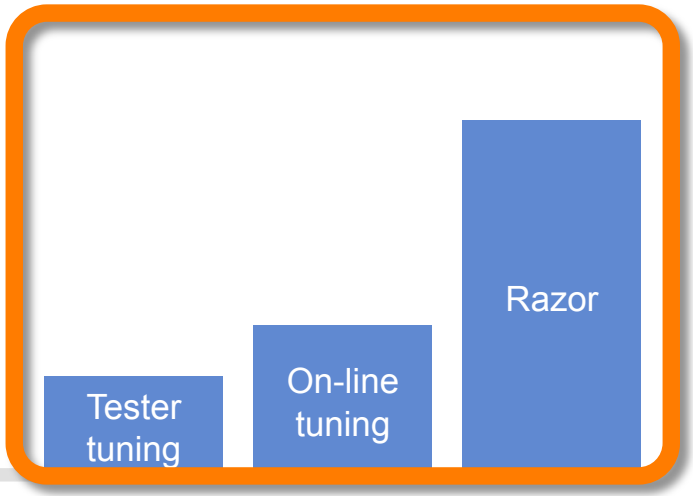
*3 sigma = 99.73%

Conventional ways of reducing margins	Sources of margin overhead	Razor
Tester tuning	Inter-die process variation Intra-die process variation	Eliminates margins
Run-time tuning	IR drop Package and die v_{dd} fluctuation Ambient temperature variation Life-time degradation (NBTI,TDDB) Temperature hot-spots	
	PLL jitter Coupling noise (capacitive and Ldi/dt) Local clock jitter (IR drop in clock tree) Virtual supply transients	Tolerant of fast transients

Margin impact on circuit delay



Focus on 3 sigma yield: benefit on all chips



Frequency increase: **10%** **15%** **40%**

… and Razor gives more than **20%** beyond the best the industry can do (Intel Itanium, IBM Power6)

… and more than **30%** beyond what's cutting-edge in the wireless domain

Assumes equivalent Razored core including intrinsic overheads

*3 sigma = 99.73%

Conclusion

Large energy efficiency and performance gains are possible if we could reduce the margins deployed in designs

Challenge is to do this without compromising design integrity and increasing product cost

Razor offers a way forward by

- Invoking margins on demand only when necessary
- Enabling implementers to avoid margins for the common-case

Fin