

Hardware-software interactions in variability & reliability aware design

REALITY: Reliable and Variability tolerant System-on-a-chip Design in More-Moore Technologies

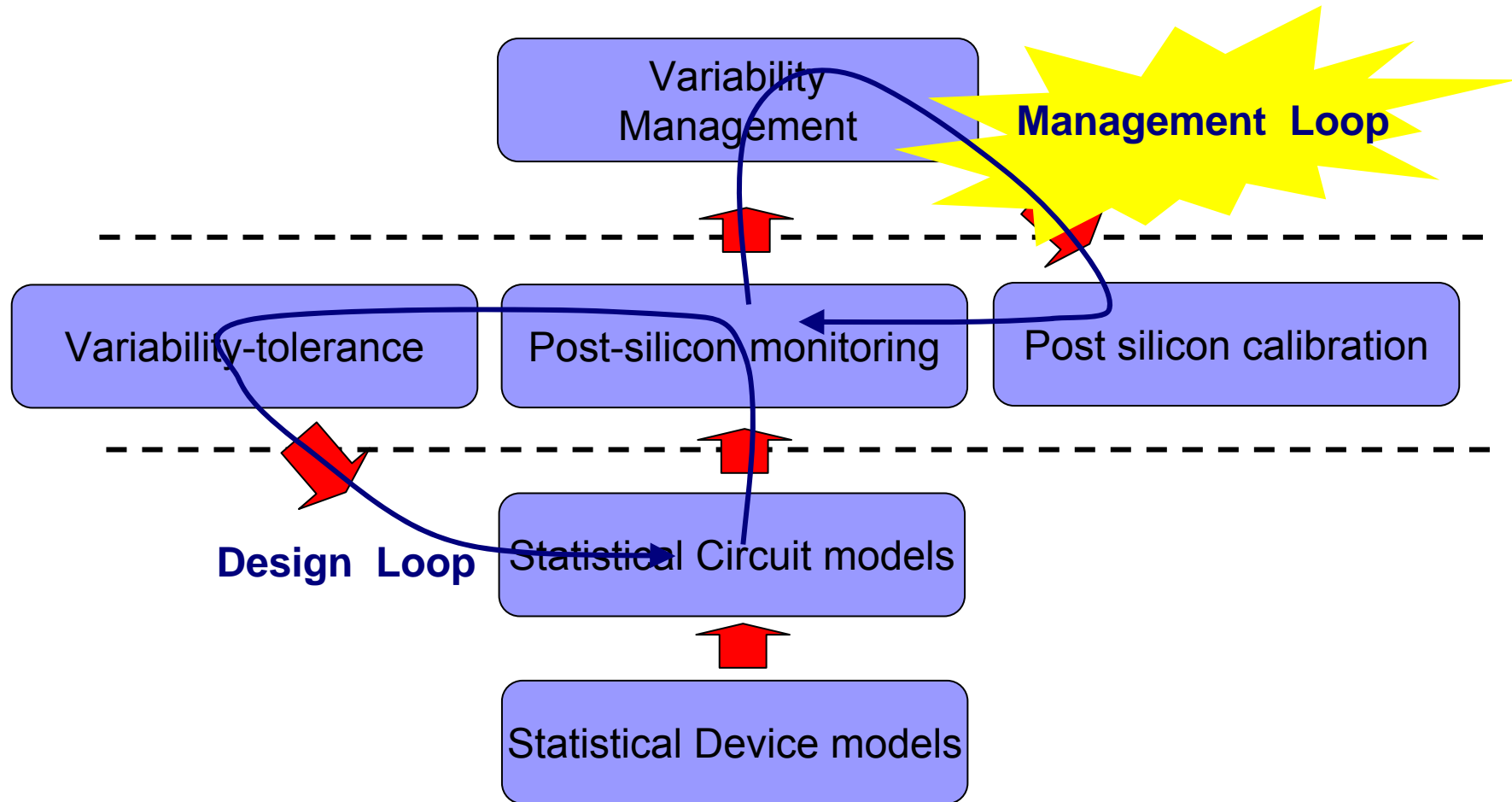
Luca Benini

DEIS Università di Bologna

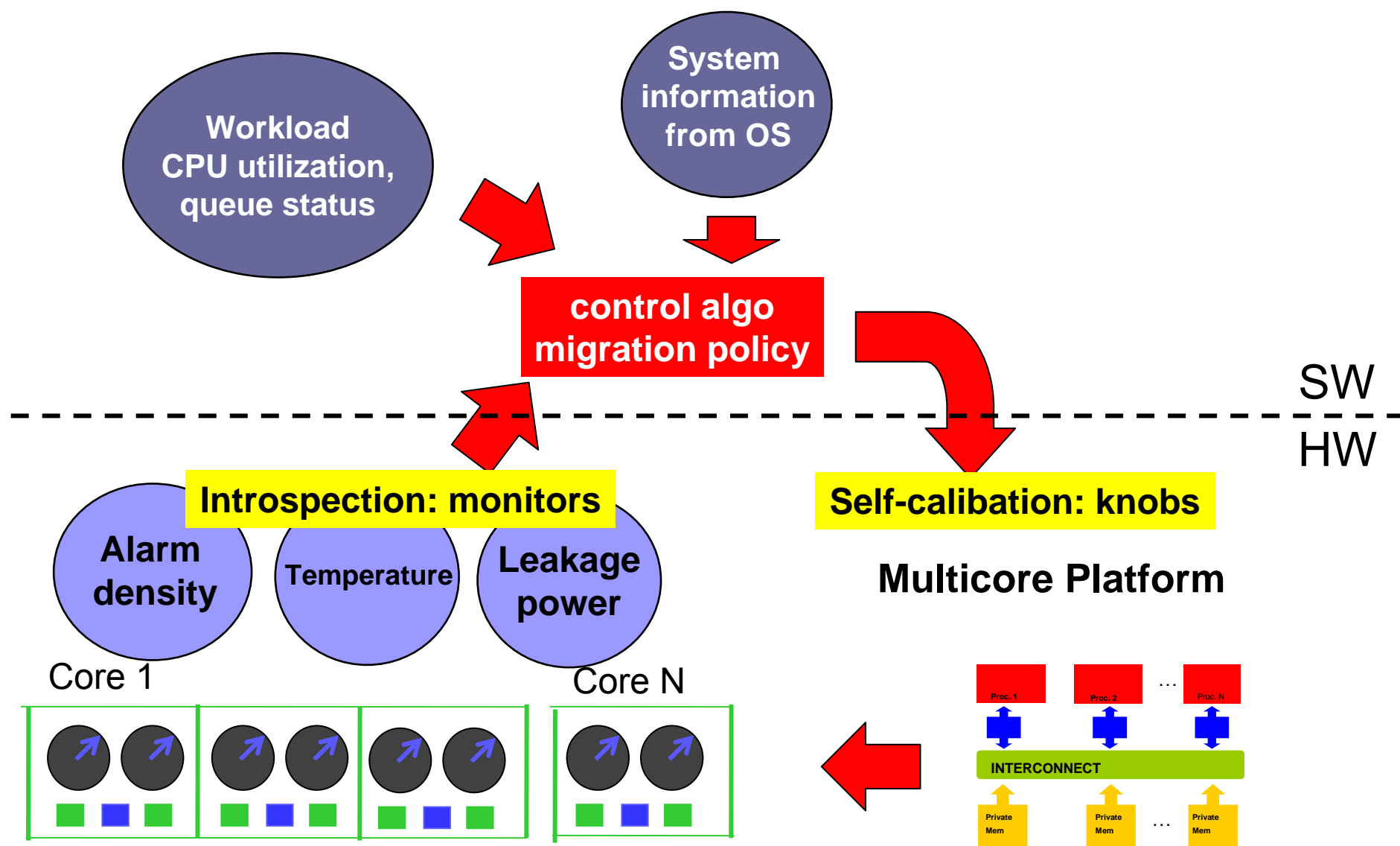
Luca.benini@unibo.it



REALITY: vertically integrated vision



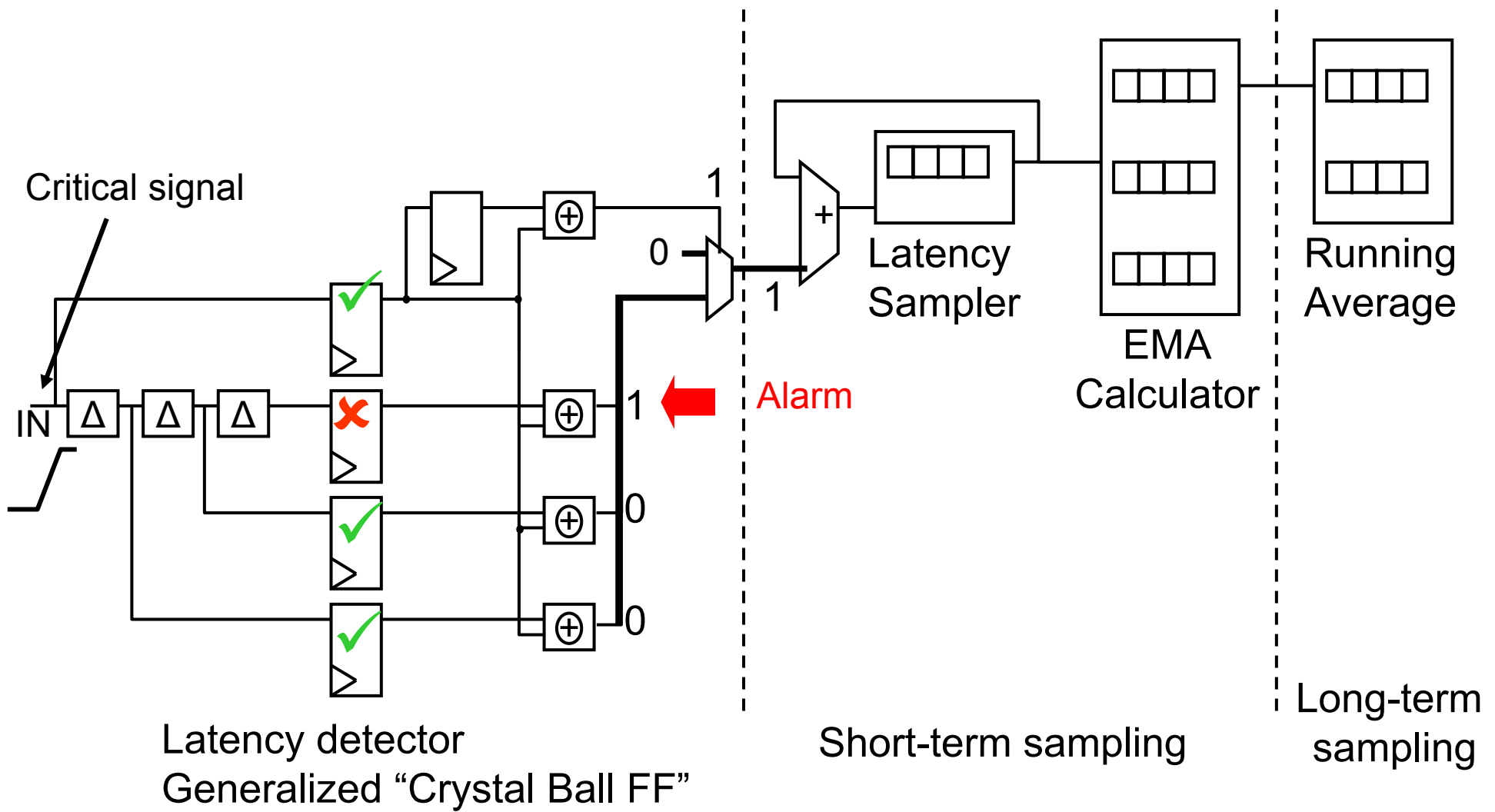
Management Loop: Holistic view



Monitors

- With variations, we know that a die is a sample of a multi-variate statistical distribution
- Where in the distribution? We don't know a priori
- Monitors help in determining this
 - Directly: i.e. measuring critical path
 - Indirectly; i.e. measuring Idd and T
- Monitors are extra hardware, and their cost must be justified and minimized
 - Design-time effort on selecting and placing monitors is required
- Monitors help decreasing uncertainty, but can never eliminate it
 - Cannot monitor everything and/or alyways!

Anatomy of a Monitor (Speed)

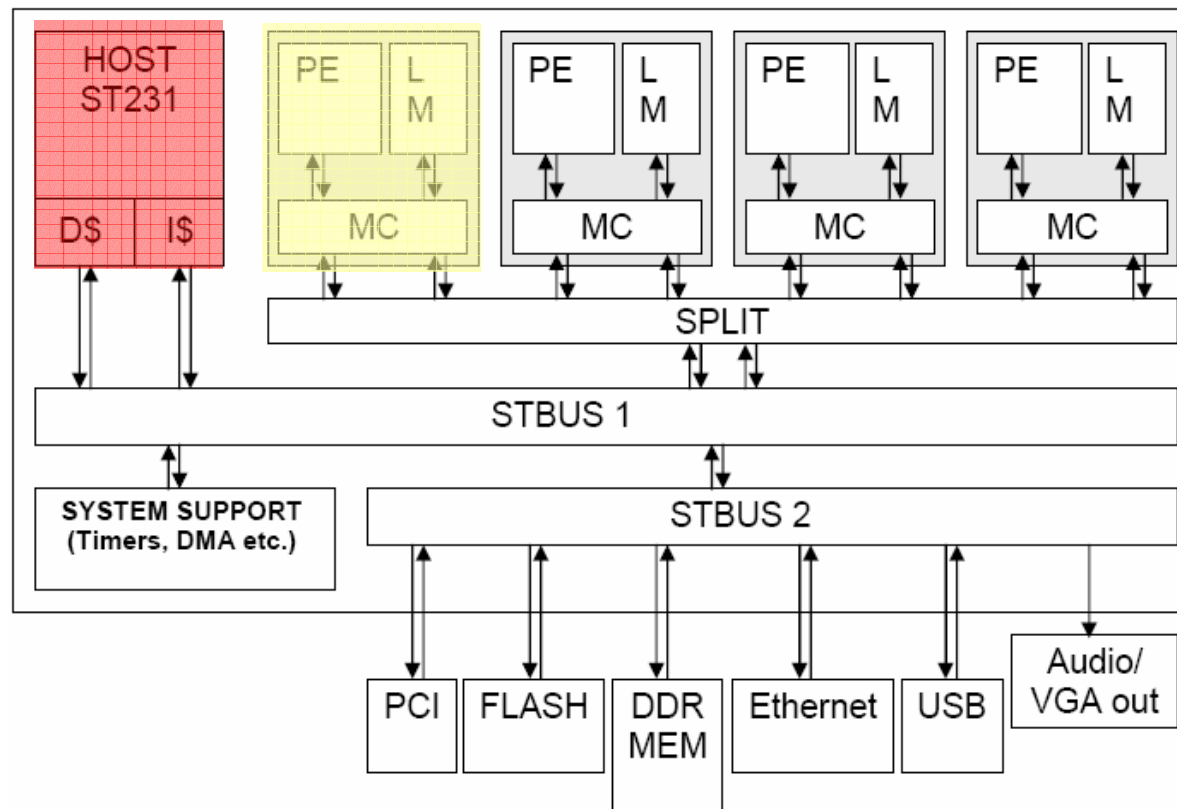


REALITY System-level demonstrator

- Realistic next-generation SoC platform
 - Multiple cores
 - Large amount of on-chip storage
- Advanced technology 45→32 nm
- HW and SW needs to be modeled
 - Introspection
 - Self-calibration
- NOTE: work in progress!

REALITY Variability-tolerant MPSoC

- STM-xStream array



ST231 Host plus N x Processing Engines (PE): PE: Processing Engine, LM: Local Memory (64 KB-128KB), MC: Memory Controller

Software Infrastructure

■ Functionality

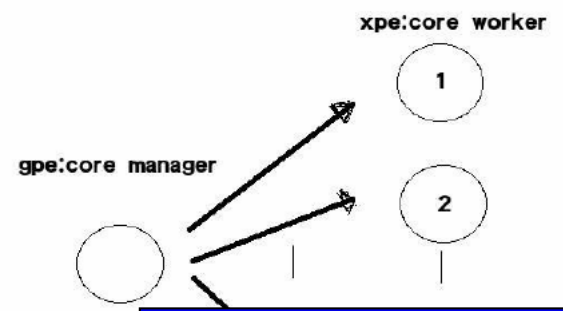
- Task dispatching (scheduling)
- Inter-processor communication library
- Synchronization library
- Task migration library (with checkpointing)

■ Representative workloads

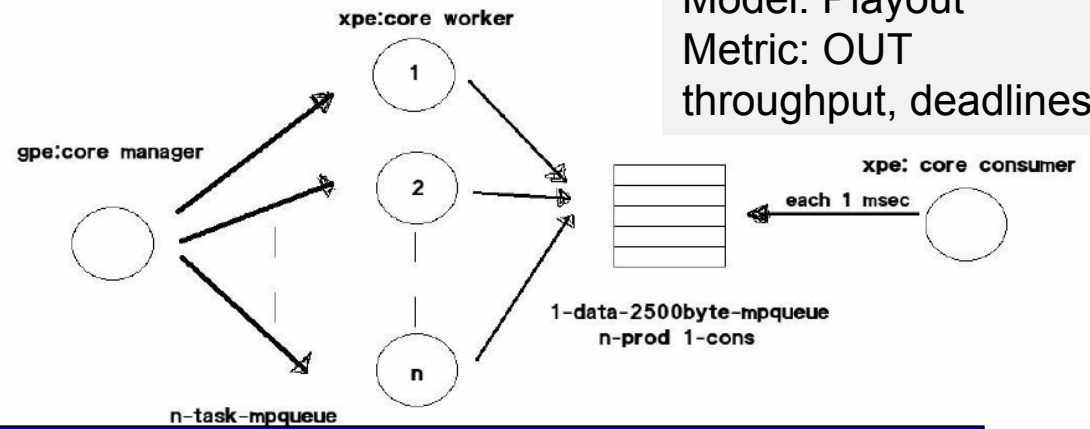
- Independent task model → completion of N tasks
- Playout model → out throughput constraint, deadline misses
- Streaming model → in/out throughput, deadline misses

Workload models

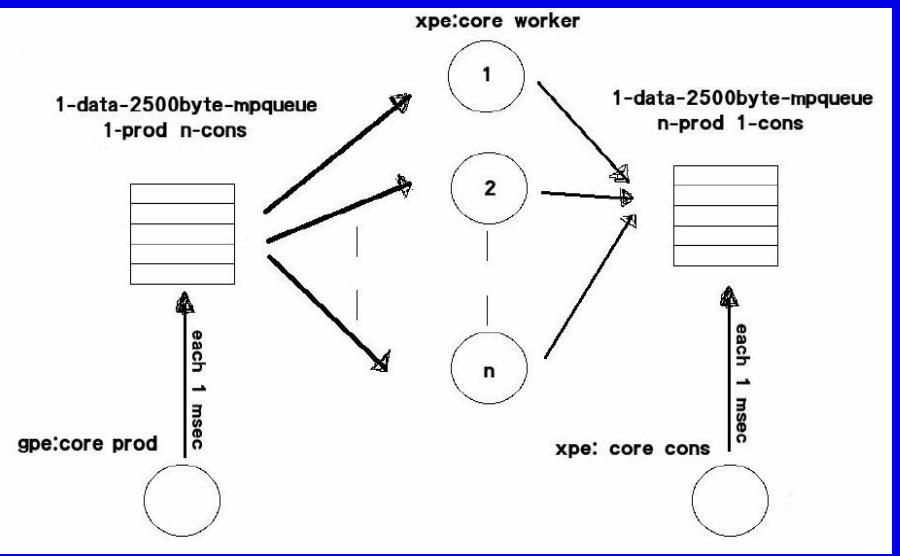
Model: Independent tasks
Metric: Completion of N tasks



Model: Playout
Metric: OUT throughput, deadlines



Model: Streaming
Metric: IN/OUT throughput, deadlines



Hardware Infrastructure


■ Monitors

- Per PE variability monitor: memory-mapped register, gives “alarm count” (AC) → slow edges-per-second
- Monitors are visible to the Host

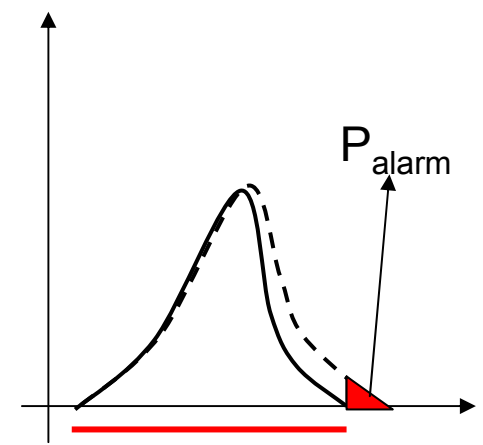
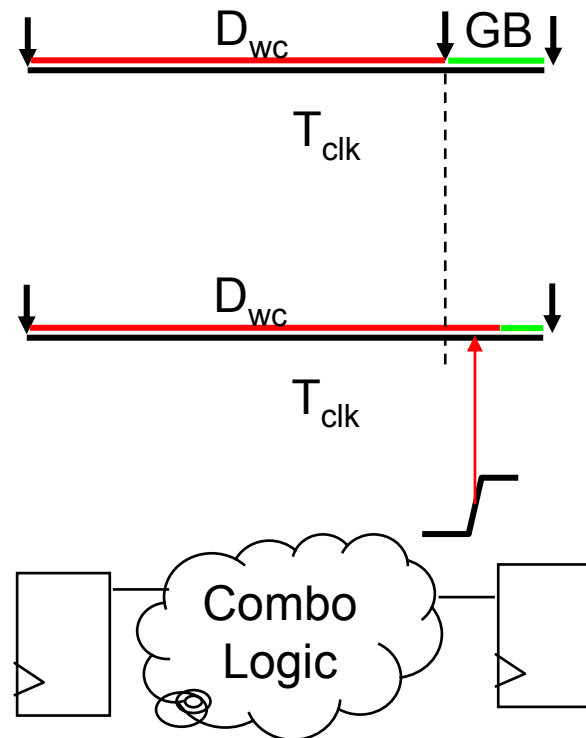
■ Monitoring scenarios

- **Offline**: PEs are run in test mode, AC are collected
- **Online**: ACs are continuously updated and sampled

■ Knobs

- NONE → AC_i is a **reliability index** for PE_i  **SW only compensation!**
- V_B → Leakage power vs. reliability tradeoff
- V_{DD} , f , V_B → Leakage & dynamic power vs. reliability tradeoff

Alarms as reliability indexes

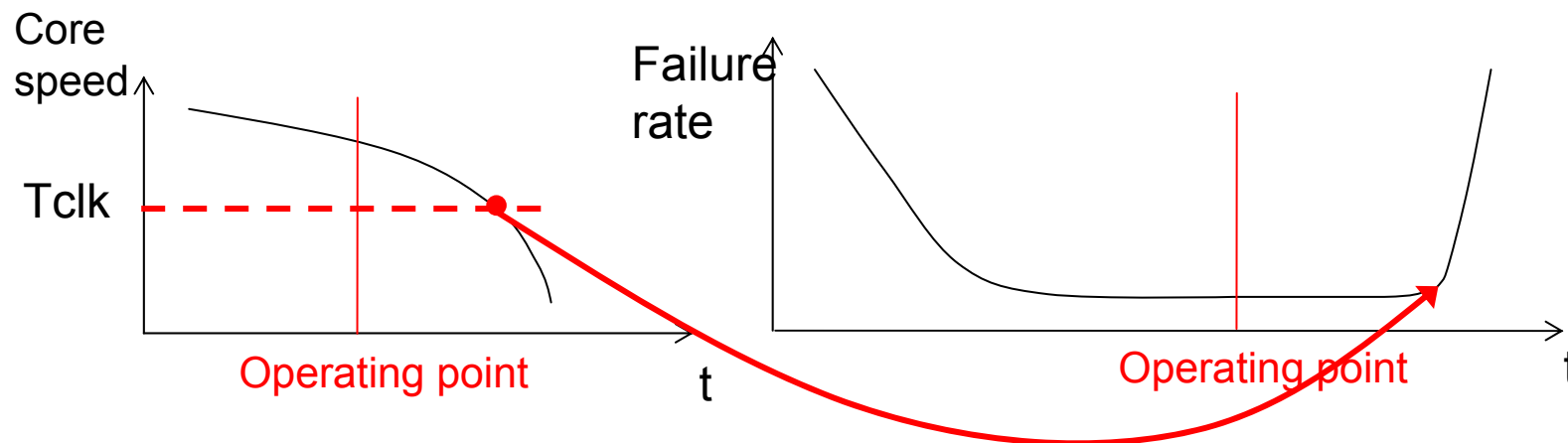


Transitions in GB interval produce alarms

Slow-corner cores produce more alarms and should be used sparingly to reduce failure probability!

Variability over time

- Time-dependent variability (i.e. aging)
 - Cores get slower with time
 - Guardband (GB) guarantees target lifetime for a nominal core

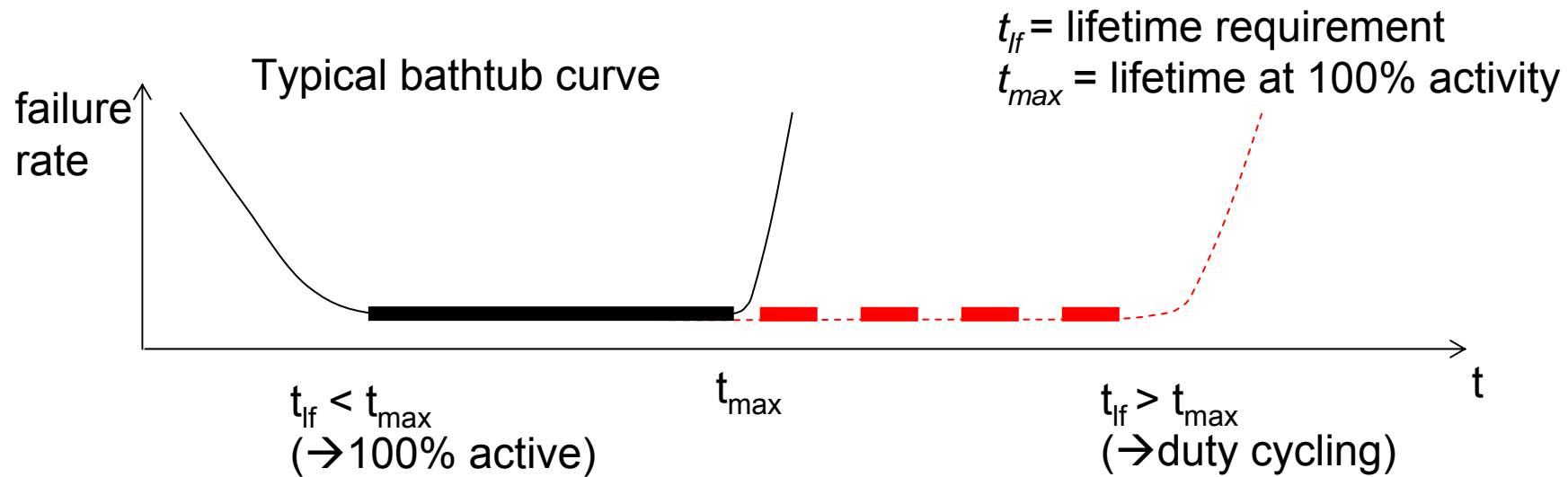


- Process variability reduces GB available for lifetime guarantees

- Extend lifetime by lowering utilization of low-GB cores (idle time can be used for healing actions – eg. power gating)

Extending the lifetime of slow cores

- Tune per-core MTTF
- *Idleness = f(lifetime requirement)*



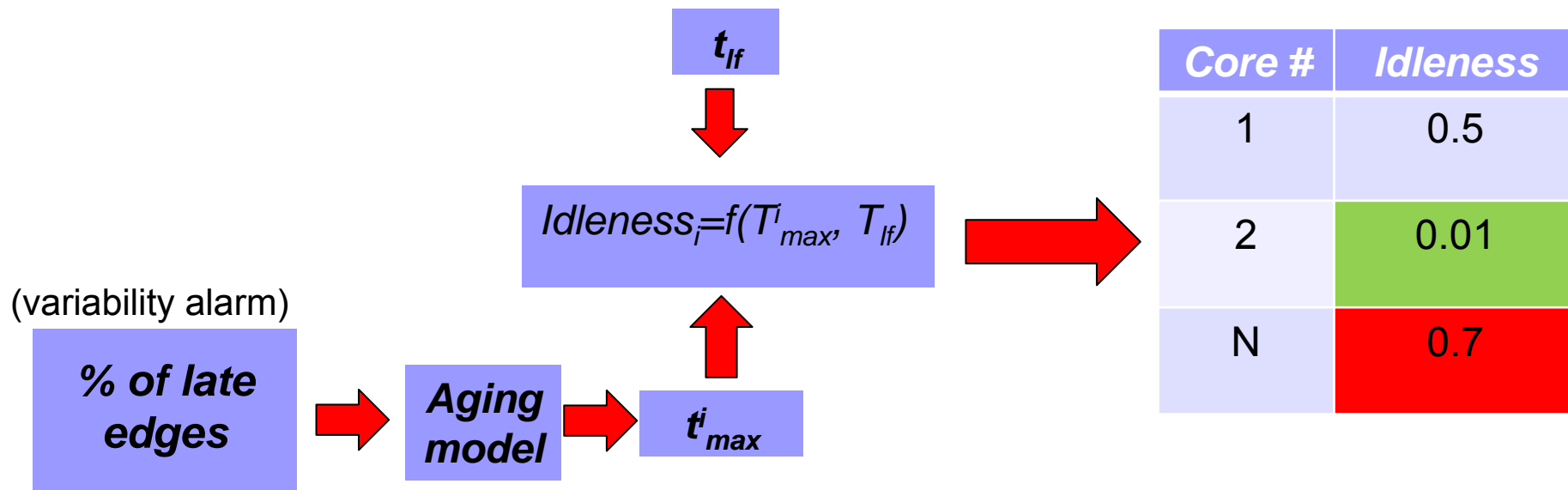
$$Idleness = 1 - (t_{max} / t_{ff}) \quad \text{if } t_{ff} > t_{max}$$

$$Idleness = 0 \quad \text{if } t_{ff} < t_{max}$$

ex: $t_{ff} = 2t_{max} \rightarrow idleness = 0.5$

Software-controlled idle-time distribution

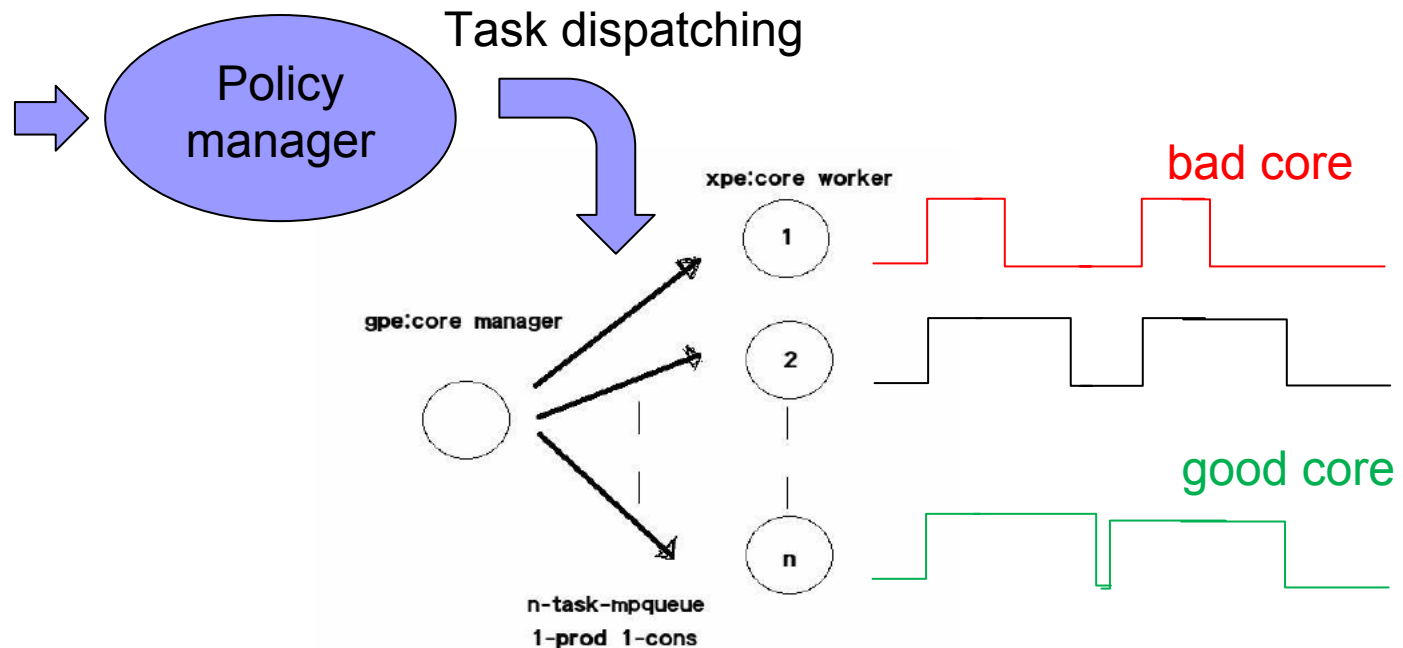
- System lifetime requirement imposes idleness for each core
 - Equalization of t_{max} of each core to target lifetime ($t_{max}^i = t_{max}$, $\forall i$)



Software Strategies: Implementation

■ Task allocation

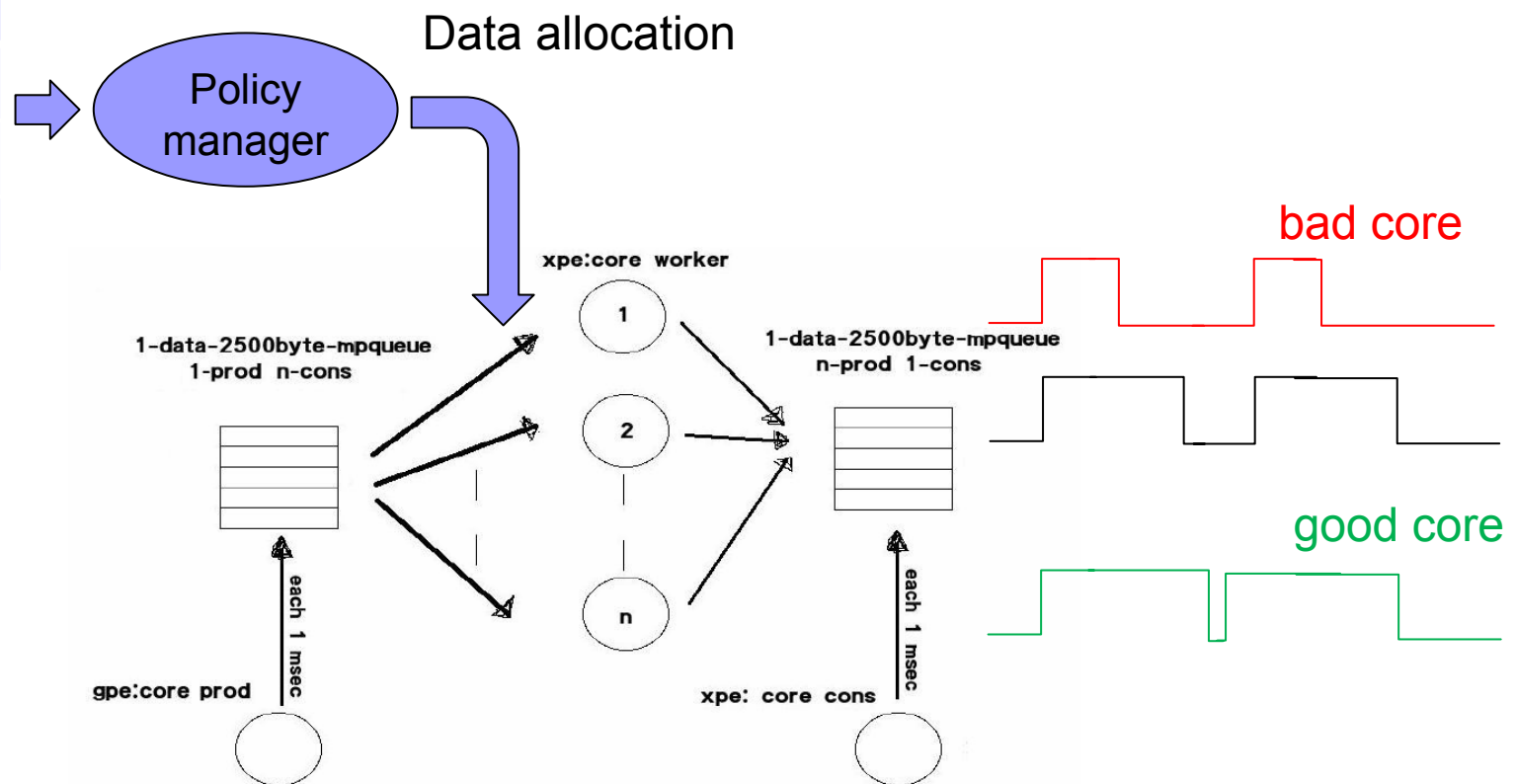
| Core # | Idleness |
|--------|----------|
| 1 | 0.5 |
| 2 | 0.3 |
| N | 0.7 |



Software Strategies: Implementation

■ Data allocation

| Core # | Idleness |
|--------|----------|
| 1 | 0.5 |
| 2 | 0.3 |
| N | 0.7 |



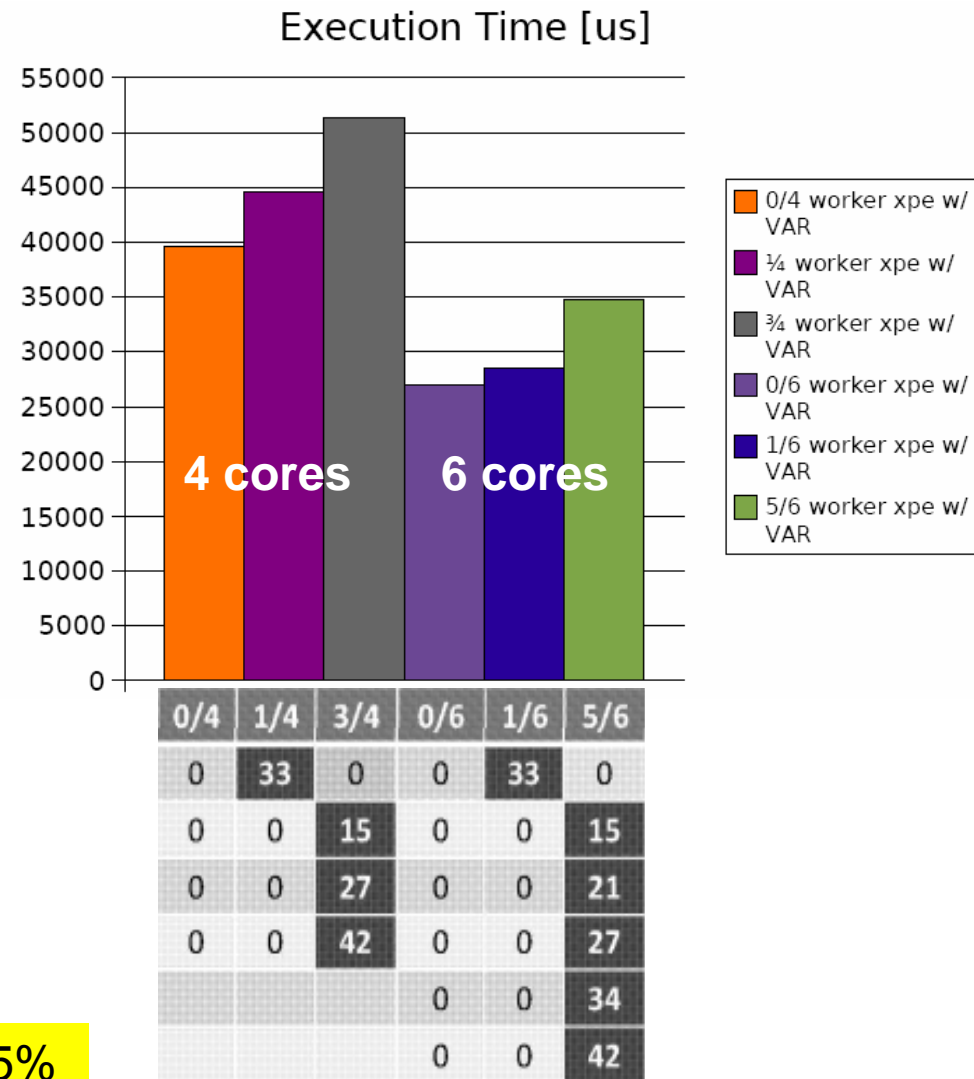
Performance Impact

- Independent task model

Impact on execution time

Idleness distribution

Avg error on idleness distribution < 0.5%



Performance Impact

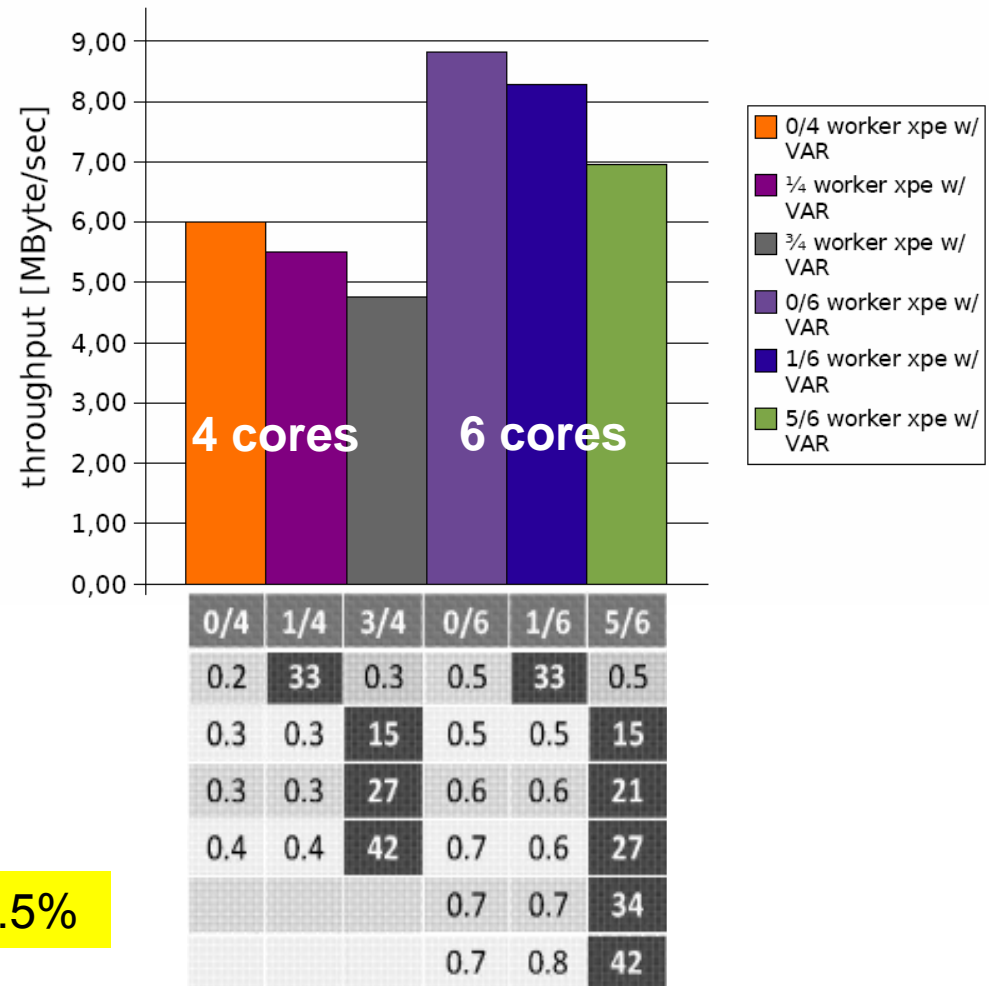
- Playout model

Impact on throughput

Idleness distribution

Avg error on idleness distribution < 0.5%

Expected Throughput 9,54 [MB/s]



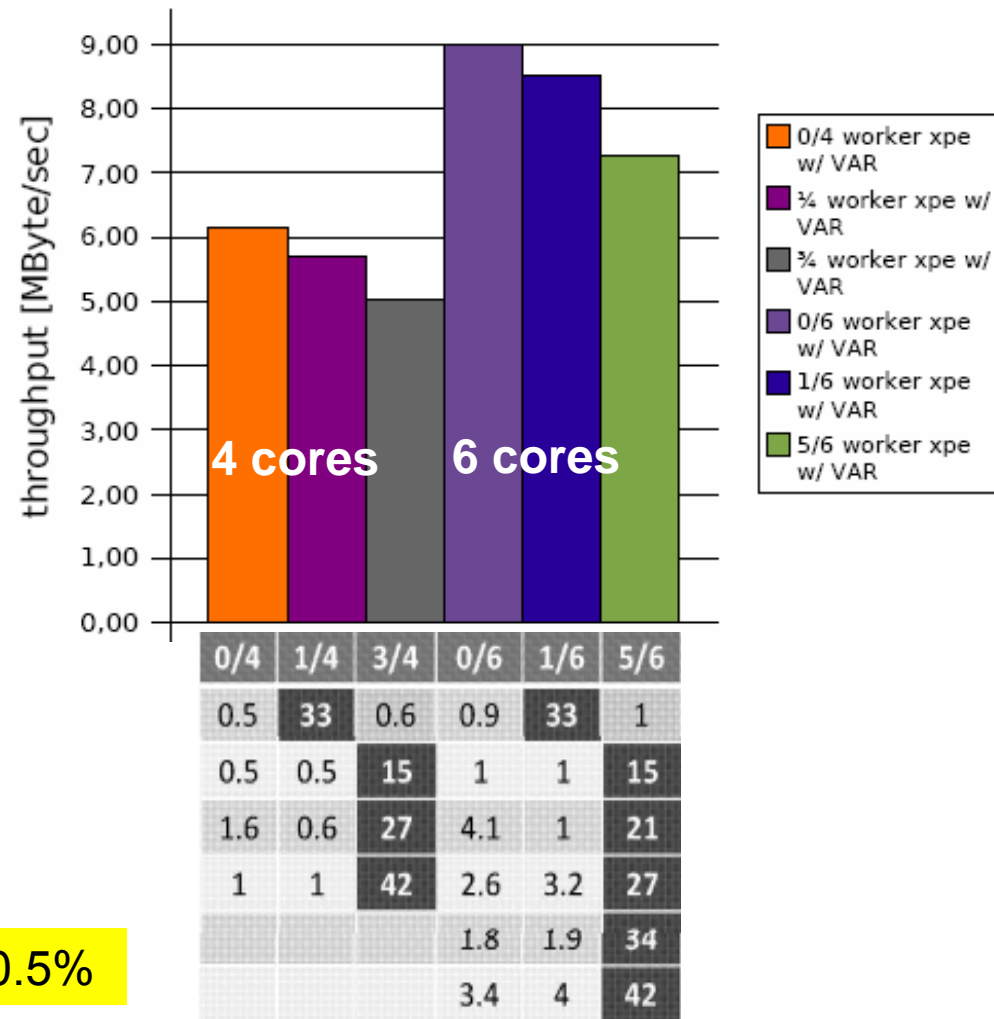
| | 0/4 | 1/4 | 3/4 | 0/6 | 1/6 | 5/6 |
|-----|-----|-----|-----|-----|-----|-----|
| 0.2 | | 33 | 0.3 | 0.5 | 33 | 0.5 |
| 0.3 | 0.3 | | 15 | 0.5 | 0.5 | 15 |
| 0.3 | 0.3 | 0.3 | | 0.6 | 0.6 | 21 |
| 0.4 | 0.4 | 0.4 | 0.4 | | 0.6 | 27 |
| | | | | 0.7 | 0.7 | 34 |
| | | | | 0.7 | 0.8 | 42 |

Performance Impact

- Streaming model

Impact on INPUT throughput

Expected Throughput In 9,54 [MB/s]

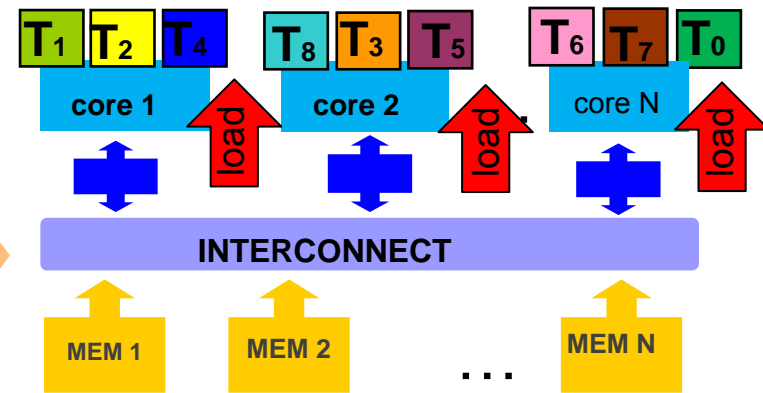
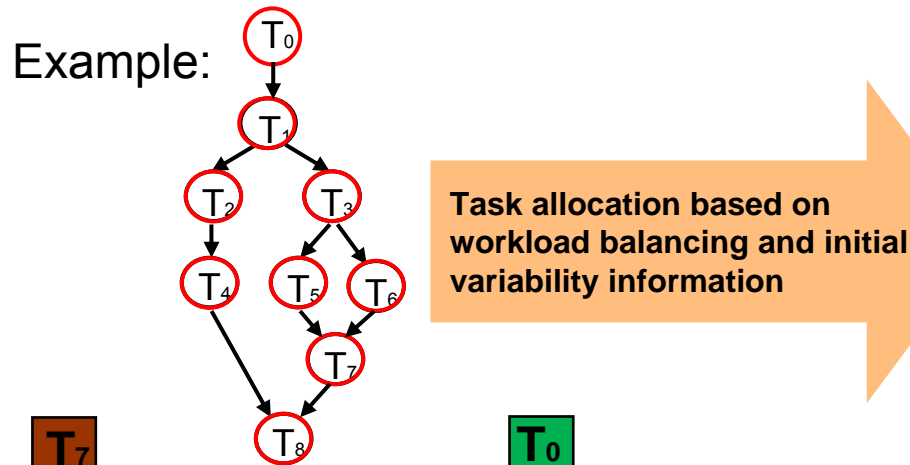


Idleness distribution

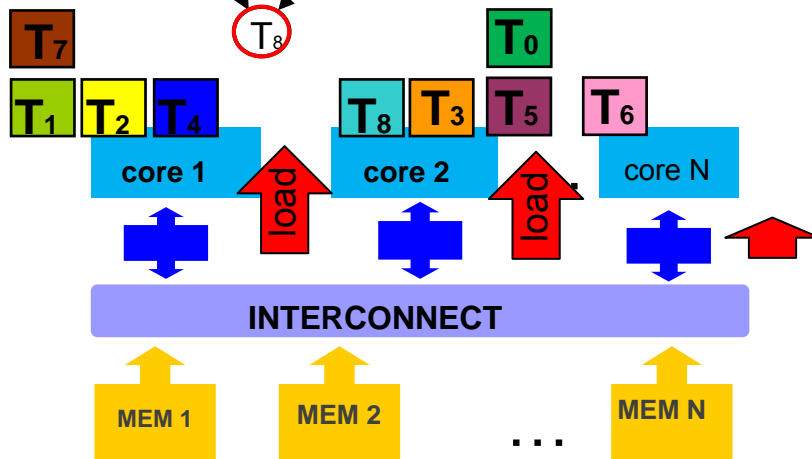
Avg error on idleness distribution < 0.5%

Dealing with static workloads

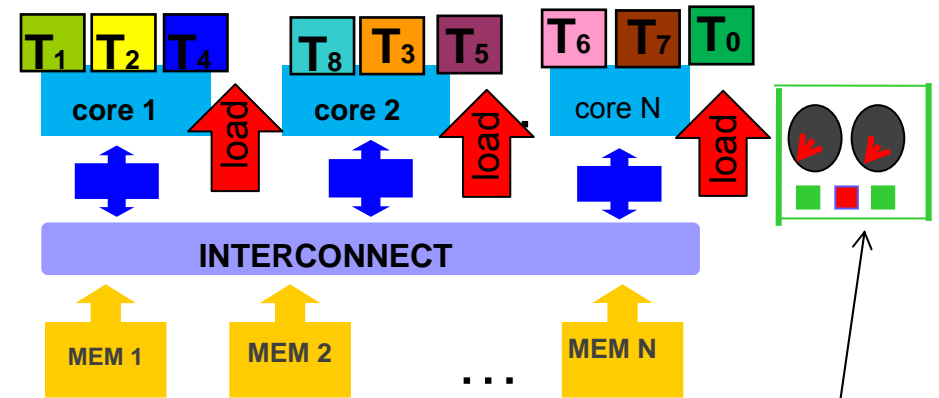
- Migration is required



a) Initial workload balanced allocation



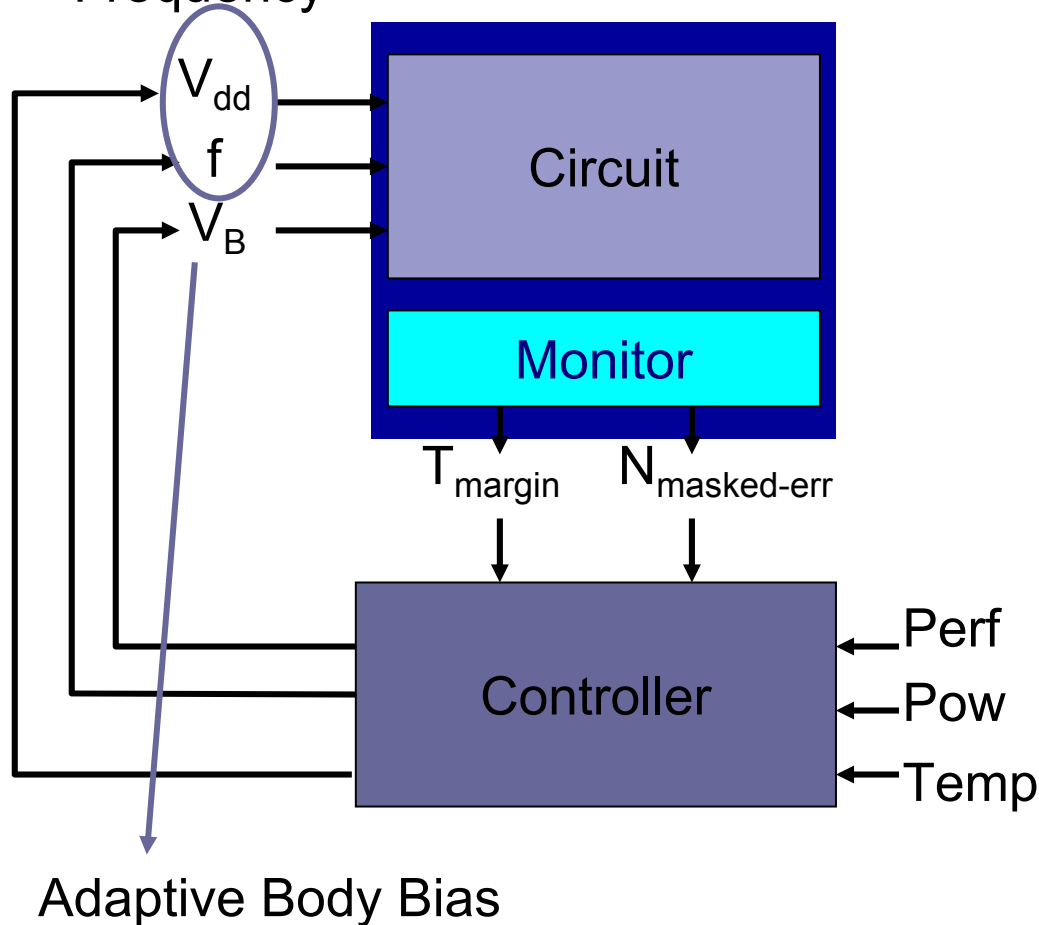
c) Workload migration strategy reduces load on core N to increase its MTTF and re-balance overall system reliability



b) On board monitor signals derating of core N that causes low performance and degrades its MTTF

Calibration techniques

Adaptive Voltage & Frequency

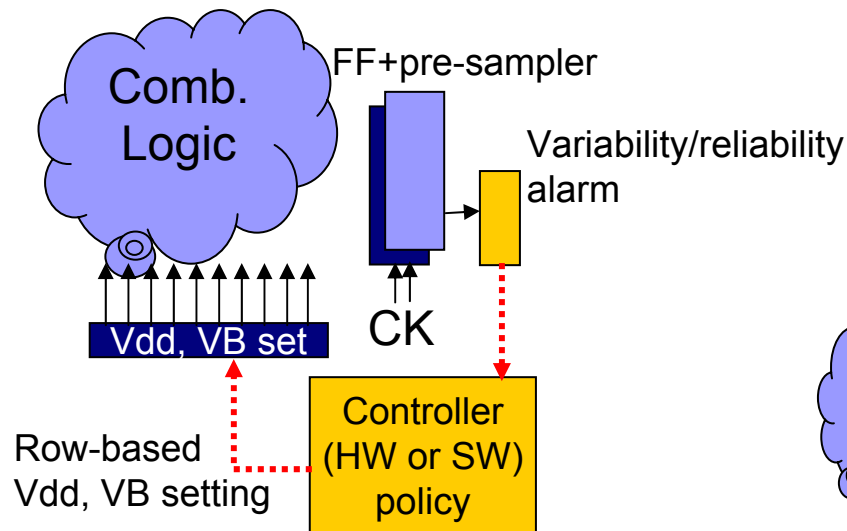


Modes of operation

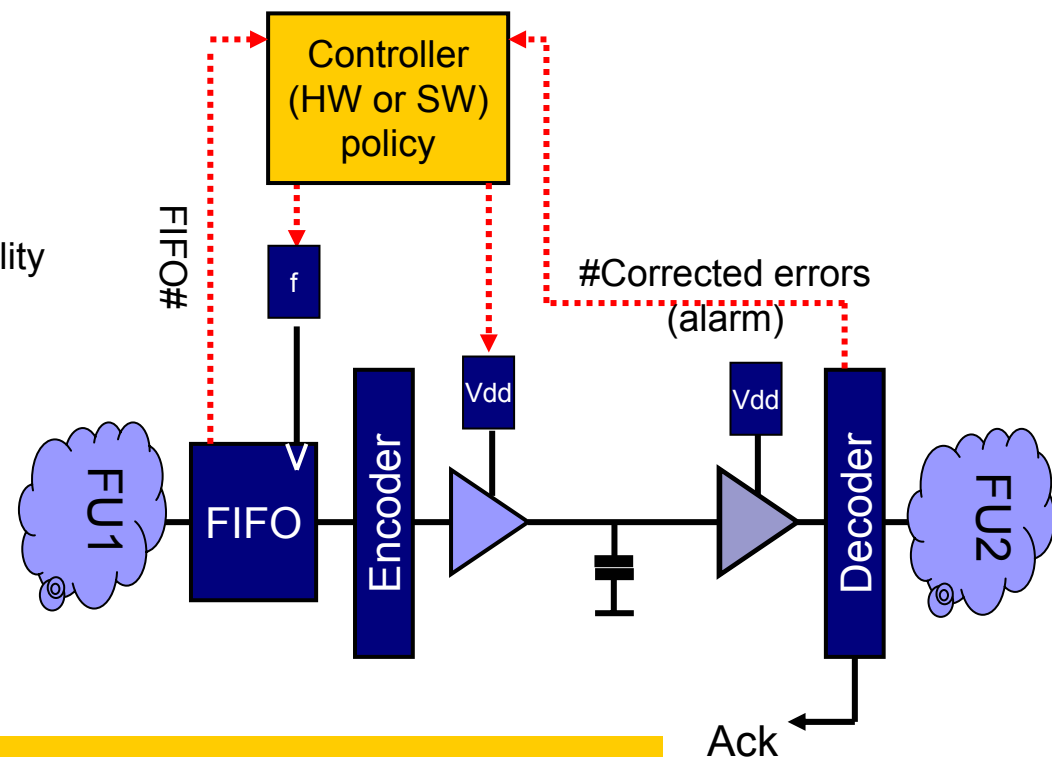
- **Post-fabrication:** regulation is performed once in post-fabrication testing → easiest & lowest overhead but limited adaptation
- **Off-line:** regulation is performed periodically, while system is in “test mode”
- **Online:** monitors and controller operate in parallel with the circuit

Implementation examples

- Reliable+Variation tolerant dpath
 - Controller examines signature and decides (based on policy) how to set V_{dd} , V_B to layout rows, with constraints on power/speed



- Reliable+Variation tolerant link
 - Controller examines signature and decides (based on policy) how to set f and V_{dd} for TX, RX, with constraints on power/speed

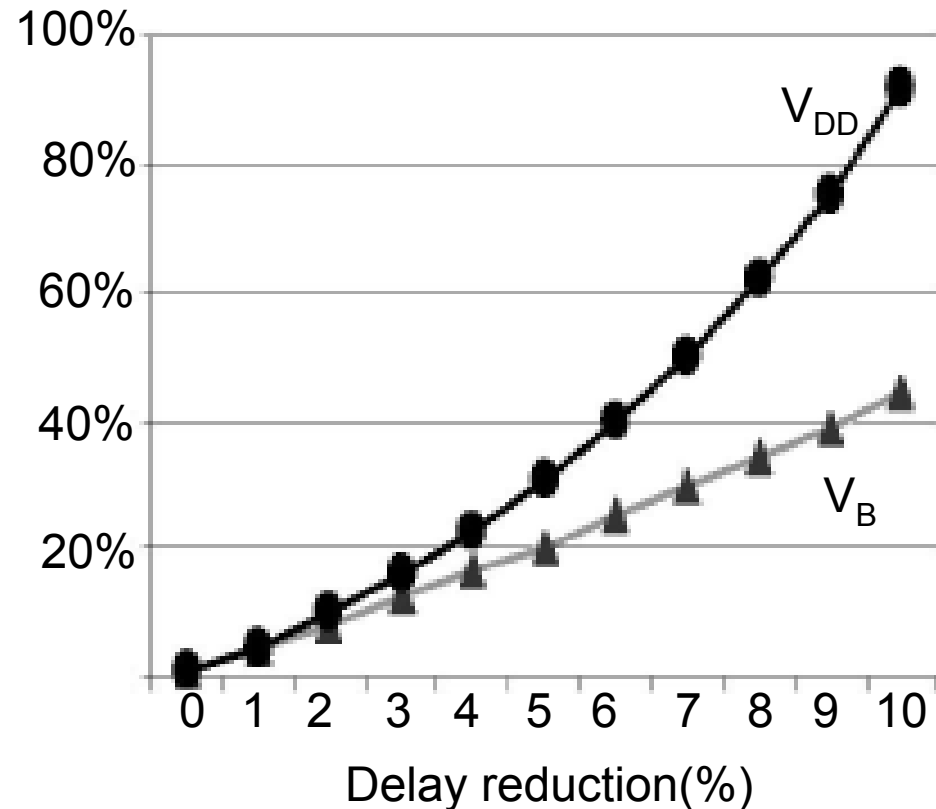


Critical issue: minimize power cost of calibration

The cost of Calibration

- Considering two techniques for datapath variability compensation:
 - AVS (Adaptive Voltage Supply)
 - ABB (Adaptive Body Bias)
- Note: speedup has a significant (static/dynamic) power price

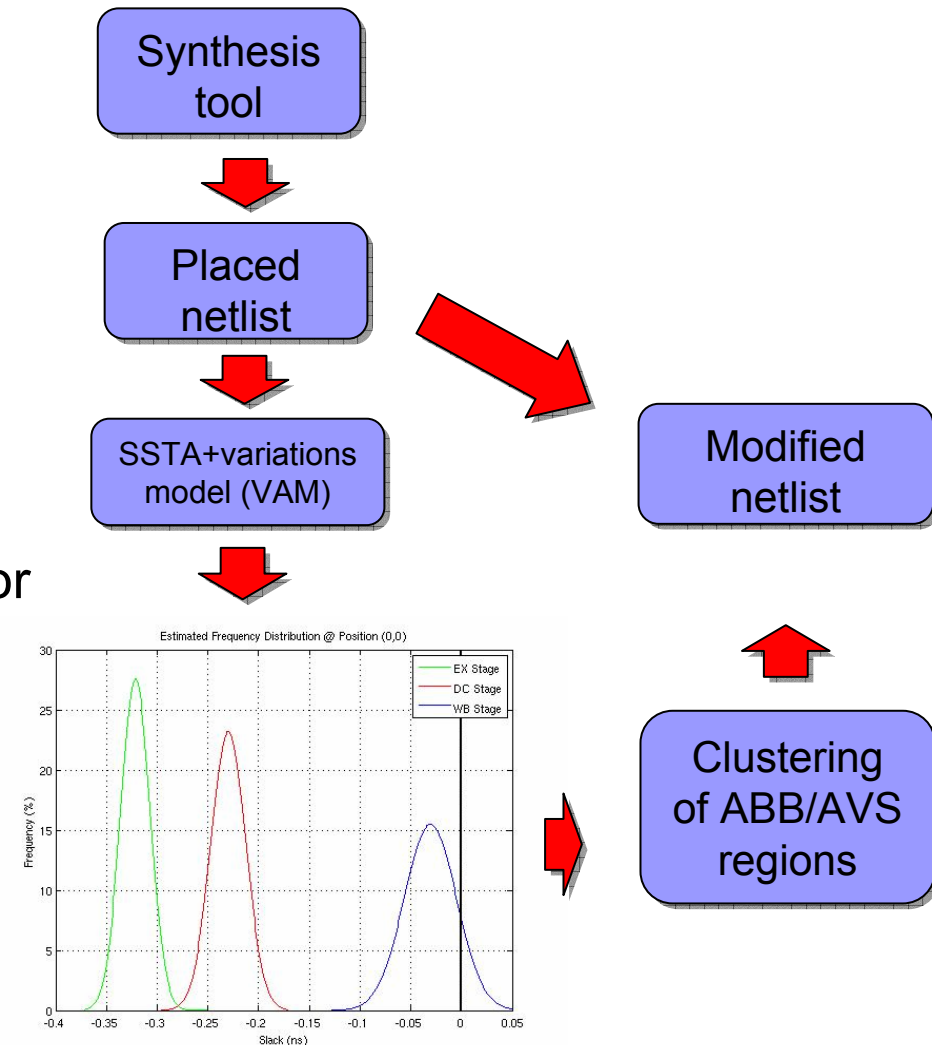
Power increase



Minimizing calibration overhead

REALITY design flow

- Integrated into a standard synthesis flow
- Take VAM data as input
 - Identifies critical gates
- Works on placed netlist:
 - No modification of placement for performance
 - Clusters the design ABB/AVS applied on critical clusters
- Minimizes overhead for a desired calibration range
 - **30% less leakage for a 10% calibration range on delay (ABB)**



Conclusions & Outlook

- Variation-tolerant systems
 - HW monitoring
 - HW/SW calibration
- Strategic for multicore in 45nm and beyond
- Ongoing work
 - Minimizing HW calibration overhead
 - Optimal synergy between HW and SW variation tolerance
 - Integrating with memory strategies (not covered here)
- Final thoughts
 - Redundancy will play an increasingly important role
 - Monitoring will become increasingly distributed and will require a structured approach
 - Design technology support is critical!