

REALITY

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

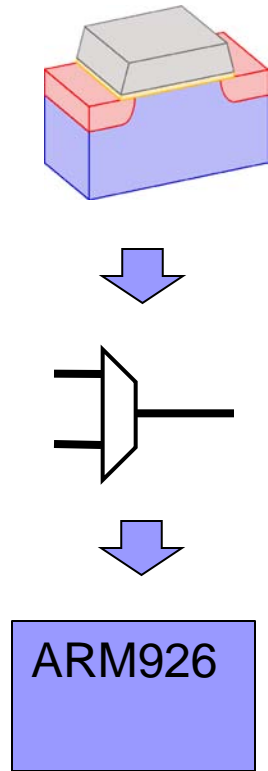
Project Workshop
System Design for Variability
UniBO Contribution
Andrea Acquaviva



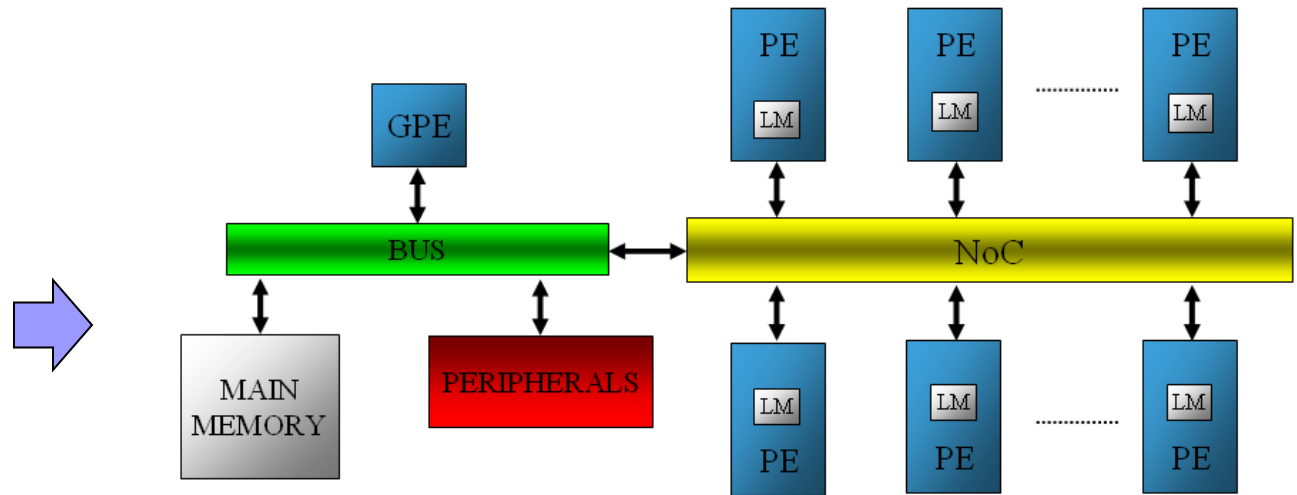
ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA



From transistor to system



- Architectural level compensation (e.g. AVS, ABB) :
variability -> failure -> diversity
- No compensation:
variability -> diversity



With or without compensation, variability leads to heterogeneity at system level

Architectural level countermeasures

■ Summary of work

- Characterization of variability impact at architectural level
- Development of AVS and ABB techniques for datapath and interconnects

■ Summary of findings

- In datapath, **ABB is more effective than AVS in providing speed-up without impacting leakage**
- In interconnect, **ABB is more effective, especially in full swing links**

AVS vs FBB Techniques

- We first explored Adaptive Voltage Supply (AVS) techniques and performed the implementation of a “Placement aware multiple voltage island design technique”.
- Then we compared with adaptive body biasing (ABB) techniques.
- We concluded that ABB overcomes AVS because of the better trade-off between overhead (for AVS is due to level shifters) and performance
- We then concentrated on ABB techniques as reported in the next slides.

Data path and controller variability

- How to **Predict** and **Compensate** circuit slow down
- Approaches:
 - Pre-design statistical optimization
 - Select optimum design time parameters to maximize timing yield
 - Can result in over design → sacrifice performance or power
 - Post fabrication, some chips might still fail
 - Post fabrication tuning
 - Insert timing sensors and alarms inside the chip
 - Detect circuit slowdown and compensate at run-time

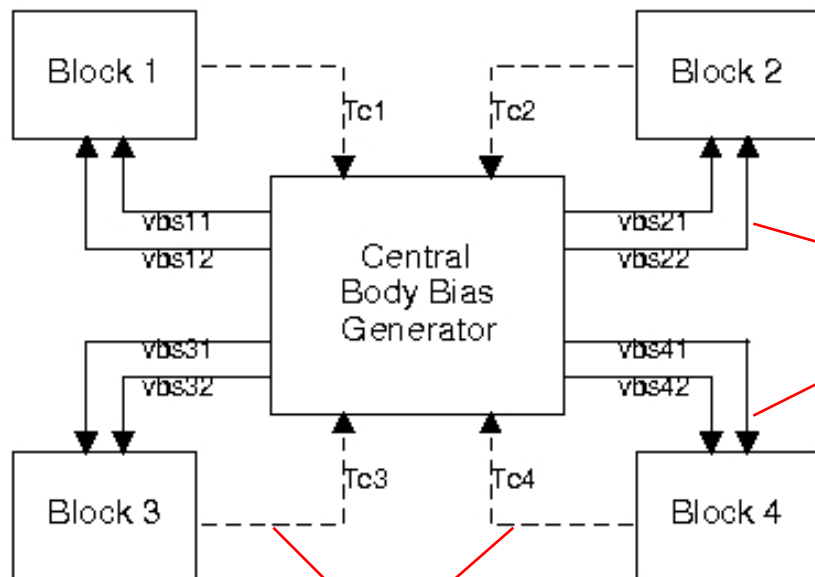
How to do run-time dynamic circuit slowdown compensation ?

- Use Adaptive Body Bias (ABB)
 - Sense circuit slowdown with timing alarms
 - Speed-up the slow circuit blocks using FBB
- Proposed methodology
 - We don't speed-up the entire block or chip but just (on the rows that contain) the most timing critical gates
 - We incur very low leakage power overhead

Variability compensation methodology

■ Compensation methodology

We activate the logic to compensate for a given amount of path slowdown



2 optimal body bias voltages

Timing Alarms

Using two *vbs* lines we can have three subsets (clusters) of rows at different *vbs*:

- no body bias
- *vbs*1
- *vbs*2

Novel Algorithms for FBB allocation

- Compensation done at **row level** granularity
 - Layout style allows access to each row for applying FBB

- Problem:

Given a percentage slowdown compensation required, find the row subsets (clusters) with optimum Body Bias Voltages to speed up the timing critical gates at the least expense of leakage power.

Results

Benchmark	#gates	Energy savings (Optimal)	Energy savings (Heuristic)
C1355	439	11.76	11.76
C3540	842	23.08	11.54
C5315	1308	21.43	16.67
C7552	1666	19.05	17.46
Adder_128bits	2026	26.57	23.08
C6288	2740	4.6	3.45
industrial	23898	-	15.67

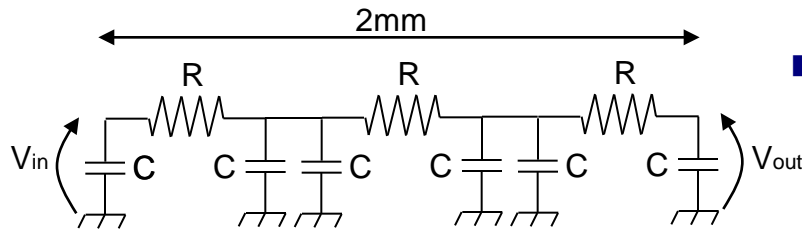
ISCA 85
and
ISCA89

ST SoC

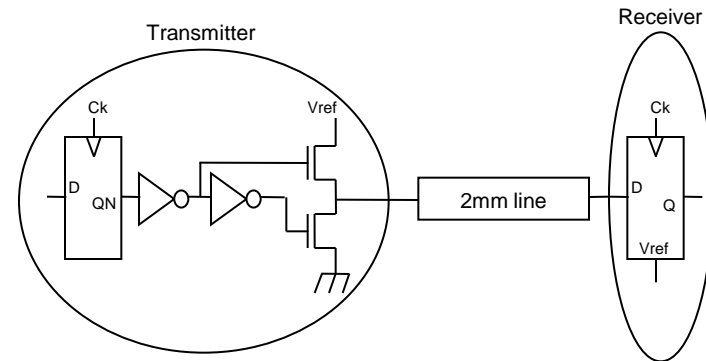
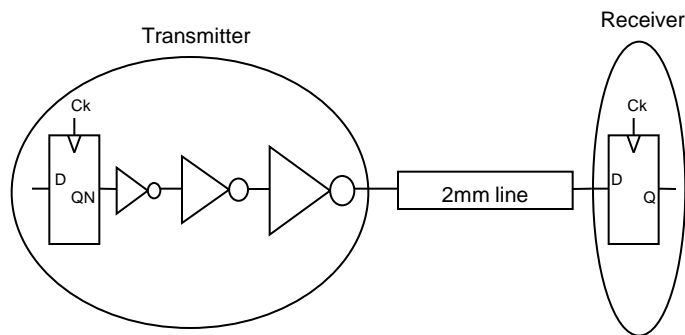
45nm ST technology library

Energy savings are w.r.t. single Body Bias technique

Interconnect: Link modeling



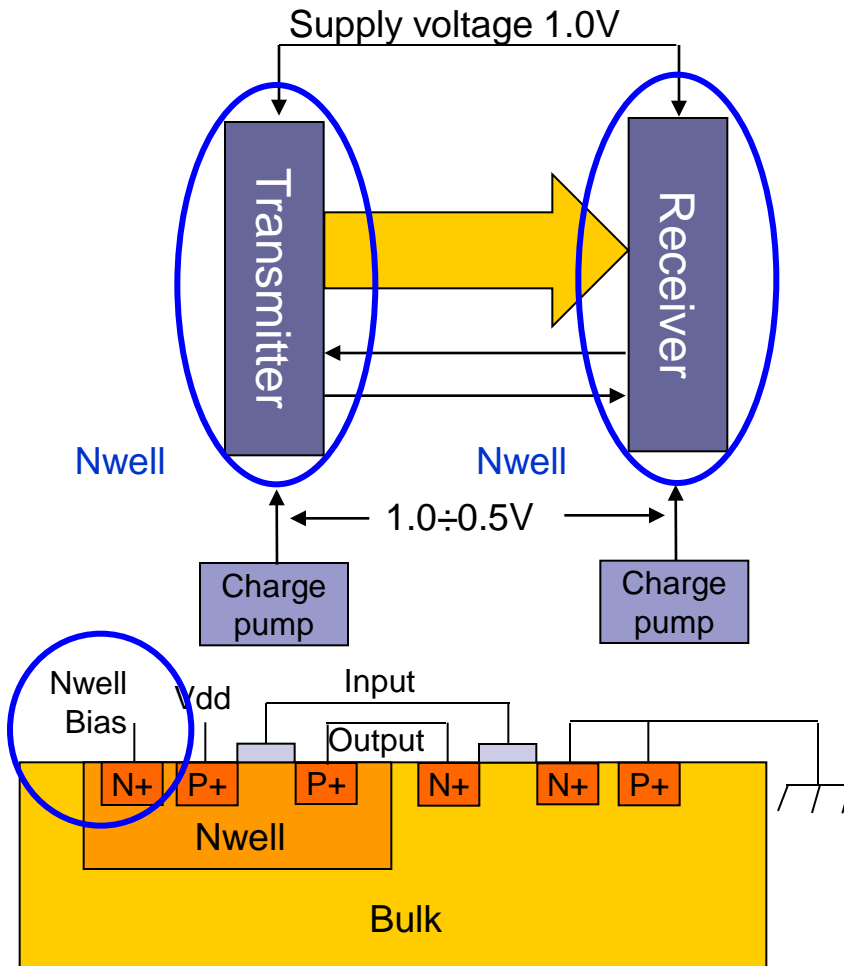
- 2mm line model:
 - RC value from PTM website
 - Intermediate connection case
 - 65nm technology



- State-of-the-art solution
 - High noise margin
 - High power consumption
- More intrinsically sensitive to var

- Low power consumption
 - Same performance
 - Low noise margin
- Less intrinsically sensitive to var

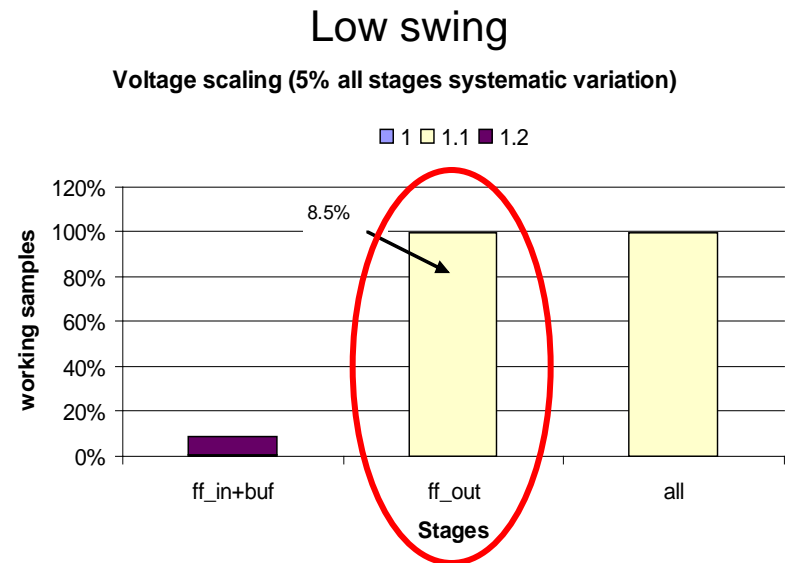
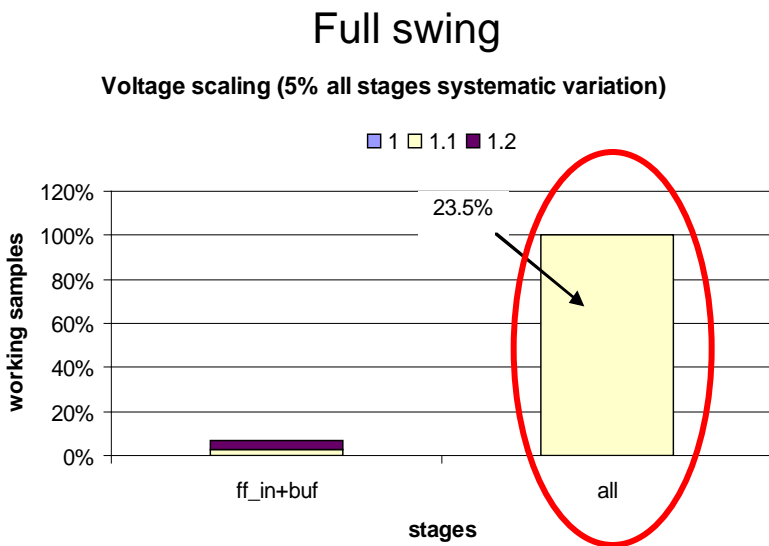
Adaptive Body Bias (ABB)



■ Adaptive Body Bias

- Nwell
 - It is divisible in islands
- Nwell body bias
 - Charge pump decreases the bias voltage from 1.0 to 0.5V with 0.1V step
 - Lower threshold voltage
 - Higher speed
 - Higher leakage
- Low knobbing capacitance
- Low power cost

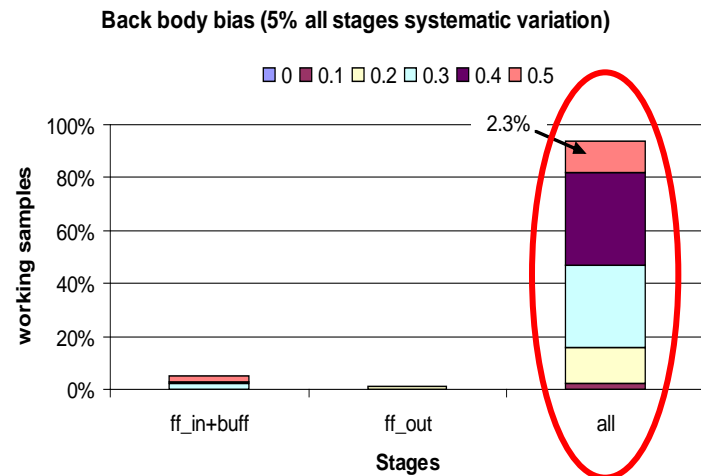
ASV efficiency



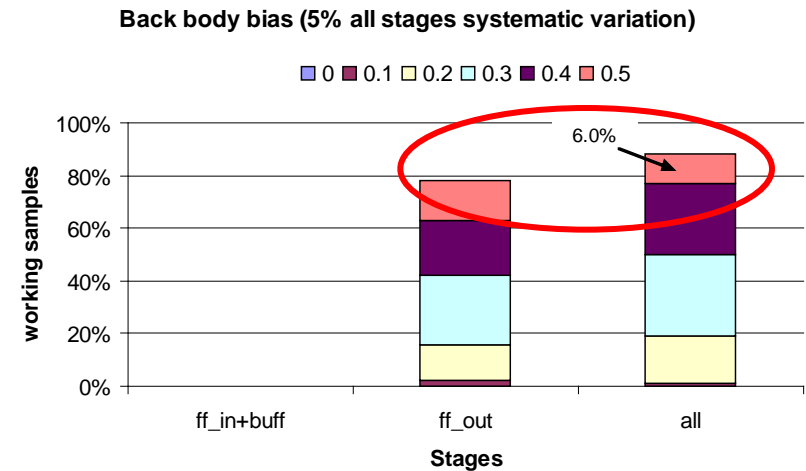
- In the full swing link
 - The ASV restores the nominal performance acting to the whole link requiring 23.5% extra power
- In the low swing link
 - The ASV restores the nominal performance acting only to the receiver requiring 8.5% extra power

ABB efficiency

Full swing



Low swing



■ Full swing link

- The ABB restores the nominal performance acting to the whole link requiring only 2.4% extra power
- With higher random variation the ABB does not restore the nominal performance of the all samples

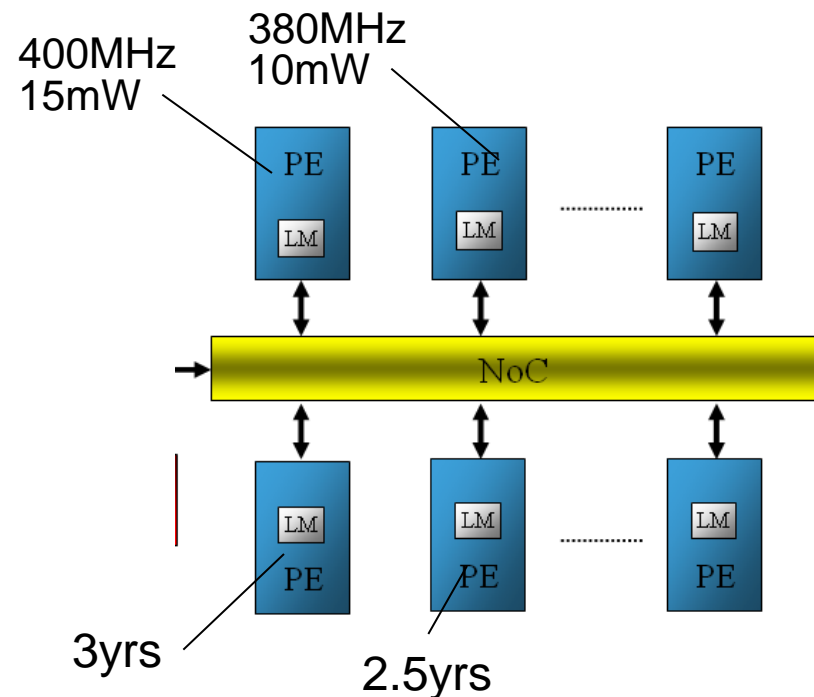
■ Low swing link

- The ABB does not restore the nominal performance of the all samples
- Acting to the receiver provides almost the same performance than acting to the whole link

Heterogeneity

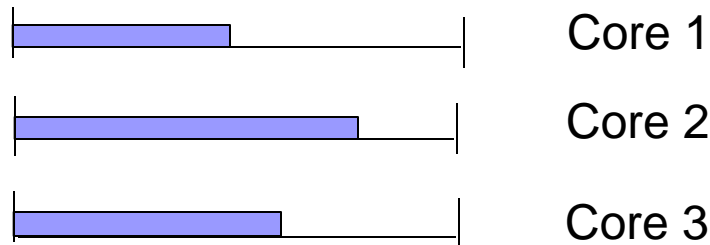
- In a multicore, variability leads to cores having different:

- Delay critical path
- Clock frequency
- Dynamic energy
- Leakage energy
- Reliability



Platform scenarios (1)

Single frequency domain

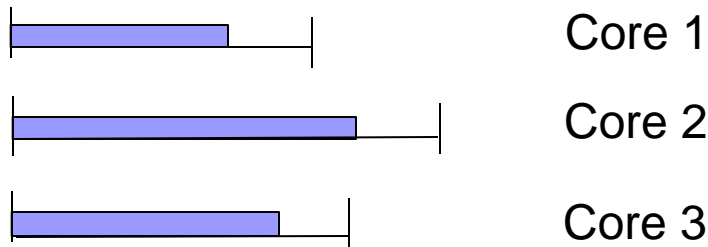


Compensation using guard-bands
Equalization of clock frequencies to the lowest

- Equalize core speed
- Slower platform
- No need for OS/app awareness
- Energy diversity remains
- Reliability dictated by the slowest core

Platform scenarios (2)

Multiple frequency domains

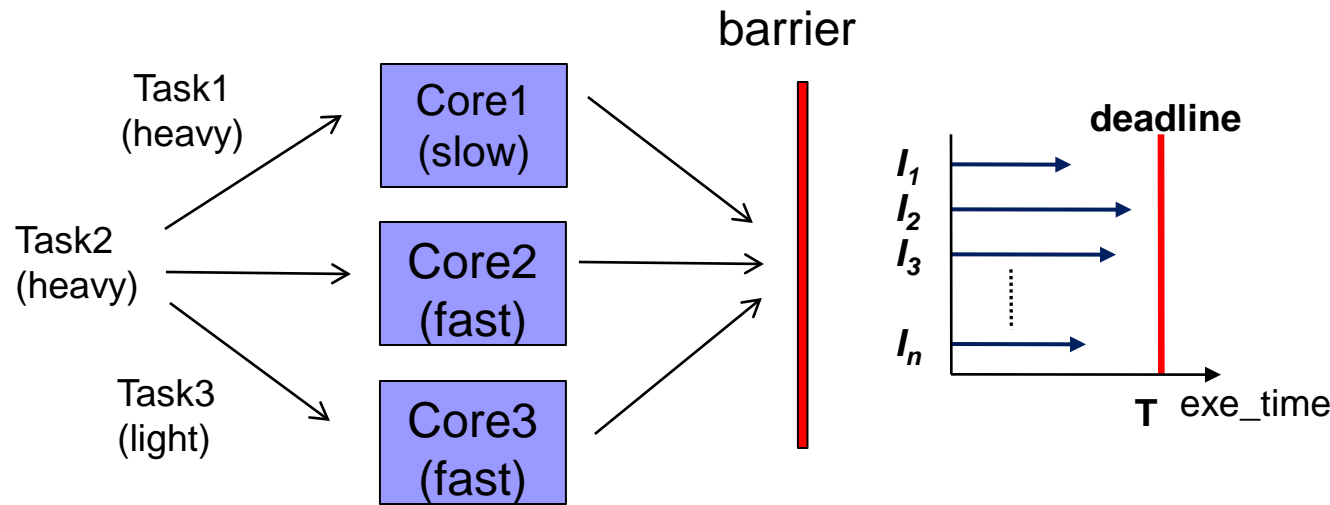


Compensation using guard-bands
Constant guard band margins

- Clock diversity
- Faster (on average) platform
- Need for OS/app awareness
- OS/app must compensate for clock/energy diversity to improve performance and minimize power
- Uniform reliability
- With multiple voltage domains, additional power diversity

System level variability impact

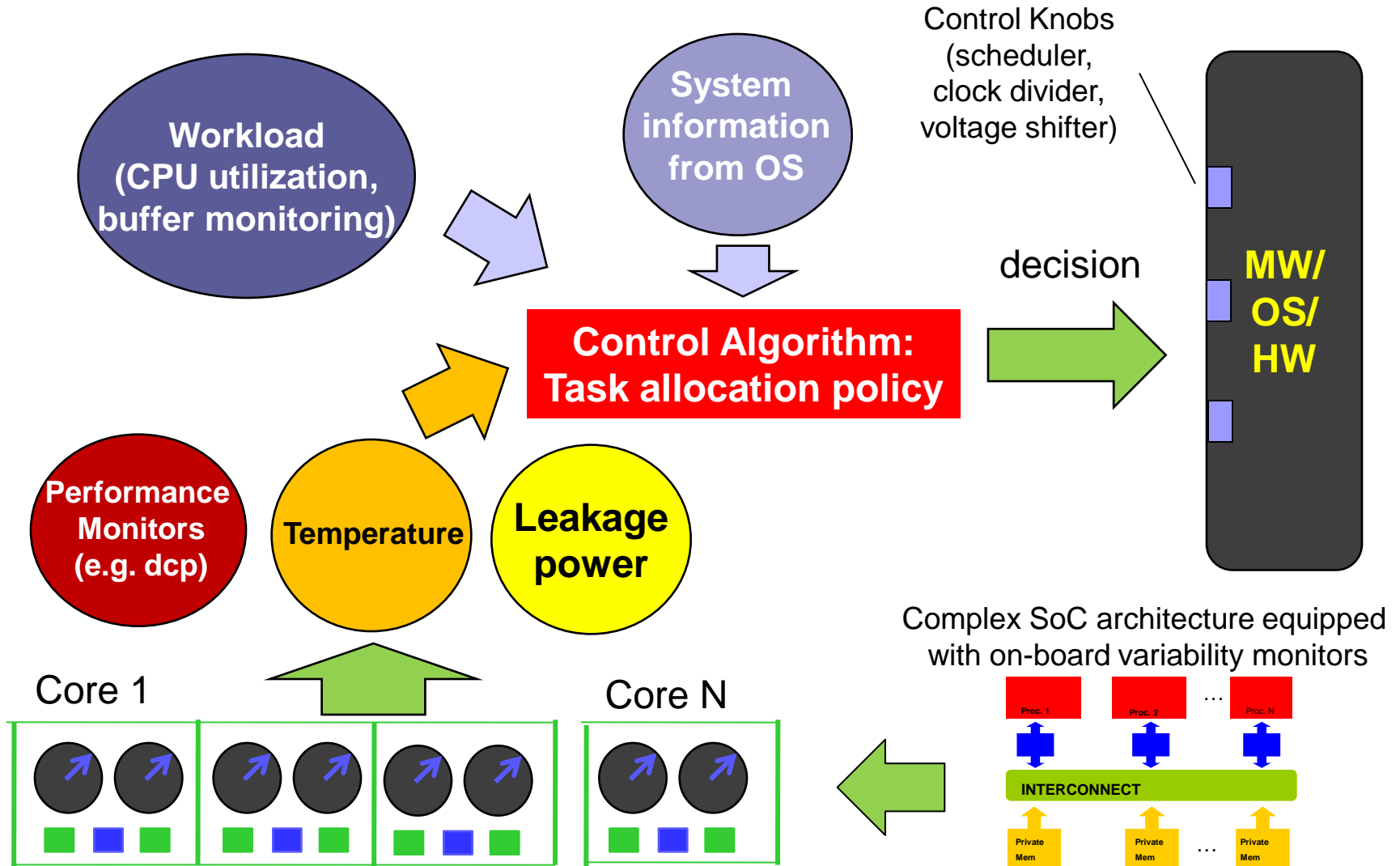
- Consider the most relevant industrial scenario:
 - Multiple frequency domains
 - Single voltage domain (for the cores)
- Without OS/app awareness



The best allocation **depends on the speed of cores** and task characteristics

How can we get the best performance and minimize energy?

Infrastructure to support system level countermeasures



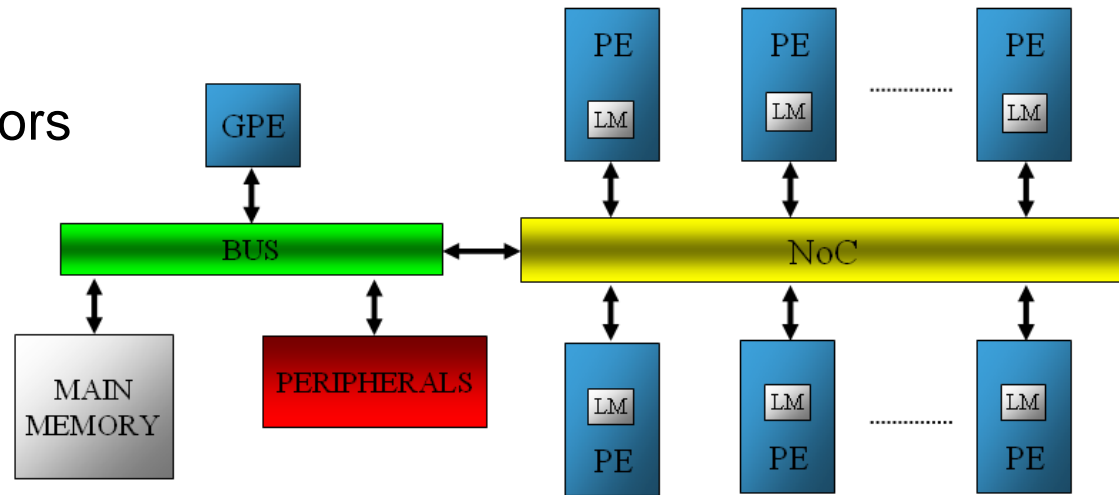
Summary

- **Motivation**: variability leads to QoS degradation and increase in power consumption. Performance and power consumption of the platform can be improved by allocating tasks in a variability-aware manner
- **Problem**: how to decide optimal task allocation on a variability affected platform where cores have different speed and power characteristics?
- **Opportunity**: Monitoring circuits are used to keep variations under control and to expose them to software level
- **Solution**: we propose workload allocation techniques to allocate independent tasks onto multicore while meeting deadlines and minimizing energy consumption;
 - We developed a runtime allocation, which is suboptimal, but it works well and it can be applied on-line

Target Architecture

Industrial multicore platform xStream provided by STMicroelectronics

- GPE is a ST231
- XPE is a VLIW accelerators

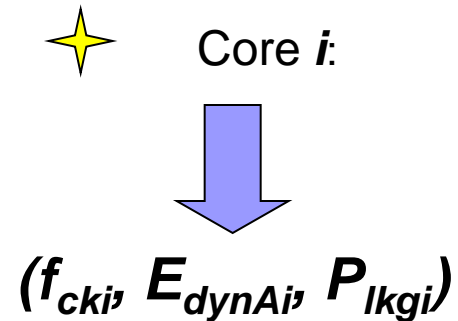
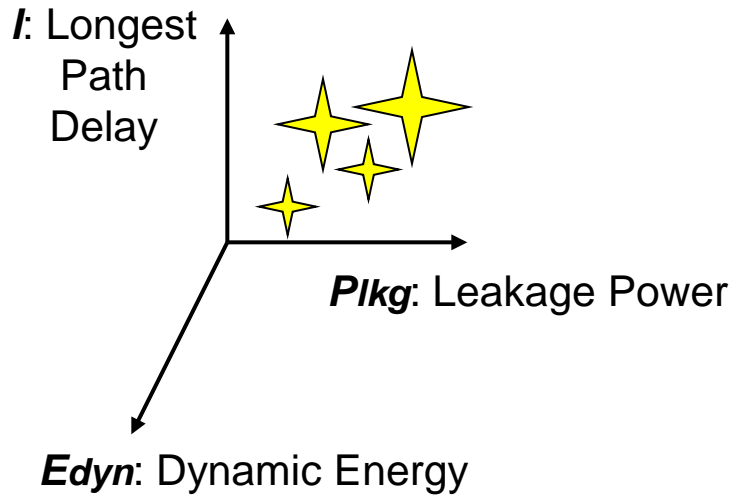


We assume to know for each core:

- Max *clock frequency* supported
- Cores have 2 states: *active* and *idle*
- *Dynamic* and *static* power consumption in any state

Variability Modeling

Variability affects performance and power leading to different clock speeds, dynamic, and static power consumption for each core

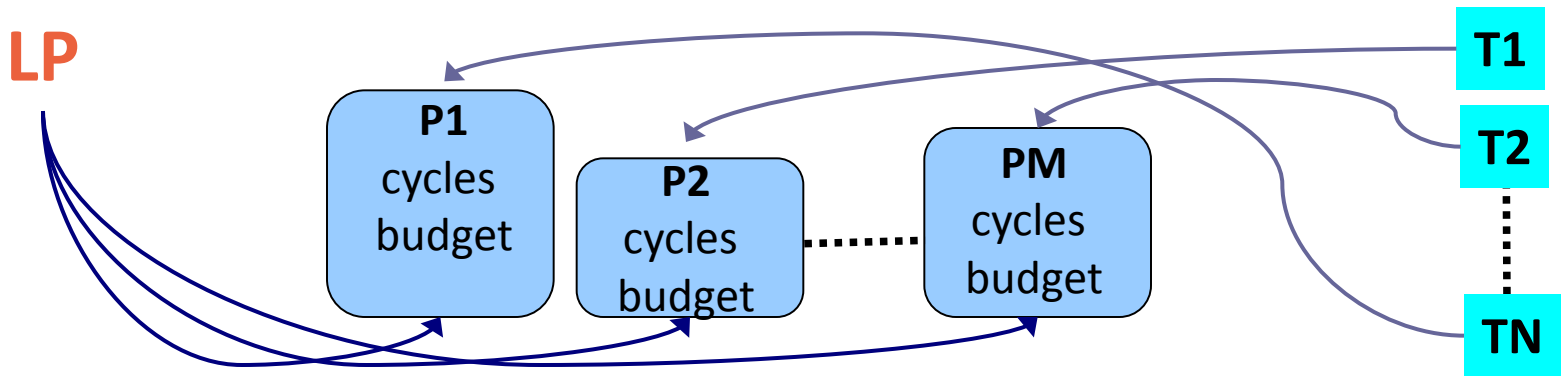


Delay critical path, leakage power and dynamic energy are frequency independent

The distribution is taken from the Variability Aware Modeling (VAM) flow provided by IMEC

Runtime Task Allocation Policy

- 1° Step Linear Programming (LP)
 - **Inputs:** cycles budget of the whole application, time constraint, platform configuration (cores frequencies and powers)
 - **Outputs:** activity cycles budget for each processor P_i
- 2° Step Bin Packing (BP)
 - **Inputs:** LP output, required cycles of each independent task T_j
 - **Output:** Tasks mapping on cores



Comparison

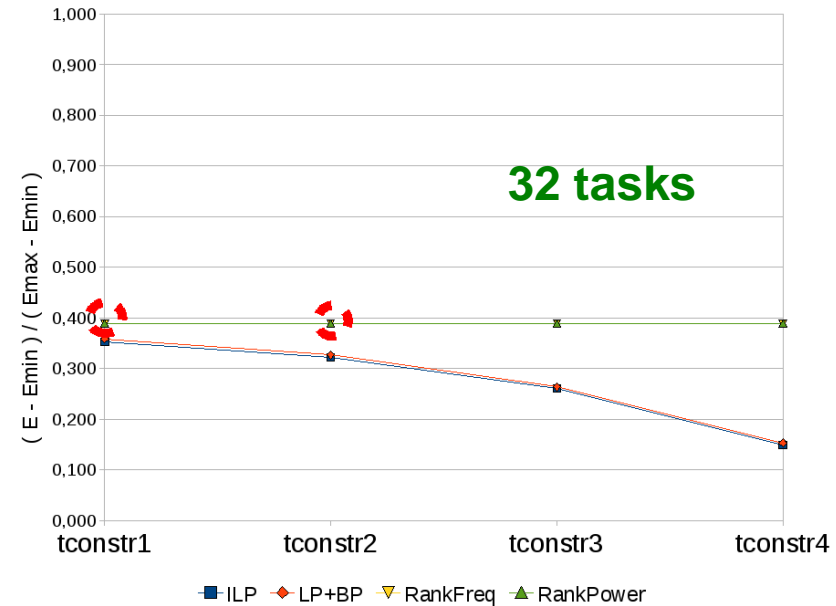
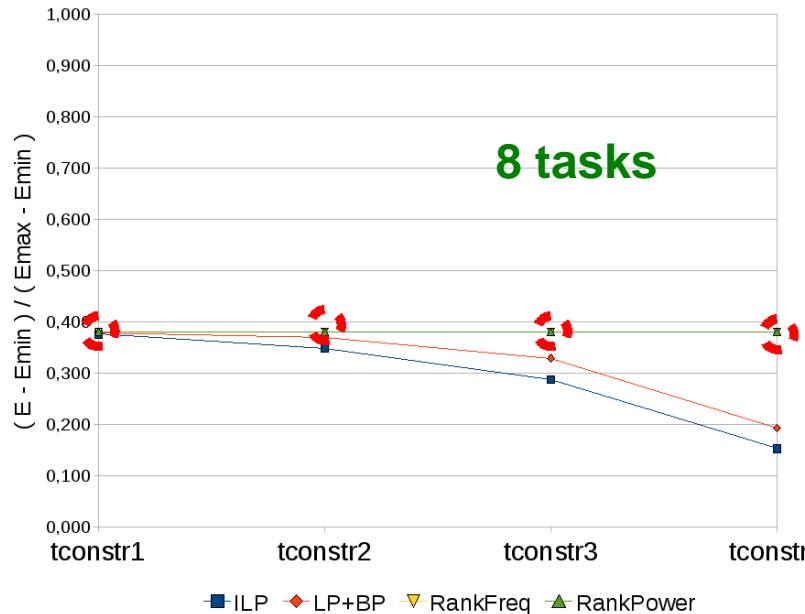
Results using 4-core platform

- E_{min} and E_{max} using (respectively) only the core with minimum and maximum energy
In these situations, deadlines are not met!
- Energy normalization: $(E - E_{min}) / (E_{max} - E_{min})$

Task sets with std_dev / average = 0.5

Circles: Some deadline are not met

Energy Comparison



Deadline Miss Rate

	tconstr1	tconstr2	tconstr3	tconstr4
ILP	0,13	0,00	0,00	0,00
LP+BP	0,75	0,25	0,00	0,00
RankFreq	1,00	1,00	0,75	0,38
RankPower	1,00	1,00	0,75	0,38

	tconstr1	tconstr2	tconstr3	tconstr4
ILP	0,00	0,00	0,00	0,00
LP+BP	0,00	0,00	0,00	0,00
RankFreq	0,88	0,38	0,00	0,00
RankPower	1,00	0,38	0,00	0,00

Comparison

Results using 8-cores platform

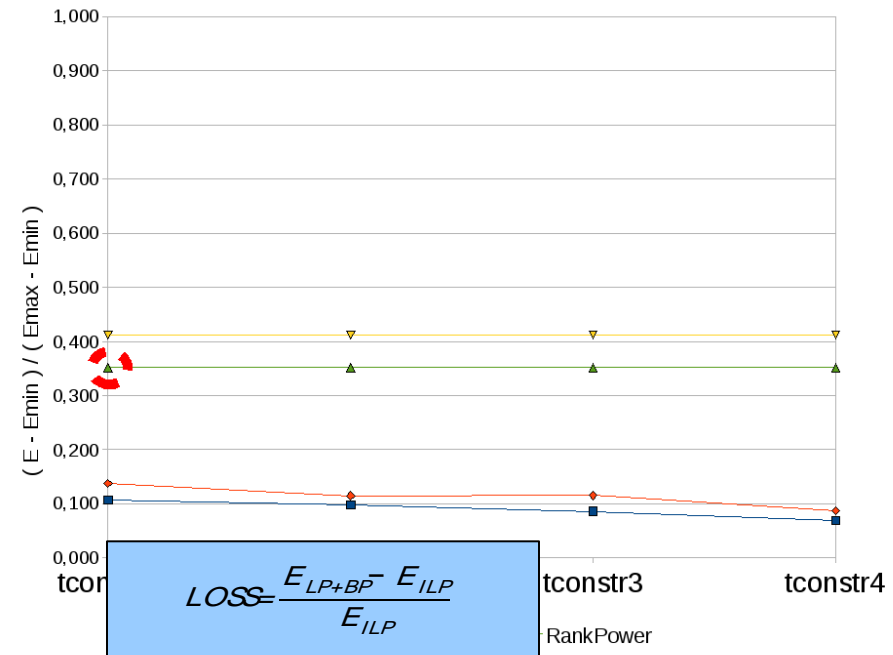
Task sets with std dev / average = 0.5 and 8 tasks

Circles: Some deadline are not met

Platform Utilization Percentage

	tconstr1	tconstr2	tconstr3	tconstr4
ILP	42%	47%	50%	63%
LP+BP	36%	42%	45%	58%
RankFreq	100%	100%	100%	100%
RankPower	100%	100%	100%	100%

The 42% value means that the 58% of cores are never used: *VA policies reduce platform utilization*



LP+BP energy loss related to the ILP:

4 cores, 8 tasks, standard deviation / average = 0.5: **LOSS** = 4.43%

8 cores, 8 tasks, standard deviation / average = 0.5: **LOSS** = 8.90%

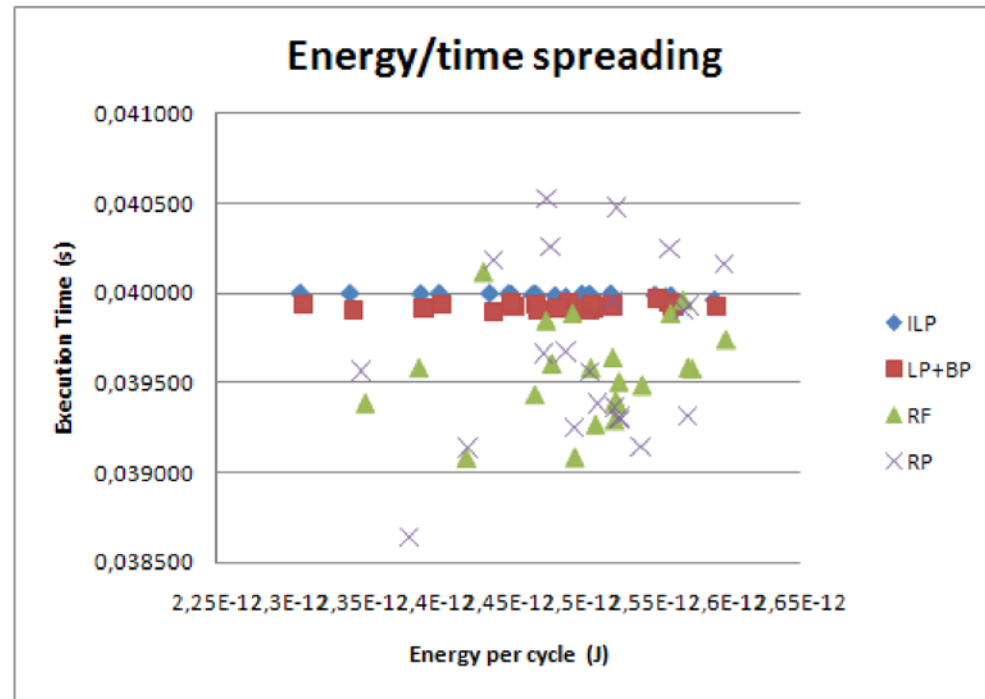
In relaxed conditions, VA policies find allocations that save more energy than rank policies

Variability Compensation Analysis

- Analysis
 - Impact of variations in terms of performance and energy
 - Effectiveness of variability-aware task allocation policies to compensate both energy with performance constraints
- Setup
 - 23 different variability affected platform by VAM
 - A group of 8-task set with gaussian distribution of cycles
- Metrics
 - Energy consumption percentage: $(E - E_{\min}) / (E_{\max} - E_{\min})$
 - Deadline miss-rate
 - For each generated platform we computed MAX e MIN energy for the given cycle budget

Energy vs Performance Spreading

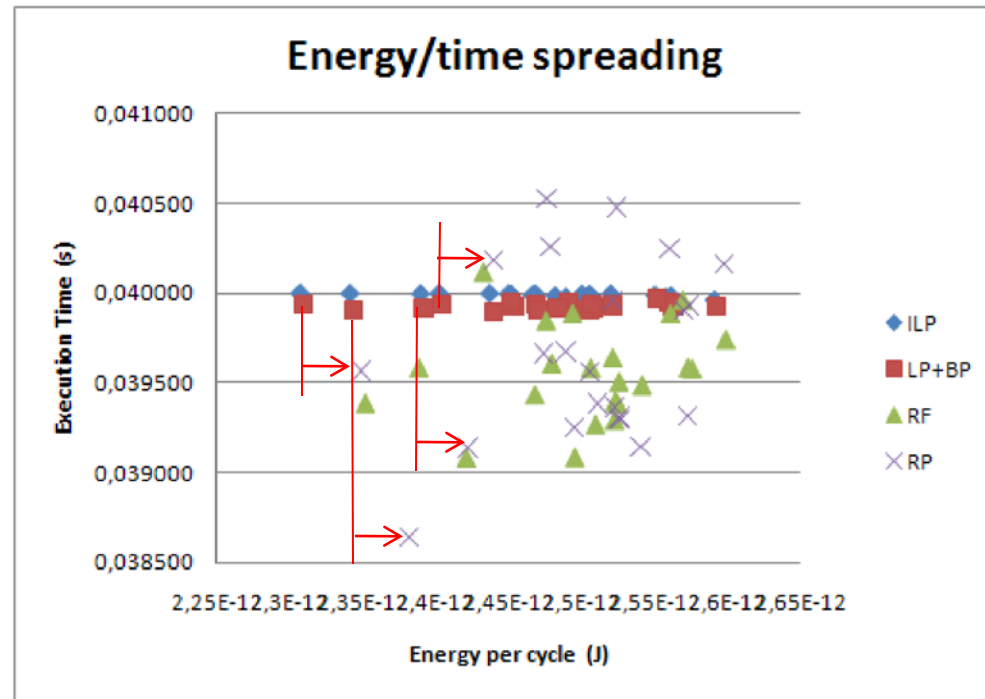
Deadline is 40ms



- VA policies compensate variations by reducing time violations due to variability effects
- Execution time of ILP and LP+BP are very close to the deadline independently from the platform
- Rank policies lead to much more variable execution times
- VA policies provide always lower energy while matching time constraints

Energy vs Performance Spreading

Deadline is 40ms



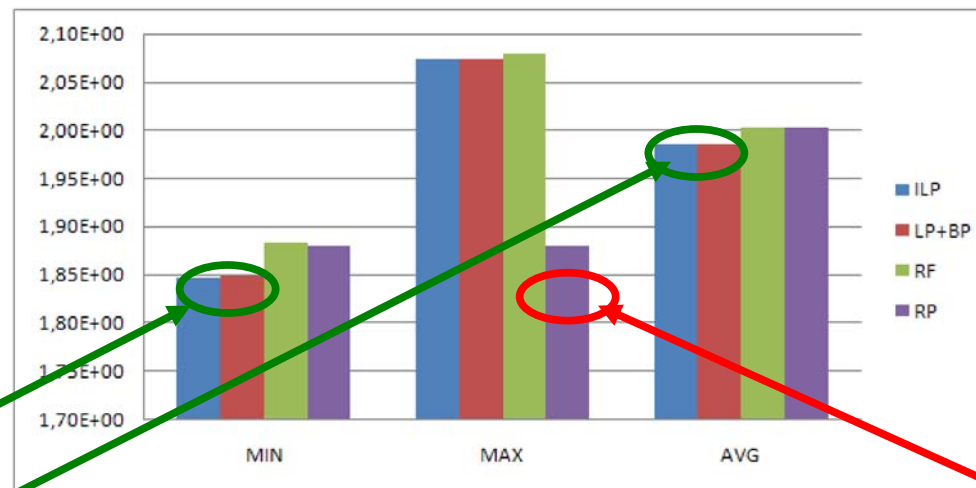
- VA policies compensate variations by **reducing time violations due to variability effects**
- VA achieve **variability compensation independently from the platforms**
- VA policies **minimize energy while matching time constraints**
- **Can be applied at runtime on streaming multimedia applications**

Thank you!

QUESTIONS?

Cumulative Energy Comparison

Minimum, maximum, and average energy consumption across all platforms.



ILP and LP+BP:
lowest energy
consumptions

RP pays lost of
deadline violations

The amount of energy saved by ILP and LP+BP must be evaluated considering the met deadlines and the maximum energy variations per platform