

SYNERGISTIC DESIGN OF ENERGY-EFFICIENT HETEROGENEOUS COMPUTE NODES

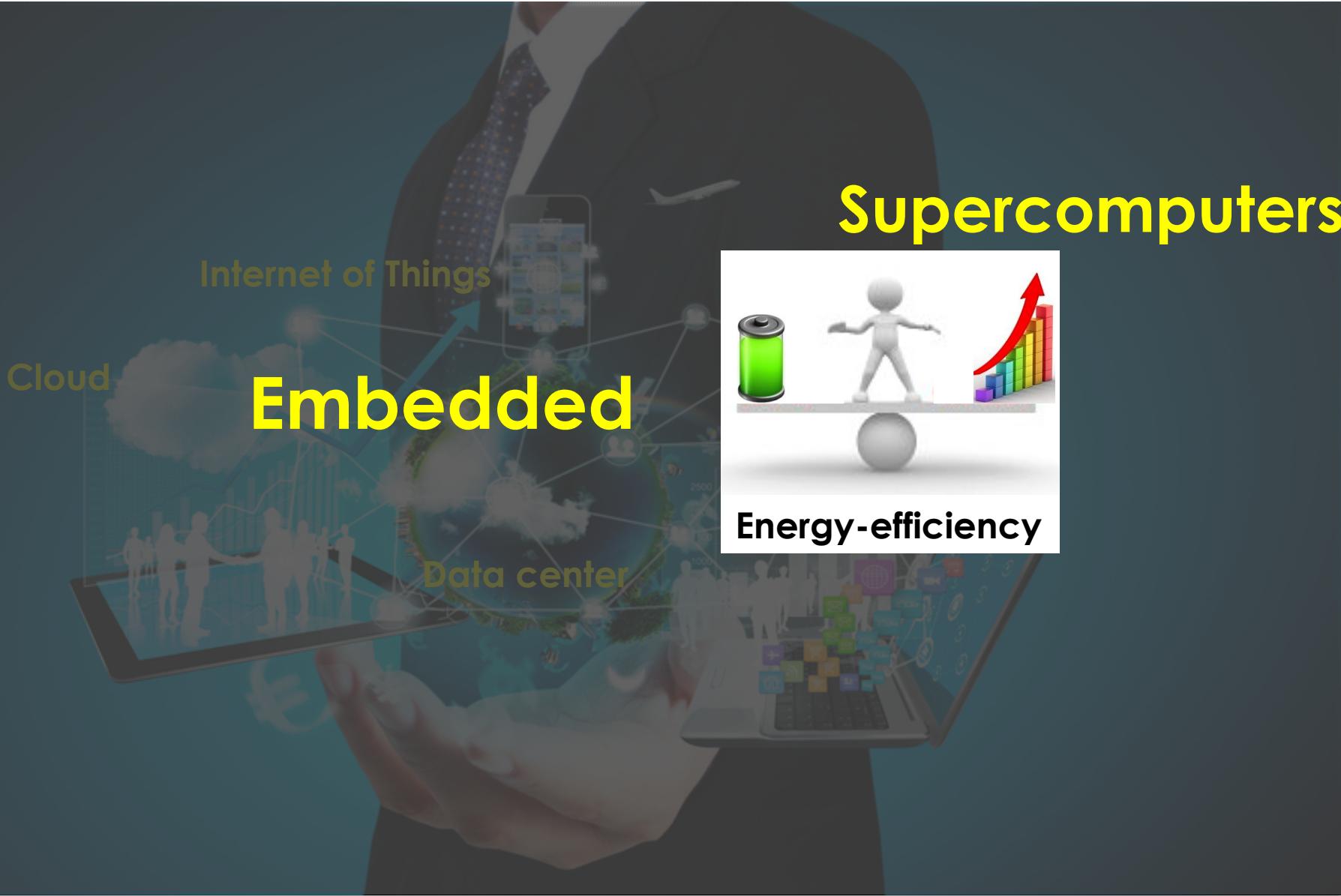
Abdoulaye Gamatié

LIRMM / CNRS-UM, Montpellier

Colloque National du GDR SOC2, June 2017, Bordeaux

*** Joint work: F. Bruguier, A. Butko, M. Novaes, P.Y. Péneau, F. Pereira, M. Robert, G. Sassatelli, S. Senni, and L. Torres

Nowadays technological ecosystem



FOCUS ON SUPERCOMPUTING DOMAIN

The quest of Exascale Supercomputers

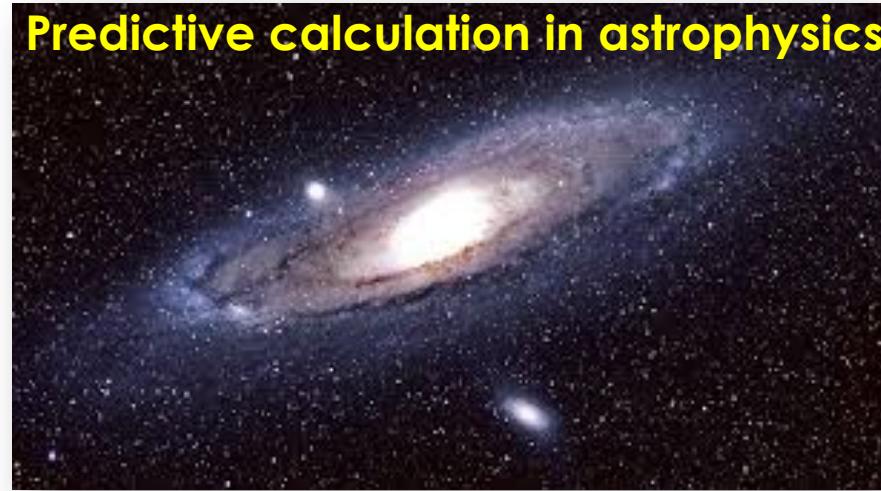
Scientific challenges for HPC: some applications

Hurricane prediction



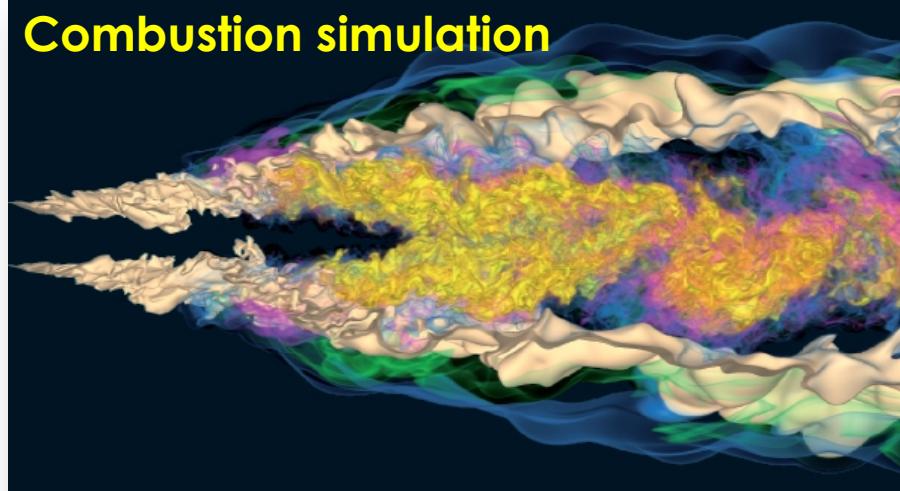
Avoid thousands human victims, e.g. Katrina, Sandy...

Predictive calculation in astrophysics



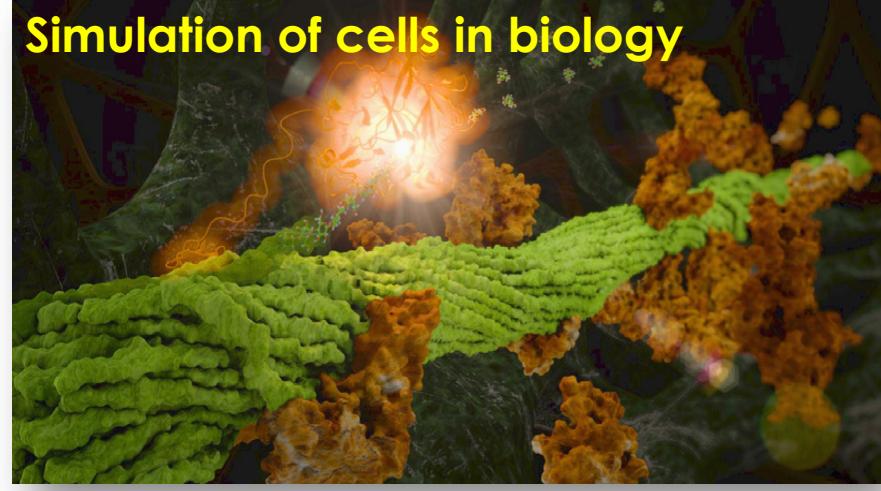
Understanding the formation and evolution of the universe

Combustion simulation



Low temperature combustion for efficient and low emission automobile engines

Simulation of cells in biology



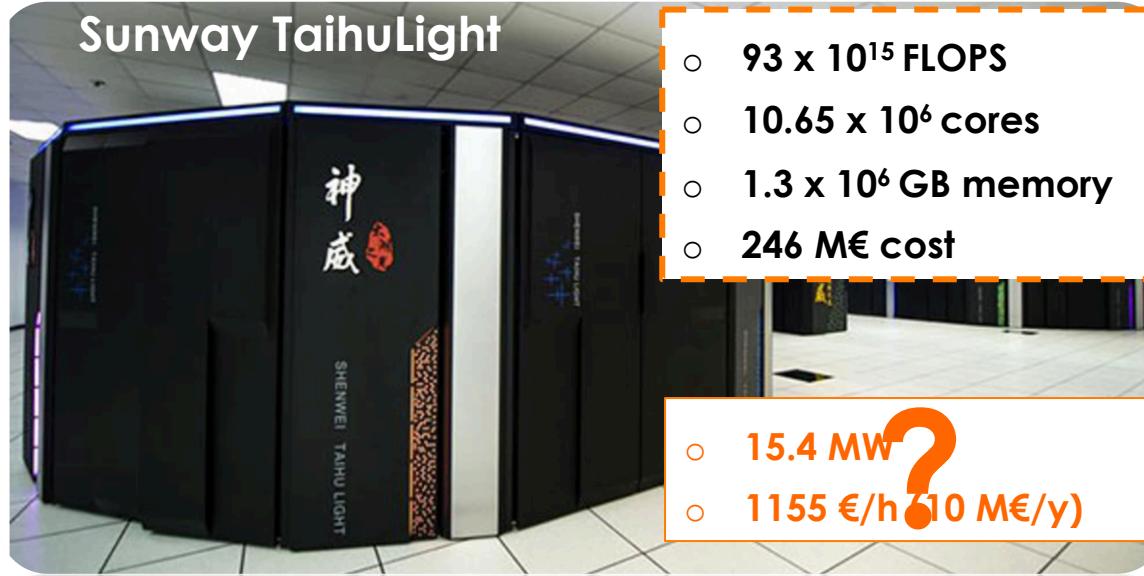
Advance comparative genomics and underlying phylogenics, cell degradation

Goal: exascale supercomputers

10^{18} floating point operations per second (FLOPs)

Target power: maximum 20 MW around 2020

Today:



#1

With current technology → 3ExaFLOPs is 1GW budget



Funny facts: towards reduction of energy cost

World's first Petaflop supercomputer Roadrunner is shut down

Was using too much power

By [Lee Bell](#)

Tue Apr 02 2013, 11:03

THE WORLD'S FIRST Petaflops supercomputer Roadrunner has been switched off after five years of operation due to its high energy consumption.

The IBM built supercomputer cluster, which is housed at the US Los Alamos National Laboratory, was the first to break the Petaflops barrier of one quadrillion calculations per second when it was launched in 2008.



Funny facts: towards reduction of energy cost (cont'd)

[Store](#)[Products](#)[Support](#)[News Center](#)[Our Company](#)[Our Products](#)[Blogs & Communities](#)[Press Tools](#)

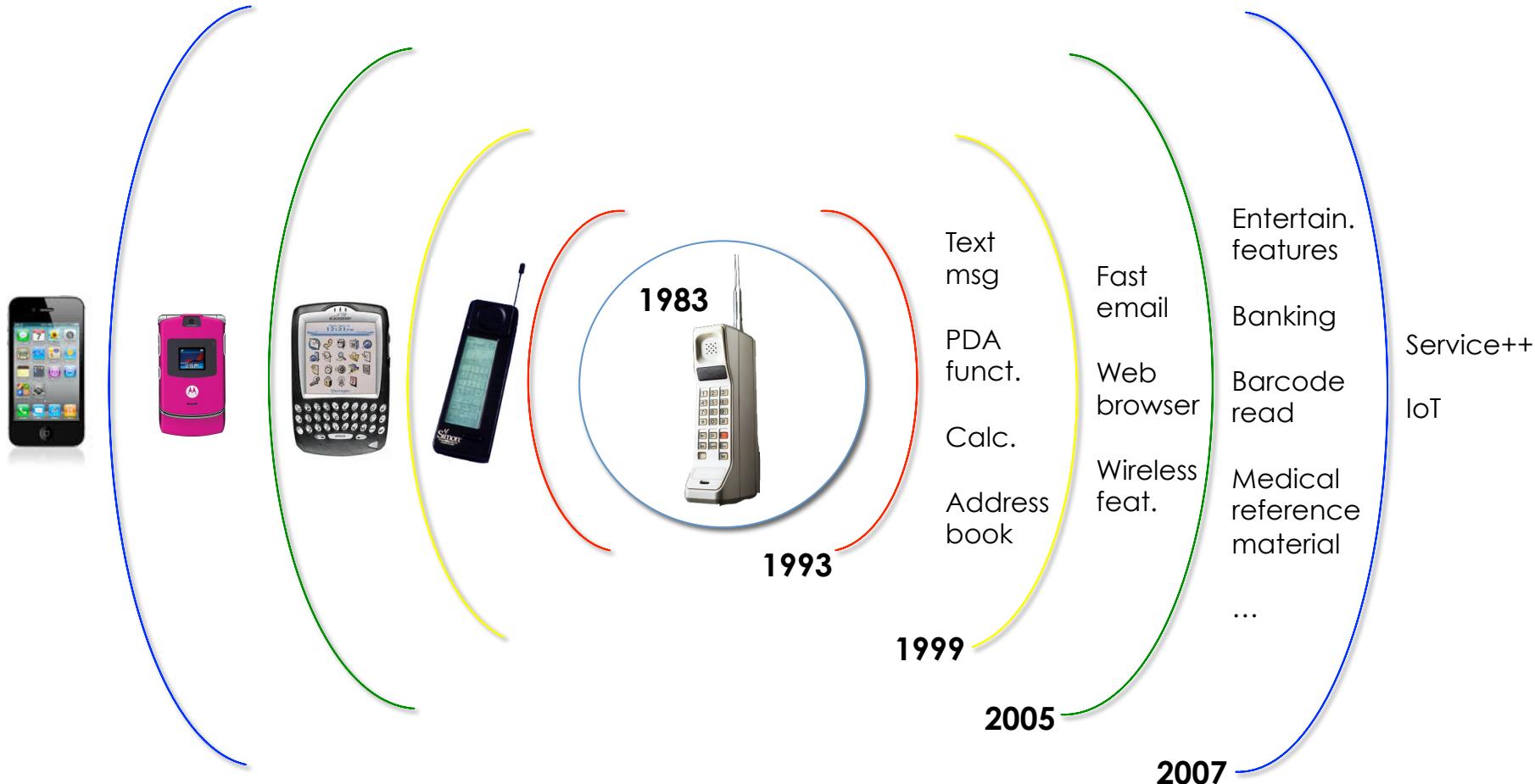
Microsoft research project puts cloud in ocean for the first time

FOCUS ON EMBEDDED DOMAIN

Smarter world at minimized energy budget

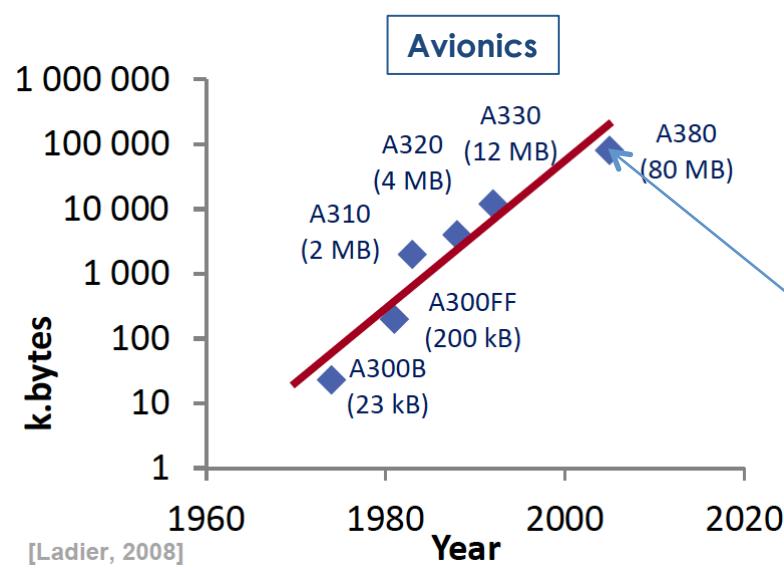
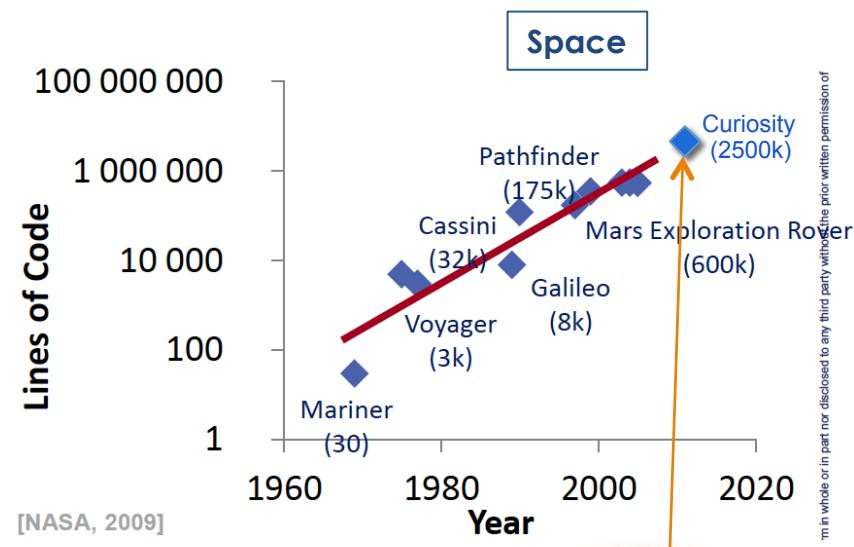
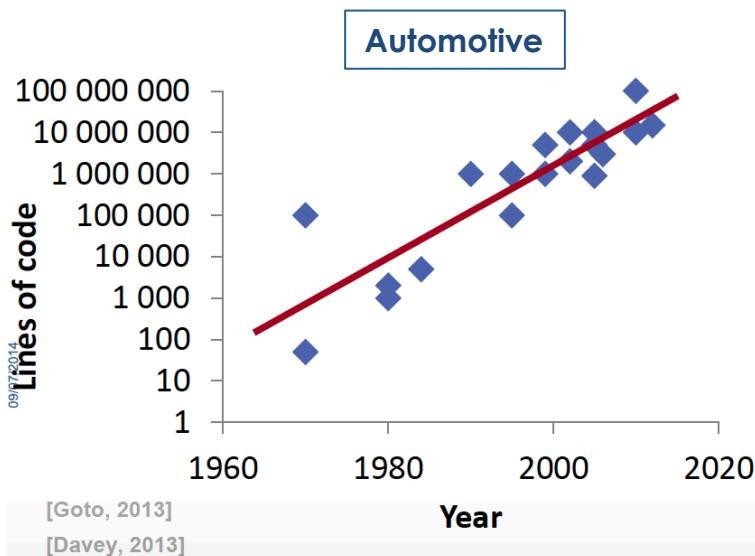
Trends in embedded computing domain

- Continuous integration of new functionalities

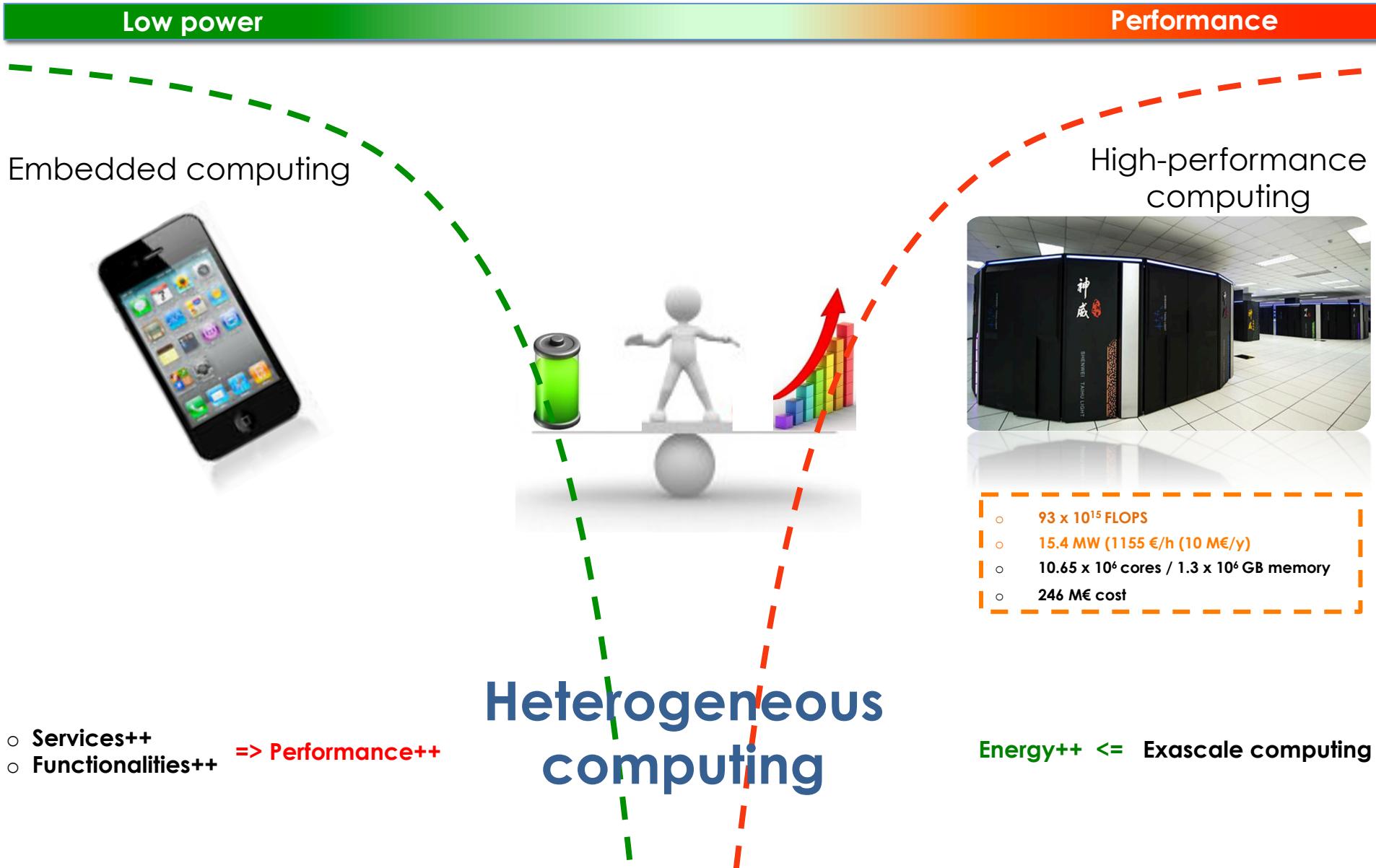


- Increasing demand in performance: on-chip multicore systems

Increasing performance demand: some numbers



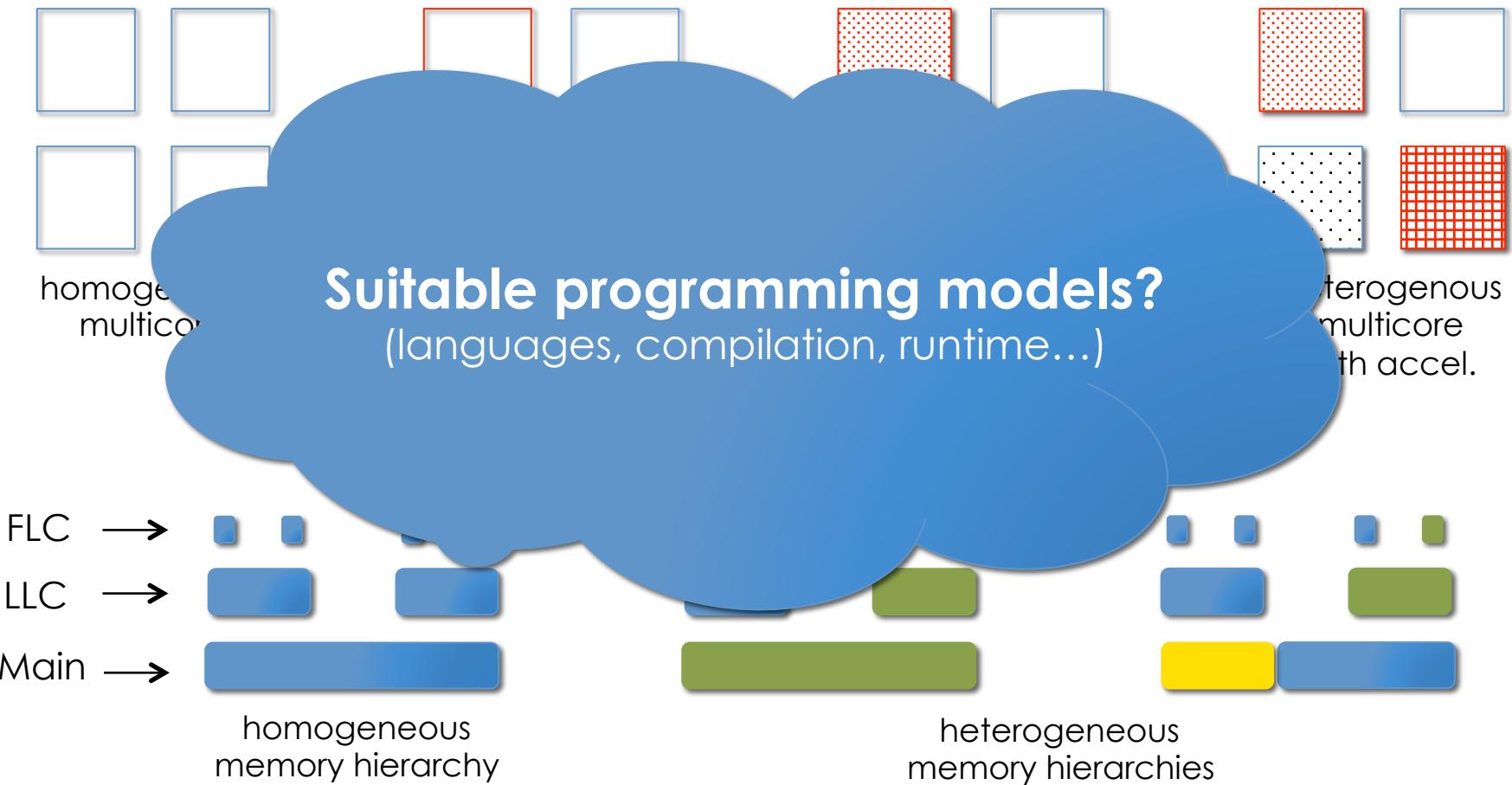
The quest of energy-efficiency



Outline of this talk

- **Synergistic design of heterogeneous compute nodes**
 - Motivation and requirements
- **Examples of synergistic designs for energy-efficiency**
 - Adaptive multicore compute node design
 - Leveraging non volatile memory with compiler optimization
- **Conclusion**

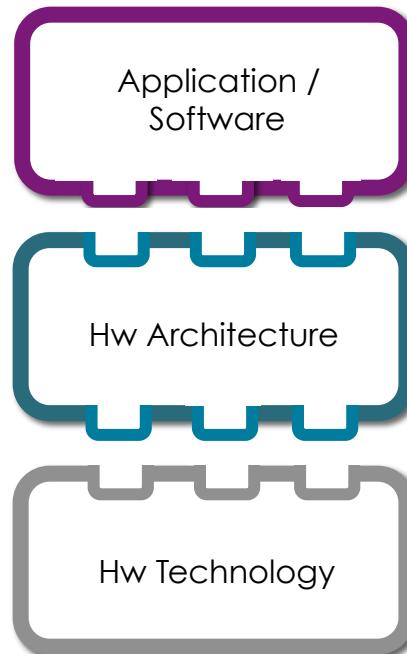
Compute node design: quick brainstorming



Synergistic design

Key idea: dealing with multi-level design concerns

- **Requirement #1:** suitable integrated frameworks
 - design and evaluation
- **Requirement #2:** cross-domain interaction



computer science

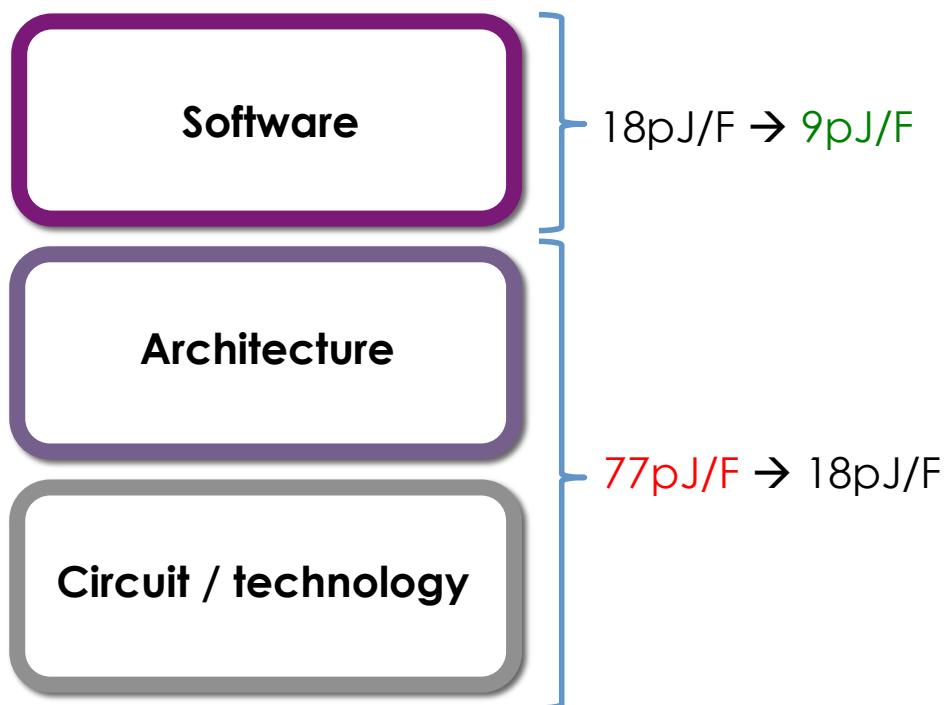
computer engineering

electrical engineering

...

Example: data movement issue

- On 22nm CMOS technology*
 - Toggling a single bit: **10^{-6} pJ**
 - Moving a single bit: **1 pJ/mm** (1 mW/mm at 1 GHz)
- Synergistic optimization for energy-efficiency: more than **80% gain**



Bill Dally
Chief Scientist NVIDIA
Professor Stanford

*

M. Duranton, D. Black-Schaer, K. De Bosschere, and J. Maebe. « **The HiPEAC vision for advanced computing in horizon 2020** », 2013.

Outline of this talk

- **Synergistic design of heterogeneous compute nodes**
 - Motivation and requirements
- **Examples of synergistic designs for energy-efficiency**
 - Adaptive multicore compute node design
 - Leveraging non volatile memory with compiler optimization
- **Conclusion**

DESIGN EVALUATION FRAMEWORK

For heterogeneous compute nodes

Synergistic design: which frameworks?

Manycore Architecture enerGy and Performance evaluation environment: **MAGPIE***

- **gem5***: quasi-cycle accurate simulator
 - IPs: cores (x86, ARM...), memory, interconnect...
 - can boot a complete linux OS
 - detailed microarchitecture details / performance statistics
- **McPAT****: area, power and timing modeling for CMOS, SOI technologies
- **NVSim*****: area, power and energy estimator for non-volatile memory technologies



accuracy does not come for free!

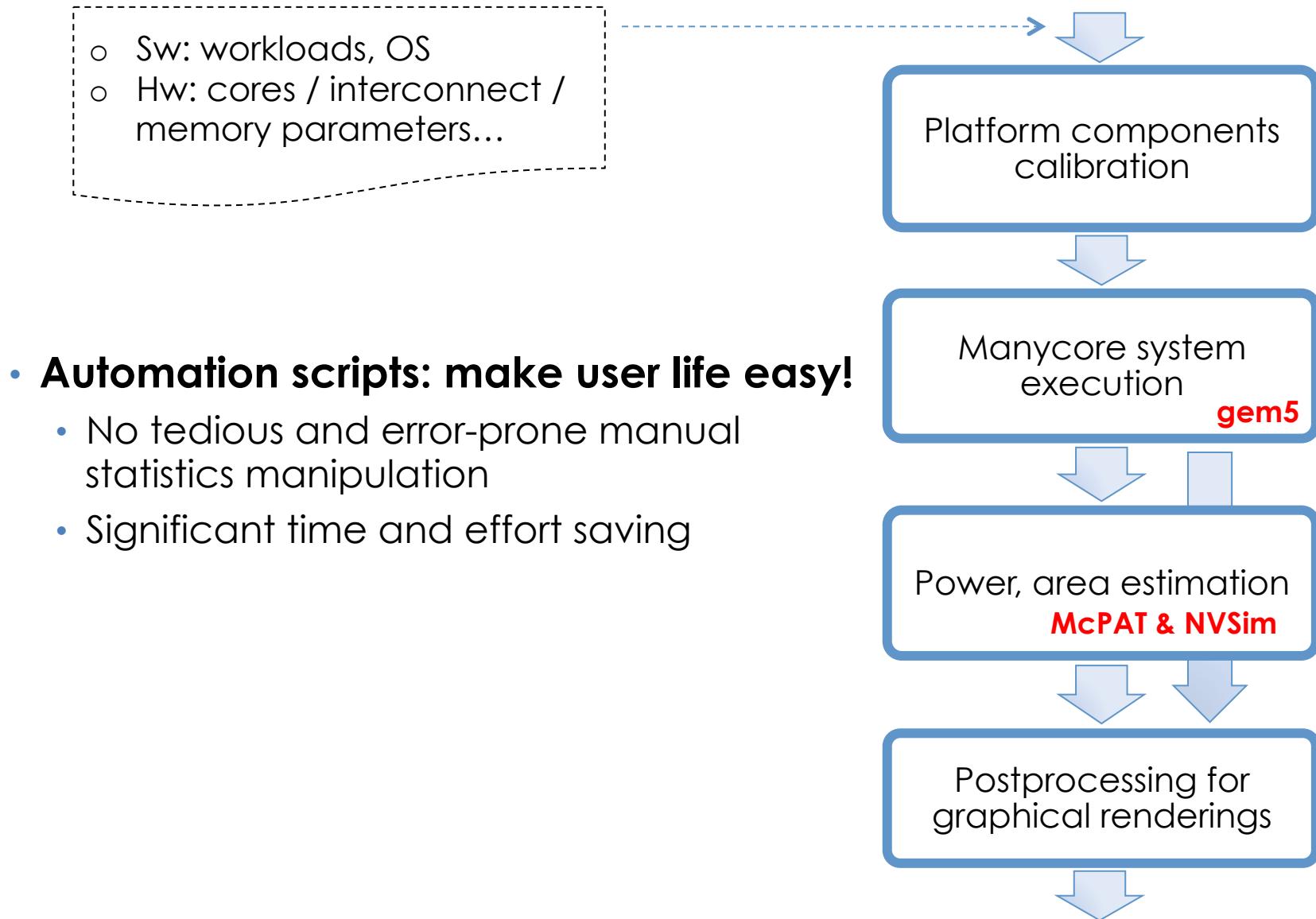
* <http://www.lirmm.fr/continuum-project/pages/magpie.html>

** <http://www.gem5.org>

*** www.hpl.hp.com/research/mcpat

**** <http://nvsim.org>

Synergistic design: MAGPIE automated flow

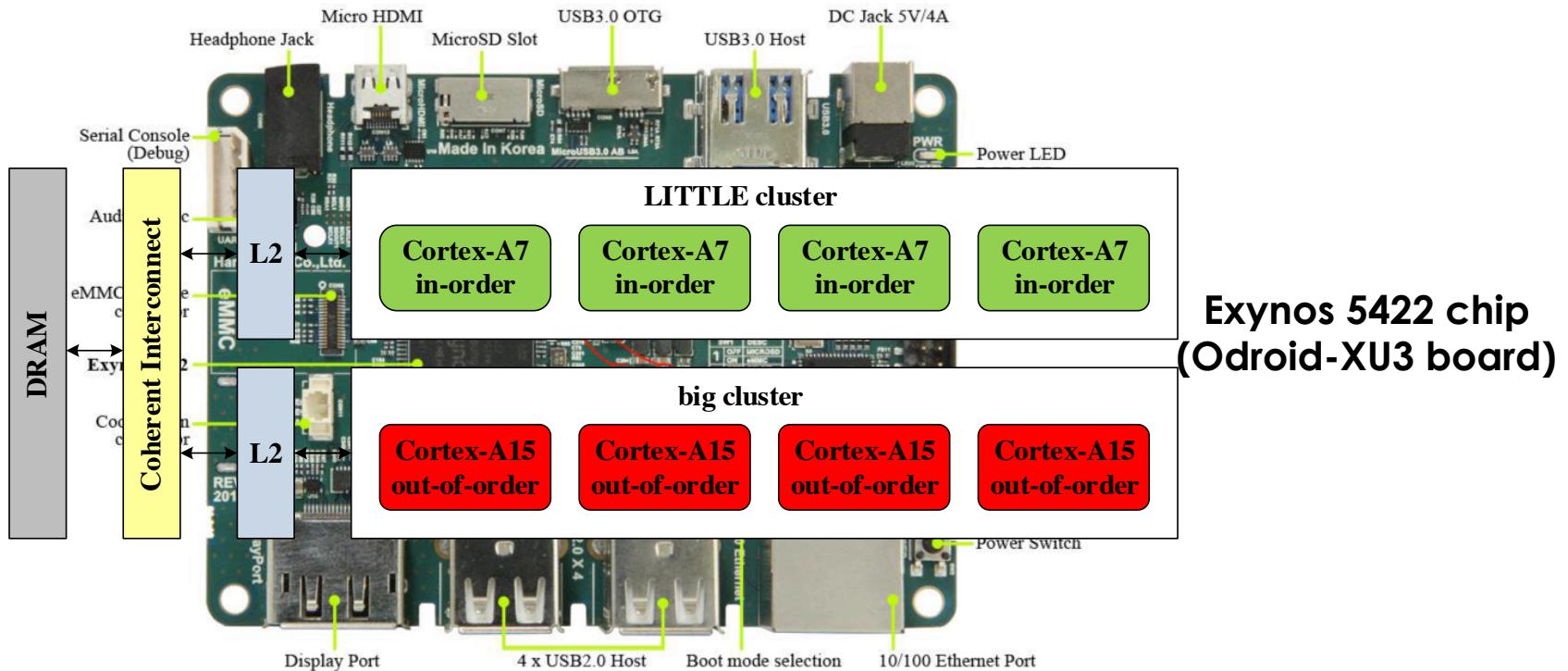


ENERGY-EFFICIENT COMPUTE NODE DESIGN

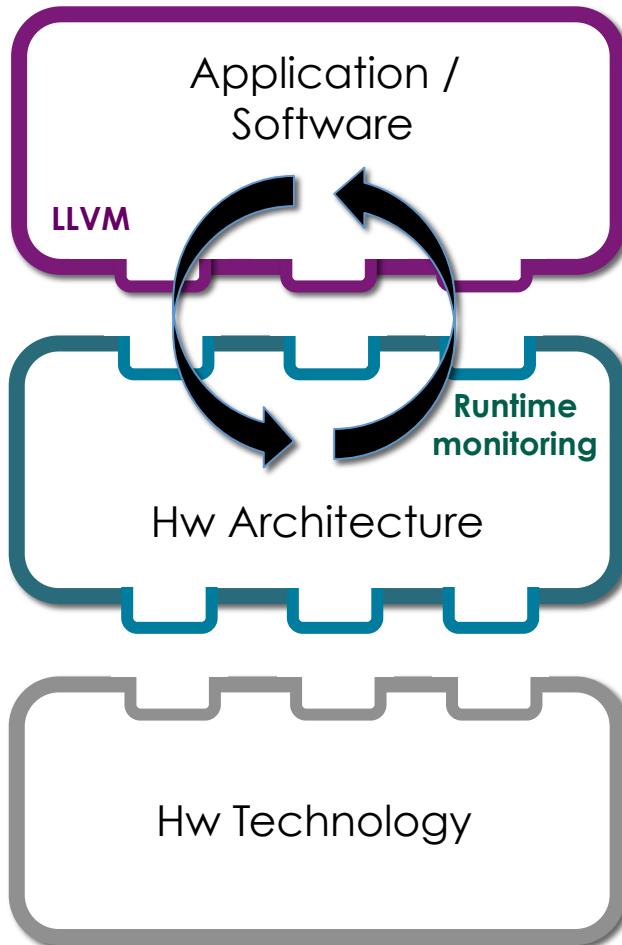
Software / hardware design tradeoff: part 1

Example of heterogeneous compute node

Single-ISA multicore



How to reach energy-efficiency?



CONTINUUM ANR project

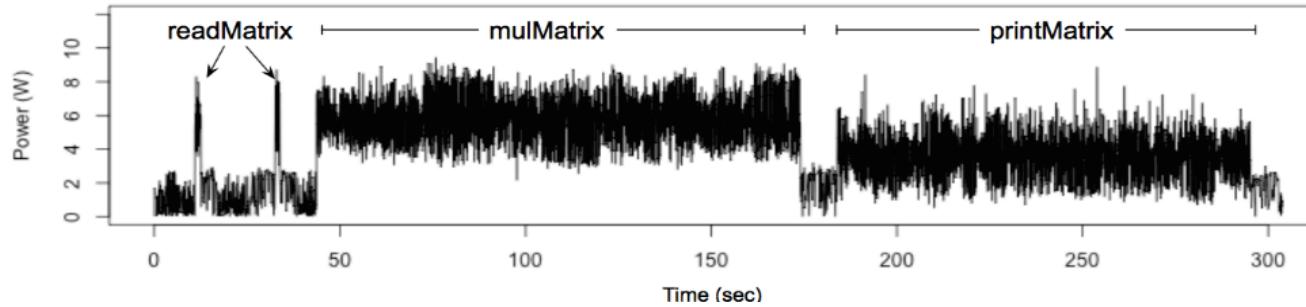
- leveraging compilation in adaptive compute node design
- <http://www.lirmm.fr/continuum-project>



Motivational example: matrix multiplication

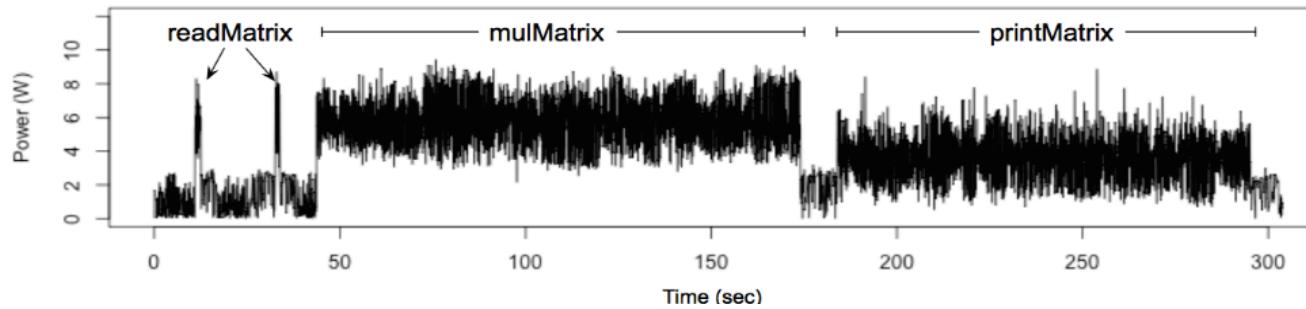
Allocation directives

```
int main(int argc, char** argv){  
    int M1, N1, M2;  
    setCoreAllocation(_config);  
    int** m1 = reaMatrix(argv[1], &M1, &N1);  
    ...  
    int** m2 = reaMatrix(argv[2], &N1, &M2);  
    ...  
    setCoreAllocation(_config);  
    int** m3 = multMatrix(m1, m2, M1, N1, M2);  
    ...  
    setCoreAllocation(_config);  
    printMatrix(m1);  
    printMatrix(m2);  
    printMatrix(m3);  
}
```



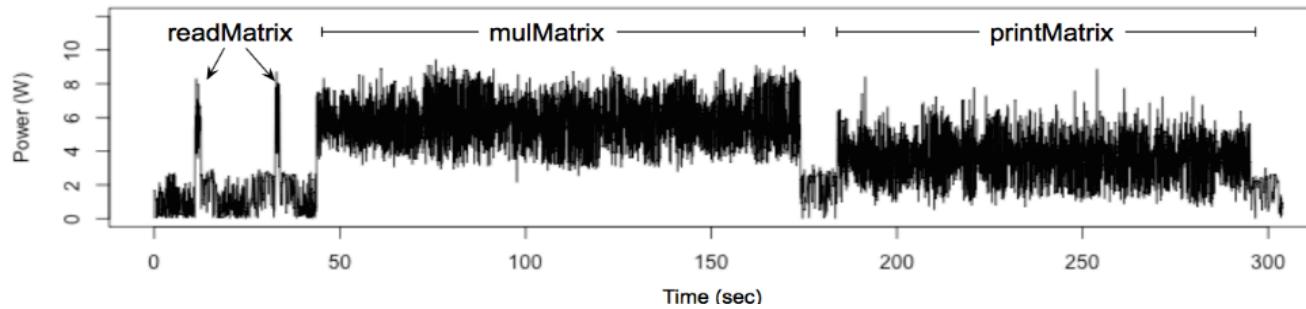
Motivational example: design-time analysis

```
int main(int argc, char** argv){  
    int M1, N1, M2;  
    setCoreAllocation(LOW-POWER); // IO-intensive  
    int** m1 = reaMatrix(argv[1], &M1, &N1);  
    ...  
    int** m2 = reaMatrix(argv[2], &N1, &M2);  
    ...  
    setCoreAllocation(HIGH-PERF); // Compute-intensive  
    int** m3 = multMatrix(m1, m2, M1, N1, M2);  
    ...  
    setCoreAllocation(...);  
    printMatrix(m1);  
    printMatrix(m2);  
    printMatrix(m3);  
}
```



Motivational example: runtime analysis

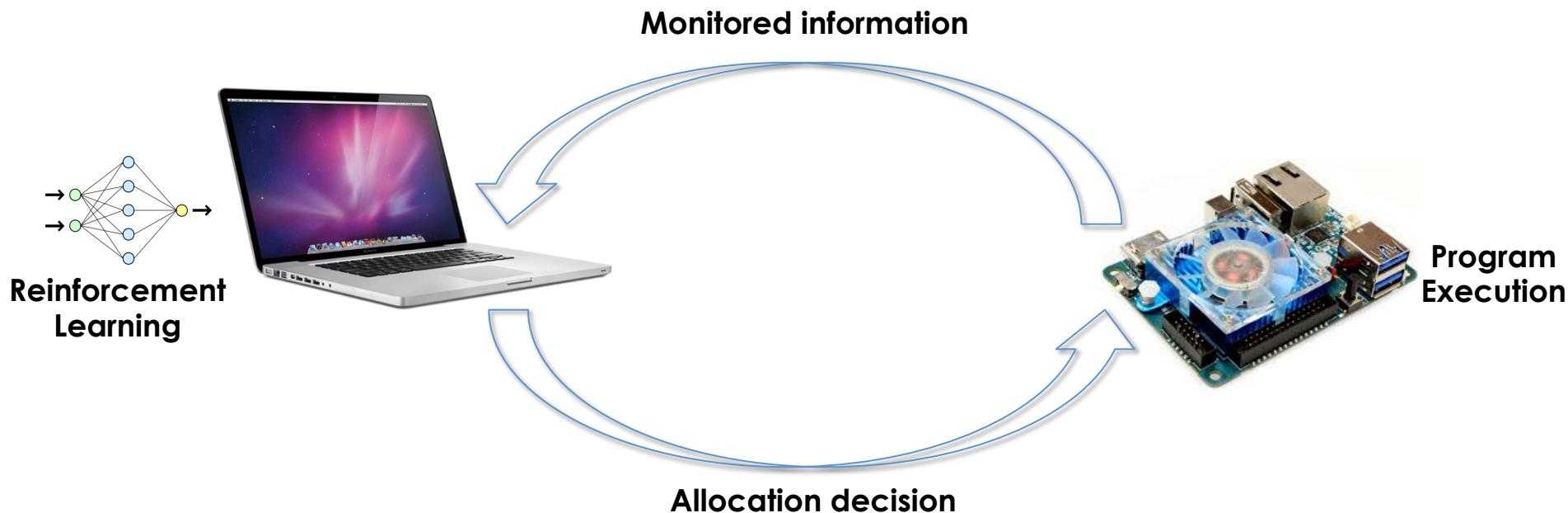
```
int main(int argc, char** argv){  
    int M1, N1, M2;  
    setCoreAllocation (LOW-POWER); // IO-intensive  
    int** m1 = reaMatrix(argv[1], &M1, &N1);  
    ...  
    int** m2 = reaMatrix(argv[2], &N1, &M2);  
    ...  
    setCoreAllocation (?); // Depends on M1, N1 & M2  
    int** m3 = multMatrix(m1, m2, M1, N1, M2);  
    ...  
    setArchitecture(...);  
    printMatrix(m1);  
    printMatrix(m2);  
    printMatrix(m3);  
}
```



Which approach to consider?

	<u>Design-time</u> (e.g., compiler)	<u>Runtime</u> (e.g., OS)	<u>Hybrid adaptive</u>
pros	Leverage program static features at lower runtime cost	Leverage unpredictable runtime behavior for better choice	Leverage advantages of both Static and Dynamic
cons	Cannot address unpredictable behavior: e.g., input-dependent or environment-dep. executions	<ul style="list-style-type: none"> - Overhead of runtime monitoring - Getting all suitable insights from program behavior not easy 	Requires a careful combination of Static and Dynamic decisions

Hybrid adaptive allocation approach

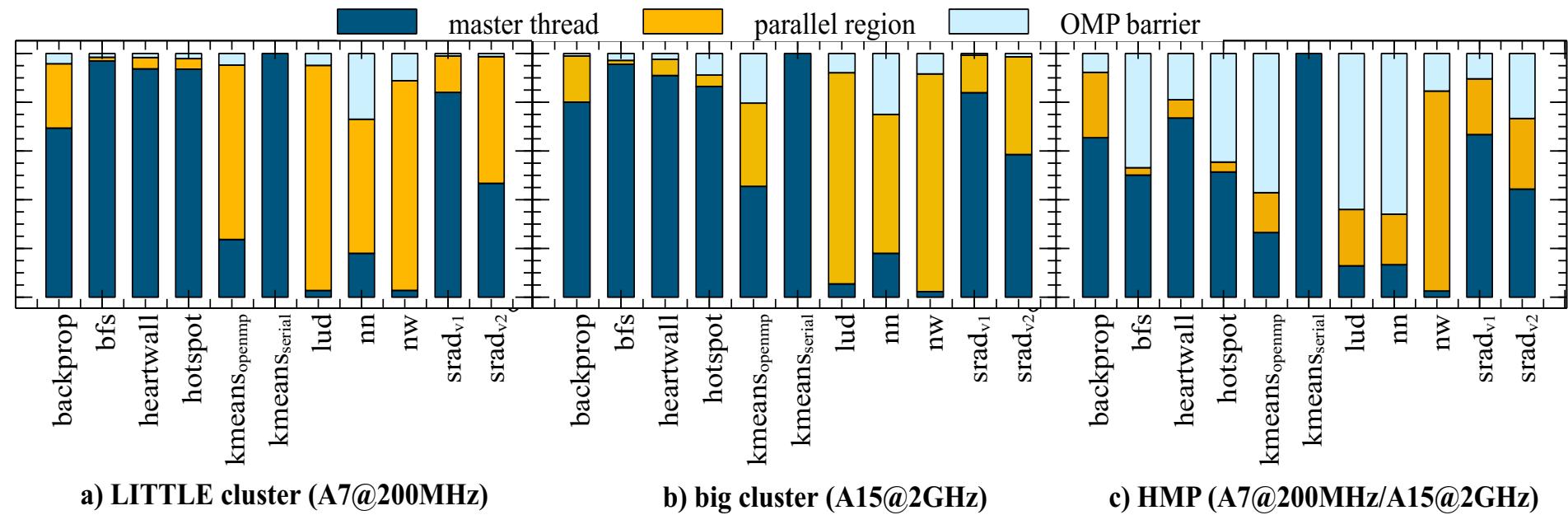


ENERGY-EFFICIENT COMPUTE NODE DESIGN

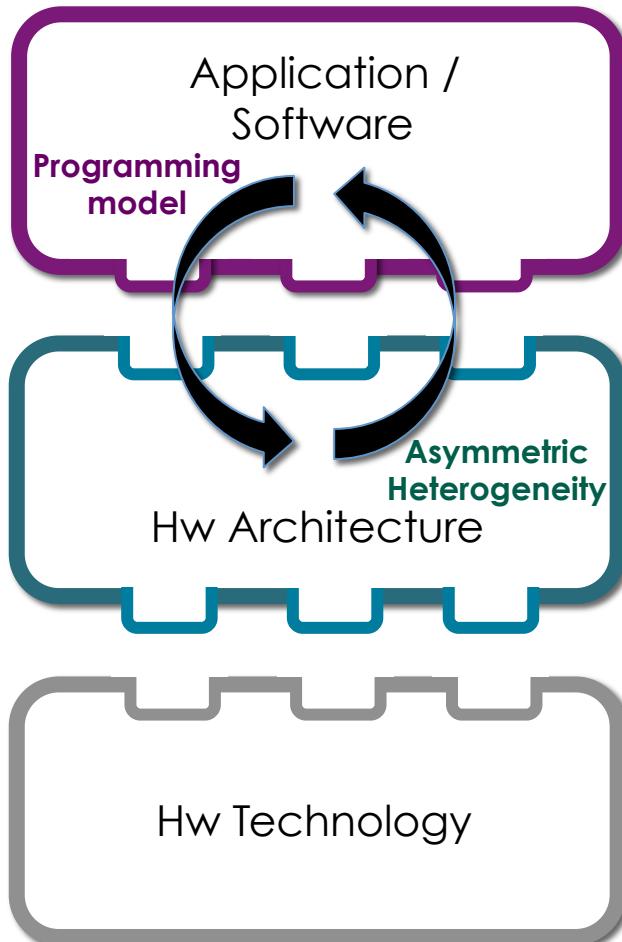
Software / hardware design tradeoff: part 2

Let us reconsider things...

Runtime breakdown



How to reach energy-efficiency?

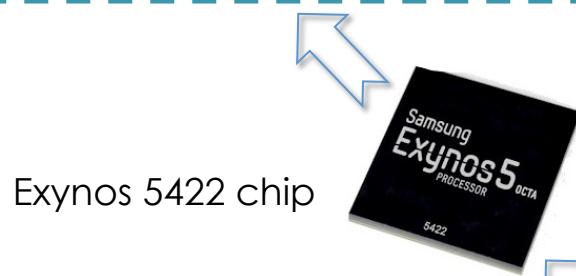
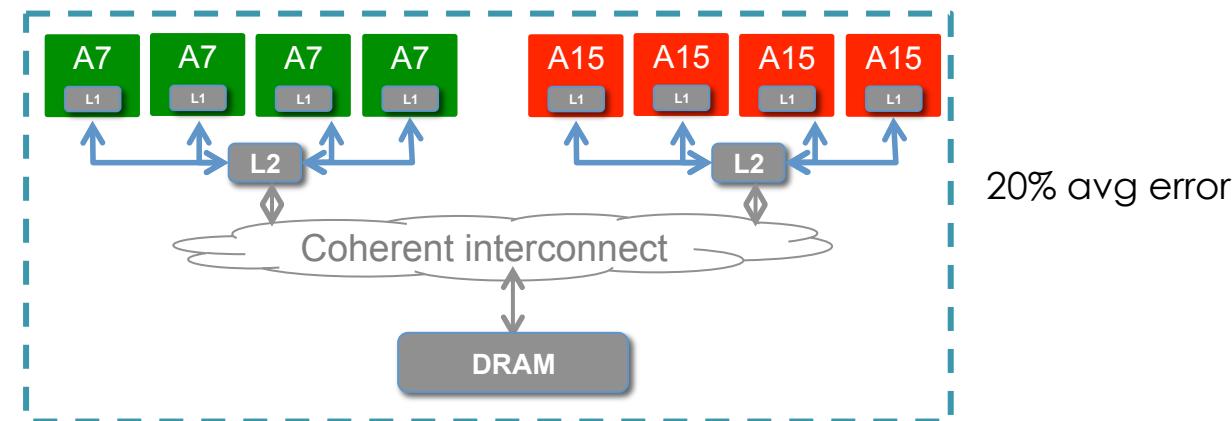


MontBlanc EU projects

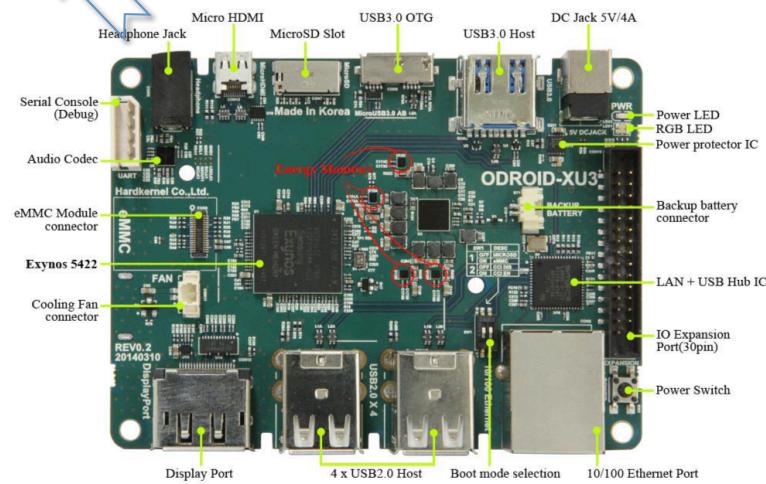
- leveraging programming models in heterogeneous compute node design
- <http://montblanc-project.eu>



big.LITTLE architecture: modeling with gem5



Exynos 5422 chip

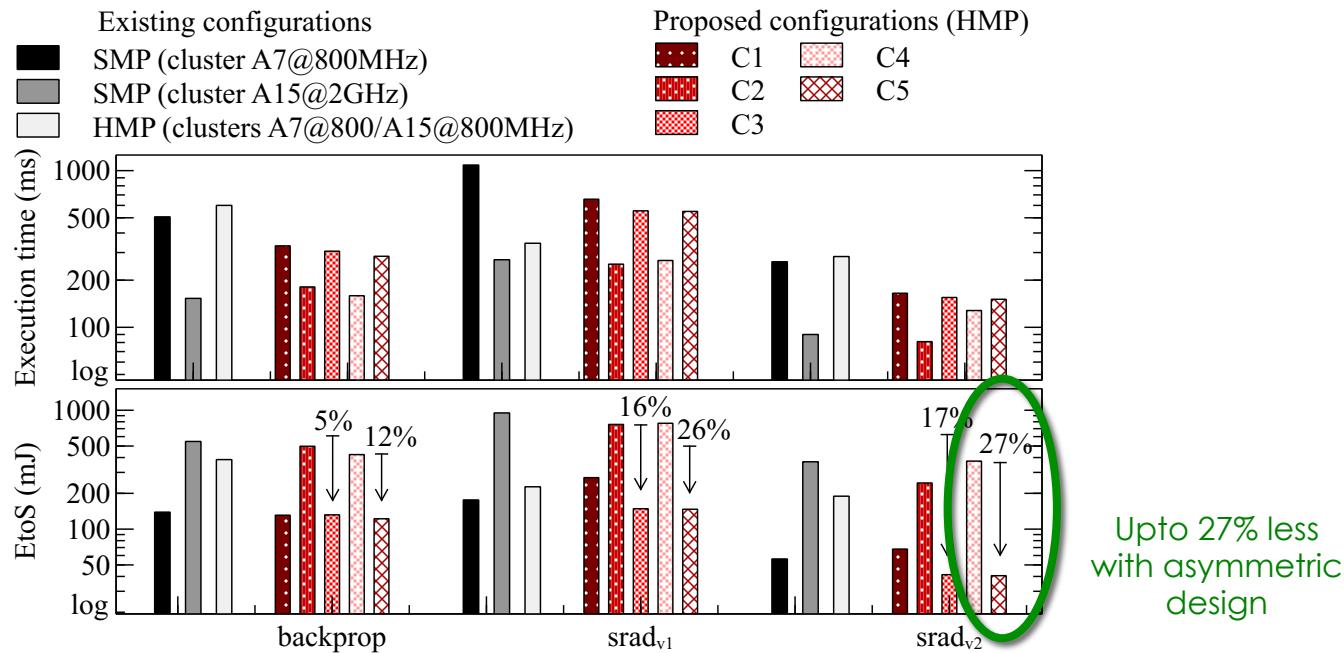


Towards big.LITTLE asymmetric architecture

Energy-efficient designs based on Rodinia kernels

	Cortex-A7 cluster		Cortex-A15 cluster		
	Count	Clock	Count	Clock	L2
Conf. 1	4	800 MHz	1	800 MHz	2 MB
Conf. 2	4	800 MHz	1	2 GHz	2 MB
Conf. 3	7	800 MHz	1	800 MHz	2 MB
Conf. 4	7	800 MHz	1	2 GHz	2 MB
Conf. 5	7	800 MHz	1	800 MHz	512 kB

Variable gain =>
programming model matters?

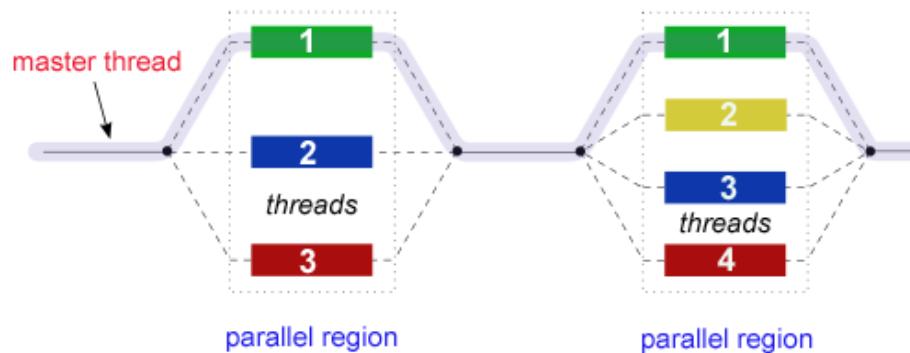


Up to 27% less
with asymmetric
design

Leveraging programming models

OpenMP 3.0 thread-based fork-join model

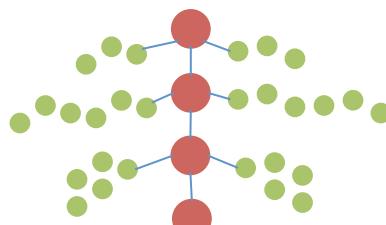
- Symmetric worker threads



OmpSs (or OpenMP 4.0) task-based model

- Design-time task criticality analysis
- Criticality-aware dynamic task scheduling

Task Dependency Graph



Critical task queue

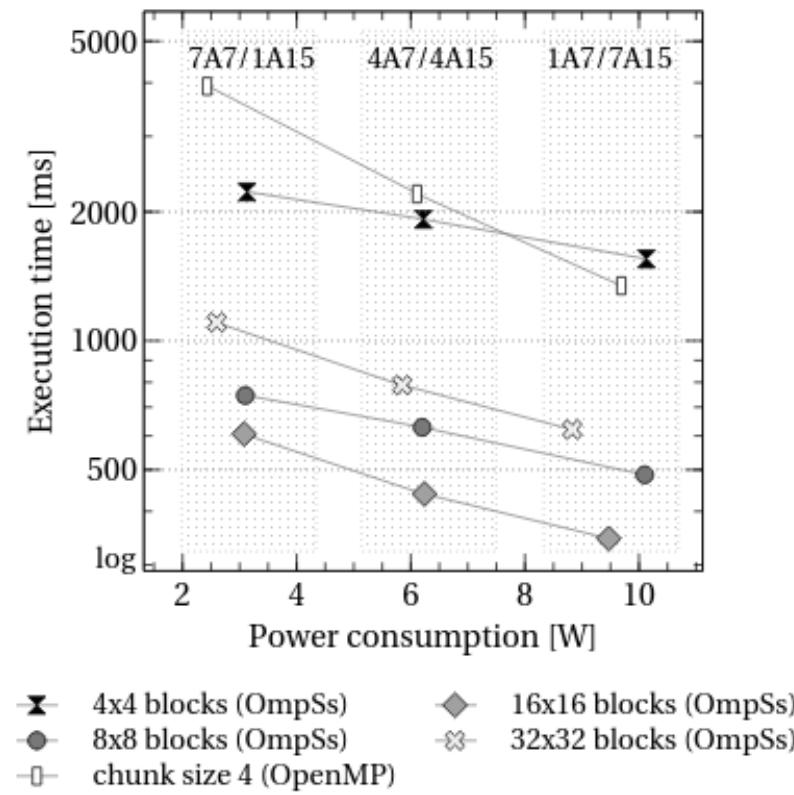


Non-Critical task queue



Leveraging programming models (cont'd)

Example of empirical design evaluation (Cholesky)

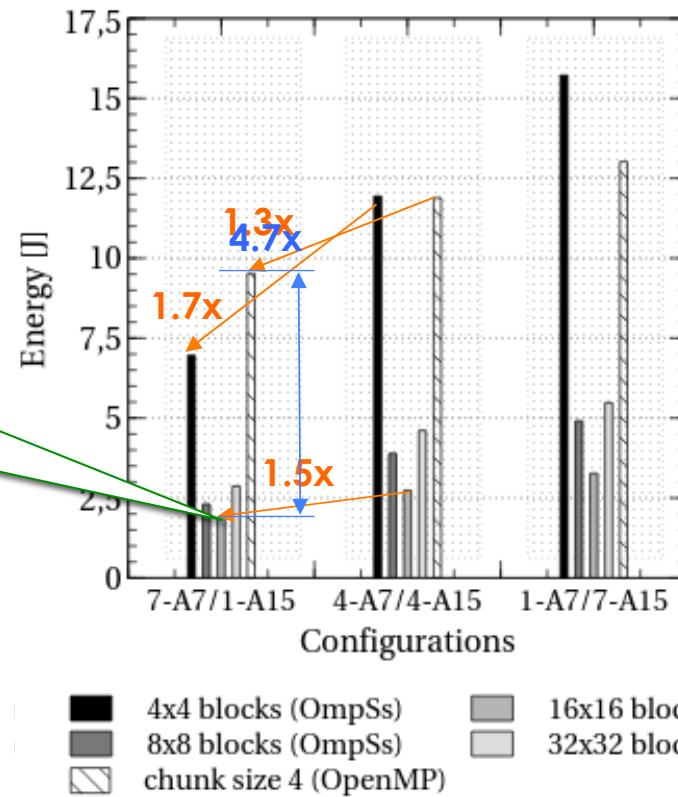


Leveraging programming models (cont'd)

Example of empirical design evaluation (Cholesky)

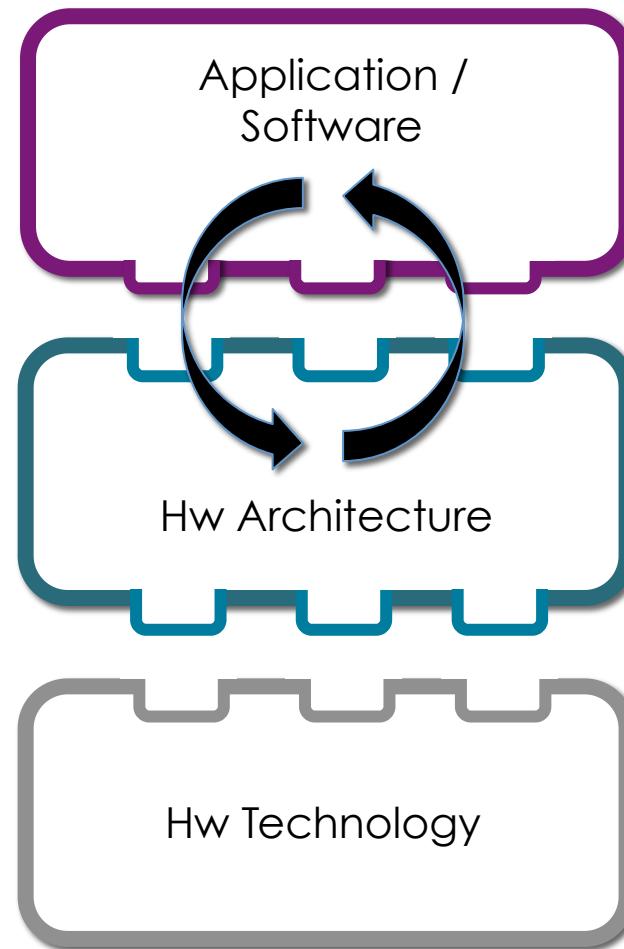
- Architecture design impact
- Programming model impact

Best synergistic
energy-efficient
configuration



Insight #1

Energy efficiency is sensitive to careful combination of heterogeneous architecture configurations with programming models / runtime



ENERGY-EFFICIENT COMPUTE NODE DESIGN

Exploiting emerging memory technologies

Emerging memory technologies

- Magnetic memories: STT-RAM...
- Non volatility, high density, good endurance...
- **Low leakage => negligible static power**

	SRAM	DRAM	Flash (NOR)	Flash (NAND)	FeRAM	MRAM	PRAM	RRAM	STT-RAM
Non-volatile	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Cell size (F²)	50–120	6–10	10	5	15–34	16–40	6–12	6–10	6–20
Read time (ns)	1–100	30	10	50	20–80	3–20	20–50	10–50	2–20
Write / Erase time (ns)	1–100	15	1 µs / 10 ms	1 ms / 0.1 ms	50 / 50	3–20	60 / 120	10–50	2–20
Endurance	10 ¹⁶	10 ¹⁶	10 ⁵	10 ⁵	10 ¹²	>10 ¹⁵	10 ⁸	10 ⁸	>10 ¹⁵
Write power	Low	Low	Very high	Very high	Low	High	High	Low	Low
Other power consumption	Current leakage	Refresh current	None	None	None	None	None	None	None
High voltage required	No	3 V	6–8 V	16–20 V	2–3 V	3 V	1.5–3 V	1.5–3 V	<1.5 V

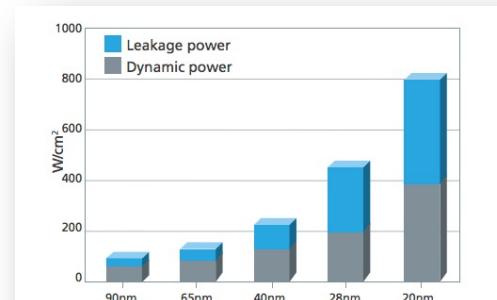
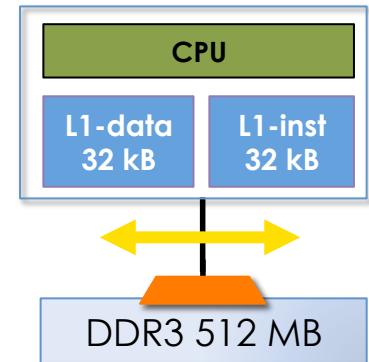


Figure 1: Leakage power becomes a growing problem as demands for more performance and functionality drive chipmakers to nanometer-scale process nodes (Source: IBS).

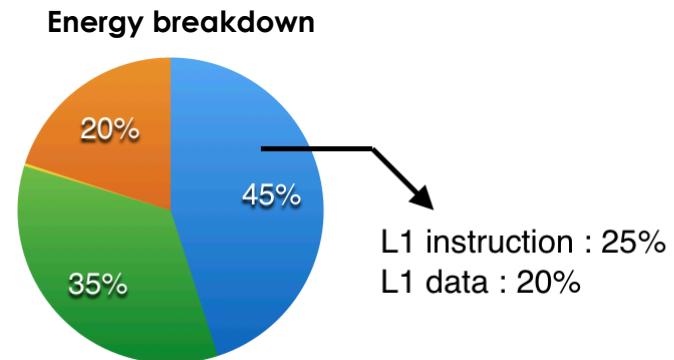
Motivational example

```
volatile int x = 0;
for (i = 0; i<N; i++) {
    x = 0;
}
```



$(N = 5 \cdot 10^7)$	Exec. time (ms)	Energy (mJ)
full-SRAM L1 cache	305.13	79.5
full-STTRAM L1 cache	305.31 (+0.06%)	67.8 (-14.7%)

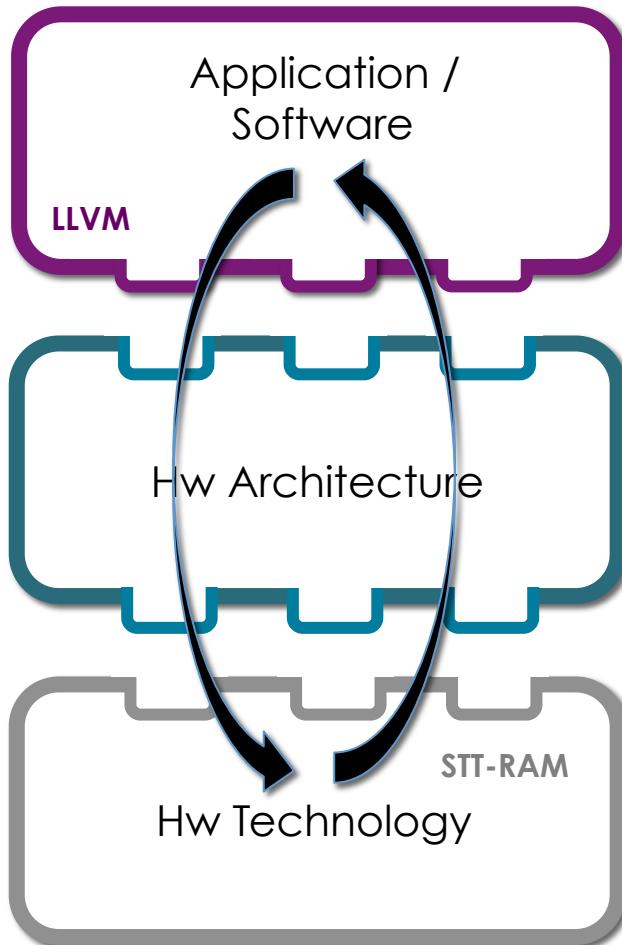
Negligible
static power



Possible gain in dynamic power?

➤ focus on the **expensive writes** in STT-RAM

How to reach energy-efficiency?



- Compiler optimization
- Non volatile memory in cache hierarchy

Silent Stores

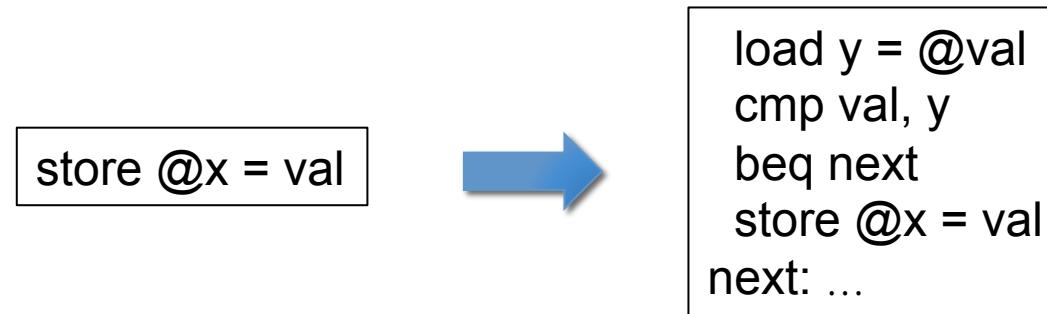
```
volatile int x = 0;  
for (i = 0; i<N; i++) {  
    x = 0; ←  
}
```

redundant (expensive) writes: silent stores

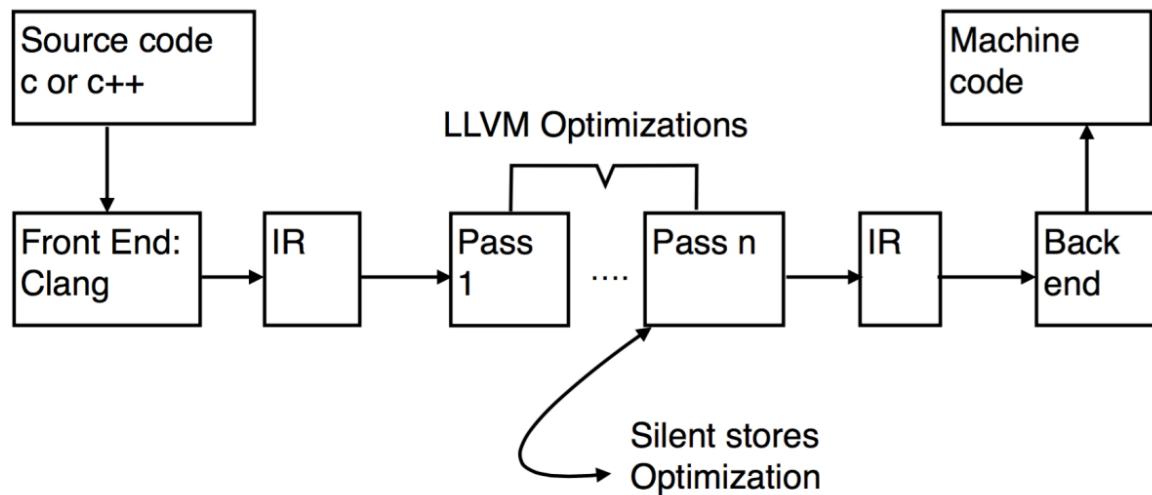
- Definition
 - a store is “**silent**” if it writes the value that is already present in memory location
- Studied in the early 2000’s
 - Kevin M. Lepak, Gordon B. Bell, and Mikko H Lipasti “Silent stores and store value locality”. In IEEE TC 50(11), 2001
 - For monoprocessor performance speedup, and multiprocessor bus traffic improvement
 - Addressed at hardware level

Silent store elimination

- Identify likely silent stores by profiling and replace...



- Code modification at IR level during code generation



Is silent store elimination always profitable?

- This transformation is beneficial iff:

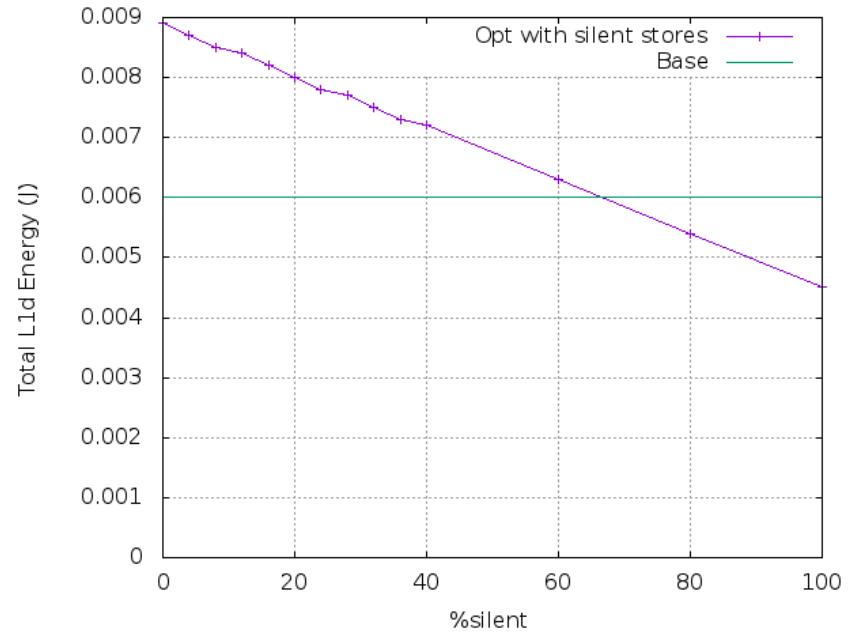
$$\text{Cost}_{\text{load}} + (1 - P_{\text{silent}}) \text{Cost}_{\text{store}} \leq \text{Cost}_{\text{store}}$$

- Example of profitability threshold evaluation

```

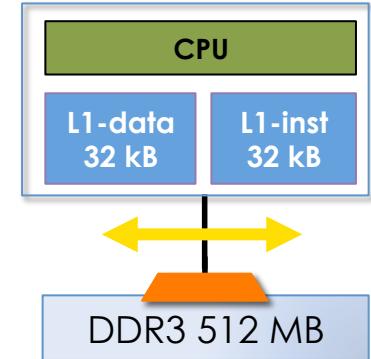
volatile int x;
// assuming N > M
For(i = 0; i<N; i++) {
    y = 0;
    if (i>M)
        y = i;
    x = y; // store
}

```



Motivational example (cont'd)

```
volatile int x = 0;
for (i = 0; i<N; i++) {
    x = 0;
}
```



	Exec. time (ms)	Energy (mJ)
full-SRAM	305.13	79.5
full-STTRAM	305.31 (+0.06%)	67.8 (-14.7%)
full-STTRAM + opt	305.31 (+0.06%)	64.6 (-18.7%)

Synergistic energy improvement:
technology + compilation

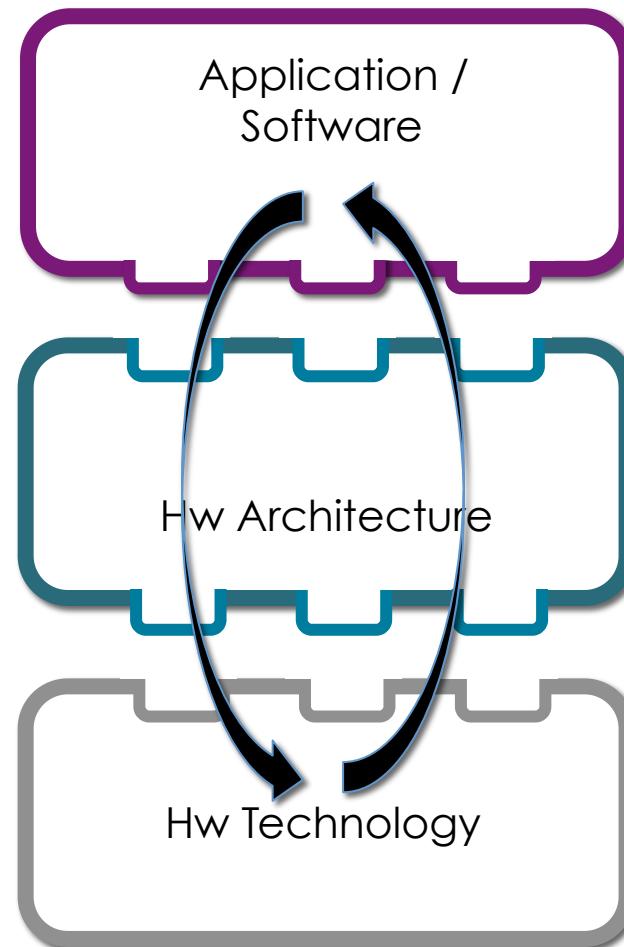
Negligible static power
+
Reduced dynamic power

Related work

- Hardware approach
 - **Early-write termination:** when new value is already in memory, a control signal is generated to switch off write circuit and terminate
 - P. Zhou et al., « Energy reduction for STT-RAM using early write termination », ICCAD, 2009
 - Y. Joo et al., « Energy-and endurance-aware design of phase change memory caches », DATE, 2010.
- Combined hardware and software approach
 - **Data allocation/migration** to reduce the cost of NVM write activities
 - **in hybrid memory:** write-intensive data blocks stored in SRAM banks, read-intensive data blocks stored in NVM banks
 - Q. Li et al. « MAC: migration-aware compilation for STT-RAM based hybrid cache in embedded systems », ISPLED'12
 - Q. Li et al. « Compiler-assisted preferred caching for embedded systems with STT-RAM based hybrid cache », LCTES'12
 - J. Hu et al. « Data Allocation Optimization for Hybrid ScratchPad Memory with SRAM and Non-volatile Memory », IEEE VLSI 2013

Insight #2

Low-leakage emerging memory technology can be leveraged with portable software-level optimizations for improved energy-efficiency

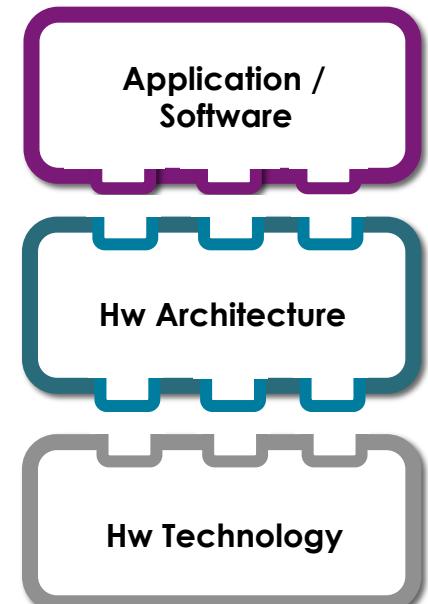


Outline of this talk

- **Synergistic design of heterogeneous compute nodes**
 - Motivation and requirements
- **Examples of synergistic designs for energy-efficiency**
 - Adaptive multicore compute node design
 - Leveraging non volatile memory with compiler optimization
- **Conclusion**

Conclusion

- Energy-efficiency is a major challenge in compute node design
- Energy-efficiency opportunities are present across different design layers
- Synergistic design as a solution
 - Design frameworks
 - Cross-domain interaction
- **Open question:** How to deal with multi-level optimization composability?



SYNERGISTIC DESIGN OF ENERGY-EFFICIENT HETEROGENEOUS COMPUTE NODES

Abdoulaye Gamatié

LIRMM / CNRS-UM, Montpellier

Colloque National du GDR SOC2, June 2017, Bordeaux

*** Joint work: F. Bruguier, A. Butko, M. Novaes, P.Y. Péneau, F. Pereira, M. Robert, G. Sassatelli, S. Senni, and L. Torres