Managing MPSoCs beyond their Thermal Design Power*

Luca Benini

Università di Bologna & STMicroelectronics

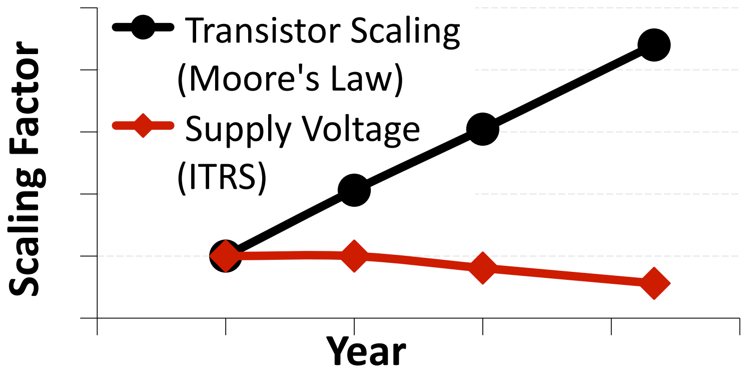
Luca.benini@unibo.it

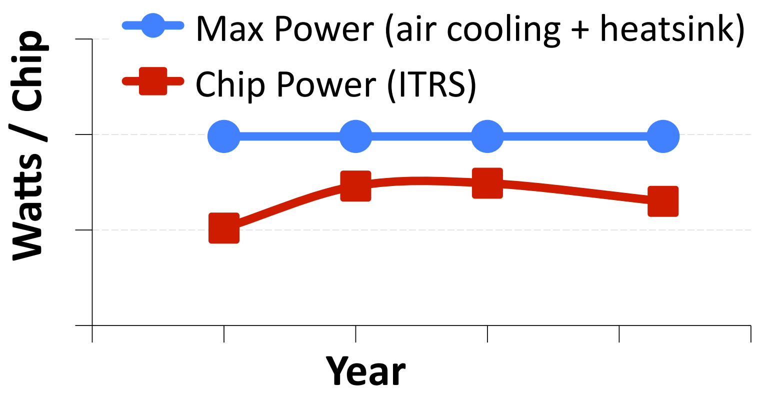
*Work supported by INTEL, FP7 THERMINATOR, FP7 Artist-Design

Thermal Power Wall

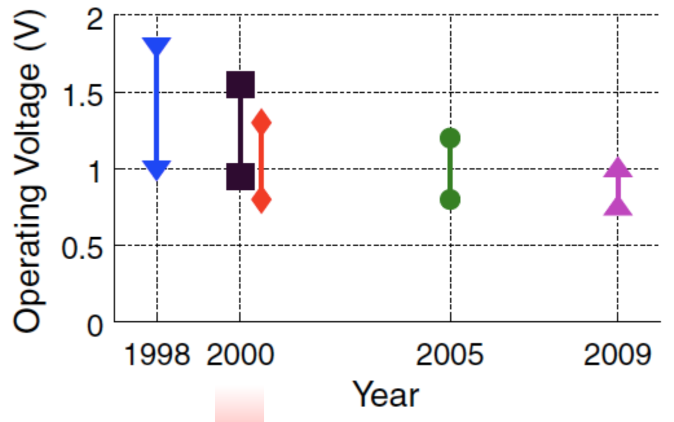
Transistor count increases exponentially, but…

Can no longer power the entire chip (voltages, cooling do not scale)





Traditional HW power-aware techniques are insufficient (e.g., voltage-freq. scaling)



[Watanabe et al., ISCA'10]

Dark Silicon !!!

[Hardavellas11]

Mobile SoCs are cool…right?

Wrong!

So.. got myself a HTC (T-Mobile) HD2 […].
As i found out the problem is pretty common: damn thing restart itself - **thermally related - the old CPU overheat problem**. By searching the net I found out that it's pretty common with some HTC models. HD2 has it, Desire has it, Nexus One has it, hell even some xperia models have it.. about half of the devices powered by anything from the Snapdragon series could have it.

June 2011 - http://forum.xda-developers.com/showthread.php?t=982454

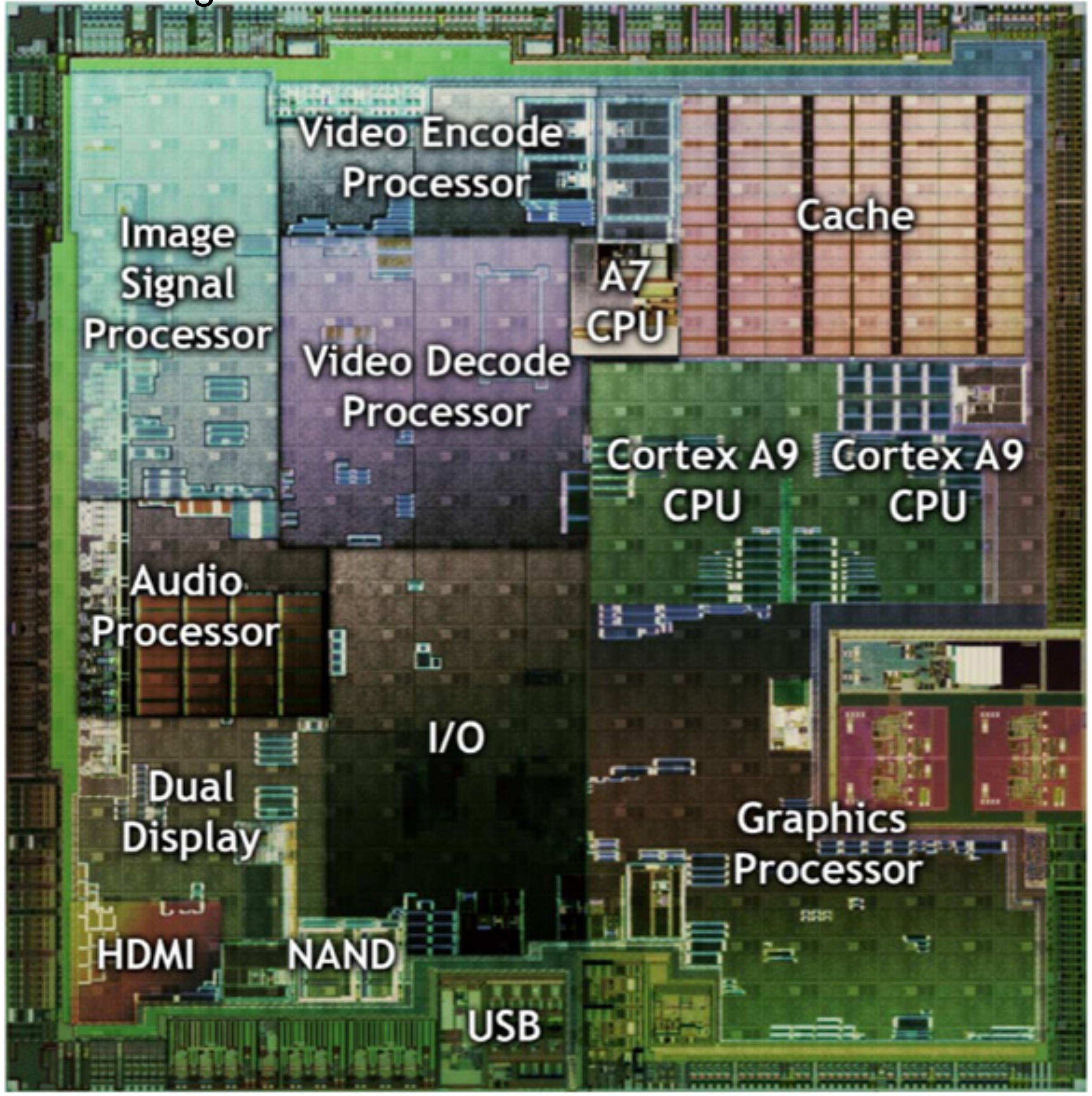
Why?

ARM has unveiled its next generation "Eagle" (Cortex A15) processor, pitching it at **everything from smartphones to energy-efficient servers**. The A15 will be initially produced at 32nm or 28nm, although ARM claims the roadmap stretches down to 20nm. It will **deliver clock speeds of up to 2.5GHz**.

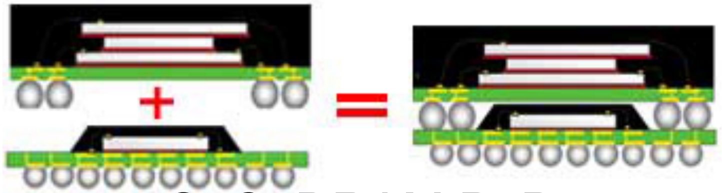
Aug. 2011 - http://www.pcpro.co.uk/news/360994/arm-preys-on-smartphones-and-servers-with-eagle

A 2011 Mobile SoC

NVIDIA Tegra II SoC



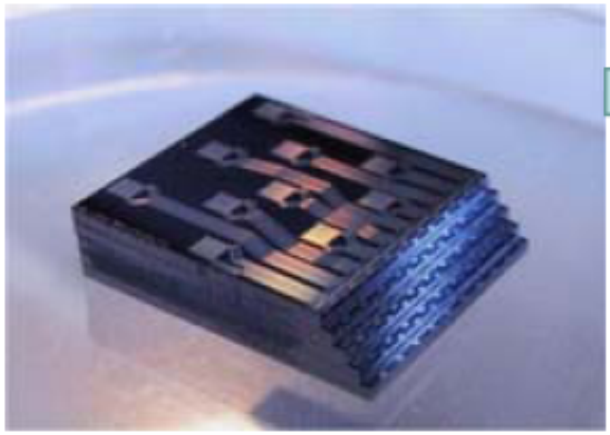
- Tegra II
  - TSMC 40nm (LP/G)
  - A9 - 1GHz (G)
  - GPU, etc. - 330MHz (LP)
  - GEForce ULV (8 shaders)
  - 2 separate Vdd rails
  - 1MB L2$
  - 32b LPDDR2 (600MHz DR)
- Tegra II 3D
  - A9 – 1.2GHz
  - GPU – 400Mhz



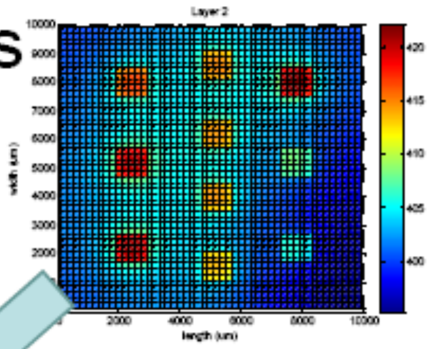
SoC+DRAM PoP

3D-SoCs are even worse

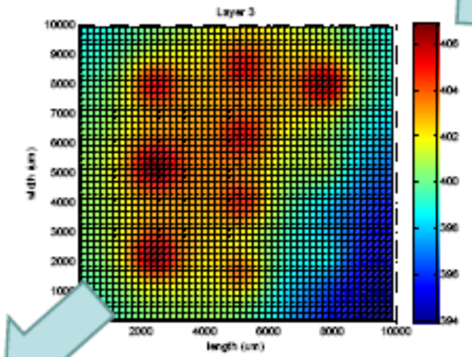
5-tier 3D stack: 10 heat sources and sensors



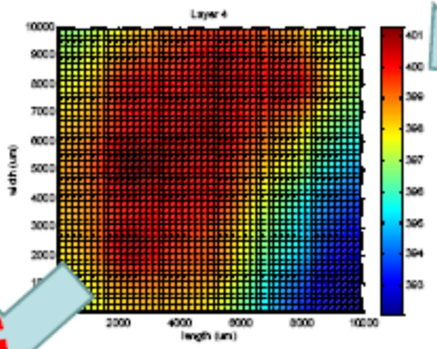
Inject between 4W – 1.5W



2nd Tier

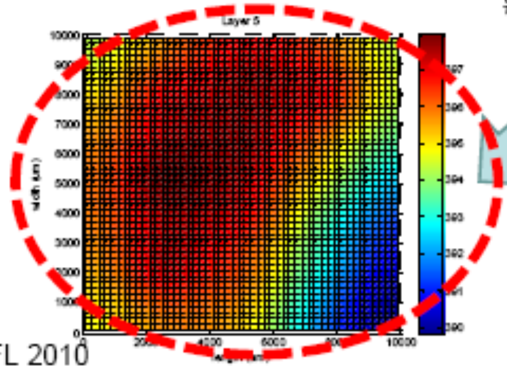


3rd Tier



4th Tier

5th Tier

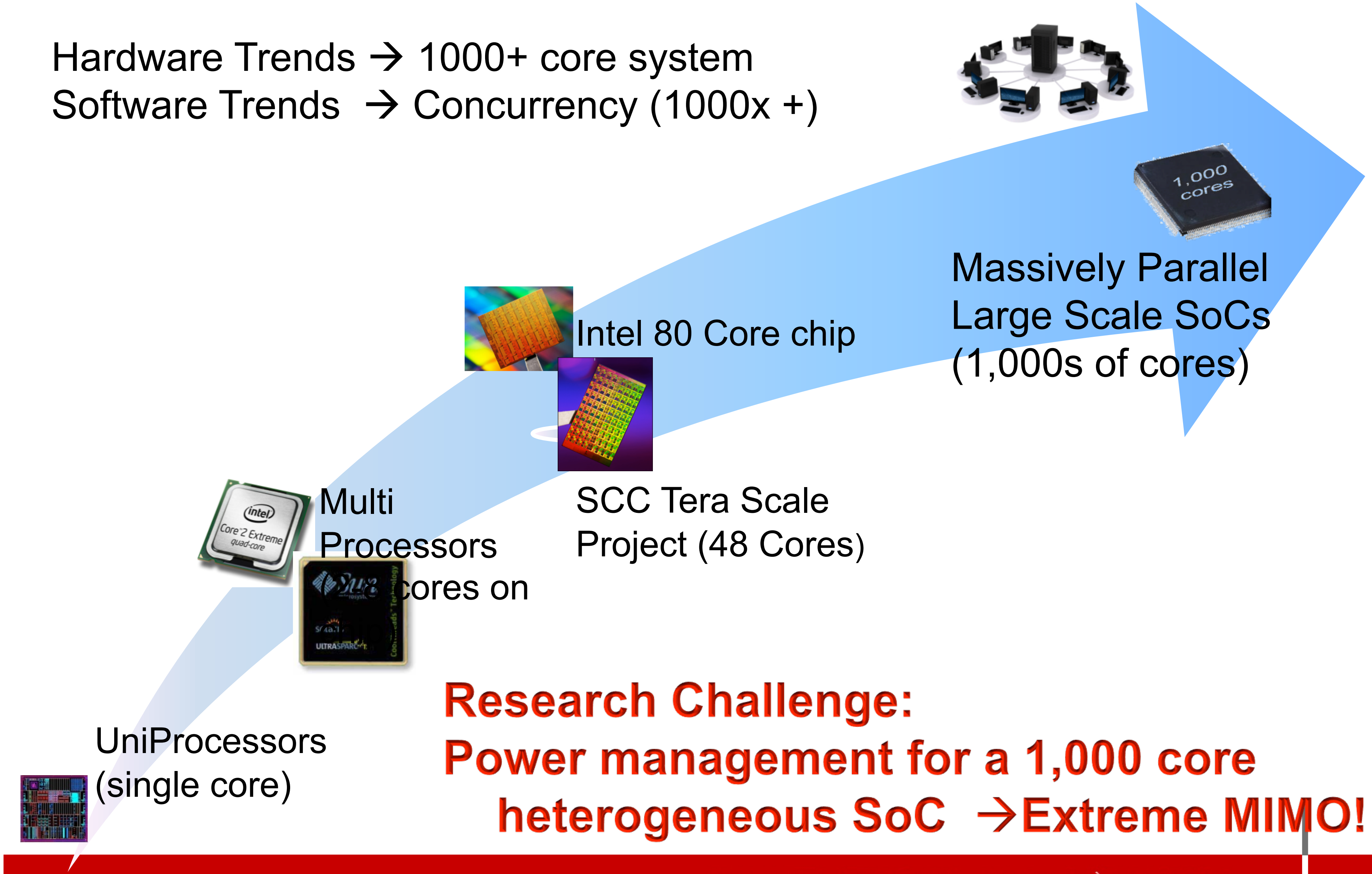


Large and non-uniform heat propagation! (up to 130° C on top tier)

ESL/EPFL 2010

Rushing to Many-Core

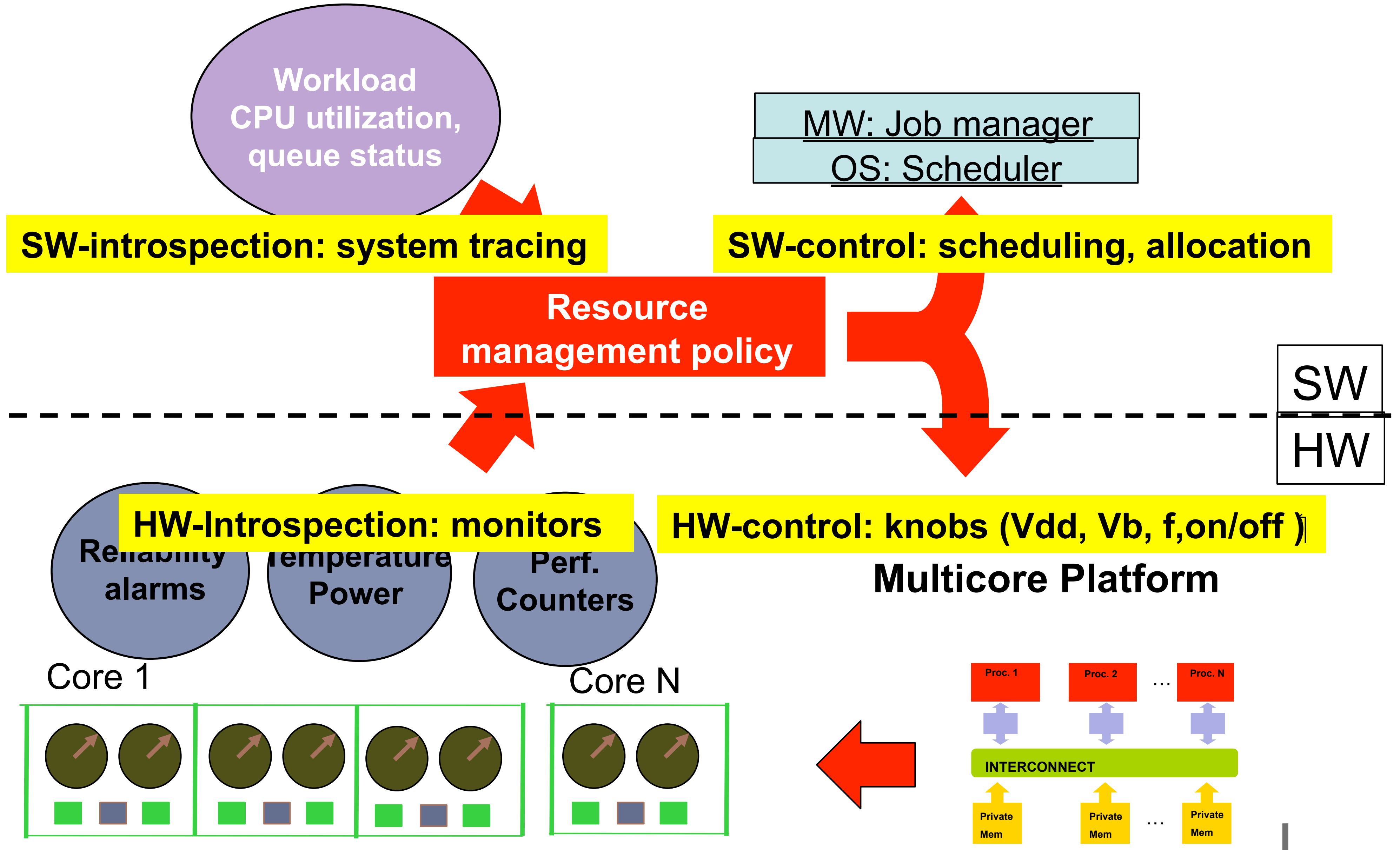
Hardware Trends → 1000+ core system
Software Trends → Concurrency (1000x +)



1,000 cores
Massively Parallel Large Scale SoCs (1,000s of cores)
Intel 80 Core chip
SCC Tera Scale Project (48 Cores)
Multi Processors cores on
intel Core 2 Extreme quad-core
Sun ULTRASPARC T
UniProcessors (single core)

Research Challenge:
Power management for a 1,000 core heterogeneous SoC →Extreme MIMO!

Management Loop: Holistic view



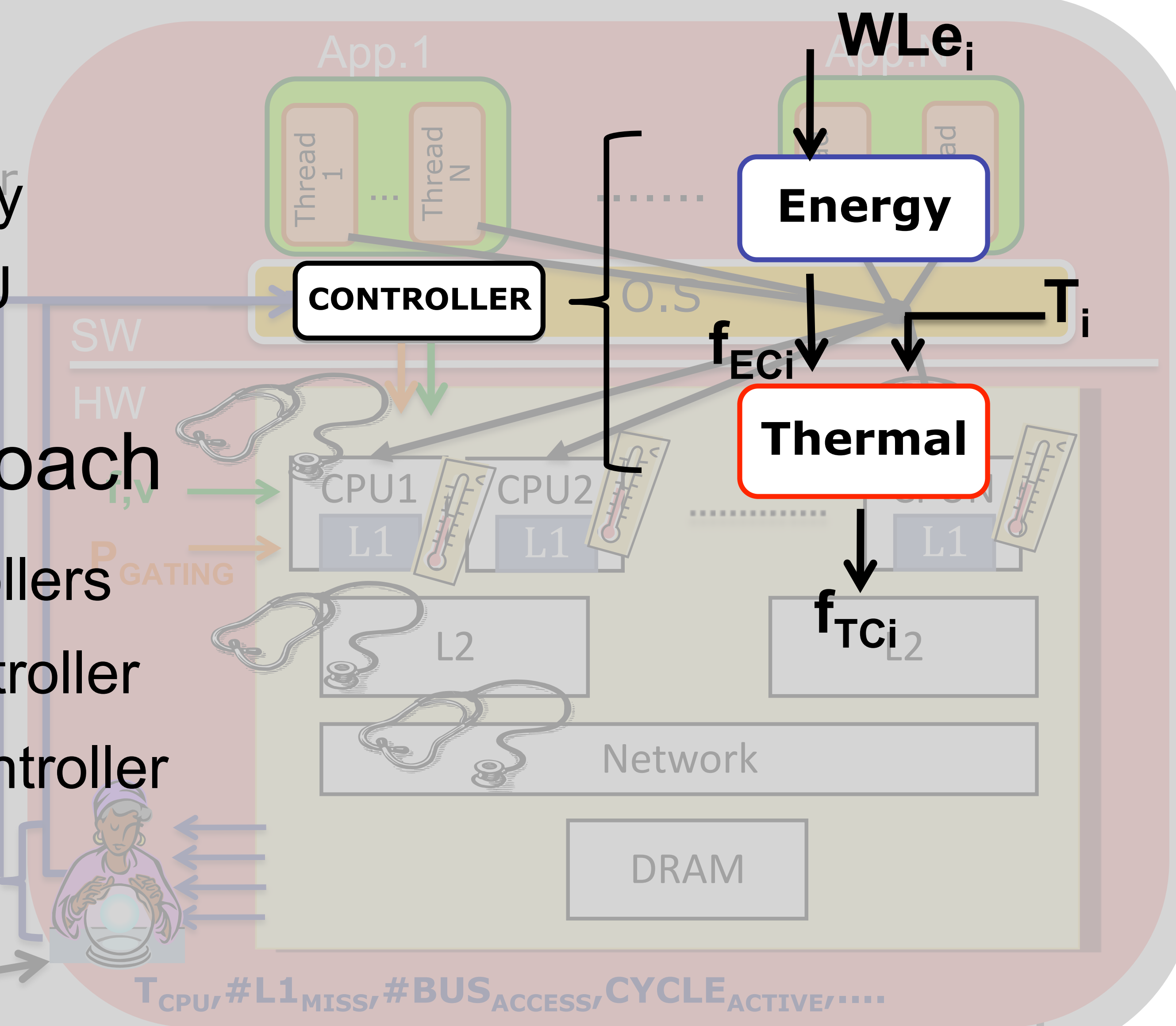
Outline

- Introduction
- Scalable Control
- Scalable model learning
- Experimental Environment
- Challenges ahead

DRM - General Architecture

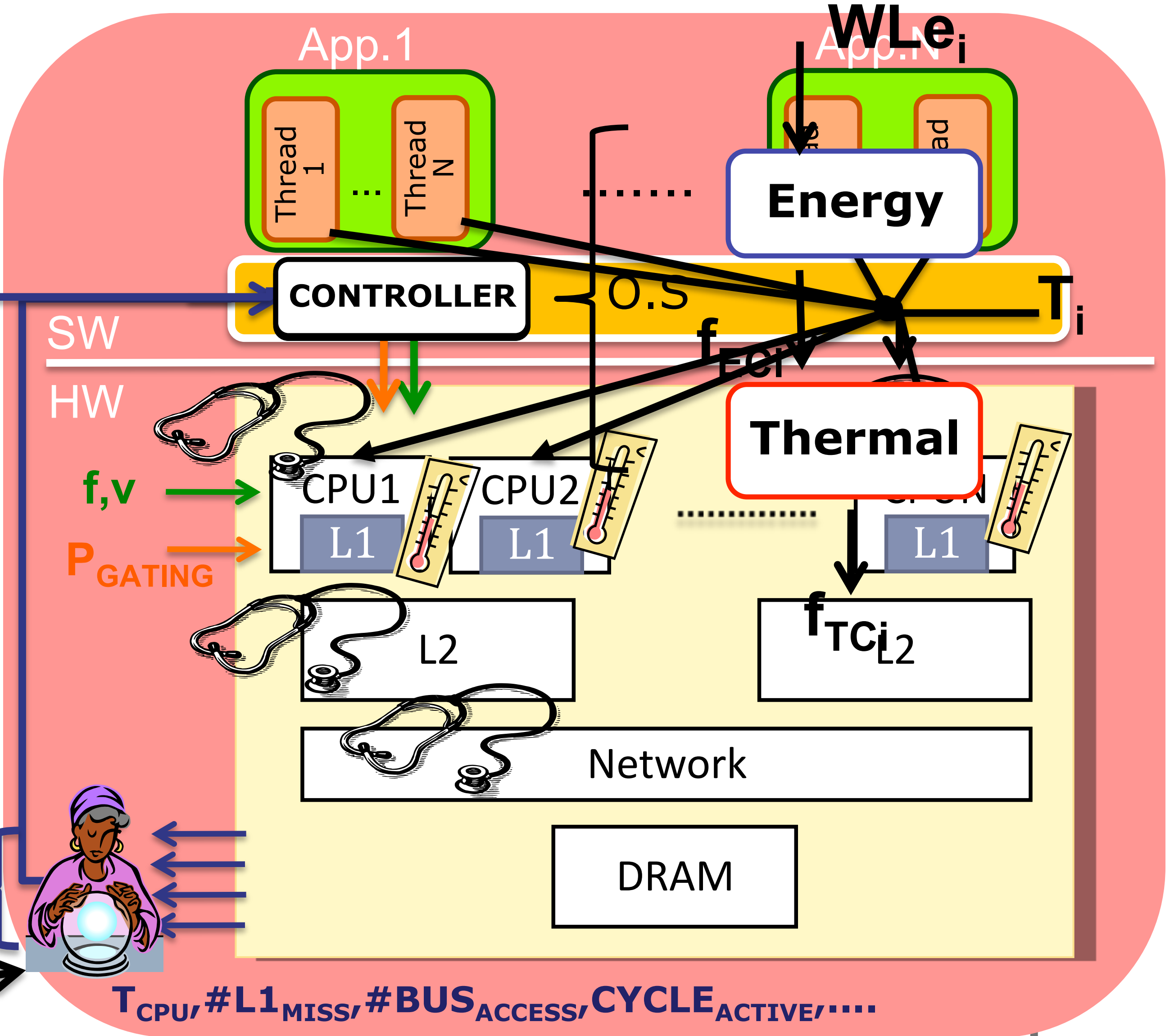
- Controller
  - Minimize energy
  - Bound the CPU temperature
- Layered approach
  - Stack of controllers
    - Energy controller
    - Thermal controller

- System (Chip Scale)
- Sensors
  - Performance counter - PMU
  - Core temperature
- Actuator - Knobs
  - ACPI states
    - P-State → DVFS
    - C-State → P_{GATING}
  - Task allocation
- Controller
  - Reactive
    - Threshold/Heuristic
    - Controller theory
  - Proactive
    - Predictors



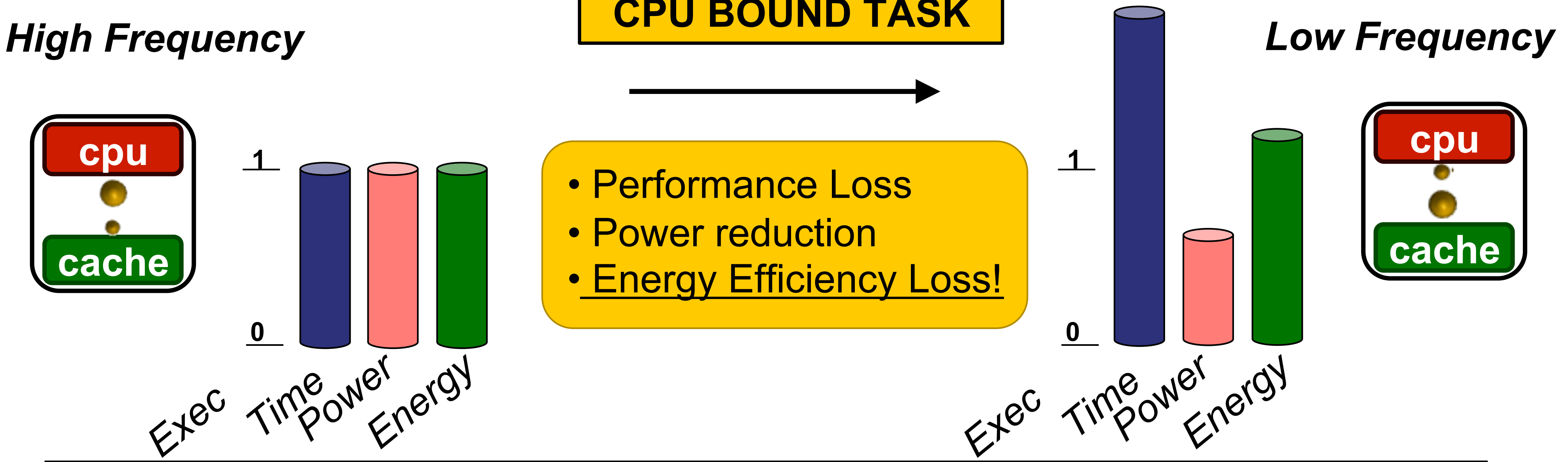
DRM - General Architecture

- System (Chip Scale)
- Sensors
  - Performance counter - PMU
  - Core temperature
- Actuator - Knobs
  - ACPI states
    - P-State → DVFS
    - C-State → P_{GATING}
  - Task allocation
- Controller
  - Reactive
    - Threshold/Heuristic
    - Controller theory
  - Proactive
    - Predictors

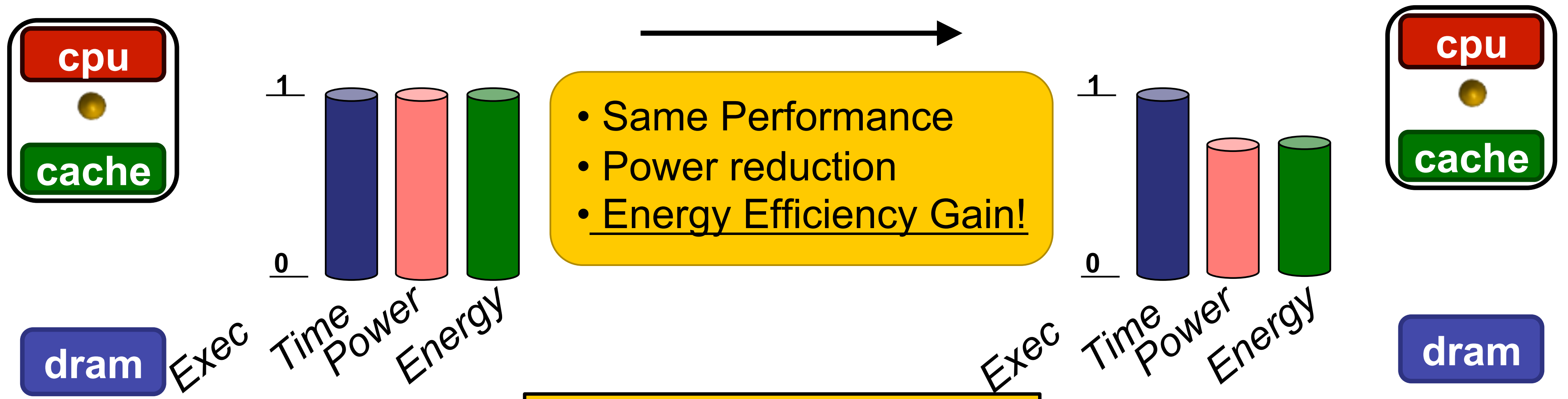


Energy Controller

CPU BOUND TASK



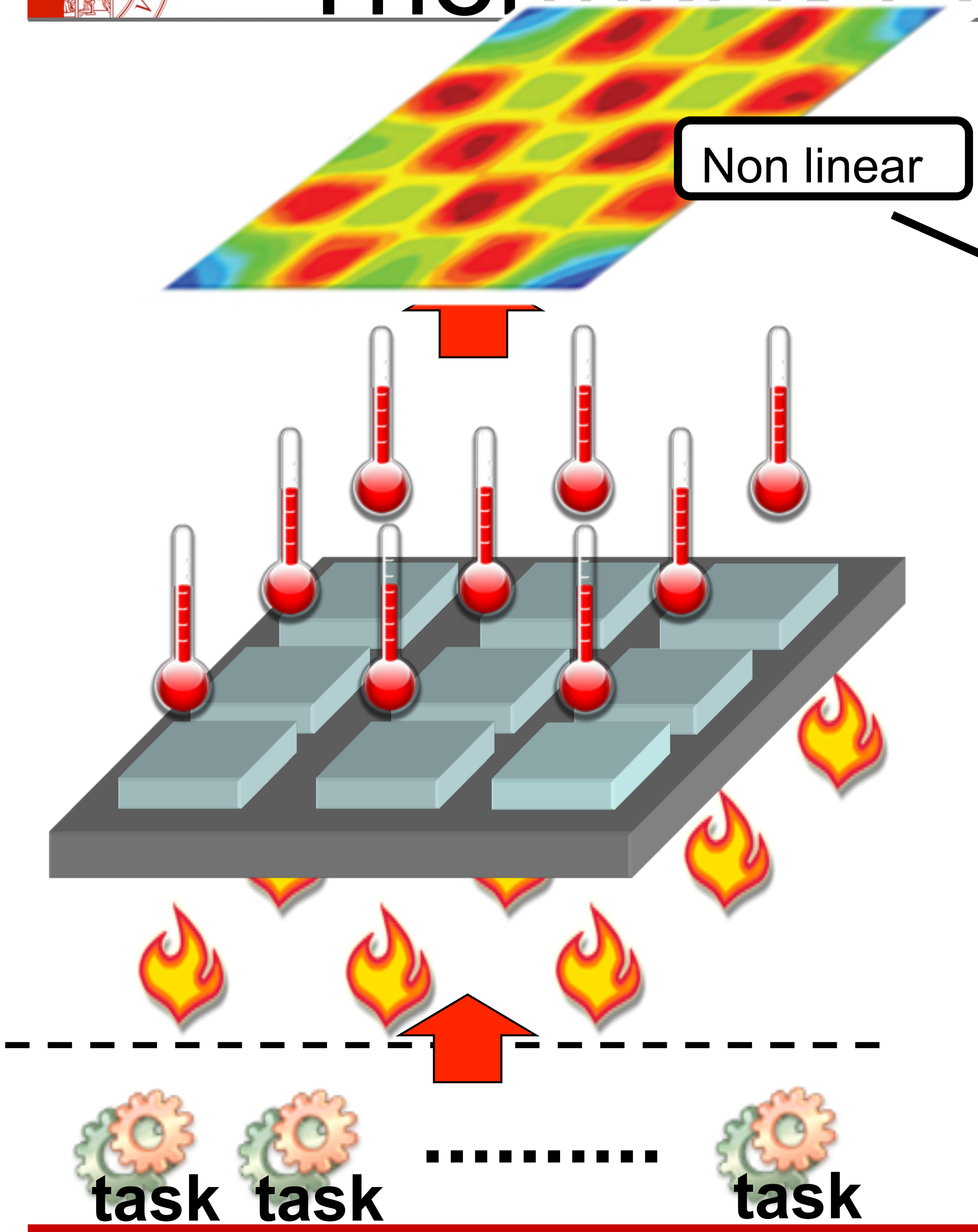
- Performance Loss
- Power reduction
- Energy Efficiency Loss!

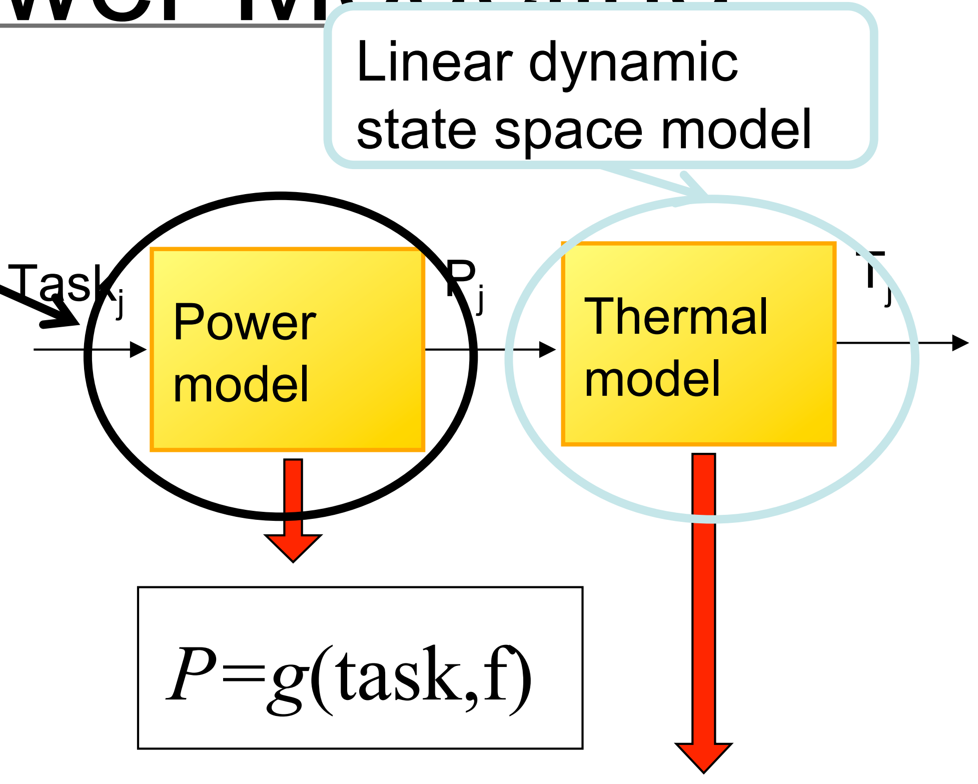


- Same Performance
- Power reduction
- Energy Efficiency Gain!

MEMORY BOUND TASK

Thermal & Power Modeling





$$\boxed{P=g(\text{task},f)}$$

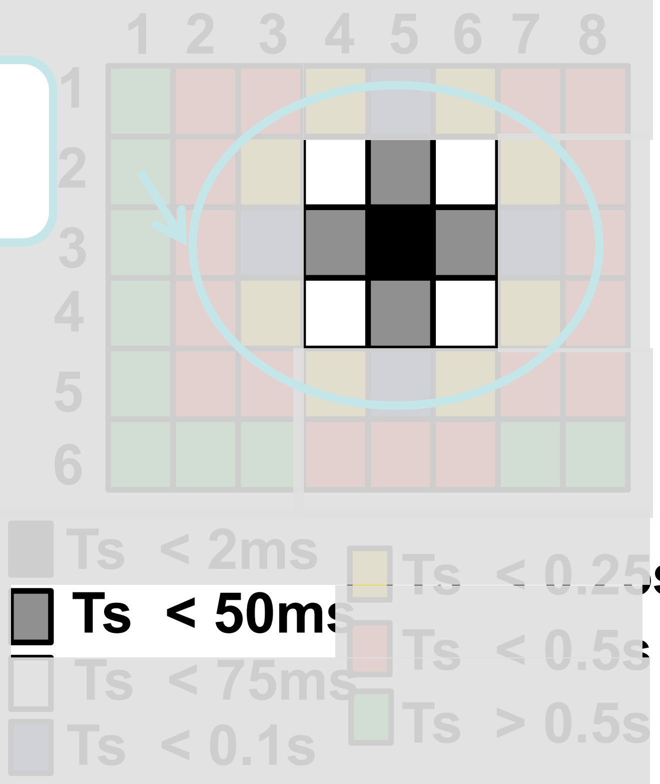
$$x_{n+1} = Ax_n + BP_{n,j}$$
$$T_{n,j} = Cx_n$$

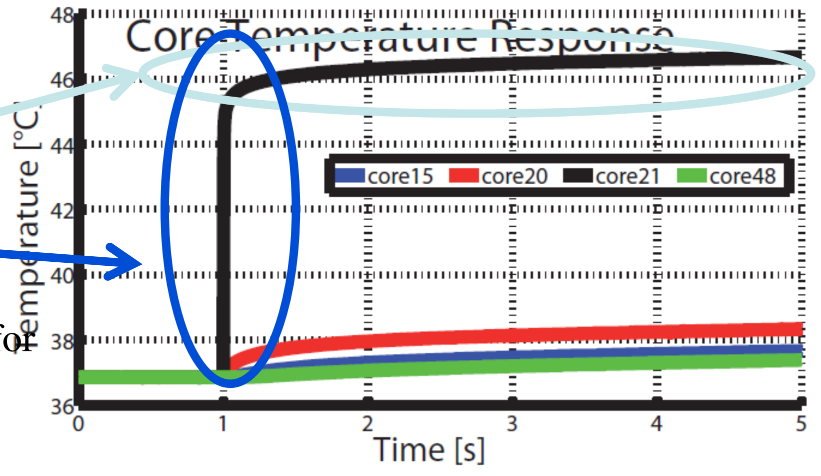
Thermal transient

Thermal locality (Direct Fourier law implications)

- Continuous model:
  - Thermal neighborhood = Physical neighborhood
- Discrete model:
  - Thermal neighborhood depends on sample time
- Hotspot simulation of 'Intel SCC like' 48core
  - Each core : Area = 11.82mm2, Pmax = 2.6W
  - We powered on only Core(5,3)
  - T neighborhood > +0.1°C
- Thermal transient – Model Order
  - Different materials reflects in different time constants [1]
    - Silicon die, heat spreader, heat sink
    - Second order model

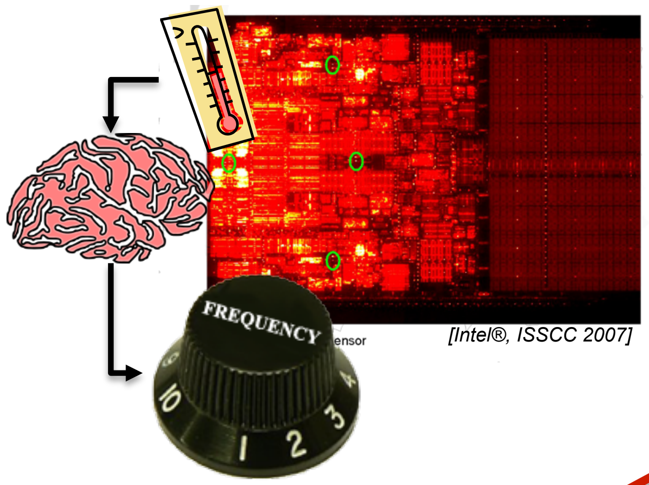
O.S time scale Neighborhood





[1] W. Huang Differentiating the roles of IR measurement and simulation for power and temperature-aware design 2009.

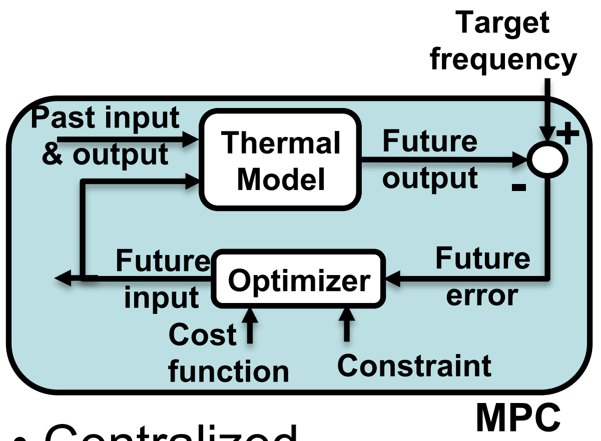
Thermal Controller


[Intel®, ISSCC 2007]

COMPLEXITY

Model Predictive Controller

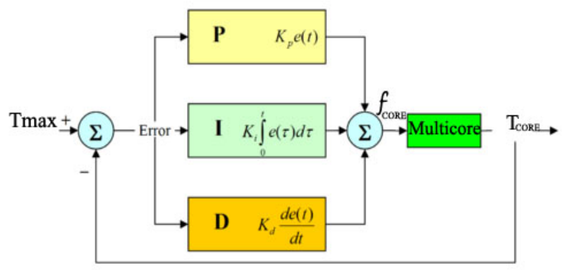
•Internal prediction:
avoid overshoot
•Optimization:
maximizes performance



- Centralized
  - aware of neighbor cores thermal influence
  - All at once – MIMO controller
- Complexity !!!

Classical feed-back controller

- PID controllers
- Better than threshold based approach
- Cannot prevent overshoot



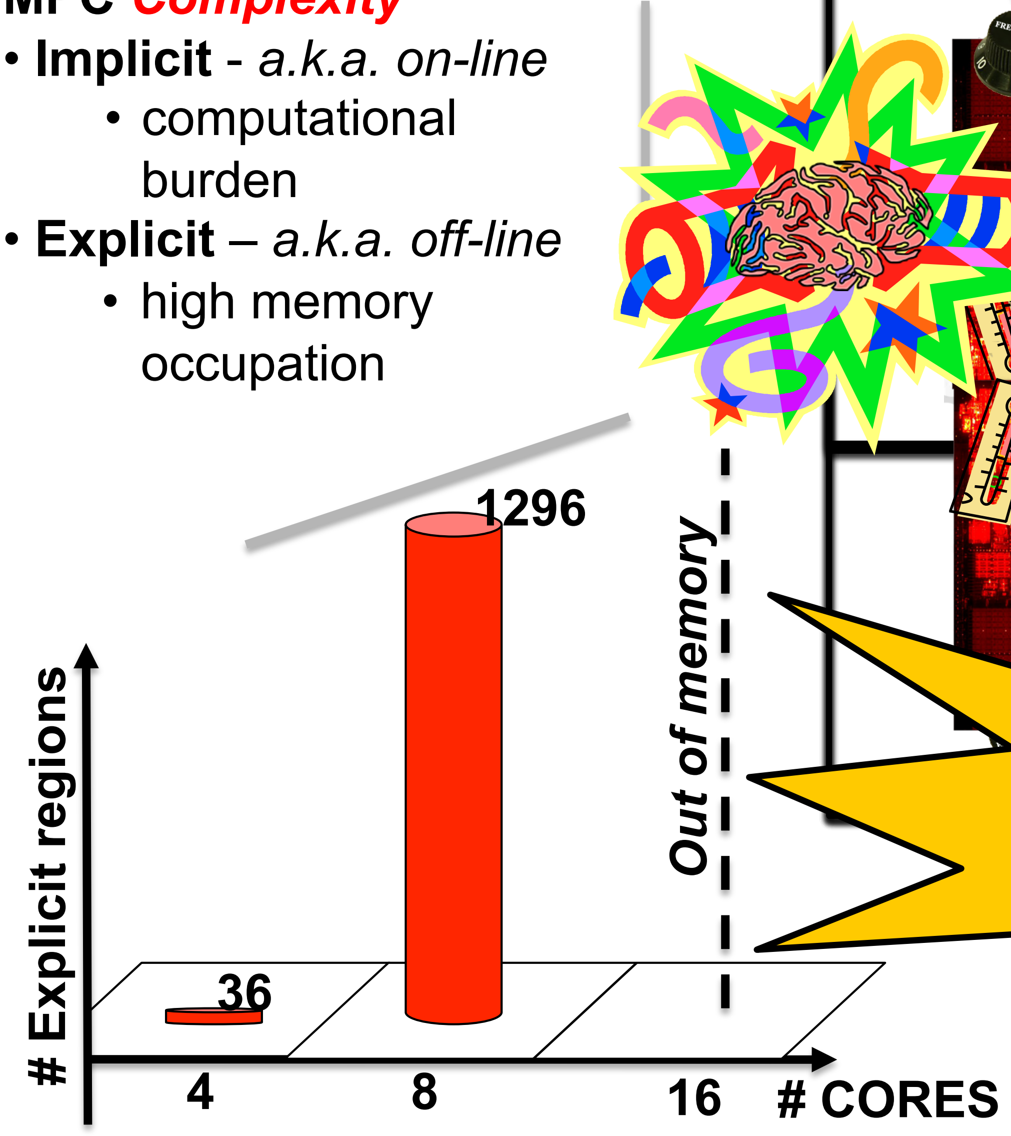
Threshold based controller

•T > Tmax → low freq
•T < Tmin → high freq
• cannot prevent overshoot
• thermal cycle

MPC Scalability

MPC *Complexity*

- **Implicit** - *a.k.a. on-line*
  - computational burden
- **Explicit** – *a.k.a. off-line*
  - high memory occupation



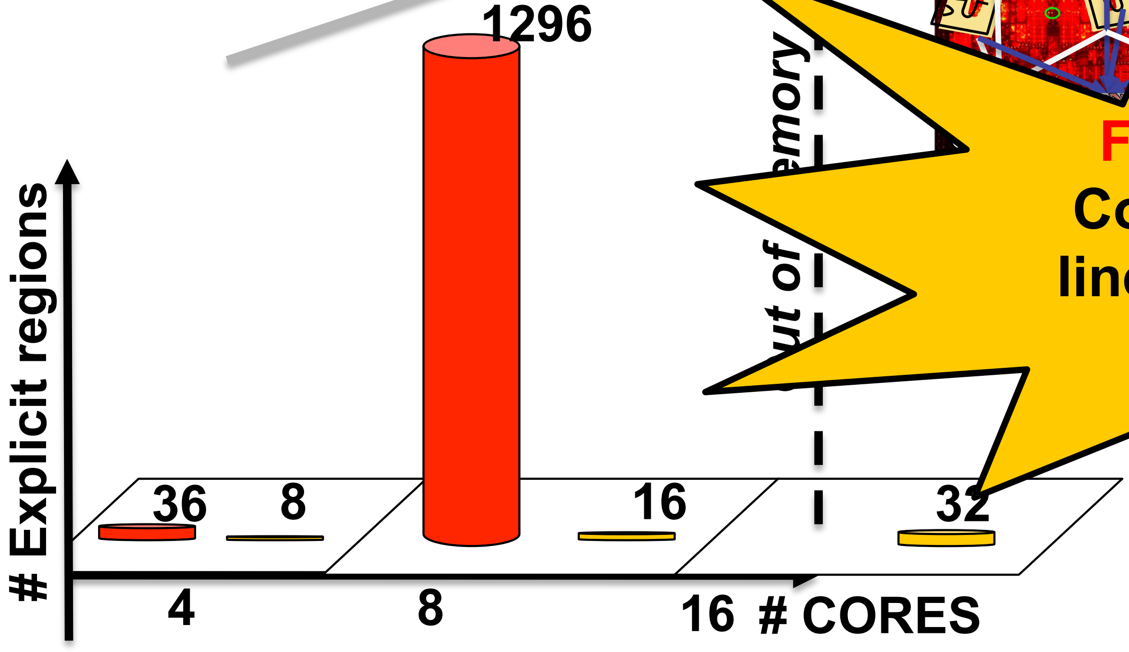
Complexity grows *superlinearly* with number of cores!!

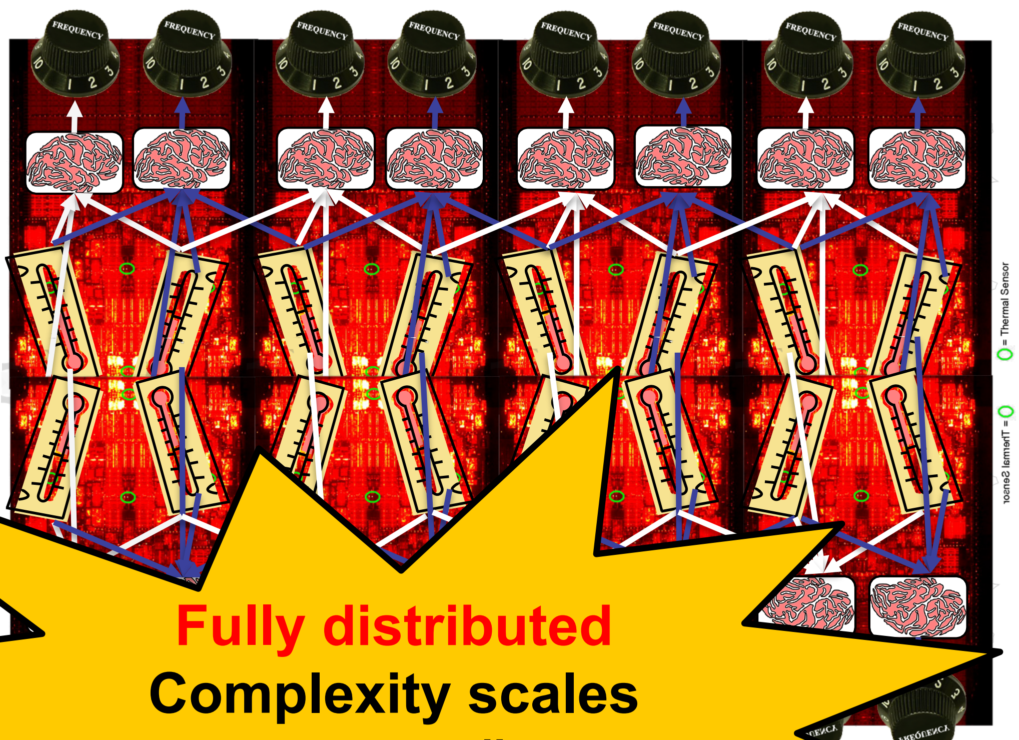
Addressing Scalability

On a short time window, power has a local thermal effect!

One controller for each core
Controller uses:

- local power & thermal model
- neighbor's temperatures



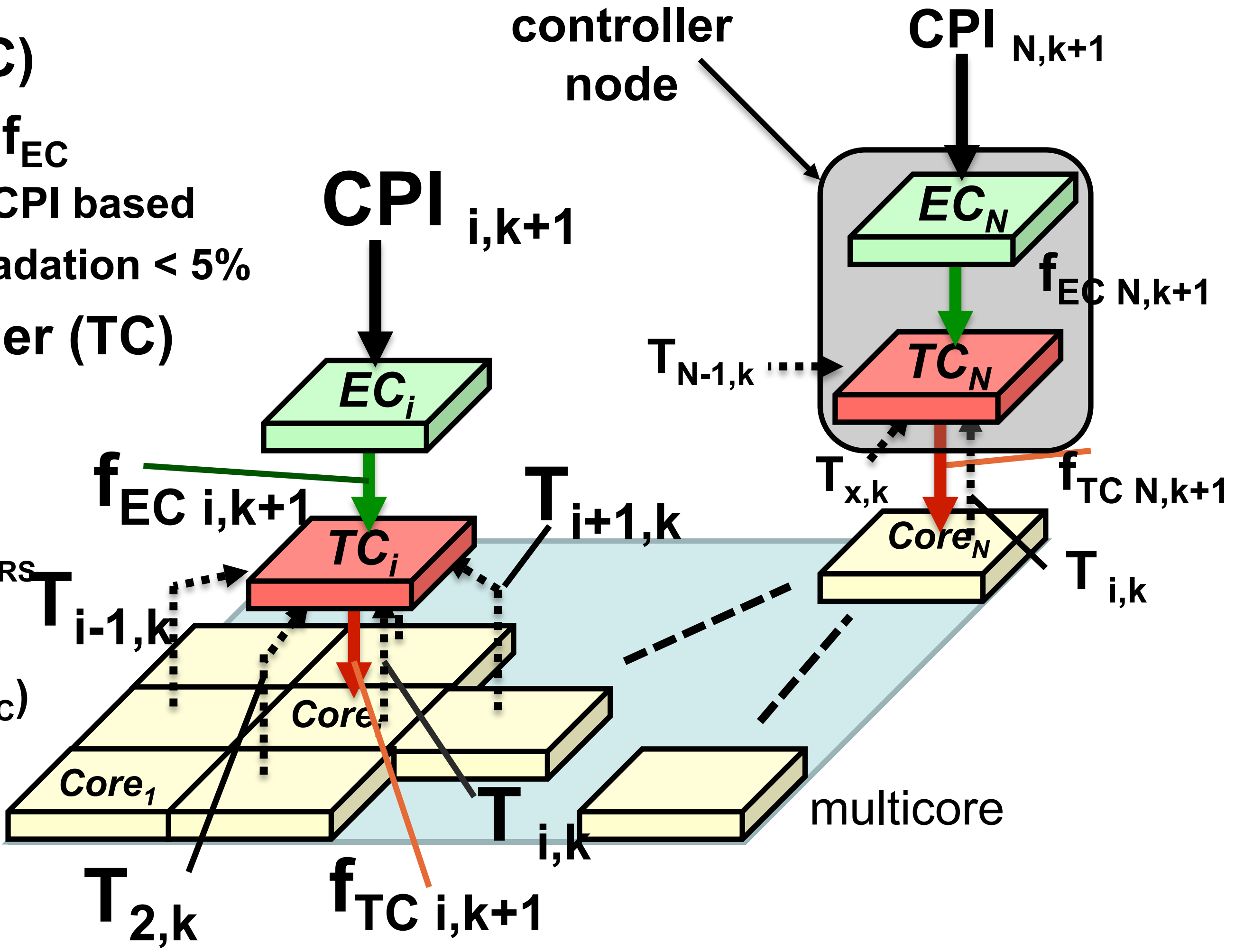


Fully distributed
Complexity scales linearly with #cores

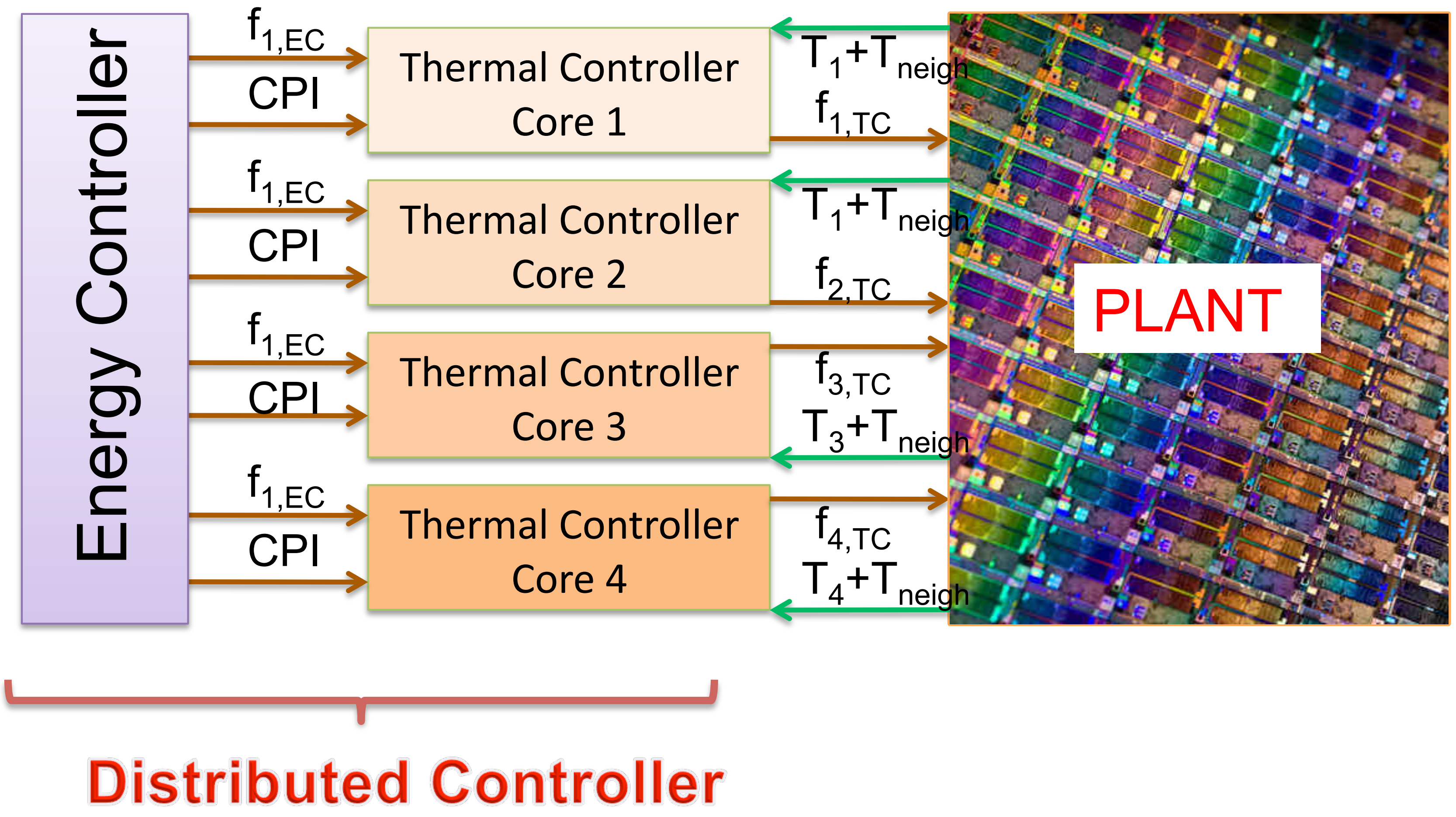
Distributed Control

Distributed and hierarchical controllers:

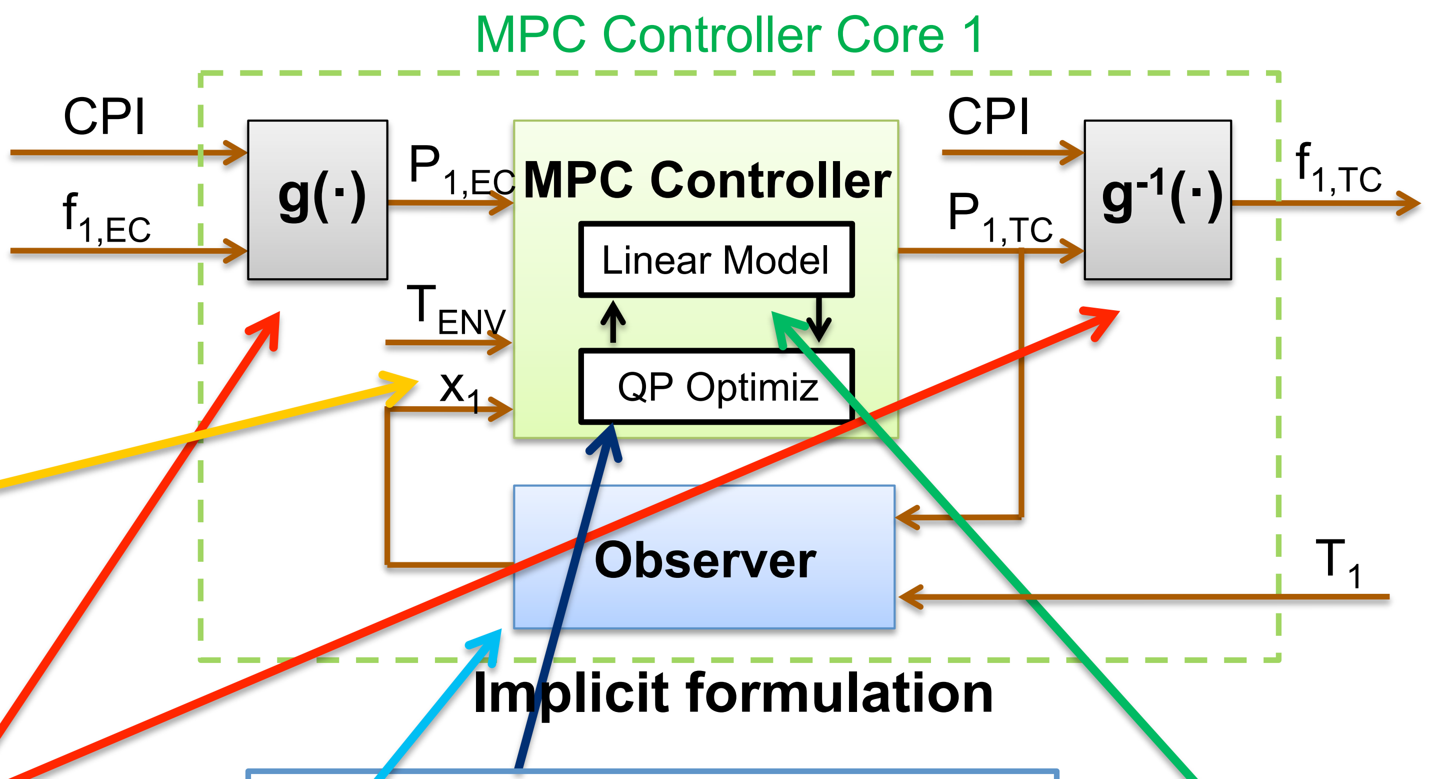
- Energy Controller (EC)
  - Output Frequency f_{EC}
    - Minimize power – CPI based
    - Performance degradation < 5%
- Temperature Controller (TC)
  - Distributed MPC
  - Inputs:
    - f_{EC}, T_{CORE}, $T_{NEIGHBOURS}$
  - Output
    - Core frequency (f_{TC})



High Level Architecture



Distributed Thermal Controller



Implicit formulation

$$\min \sum_{k=0}^{N-1} \| Q \cdot (f_{TC,1}(t+k) - f_{EC,1}(t+k)) \|_2$$

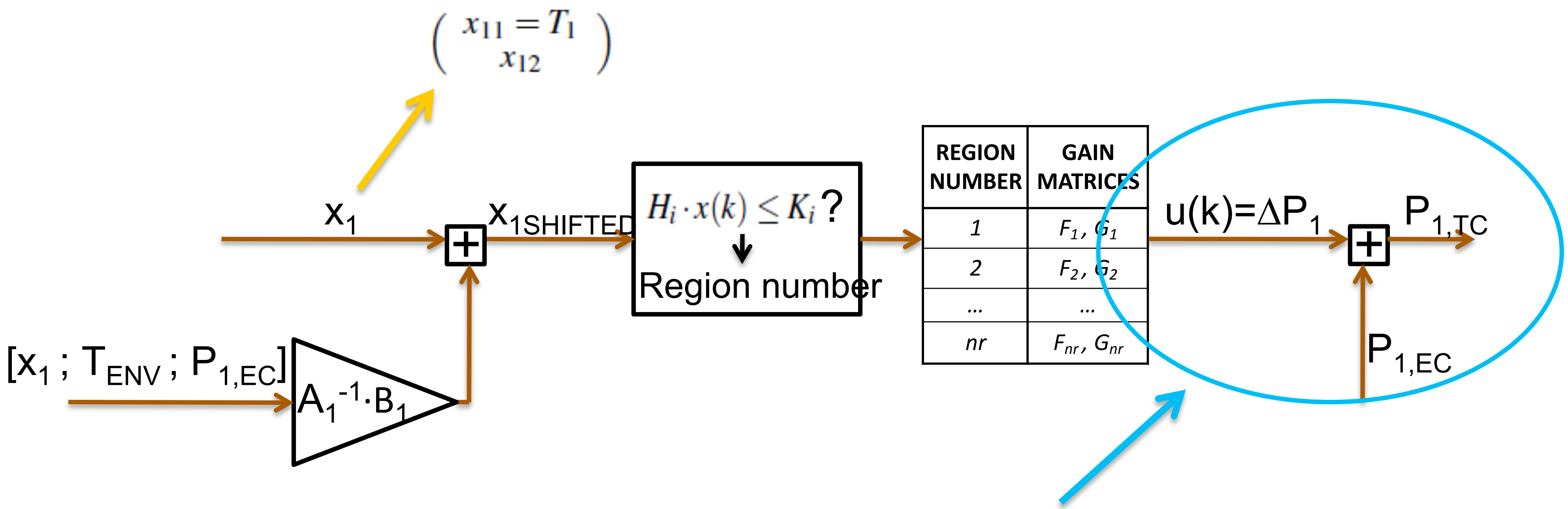
s.t

$$0 \le T_1(t+k|t) \le T_{MAX,1}$$

$$\hat{x}(k+1) = A_1 \cdot \hat{x}(k) + B_1 \cdot u(k) + L_1 \cdot (T_1 - C_1 \cdot \hat{x}(k))$$

Classic Luenberger state observer

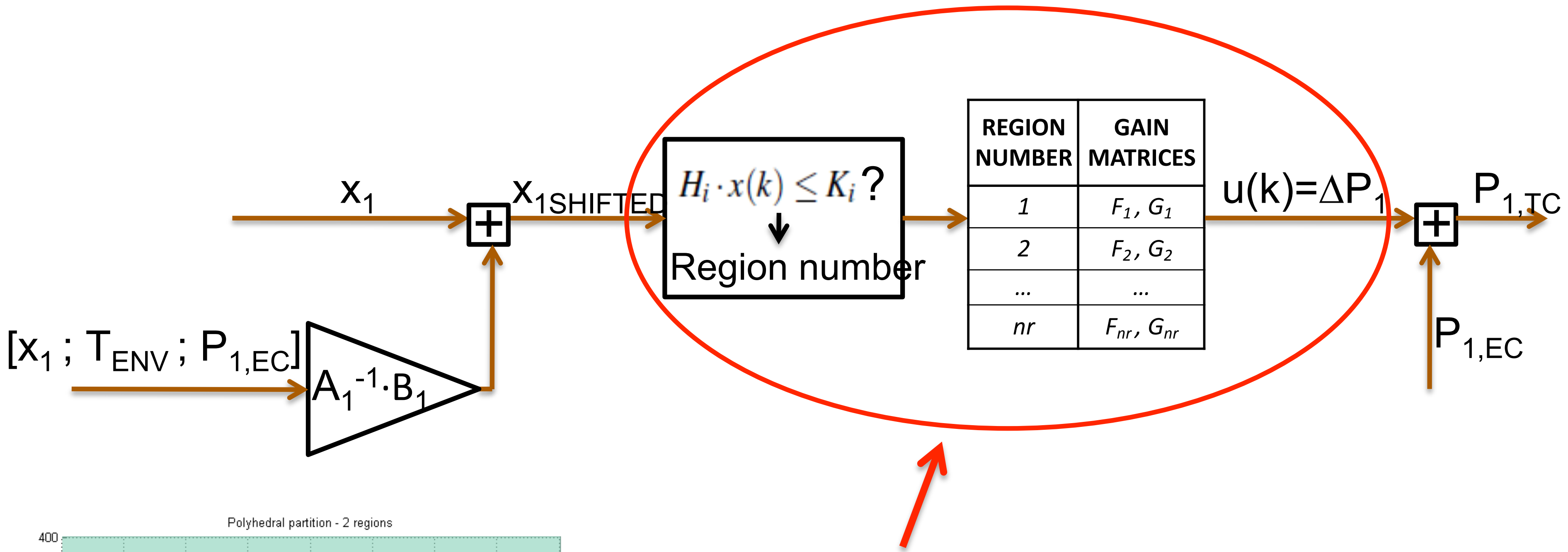
Explicit Distributed Controller



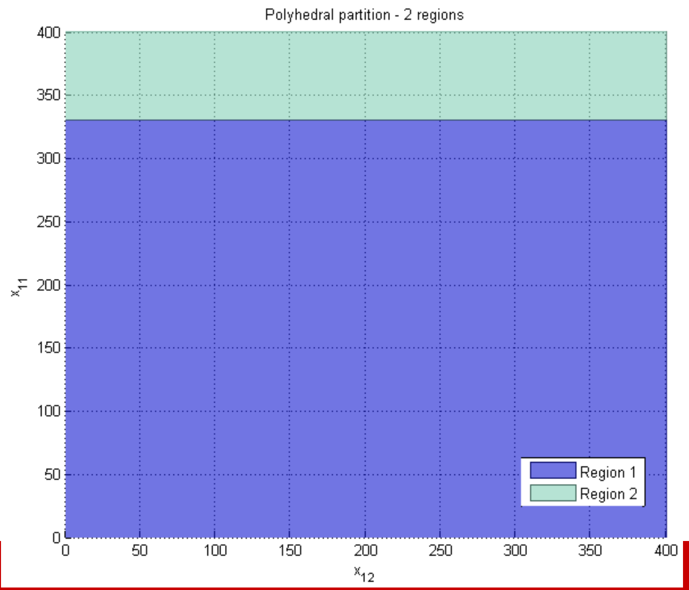
Our aim is to minimize the difference between the input $P_{1,TC}$ (also called manipulated variable MV) and the reference ($P_{1,EC}$). Our controller can only take in account a constant reference. To overcome this limitation we reformulate the tracking problem as a regulation problem consisting in taking the ΔP_1 (the new MV) to 0. The regulated power $P_{1,TC}$ is:

$$P_{1,TC} = P_{1,EC} + \Delta P_{1,EC}$$

Explicit Distributed Controller



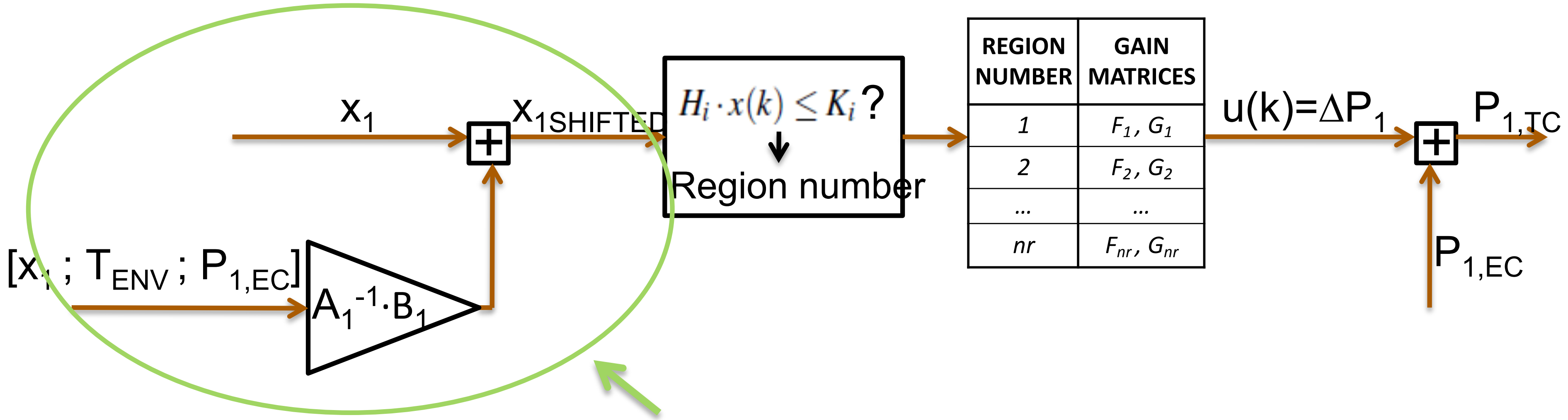
REGION NUMBER	GAIN MATRICES
1	F_1, G_1
2	F_2, G_2
…	…
nr	F_{nr}, G_{nr}



At each time instant the system belongs to a region according with its current state. On each region the explicit controller executes the following linear control law:

$$u(x) = F_i \cdot x(t) + G_i$$

Explicit Distributed Controller



REGION NUMBER	GAIN MATRICES
1	*F₁, G₁*
2	*F₂, G₂*
…	…
nr	*F_nr, G_nr*

The prediction evaluated by our explicit controller cannot take into account the measured disturbances (u_{MD}=[Tenv, P1, T_{neigh}]). Thus we exploit the superposition principle of linear systems:

$$x(k+1) = f(x(k),u_{MV}(k),u_{MD}(k)) \rightarrow x(k+1) = f(x(k),u_{MV}(k),0) + f(x(k),0,uMD(k)))$$

To remap the effect of these elements we exploit the model to modify the state (x (k) → $x_{SHIFTED}$(k)) projecting one step forward the MDs effects.

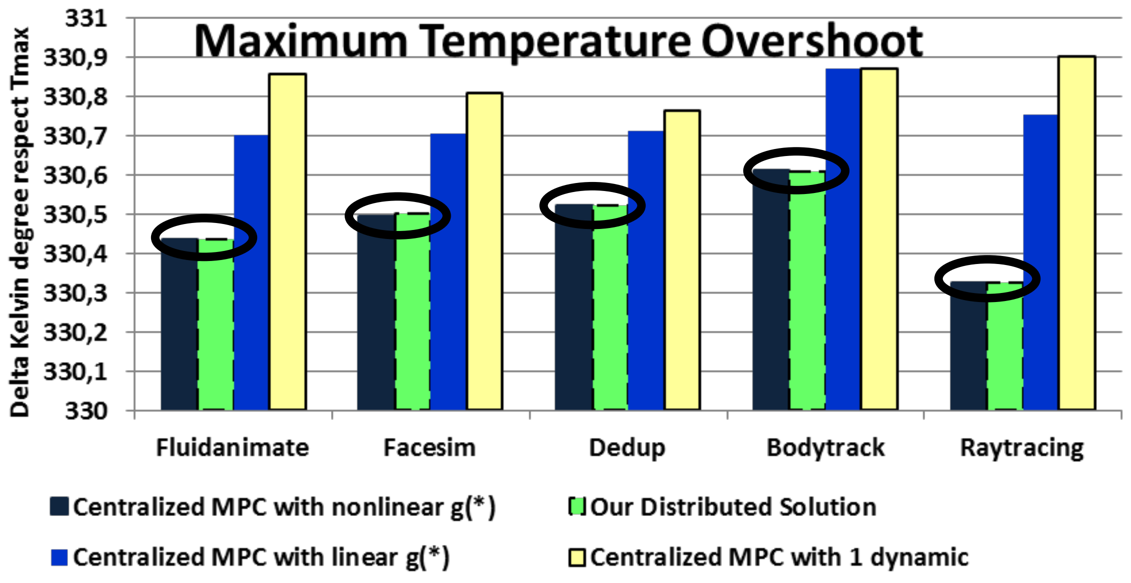
$$x(k+1) = A_1\cdot x(k) + B'_1\cdot\Delta P_1 + B''_1\cdot[P_{1,EC}\ \ T_{ENV}\ \ T_{neig}] \rightarrow x(k+1) = A_1\cdot x_{SHIFTED}(k) + B'_1\cdot\Delta P_{1,EC}$$

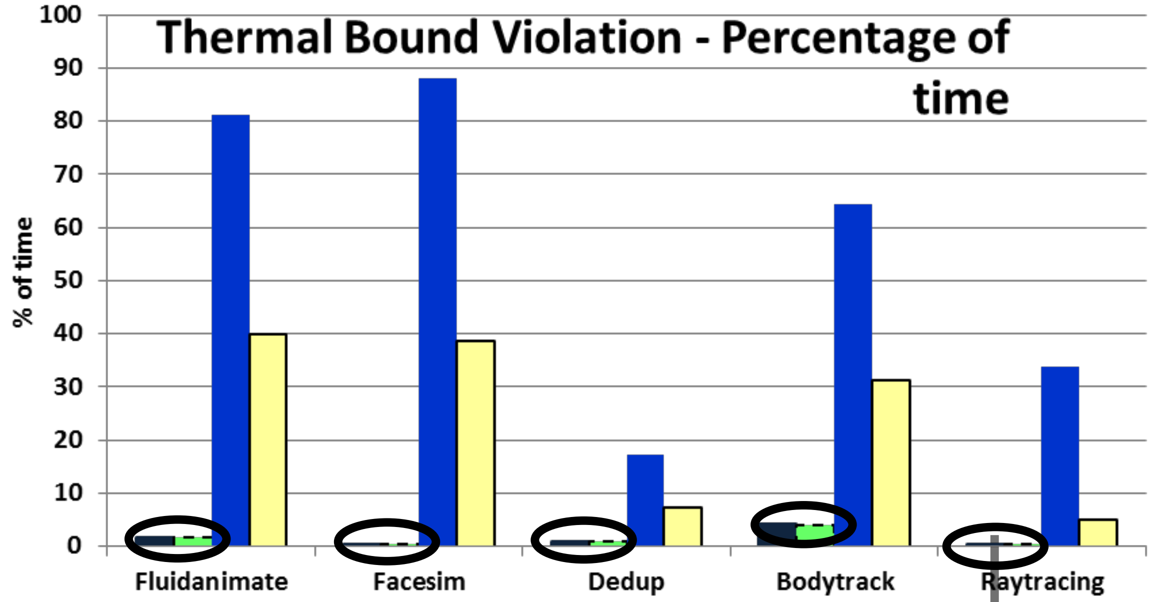
$$A_1\cdot x_{SHIFTED}(k) = A_1\cdot x(k) + B''_1\cdot[P_{1,EC}\ T_{ENV}\ \ T_{neig}] \rightarrow x_{SHIFTED}(k) = x(k) + A_1^{-1}\cdot B''_1\cdot[P_{1,EC}\ \ T_{ENV}\ \ T_{neig}]$$

MPC trade-off

Trace Driven Simulation (Matlab) – gold model

- Parsec trace obtained on real HW
- Power Model: Nonlinear vs. linear
- Thermal Model: one vs. two
- Centralized vs. Distributed



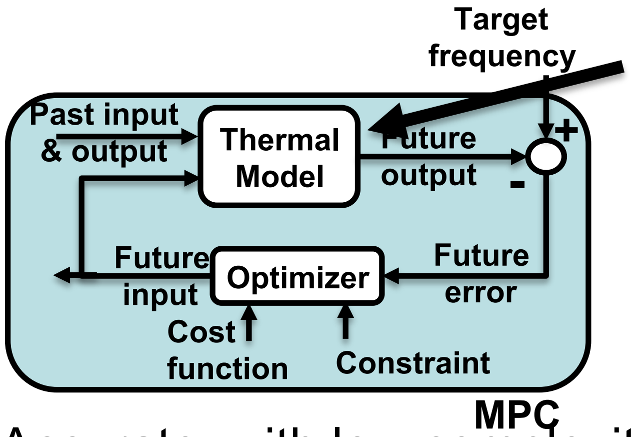


Outline

- Introduction
- Scalable Control
- Scalable model learning
- Experimental Environment
- Challenges Ahead

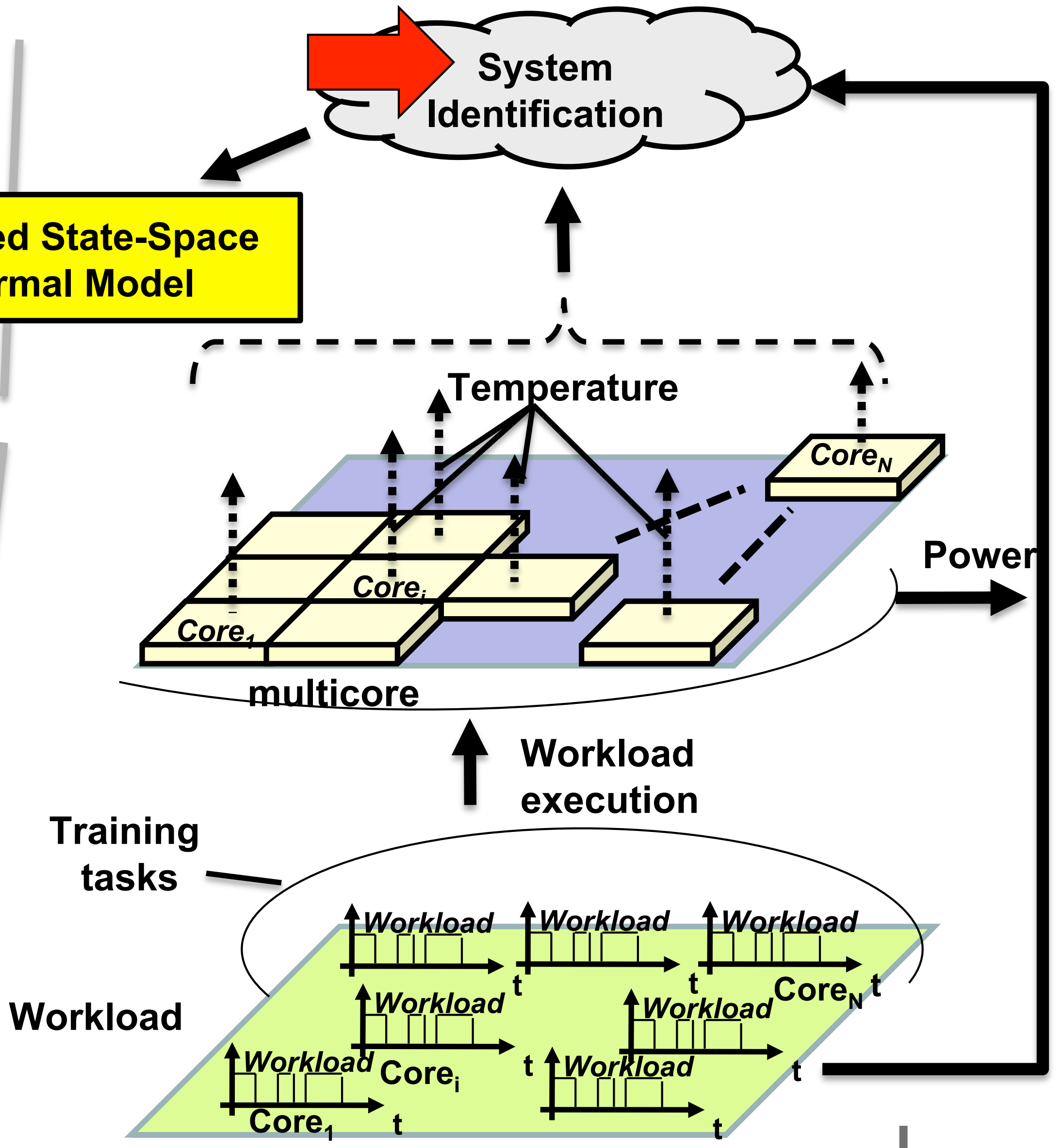
Model Identification

MPC needs a *Thermal Model*



- Accurate, with low complexity
- Must be known "at priori"
- Depends on user configuration
- Changes with system ageing

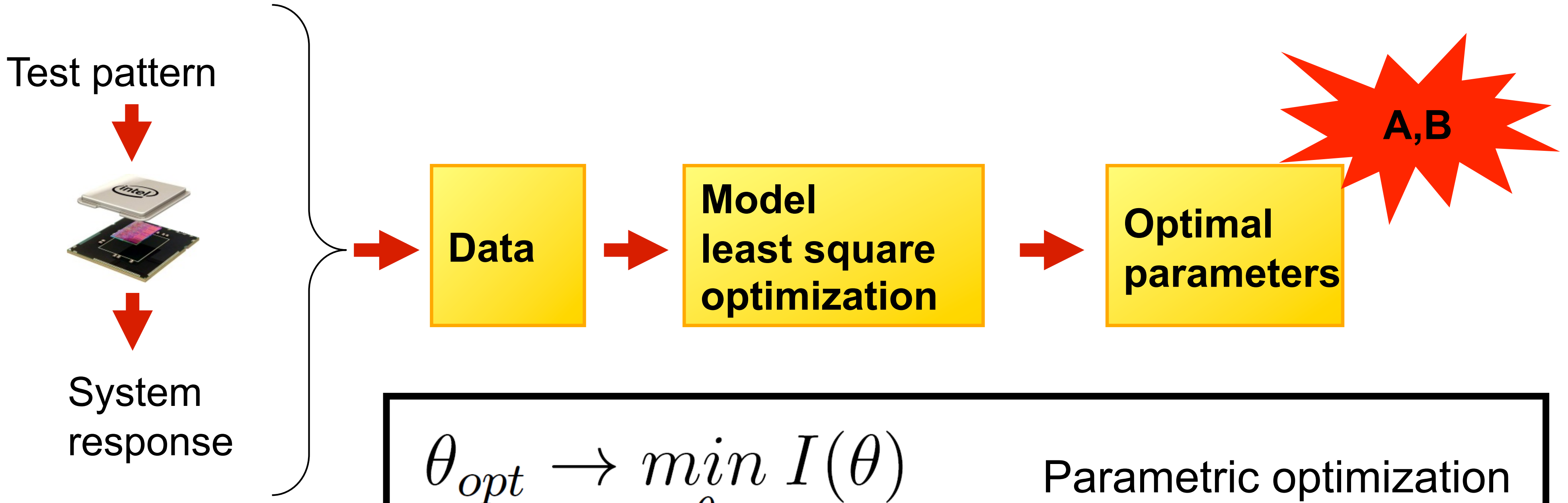
Identified State-Space Thermal Model



"*In field*" Self-Calibration

- Force test workloads
- Measure cores temperatures
- System identification

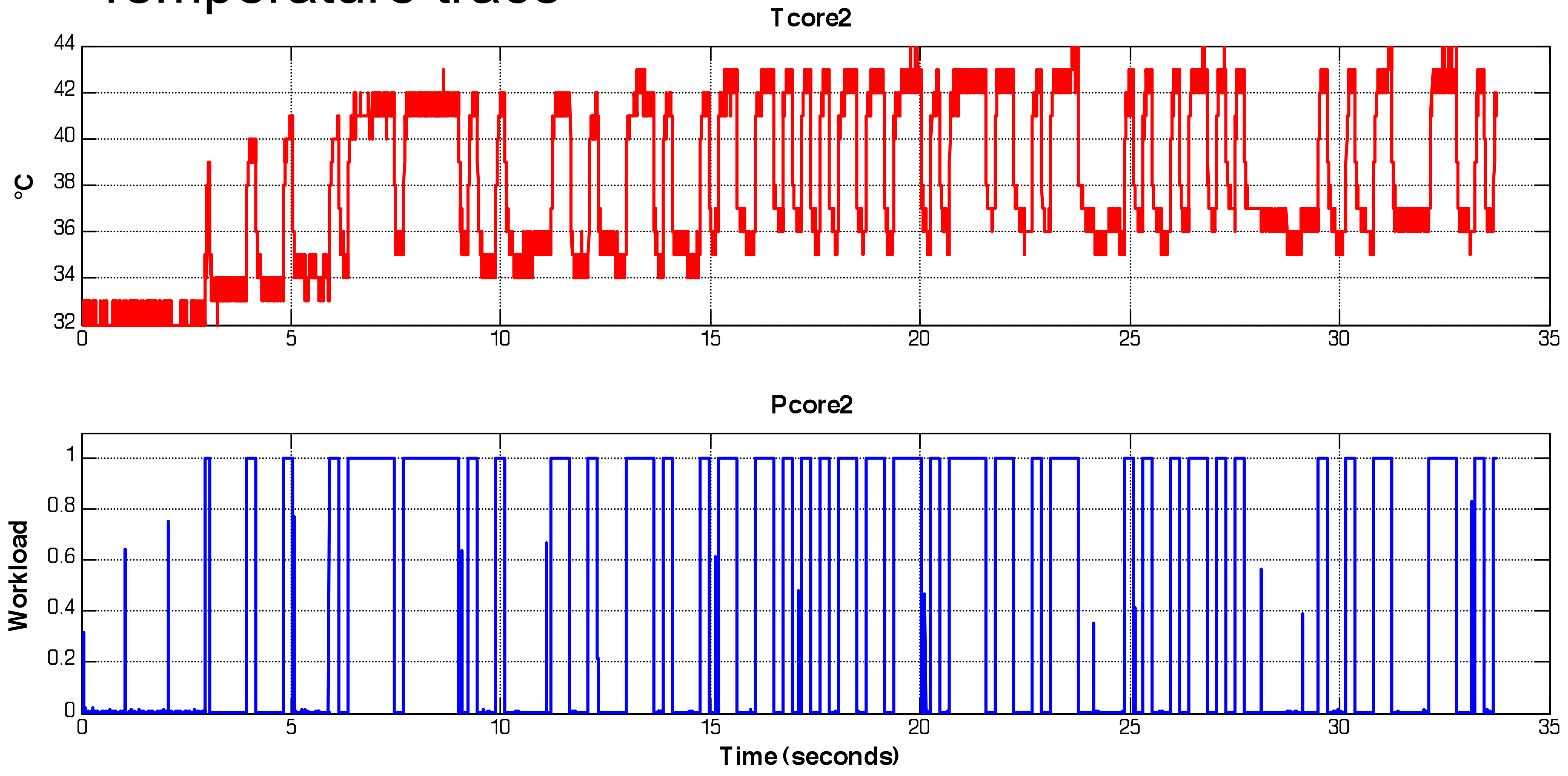
LS System Identification



$\theta_{opt} \rightarrow \min_{\theta} I(\theta)$	Parametric optimization
$\theta = \{A, B\}$	Parameters
$I(\theta) = \frac{1}{N_s} \sum_{i=1}^{N_s} e_i^2$	Cost function
$e = T_{model} - T_{real}$	Error Function

Workload & Temperature

Temperature trace

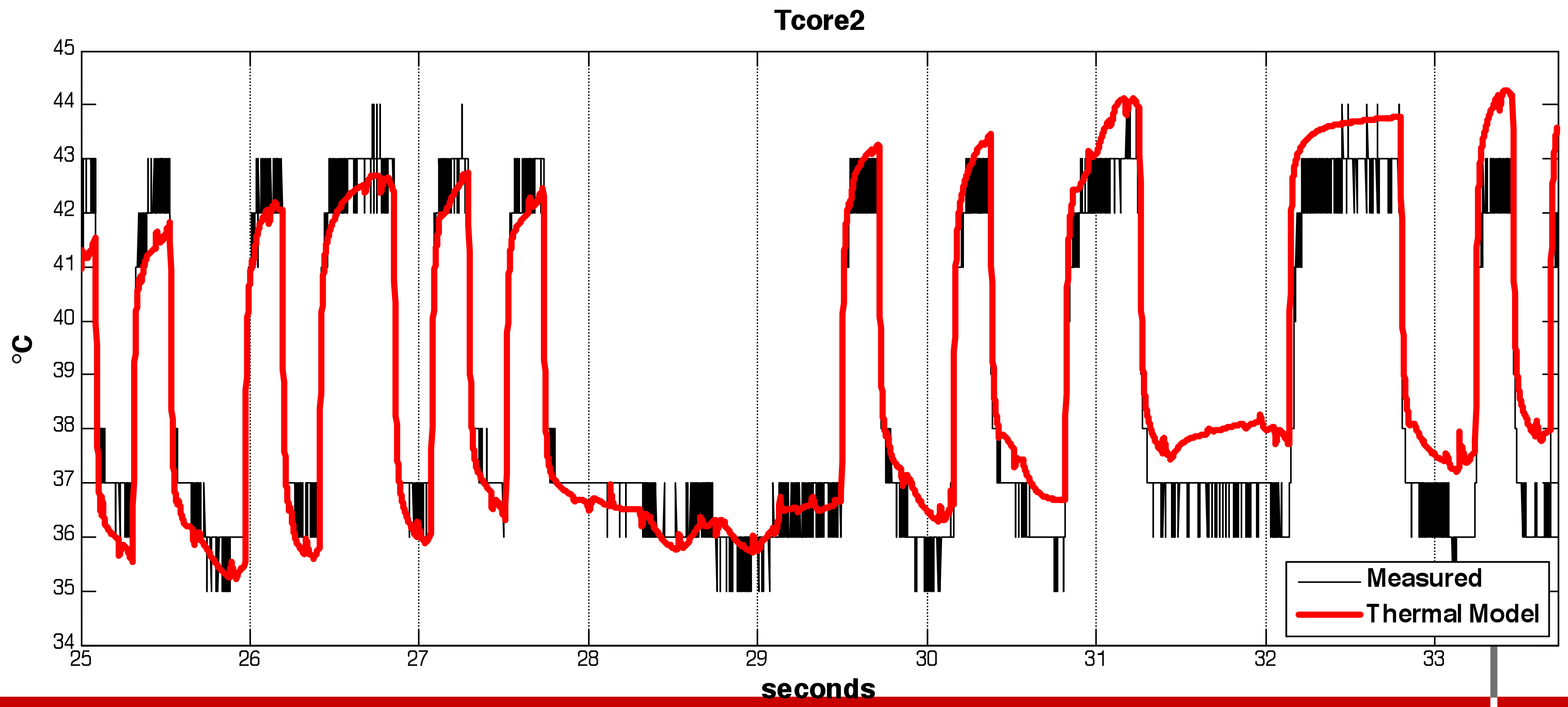


Pseudorandom workload pattern

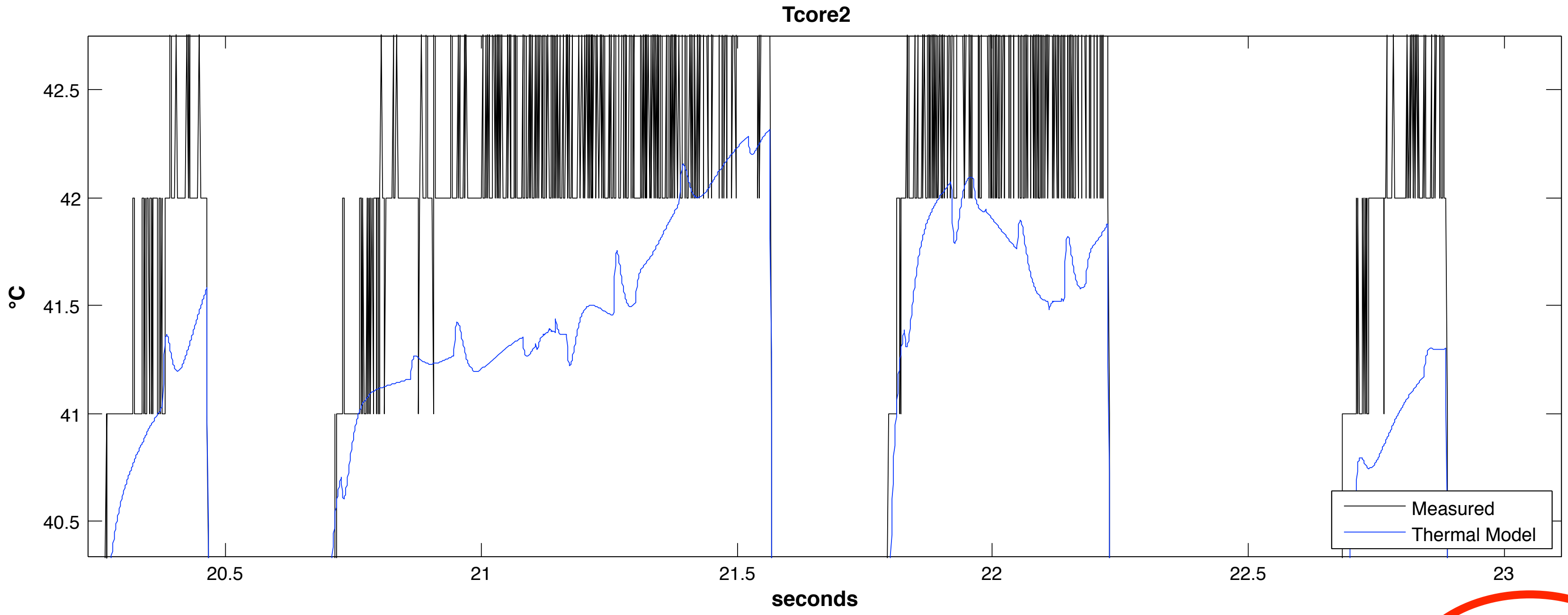
Black-box Identification

Identification based on pure LS fitting

MEASURED vs. SIMULATED TEMPERATURE



Validation

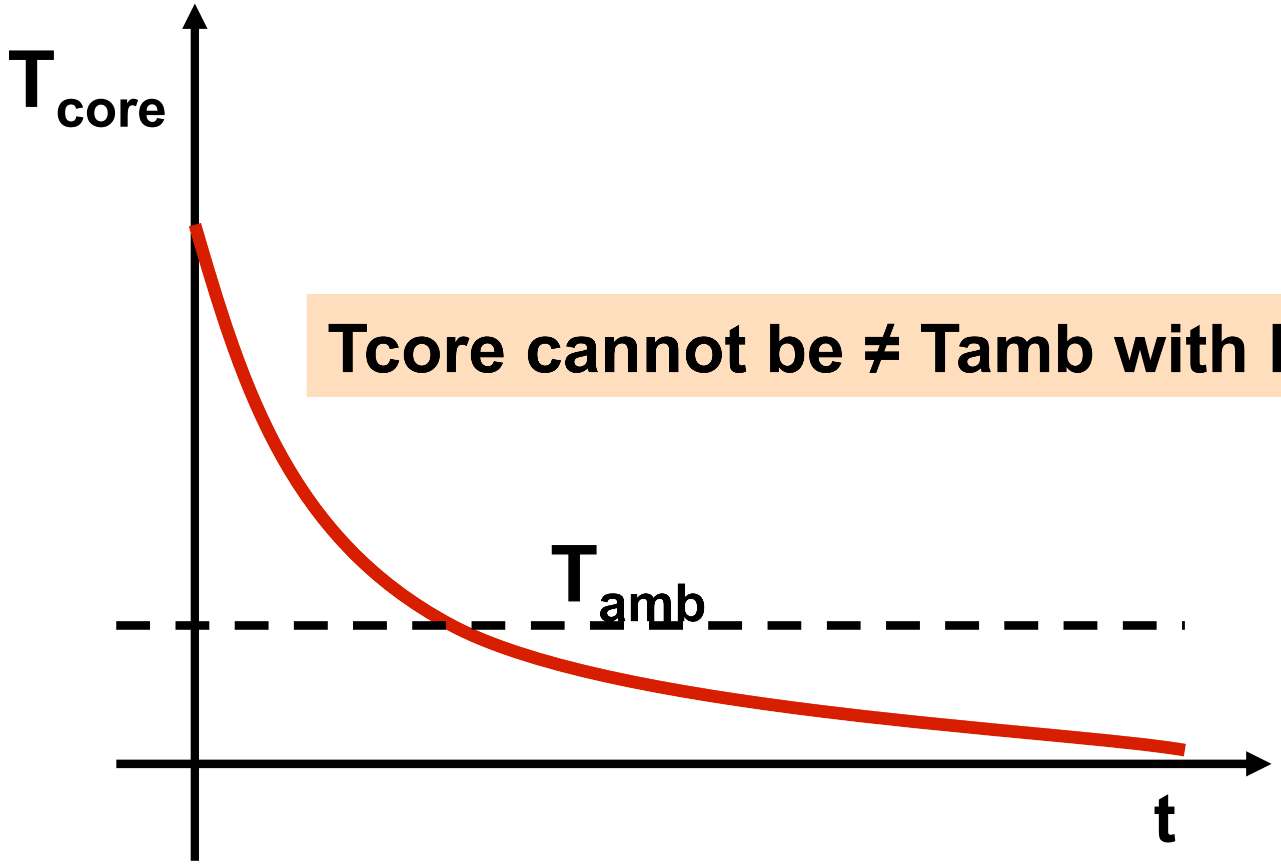


0.8557	0.8652	0.9081	0.9176	0.9896	0.9919	0.9992	1.0001

Problem: unstable Model

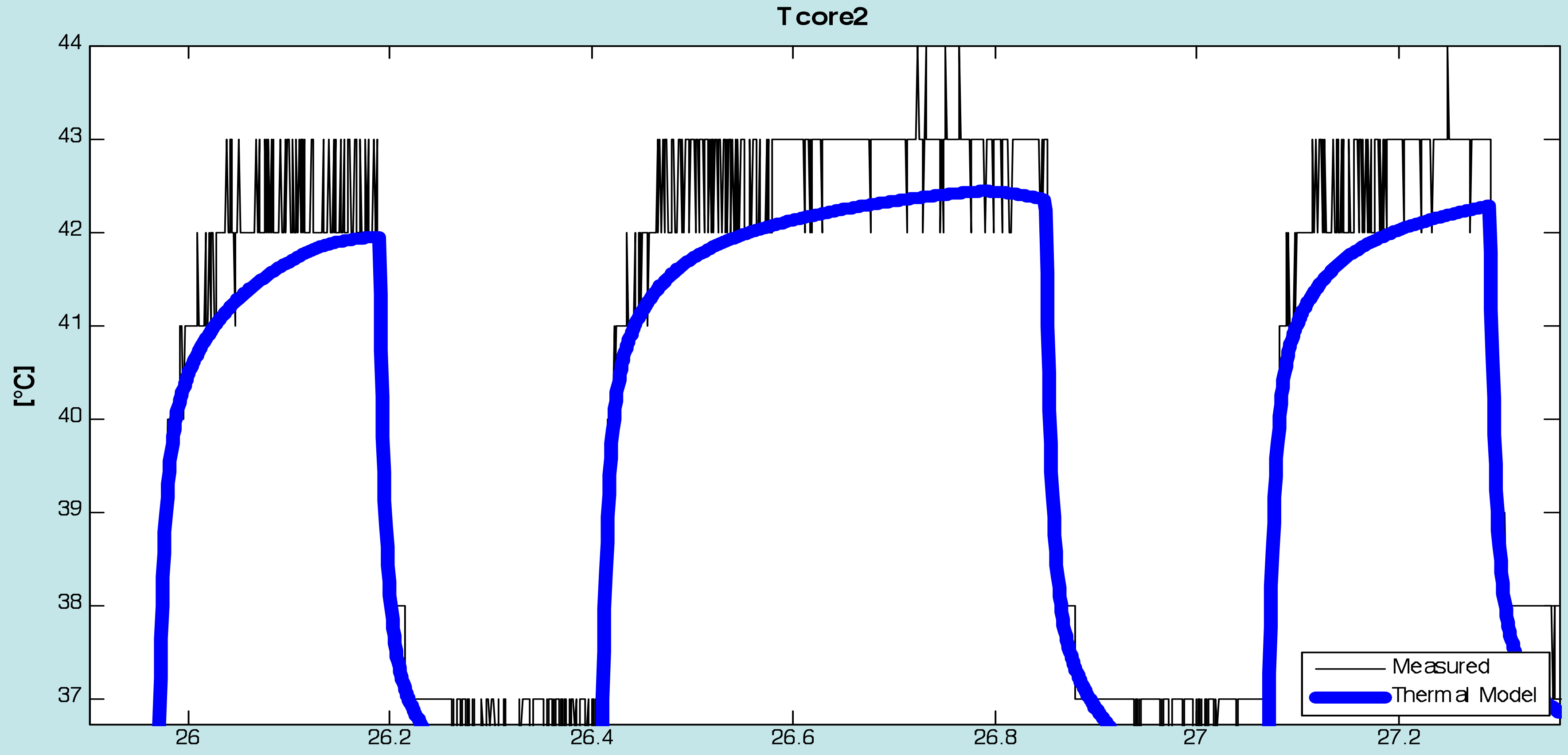
Gray Box identification

LS model must be constrained by physical properties to avoid over-fitting



Tcore cannot be ≠ Tamb with P=0

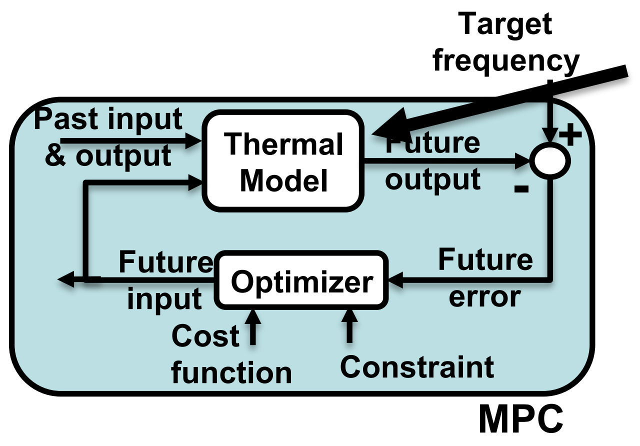
Physical Constraints



0.8393	0.9699	0.9956	0.9998	0.9418	0.9053	0.9029	0.9186

Model learning Scalability

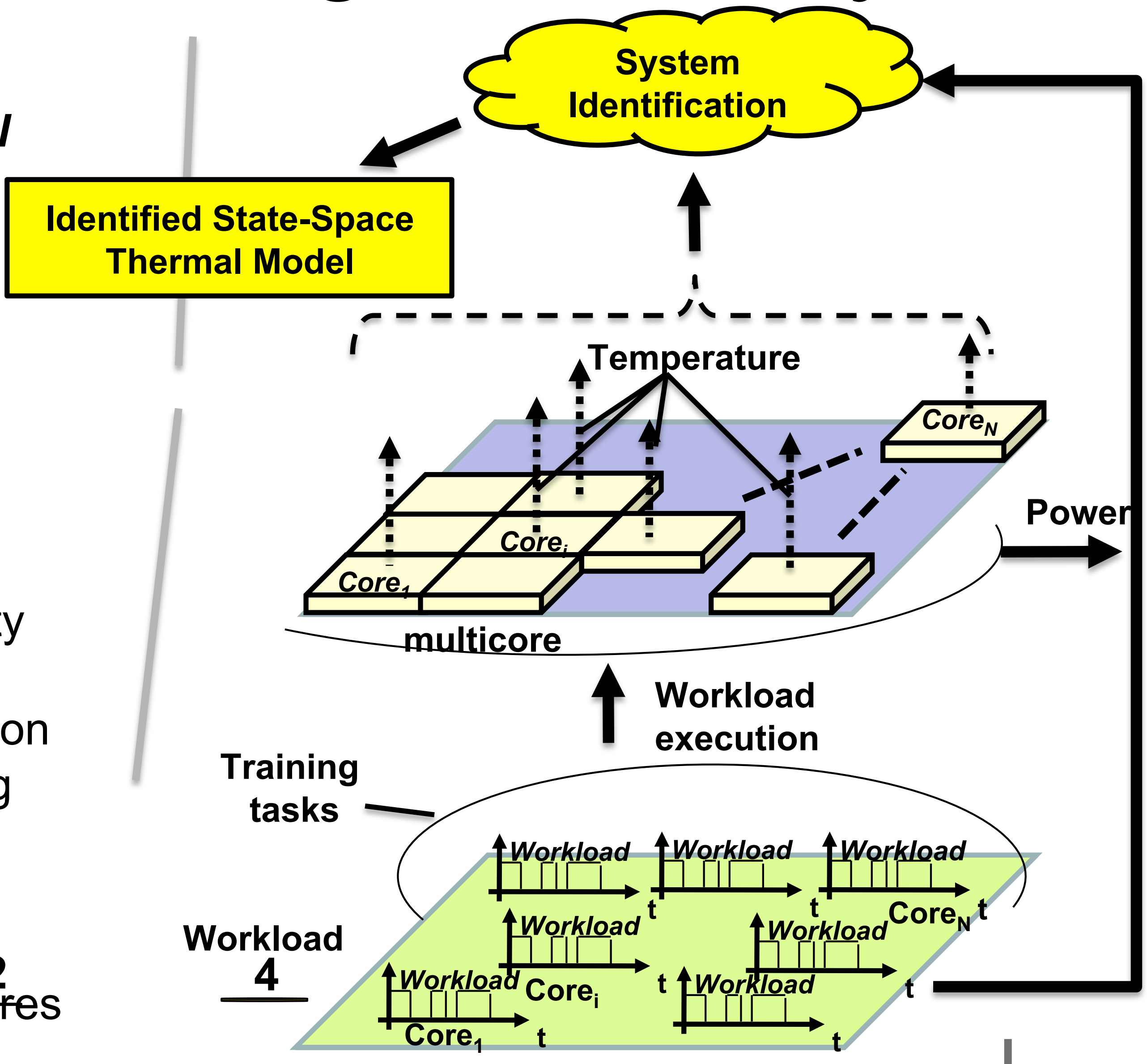
MPC Weaknesses – 2nd
Internal Thermal Model



- Accurate, with low complexity
- Must be known "at priori"
- Depends on user configuration
- Changes with system ageing

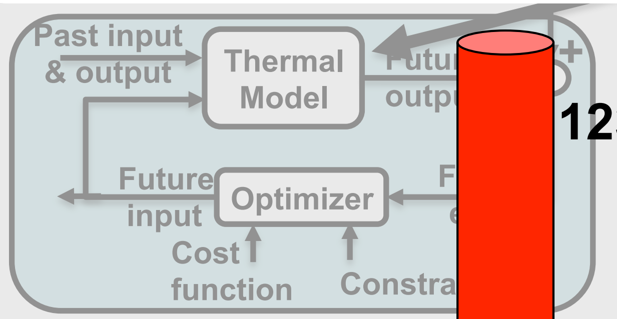
"In field" **Self-Calibration**

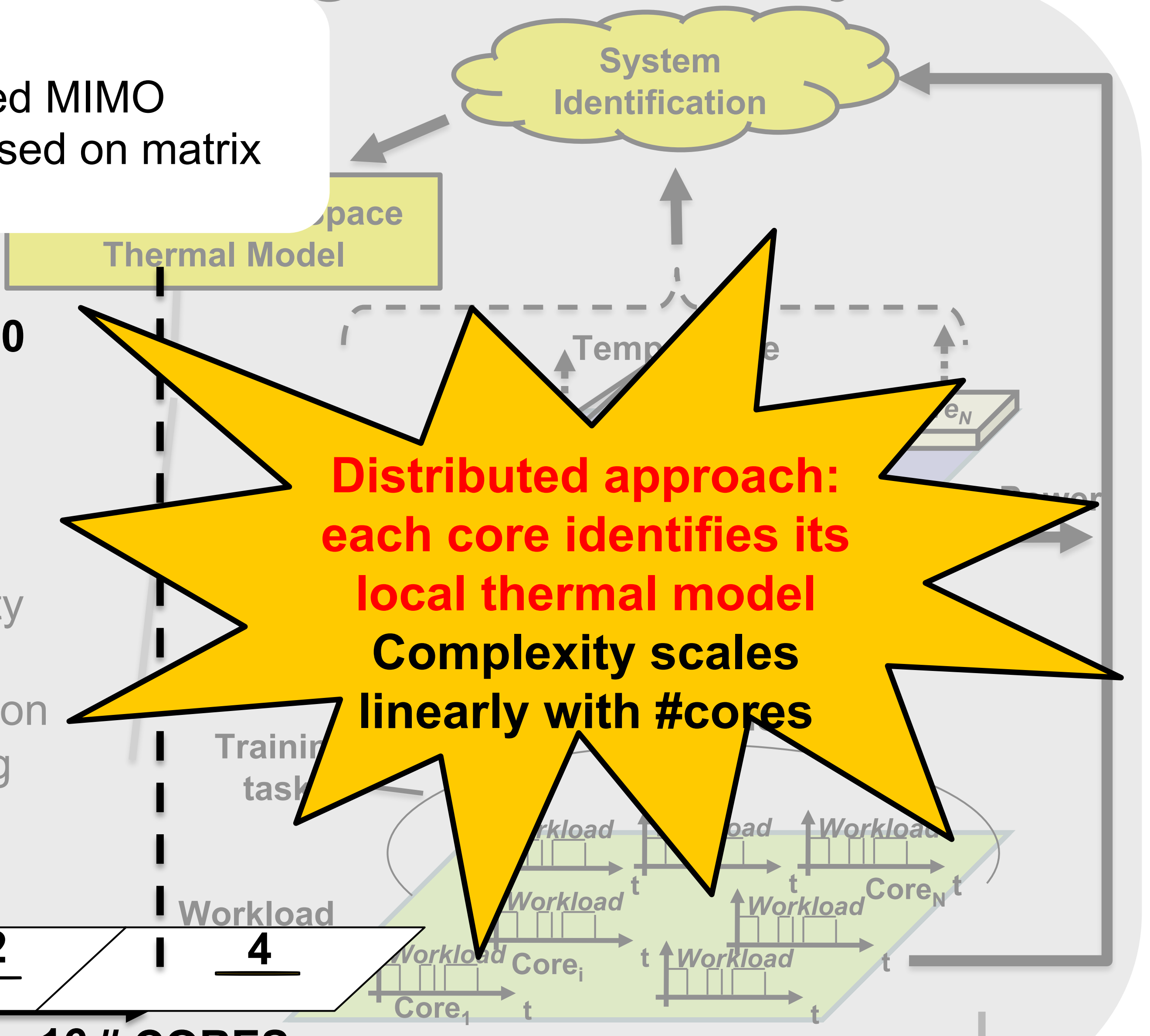
- Force test workloads
- Measure cores temperatures
- System identification



Model learning Scalability

Complexity issue
• State-of-the-art is centralized MIMO
• Least square based – is based on matrix inversion (cubic with #cores)



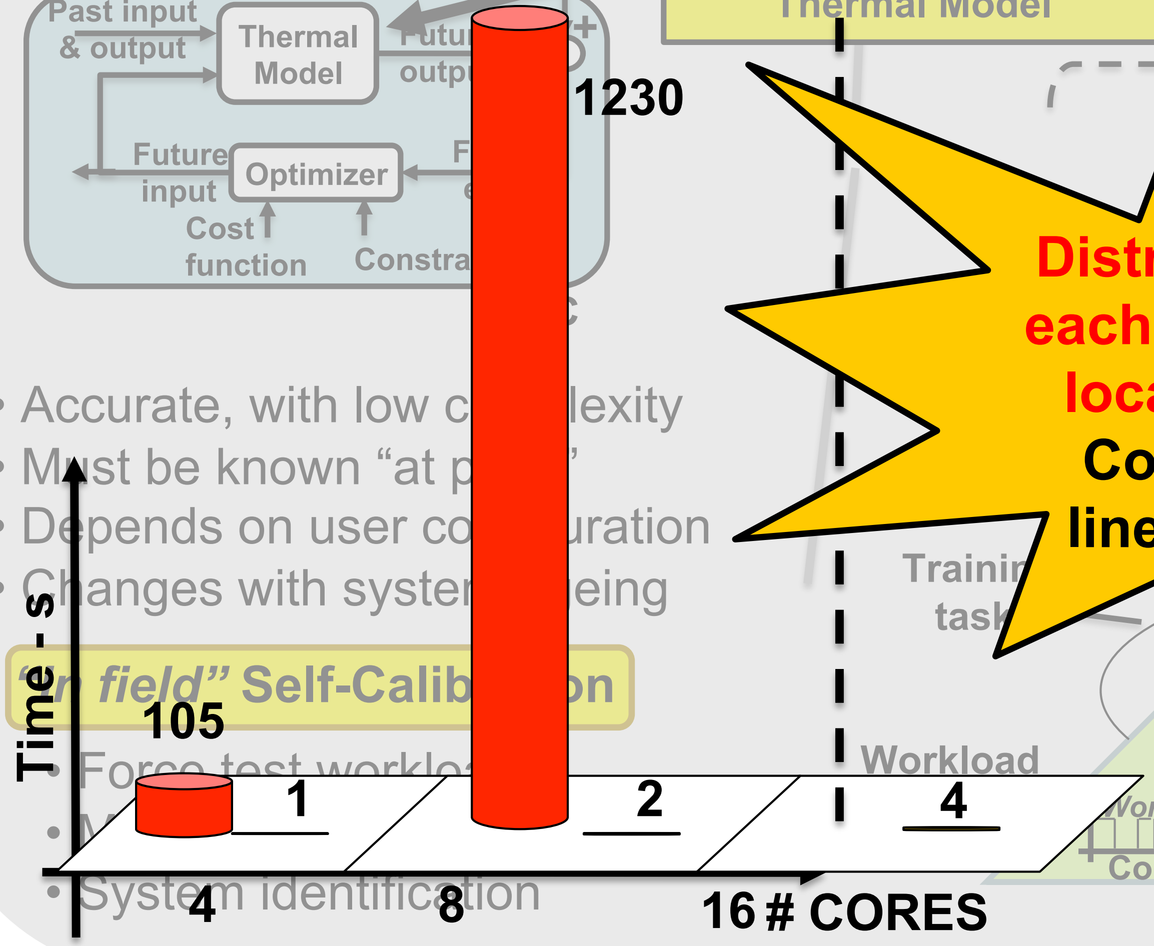


Distributed approach: each core identifies its local thermal model
Complexity scales linearly with #cores

• Accurate, with low c lexity
• Must be known "at p "
• Depends on user co uration
• Changes with syster eing

***"In field"* Self-Calib on**

• Force test worklo
• M
• System identification



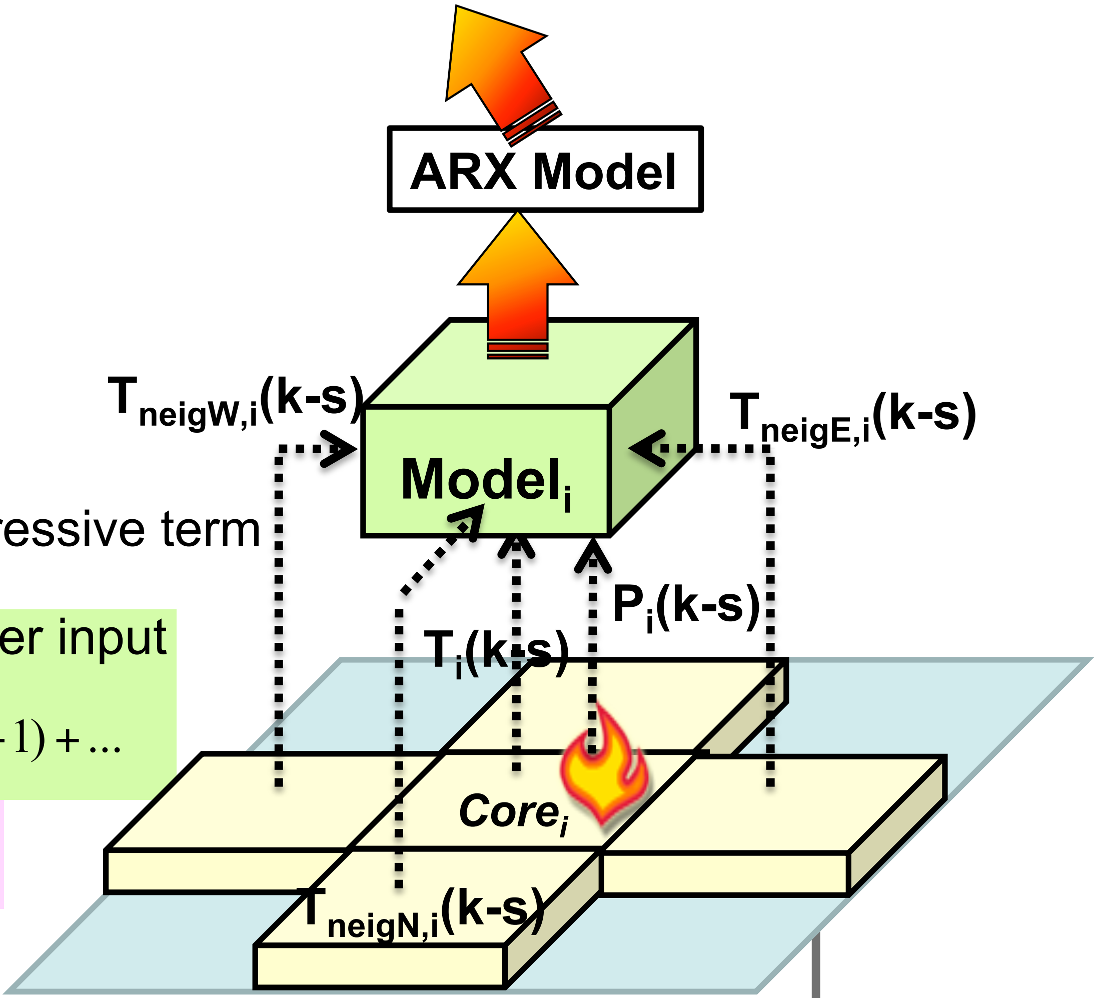
Distributed Model learning

Distributed
MISO identification

- ARX model

State Space Model

$$\begin{bmatrix} T_i(k+1) \\ x_i(k+1) \end{bmatrix} = A \cdot \begin{bmatrix} T_i(k) \\ x_i(k) \end{bmatrix} + B \cdot \begin{bmatrix} P_i(k) \\ T_{neig,i}(k) \end{bmatrix}$$



$T_i(k) = \alpha_2 \cdot T_i(k-1) + \alpha_1 \cdot T_i(k-2) +$ **Auto**Regressive term

$+ \beta_{1,1} \cdot P_i(k-2) + \beta_{1,2} \cdot P_i(k-1) +$ Power input

Neighbours temperatures $+ \beta_{2,1} \cdot T_{neig1,i}(k-2) + \beta_{2,2} \cdot T_{neigE,i}(k-1) + \ldots$

$+ e(t)$ Errors
(disturbances, measurement errors)

Distributed model-learning

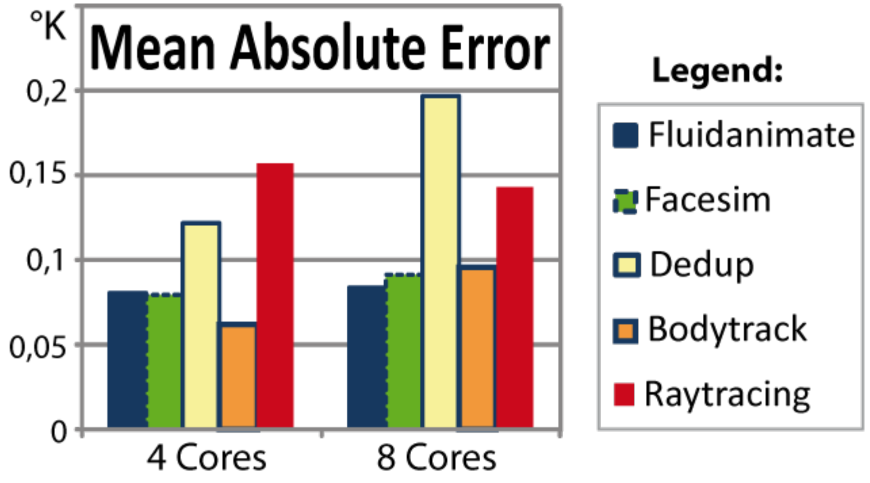
Distributed identification

- ARX model
- Computation Algorithm
- Results

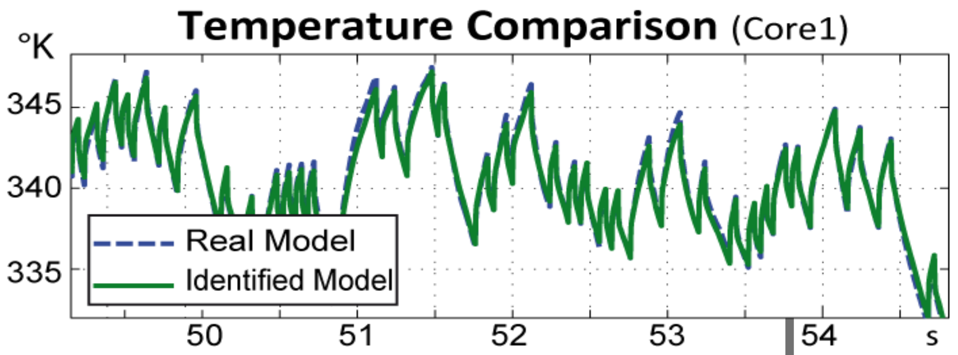
- Objective: find a_i and $b_{i,j}$
- System data collection:
- input: PRBS signals to all cores (persistently exciting inpt sequence)
- output: Temperatures of all cores (T_i^o con i= # core)
- Parameters computation:
- $T^p(a_i, b_{i,j})=T(k+1|k)$ computed with previous equation
- T^o output temperature (measured)
- Least square algorithm:

$$\min \frac{1}{L}\sum_{k=3}^{L+2}\left(T^0 - T^p\right)^2 = \min \frac{\left\|T^0 - T^p\right\|_2^2}{L}$$

1) Mean Absolute Error between original and identified models



2) Temperature response of core 1



Outline

- Introduction
- Scalable Control
- Scalable model learning
- Experimental Environment
- Challenges ahead

Simulation Strategy

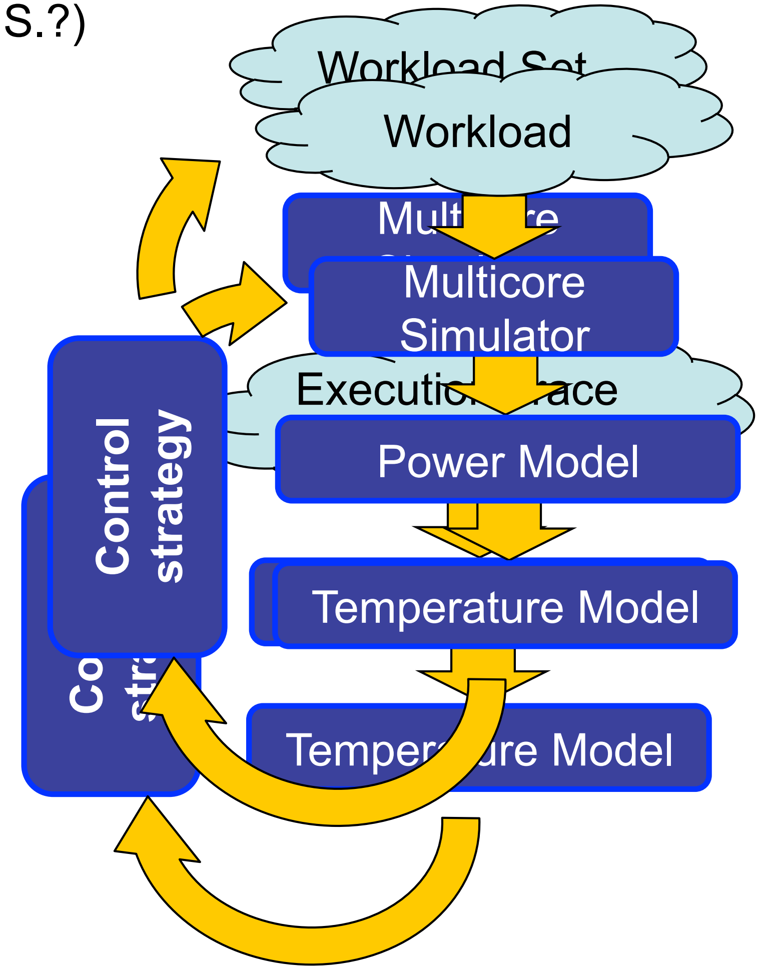
Trace driven Simulator [1]:

- Not suitable for full system simulation (How to simulate O.S.?)
- looses information on cross-dependencies

  → resulting in degraded simulation accuracy

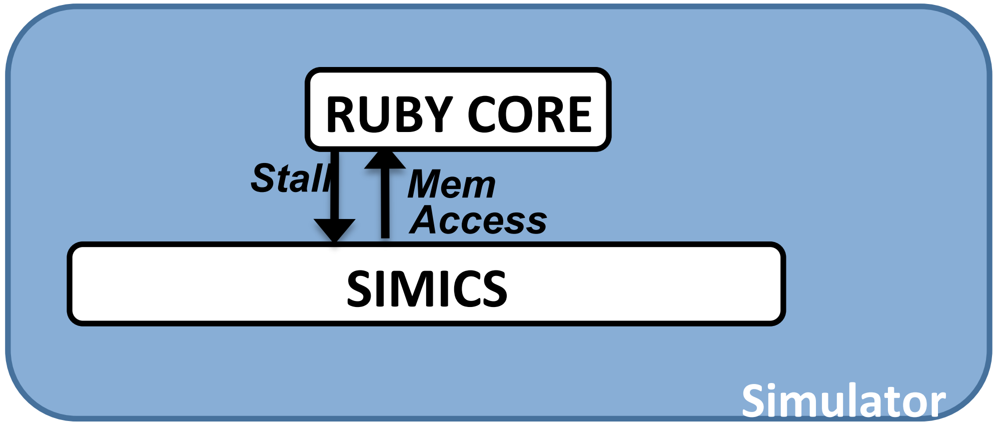
Closed loop simulator:

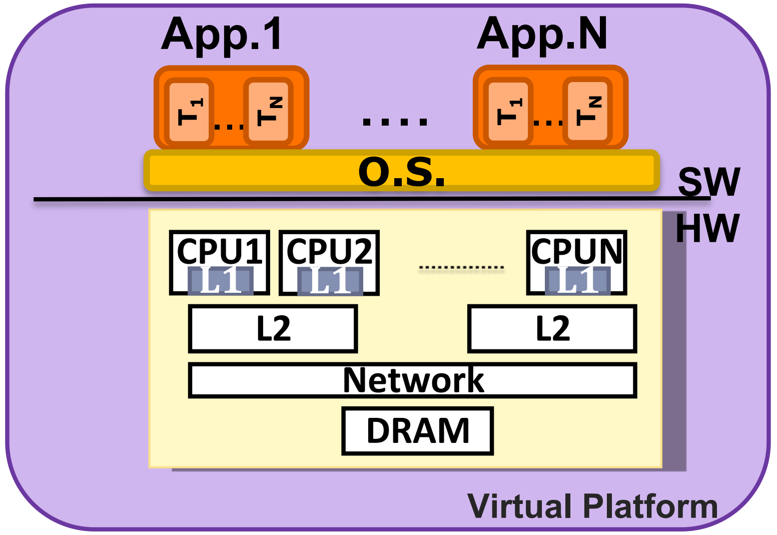
- Cycle accurate simulators [2] :
  - High modeling accuracy
  - Suppost well-established power and temperature co-simulation based on analytical models and system micro-architectural knowledge
  - Low simulation speed
  - Not suitable for full-system simulation
- Functional and instruction set simulators:
  - Allow full system simulation
  - Lower internal precision
  - Less detailed data → no micro-architectural model
  - Introduces the challenge of having accurate power and temperature physical models



[1] *P Chaparro et al.* Understanding the thermal implications of multi-core architectures. *2007*
[2] *Benini L. et al*. MPARM: Exploring the multi-processor SoC design space with SystemC 2005

Virtual Platform





Simics by Virtutech:

- full system functional simulator
- models the entire system:
  peripherals, BIOS, network interfaces, cores, memories
- allows booting full OS, such as Linux SMP
- supports different target CPU (arm, sparc, x86)
- x86 model:
  - in-order
  - all instruction are retired in 1 cycle
  - does not account for memory latency

Memory timing model

- RUBY – GEMS (University of Wisconsin)[1]
  - Public cycle-accurate memory timing model
  - Different target memory architectures
  - fully integrated with Virtutech Simics
  - written in C++
  - we use it as skeleton to apply our add-ons (as C++ object)

[1] Martin Milo M. K. et al. Multifacet's general execution-driven multiprocessor simulator (GEMS) toolset 2005

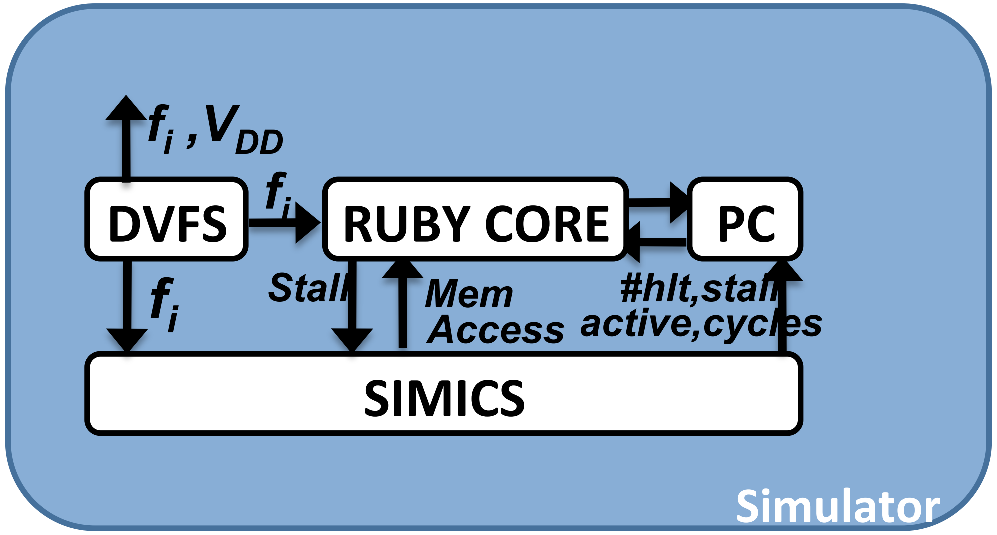
Virtual Platform

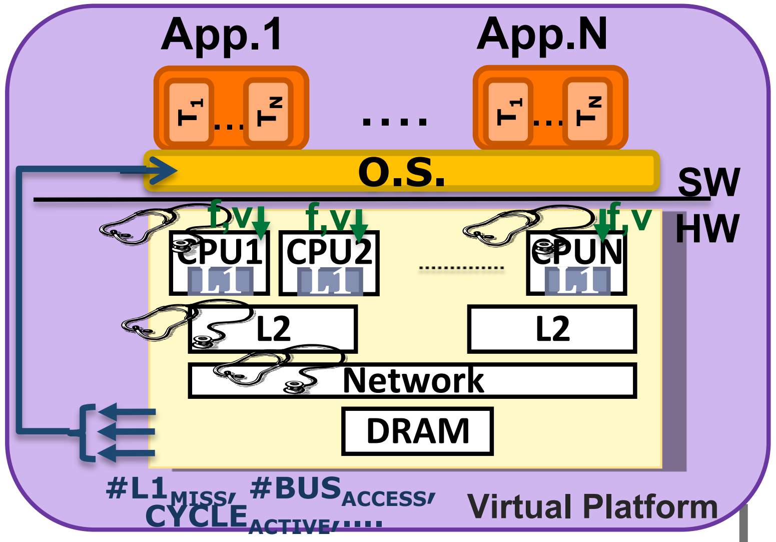
Performance knobs (DVFS) module:

- Virtutech Simics support frequency change at run-time
- RUBY does not support it
  - does not have internal knob to change frequency
- We add a new DVFS module to support it
  - ensures L2 cache and DRAM to have a constant clock frequency
  - L1 latency scale with Simics processor clock frequency

Performance counters module:

- Needed by performance control policy
- We add a new Performance Counter module to support it
  - exports to O.S. and application different quantities:
    - the number of instruction retired, clock cycles and stall cycles expired, halt instructions, ..

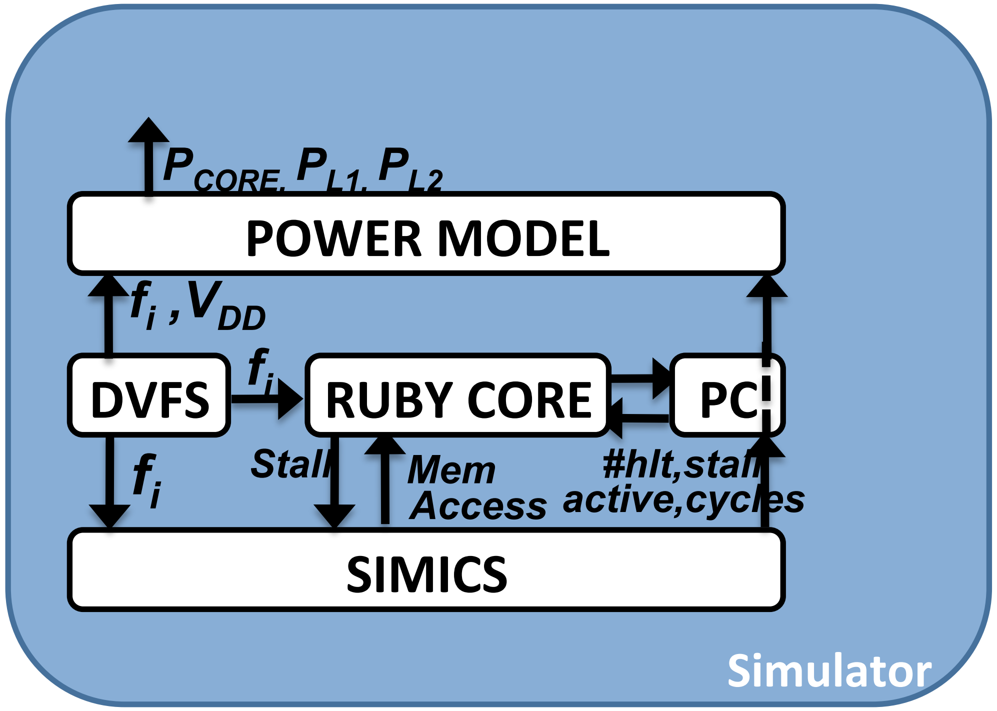


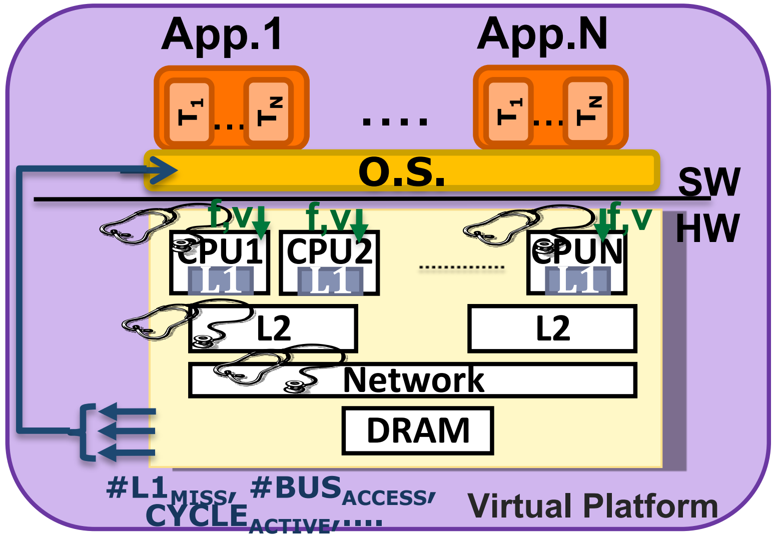


Virtual Platform

Power model module:

- At run-time estimate the power consumption of the target architecture
- Core model $P_T = [P_D(f,CPI) + P_S(T, VDD)] *(1 - idleness) + idleness *(P_{IDLE})$
- P_D experimentally calibrated analytical power model
- Cache and memory power – access cost estimated with CACTI [1]

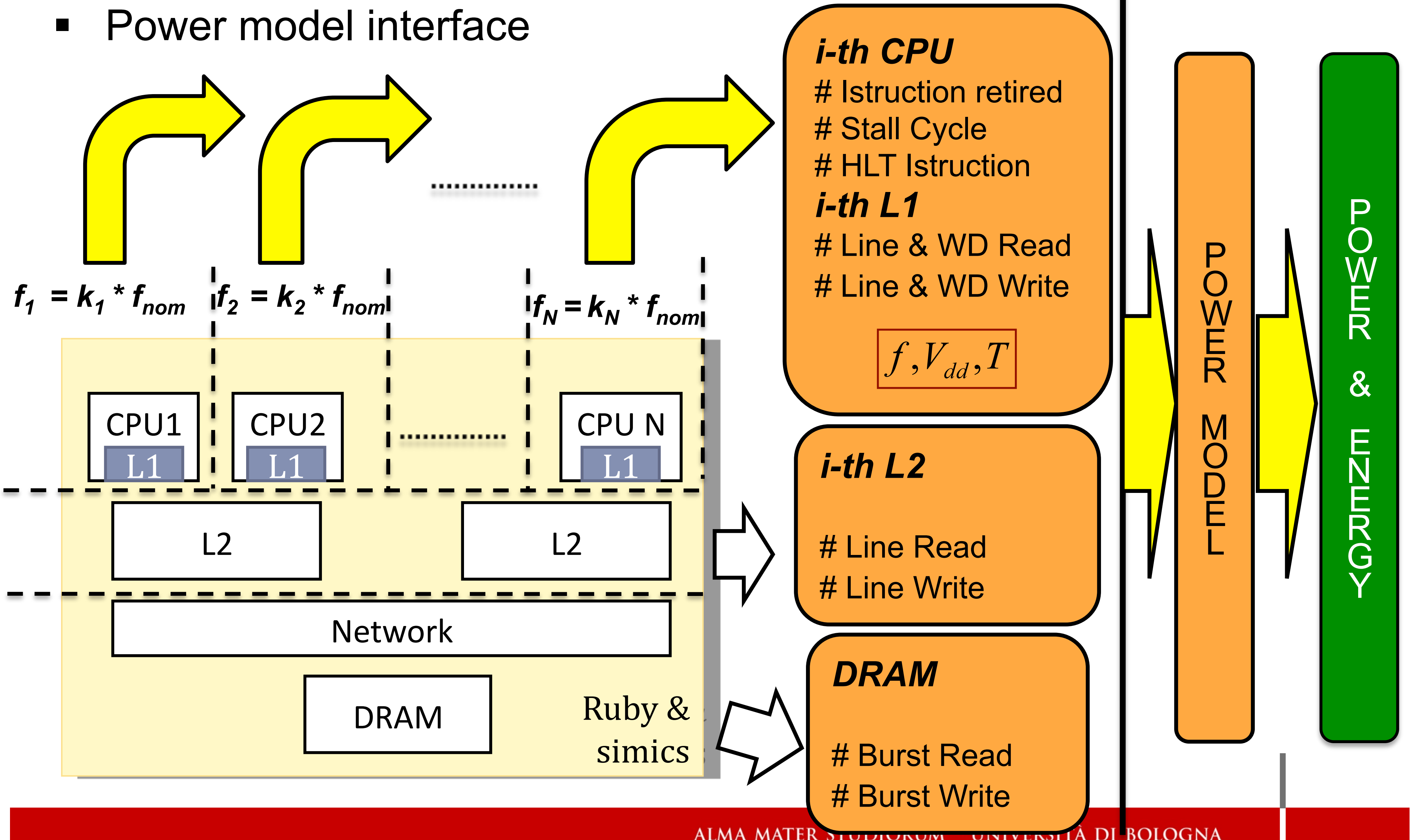




[1] Thoziyoor Shyamkumar et al. A comprehensive memory modeling tool and its application to the design and analysis of future memory hierarchies. 2008

Power Model

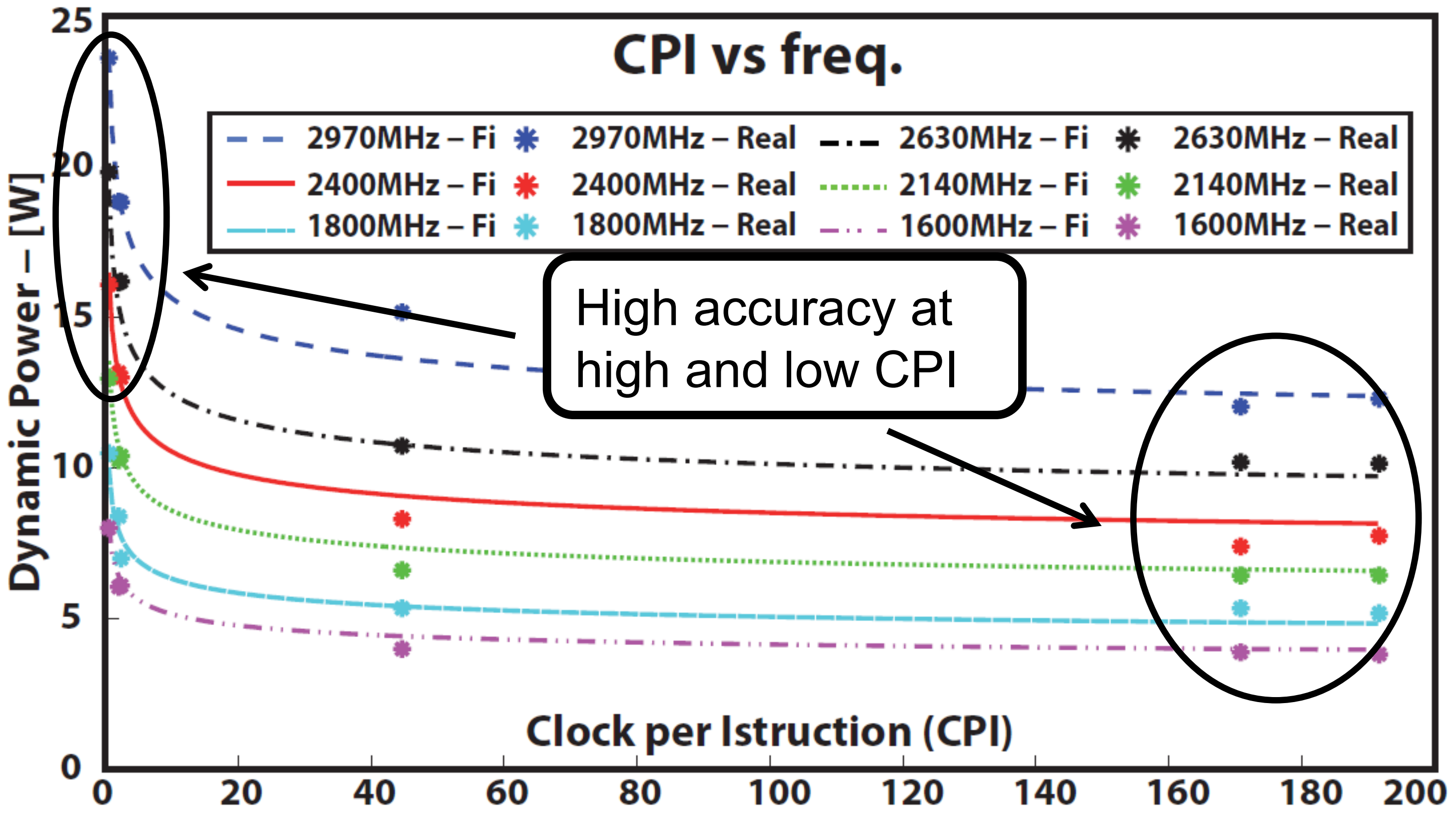
- Power model interface



Modeling Real Platform – Power

Real Power Measu

- *Intel server system*
  - 16 cores - 4 qu
  - 16GB FBDIMM
  - Intel® Core™
- *At the wall Power co*
  - test:
    - set of synt
    - forcing all t
    - for each be
    correlate it v



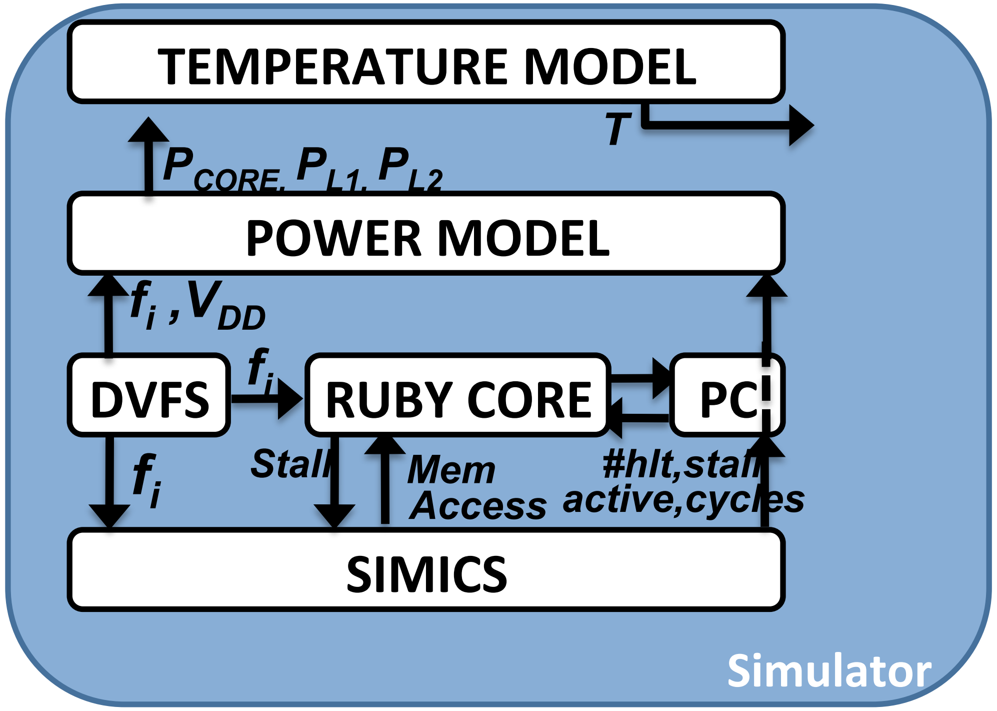
$$P_D = k_A \cdot V_{DD}^2 \cdot f_{CK} + k_B + (k_C + k_D \cdot f_{CK}) \cdot CPI^{k_E}$$

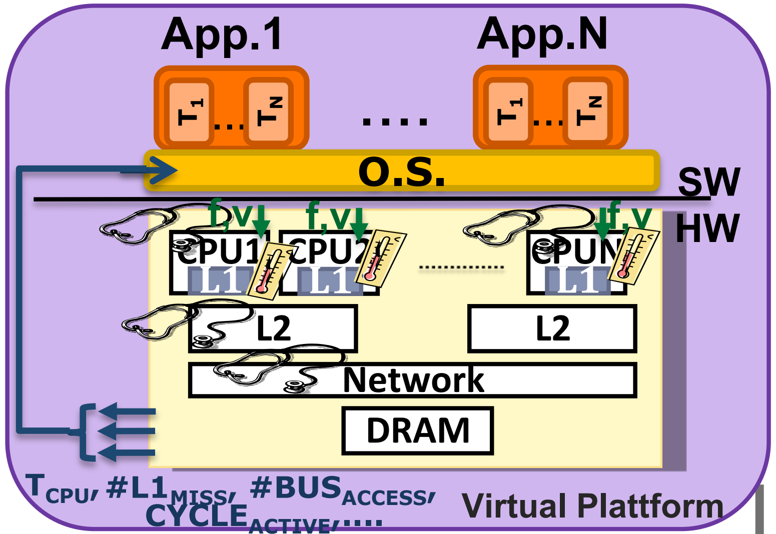
- We relate the static power with the operating point by using an analytical model

Virtual Platform

Temperature model module:

- we integrate our virtual platform with a thermal simulator [1]
- Input: power dissipated by the main functional units composing the target platform
- Output: Provides the temperature distribution along the simulated multicore die area as output

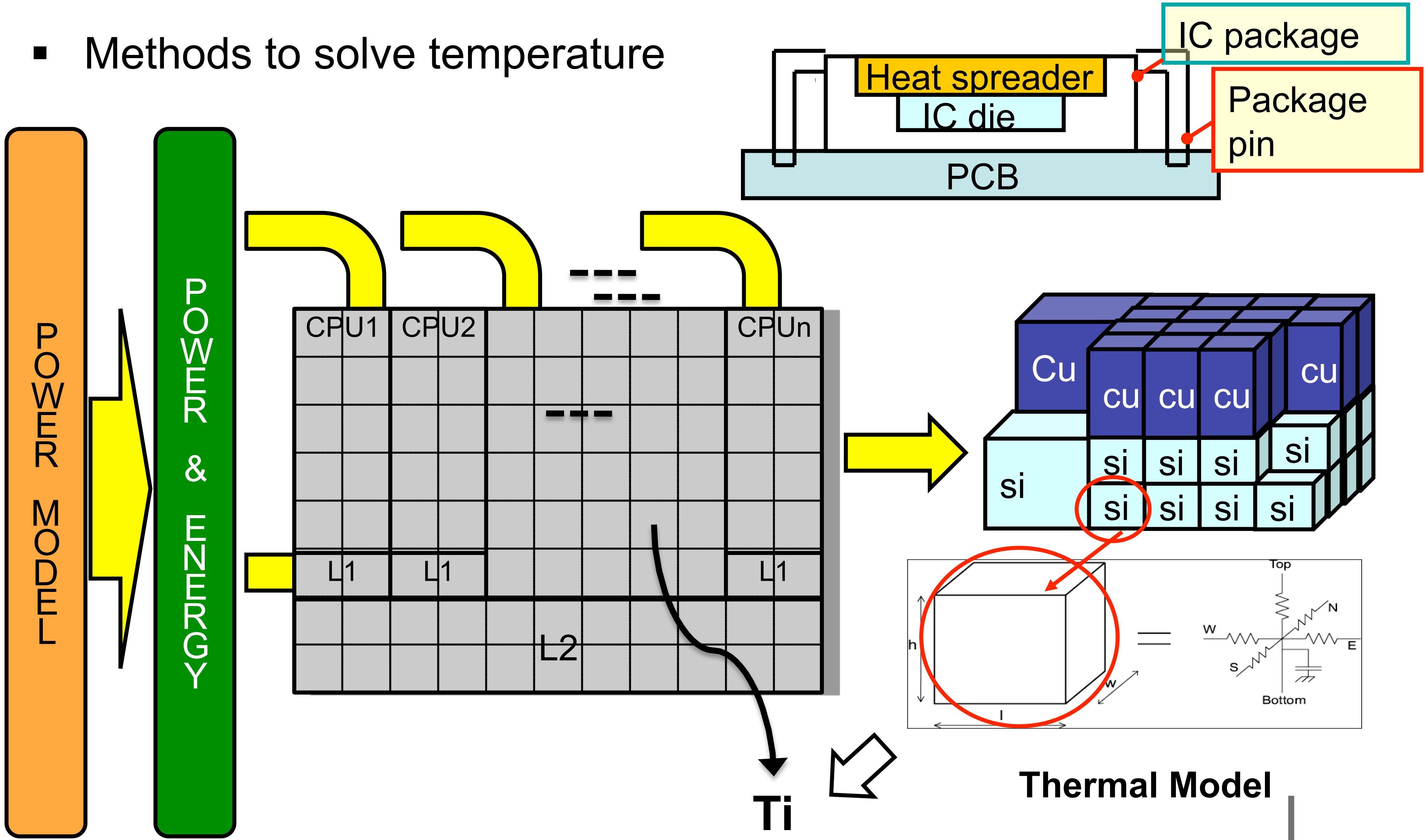




[1] Paci G. et al. Exploring "temperature-aware" design in low-power MPSoCs

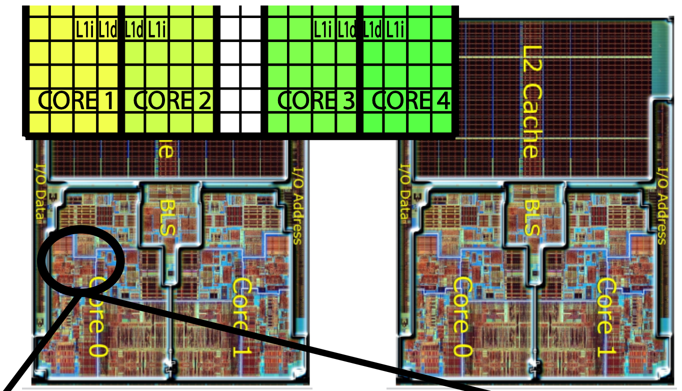
Thermal Model

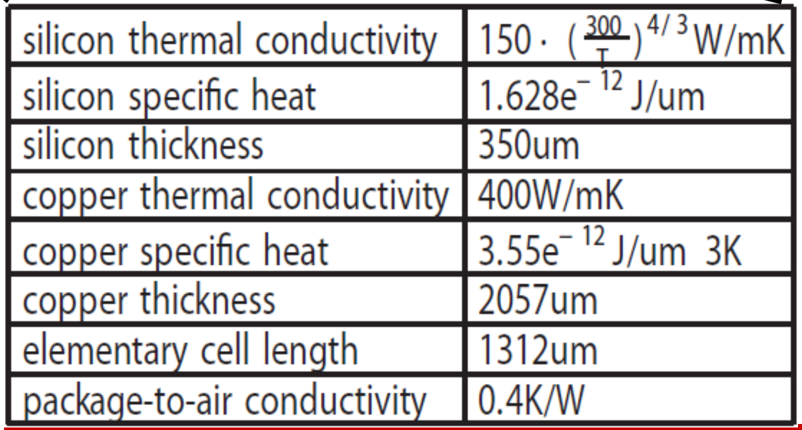
- Methods to solve temperature



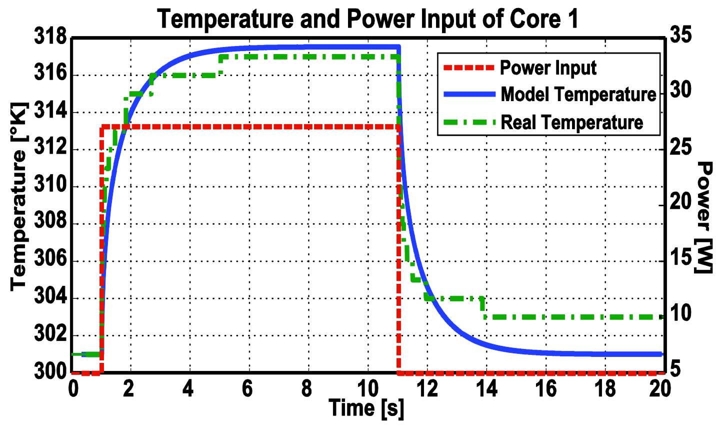
Modeling Real Platform– Thermal

- Thermal Model Calibration :
    - Derived from Intel® Core™ 2 Duo layout
    - We calibrate the model parameter to simulate real HW transient
    - High accuracy (error < 1%) and same transient behavior





silicon thermal conductivity	$150 \cdot (\frac{300}{T})^{4/3}$ W/mK
silicon specific heat	$1.628e^{-12}$ J/um
silicon thickness	350um
copper thermal conductivity	400W/mK
copper specific heat	$3.55e^{-12}$ J/um 3K
copper thickness	2057um
elementary cell length	1312um
package-to-air conductivity	0.4K/W



<1%

Virtual Platform Performance

- Target:
  - 4 core Pentium® 4
  - 2GB RAM
  - 32 KB private L1 cache
  - 4 MB shared L2 cache
  - Linux OS
- Host:
  - Intel® Core™ 2 Duo
  - 2.4 Ghz
  - 2GB RAM

1 Billion instruction

Simics + Ruby:

Tsim = 1040 s

Simics + Ruby + DVFS:

Tsim = 1045 s

Simics + Ruby + DVFS + Power:

+ 7%

Tsim = 1110 s

Compute every 13us

Simics + Ruby + DVFS + Power + Thermal interface:

Tsim = 1160 s

Simics + Ruby + DVFS + Power + Thermal Model:

+ 19.2%

Tsim = 1240 s

68 cells T = 100ns

Mathworks Matlab/Simulink

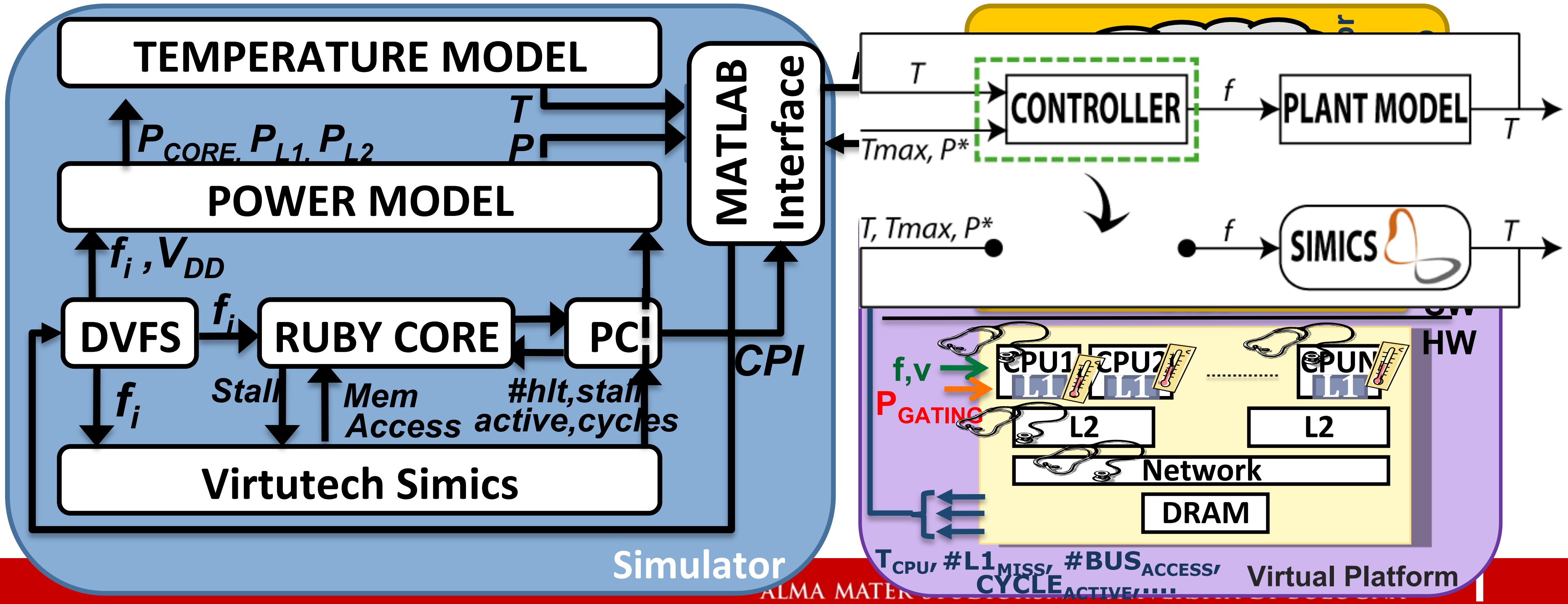
- **Numerical computing environment** developed to design, implement and test numerical algorithms
- Mathworks Simulink – for **simulation of dynamic systems**: simplifies and speedups the development cycle of control systems
- **Can be called** as **a computational engine by** writing C and Fortran **programs** that use Mathworks Matlab's engine library
- Controller design - two steps:
  - developing the control algorithm that optimizes the system performance
  - implementing it in the system

We allow a Mathworks Matlab/Simulink description of the controller to directly drive at run-time the performance knobs of the emulated system

Virtual Platform

Mathworks Matlab interface:

- New module named Controller in RUBY
- Initialization: starts the Mathworks Matlab engine concurrent process,
- Every N cycle - wake-up:
    - send the current performance monitor output to the Mathworks Simulink model
    - execute one step of the controller Mathworks Simulink model
    - propagate the Mathworks Simulink controller decision to the DVFS module



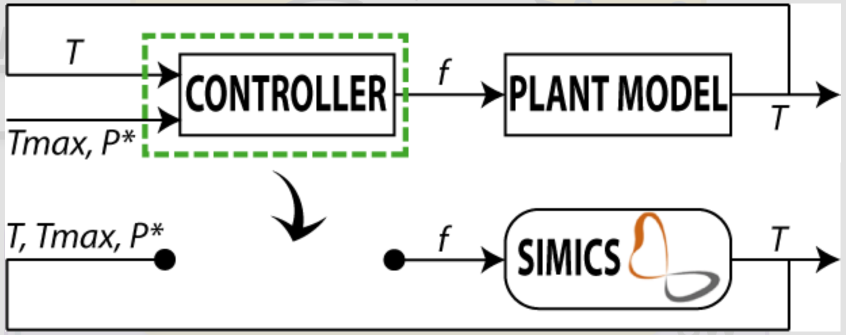
Virtual Platform

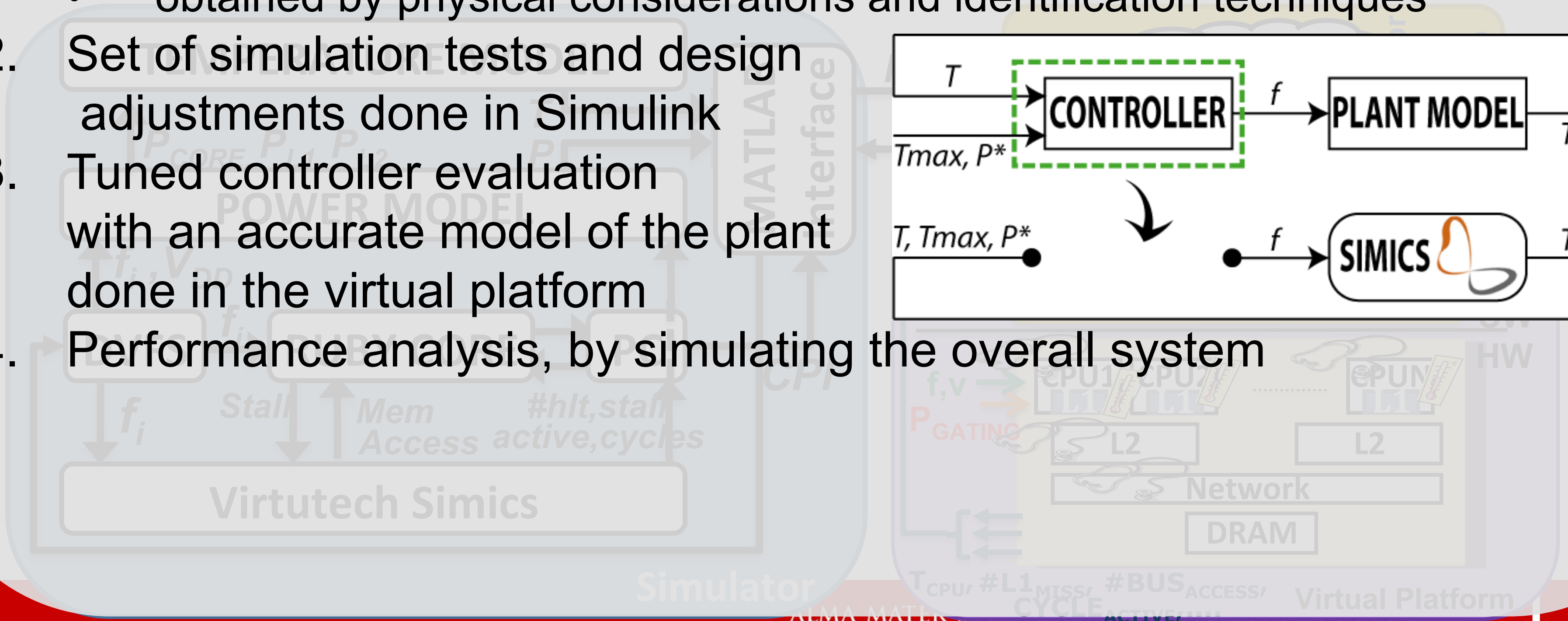
Mathworks Matlab interface:

- New module named Controller in RUBY
- Initialization: starts the Mathworks Matlab engine concurrent process,
- Every N cycle - wake-up:
  - send the current performance monitor output to the Mathworks Simulink model
  - execute one step of the controller Mathworks Simulink model
  - propagate the Mathworks Simulink controller decision to the DVFS module

CONTROL-STRATEGIES DEVELOPMENT CYCLE

1. Controller design in Mathworks Matlab/Simulink framework
   - system represented by a simplified model
   - obtained by physical considerations and identification techniques
2. Set of simulation tests and design adjustments done in Simulink
3. Tuned controller evaluation with an accurate model of the plant done in the virtual platform
4. Performance analysis, by simulating the overall system





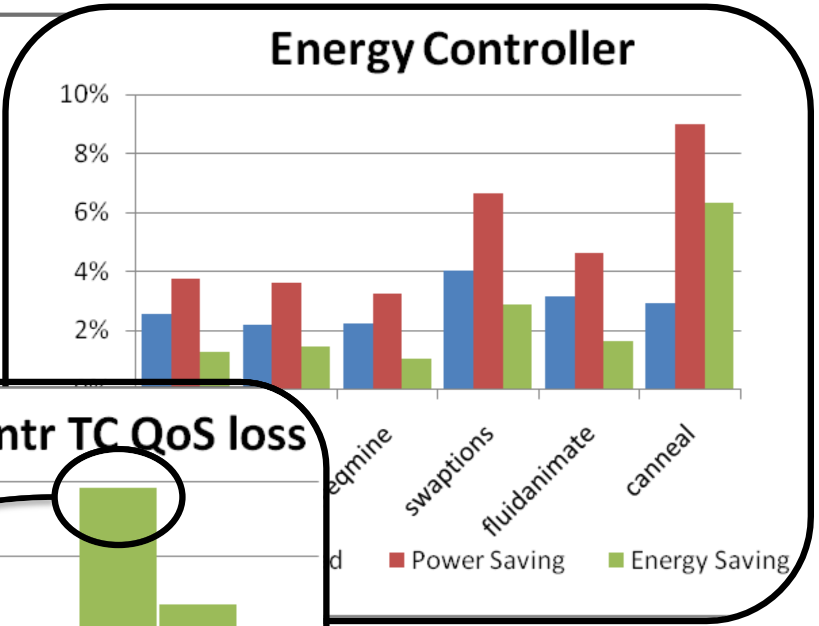
Results

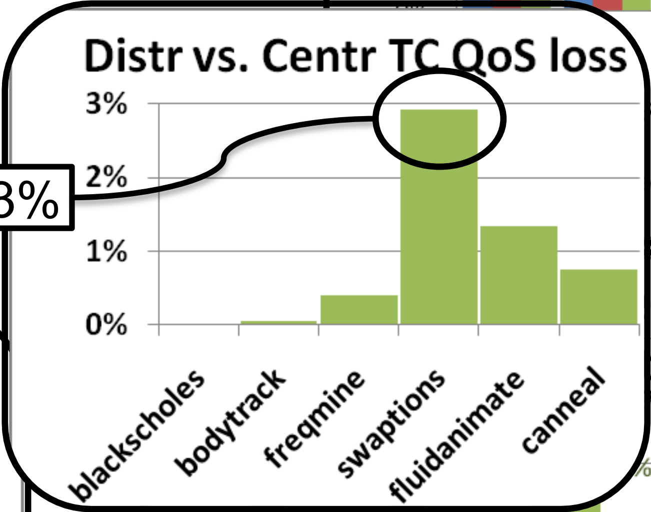
Energy Controller (EC)

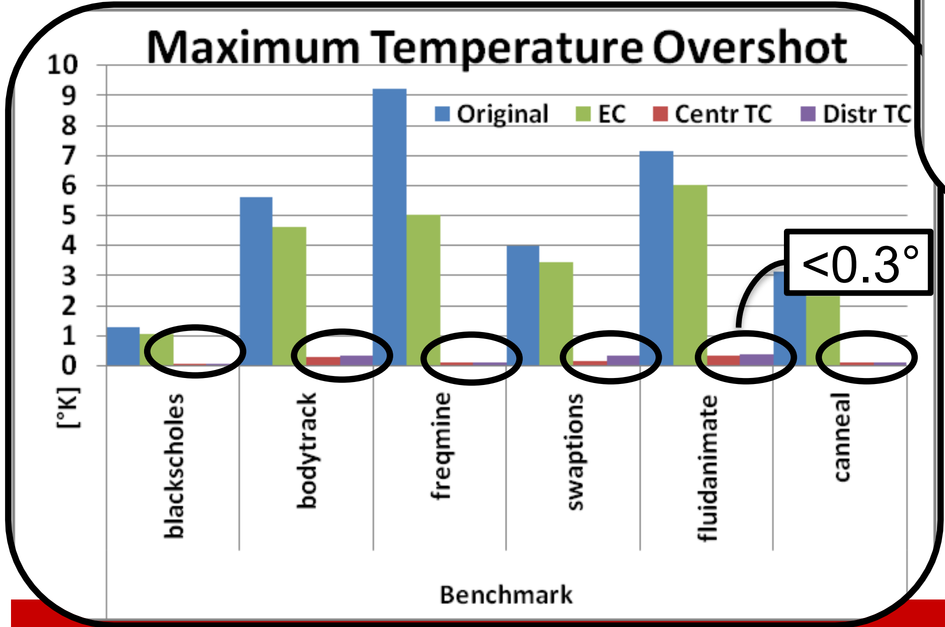
- Performance Loss < 5%
- Energy minimization

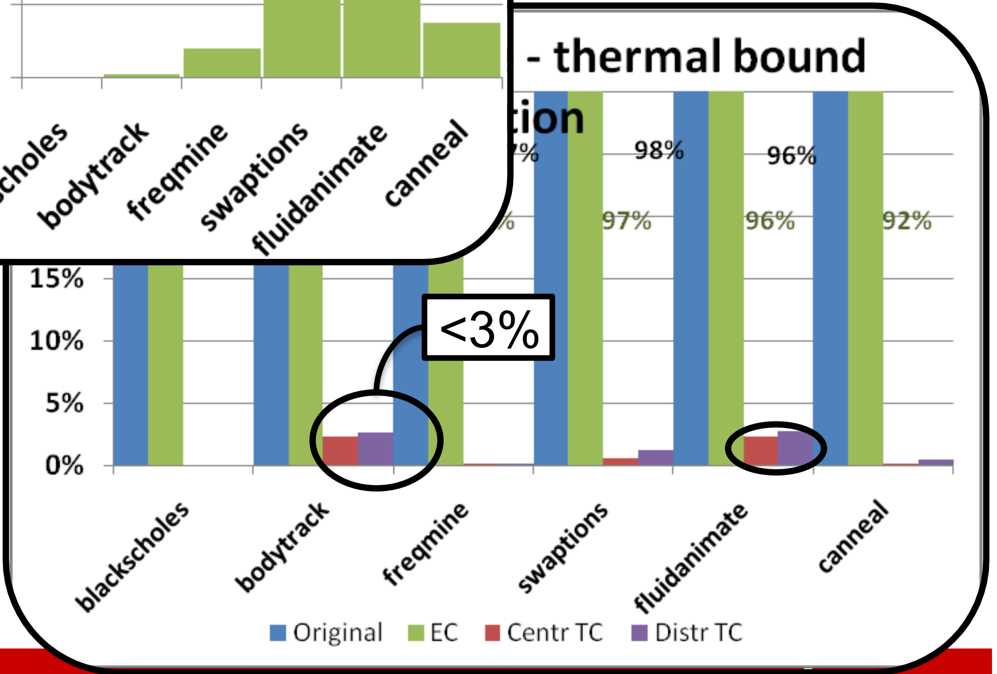
Temperature Controller (TC)

- Complexity reduction
  - 2 explicit region for controller
- Performs as the centralized <3%
  - Thermal capping

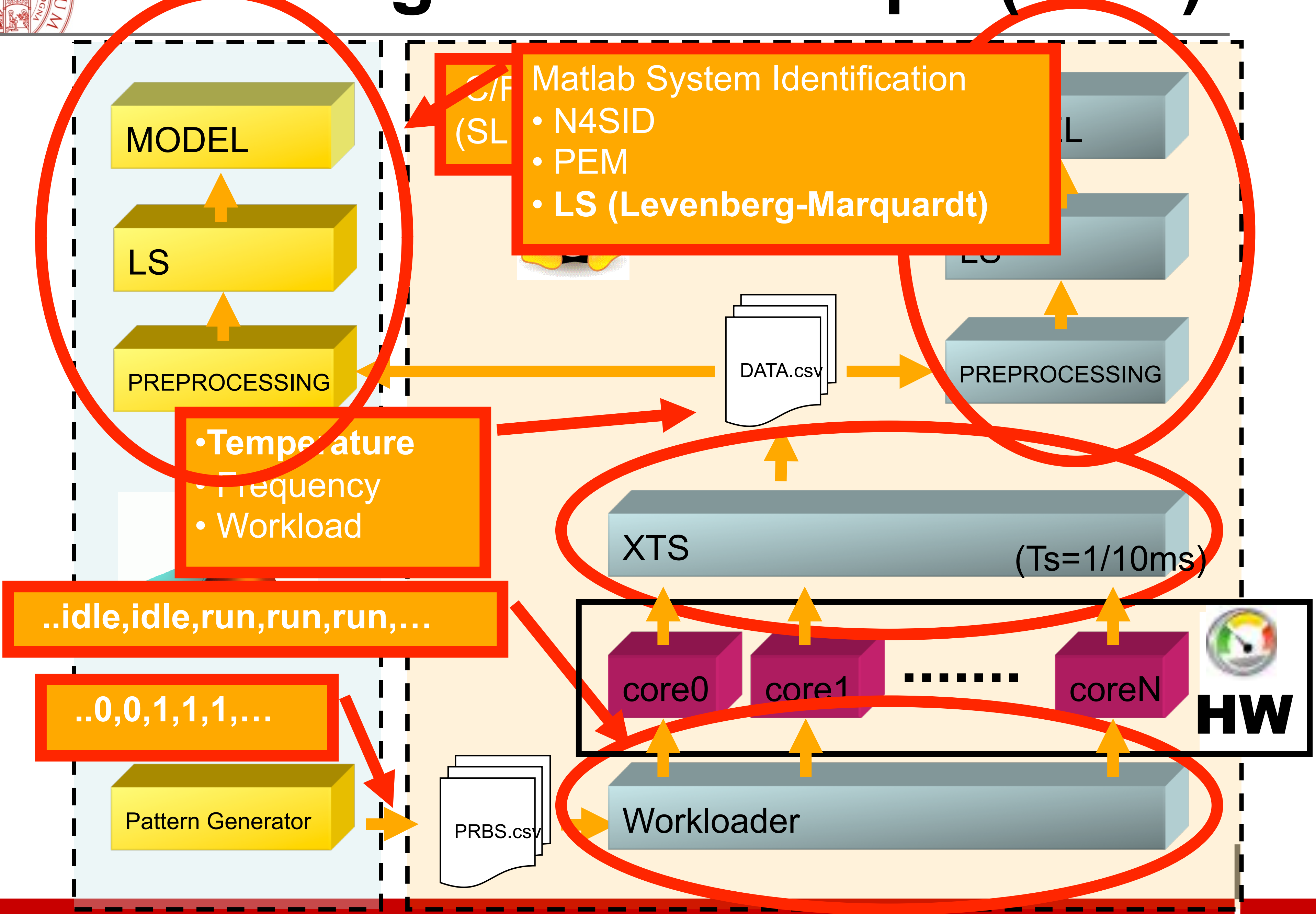








Working on Real Chips (Intel)



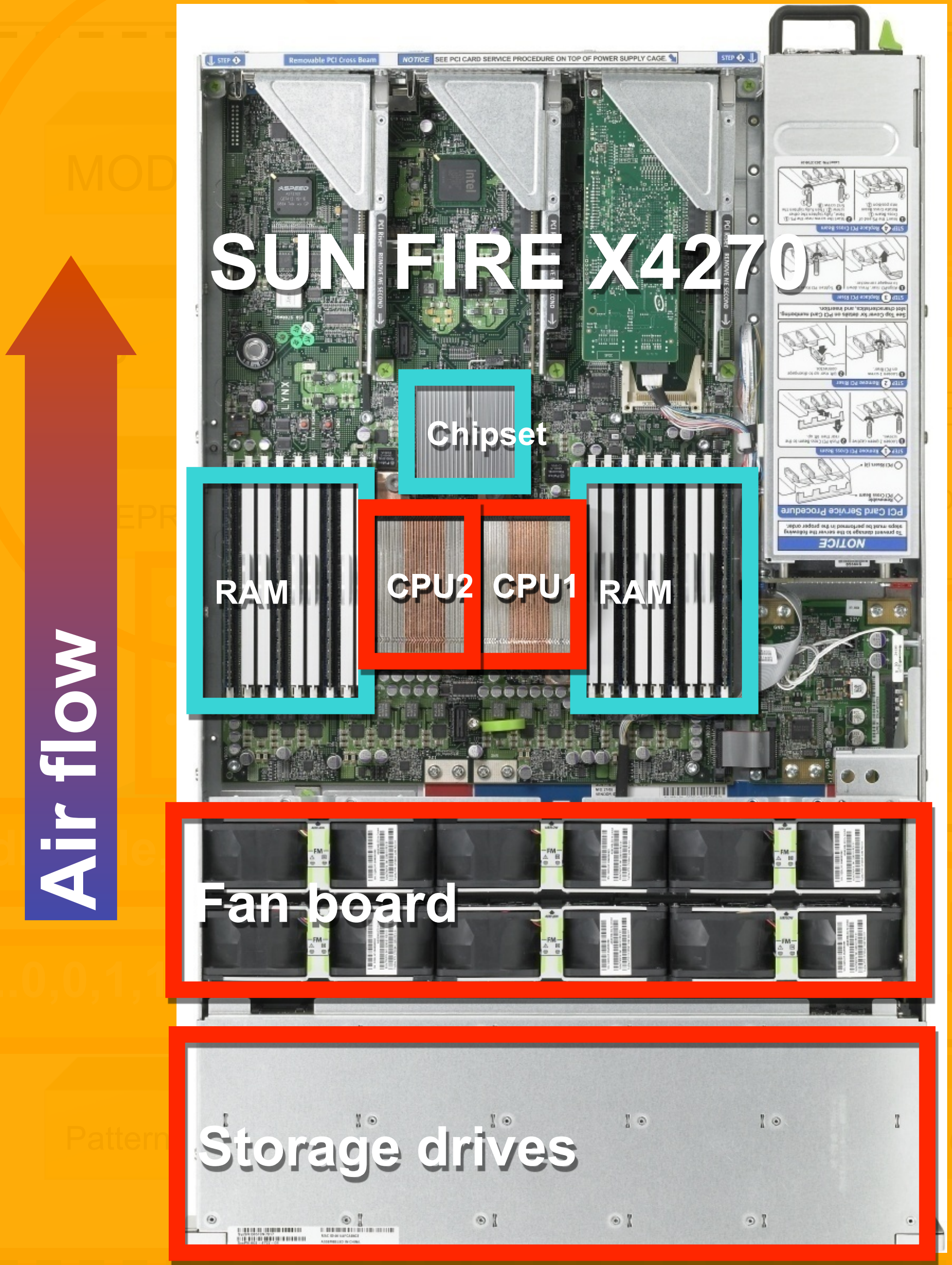
Matlab System Identification
- N4SID
- PEM
- **LS (Levenberg-Marquardt)**

- **Temperature**
- Frequency
- Workload

..idle,idle,run,run,run,…

..0,0,1,1,1,…

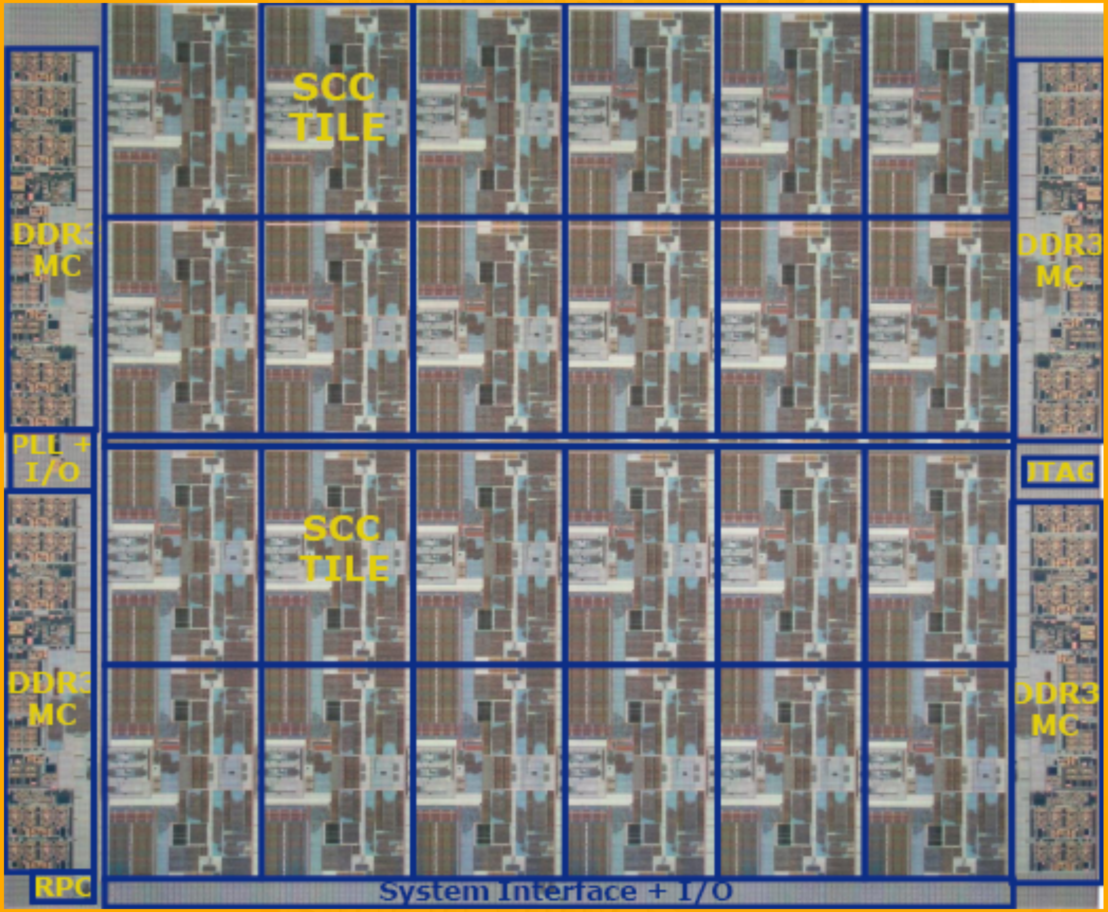
Working on Real Chips (Intel)



- Intel Nehalem
- 8core/16thread
- 2.9GHz
- 95W TDP
- IPMI

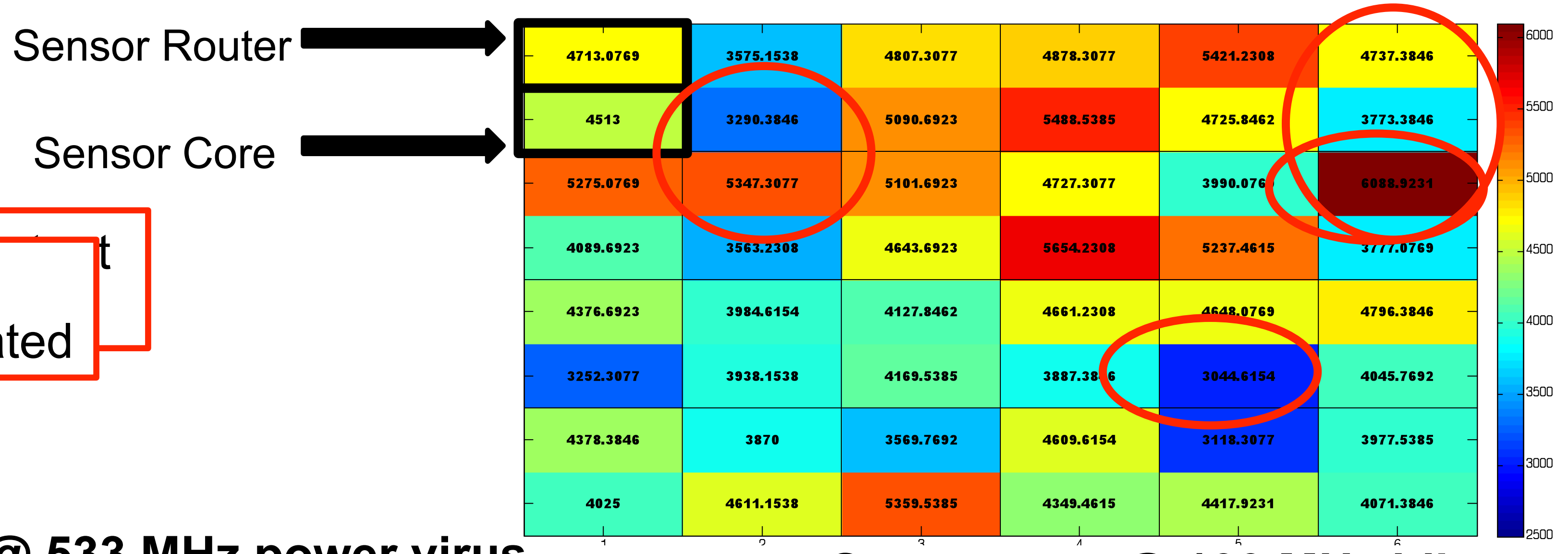
Working on Real Chips (Intel)

Single Chip Cloud (45nm)



- 567.1 mm2
- 48cores @1GHz
- 2GHz NoC
- 25-125W
- 27 (f), 8 (V) islands

Thermal Sensor Variability

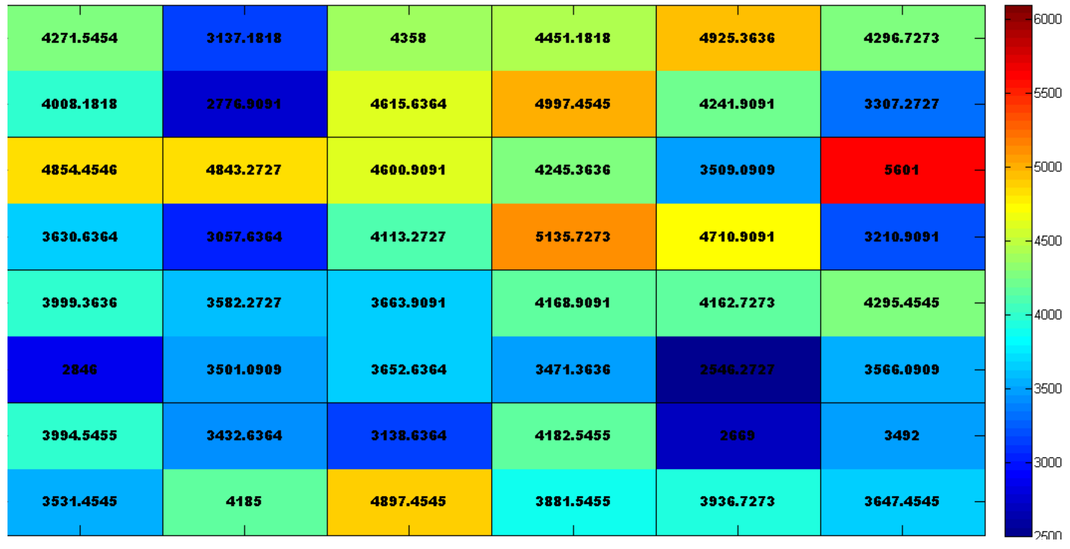


4713.0769	3575.1538	4807.3077	4878.3077	5421.2308	4737.3846
4513	3290.3846	5090.6923	5488.5385	4725.8462	3773.3846
5275.0769	5347.3077	5101.6923	4727.3077	3990.0769	6088.9231
4089.6923	3563.2308	4643.6923	5654.2308	5237.4615	3777.0769
4376.6923	3984.6154	4127.8462	4661.2308	4648.0769	4796.3846
3252.3077	3938.1538	4169.5385	3887.3846	3044.6154	4045.7692
4378.3846	3870	3569.7692	4609.6154	3118.3077	3977.5385
4025	4611.1538	5359.5385	4349.4615	4417.9231	4071.3846

Sensor map @ 100 MHz idle

Spatially
not-correlated

Sensor map @ 533 MHz power virus



4271.5454	3137.1818	4358	4451.1818	4925.3636	4296.7273
4008.1818	2776.9091	4615.6364	4997.4545	4241.9091	3307.2727
4854.4546	4843.2727	4600.9091	4245.3636	3509.0909	5601
3630.6364	3057.6364	4113.2727	5135.7273	4710.9091	3210.9091
3999.3636	3582.2727	3663.9091	4168.9091	4162.7273	4295.4545
2846	3501.0909	3652.6364	3471.3636	2546.2727	3566.0909
3994.5455	3432.6364	3138.6364	4182.5455	2669	3492
3531.4545	4185	4897.4545	3881.5455	3936.7273	3647.4545

Outline

- Introduction
- Scalable Control
- Scalable model learning
- Experimental Environment
- Challenges ahead

The 1,000 Cores Chip

- STM-CEA Platform 2012 project
  - Die with 4 16-cores tiles with L1 & L2 → few tens of mm2 (28nm)
  - SCC die → 20 of these dies: 1,280 cores
  - Thousands of Vdd, f domains
- 3D stacking currently the only technology which can provide sufficient L3 bandwidth
  - Vertical thermal dissipation!
  - Heterogeneous requirements (DRAM≠LOGIC)
- Major static and dynamic variability

Power management Challenges

- Truly scalable algorithms → O(NlogN)
- Hardware support needed (e.g. DPM NoC)
- Cross-layer algos are needed
  - Real-time intra+inter layer communication
  - Abstraction and filtering
  - Multi-scale
- The threat of non-linearity
  - Hybrid control complexity (MILP is NP-HARD)
  - Lack of robustness (Ill-conditioning) and stability proofs

SoCs as complex systems (societies/markets)
→ DPM as political sociology/finance?

Thank you!

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA