Architecture-level Thermal Behavioral Characterization for Multi-Core Microprocessors

Duo Li and Sheldon X.-D. Tan

Department of Electrical Engineering
University of California, Riverside, CA

Murli Tirumala
Intel Corporation

UCRIVERSIDE UNIVERSITY OF CALIFORNIA

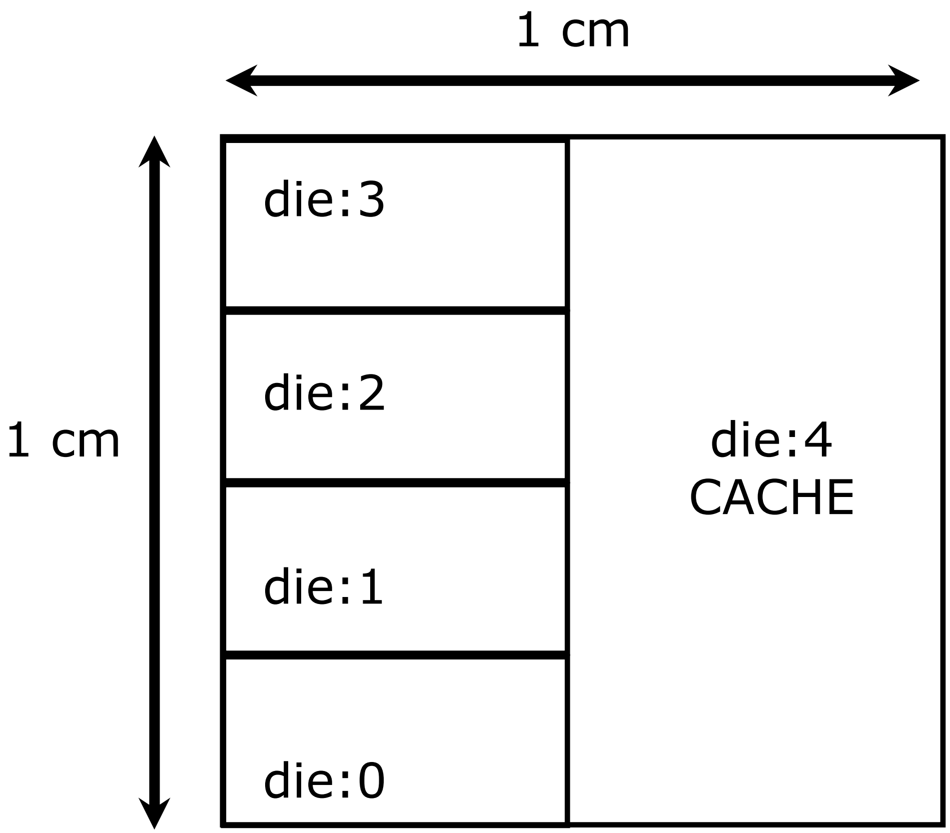
Outline

- Introduction and Motivation
  - The need for dynamic thermal management (DTM)
  - Why software thermal sensors
- Power estimation for functional units
- Architecture level thermal modeling
- Summary

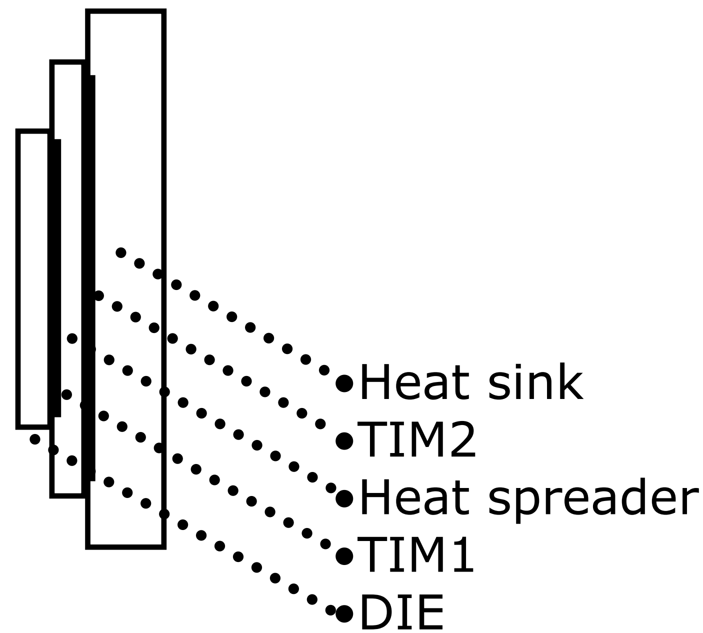
Outline

- Introduction and Motivation
- Architecture level thermal modeling
  - Intel quad-core structure
  - Transfer function
  - Matrix pencil method
  - Log-sale sampling and stabilization
  - Reduction of thermal models
  - Simulation results
- Summary

Top view: quad core

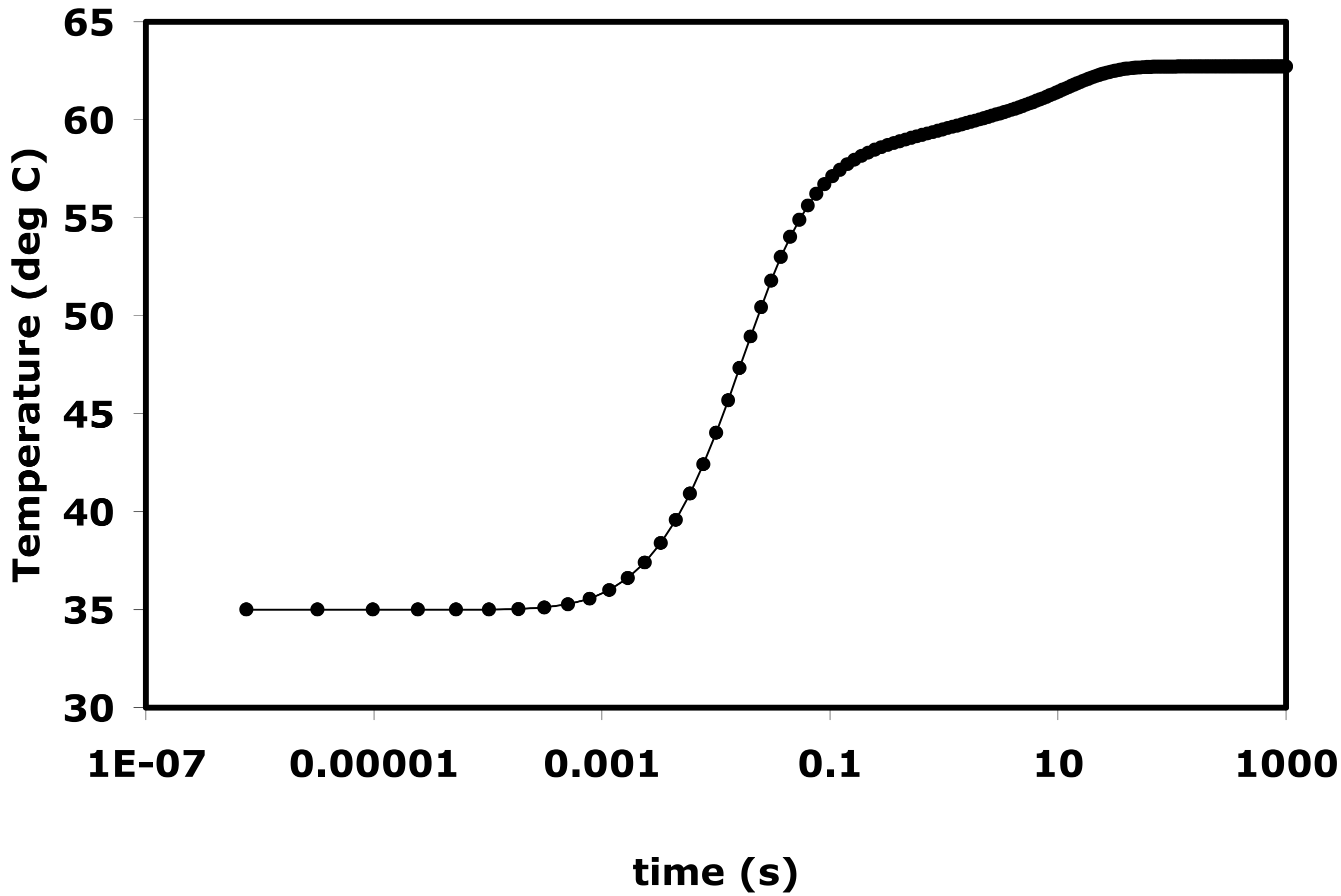


Lateral view

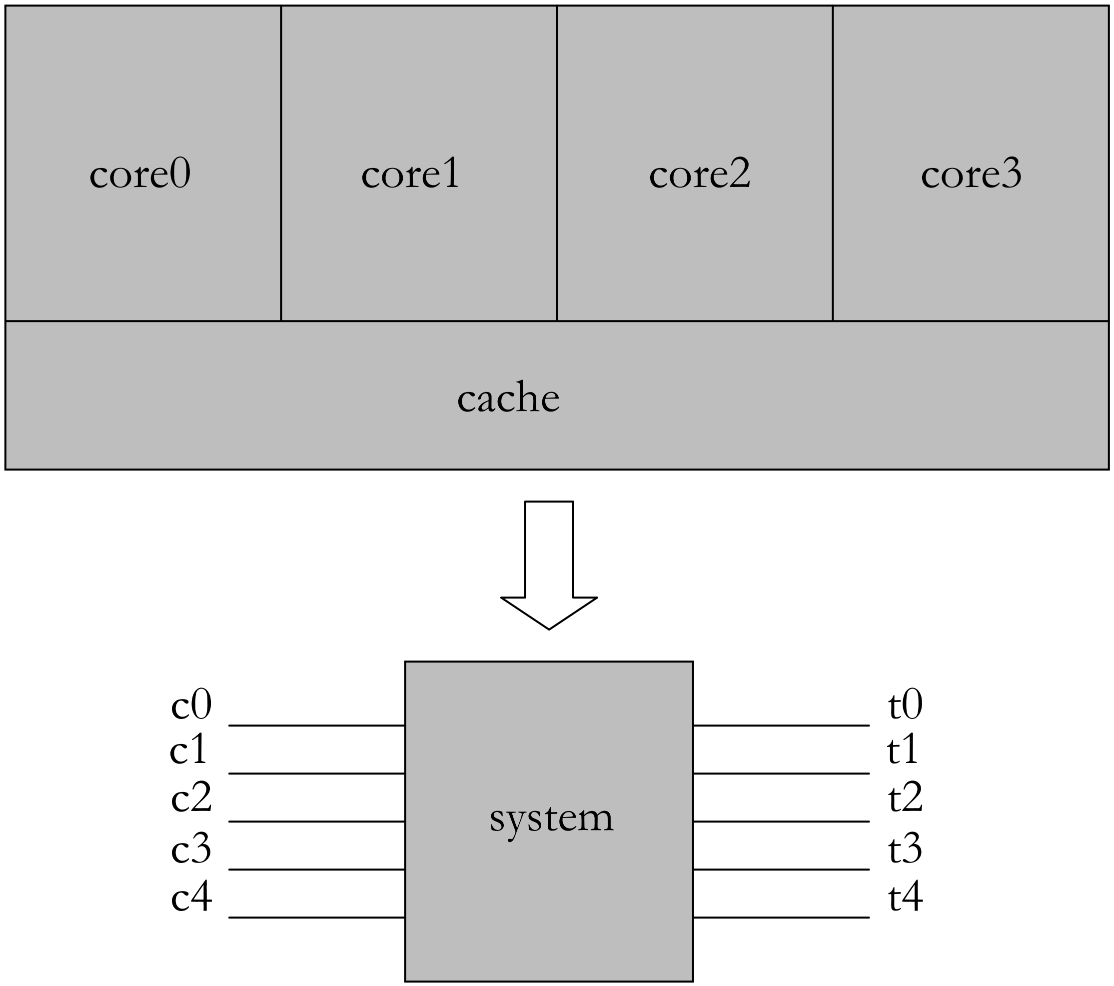


Temperatures reported are on the die bottom face
and centered with each die region

Active core 0 at 20 W: T distribution



Quad-core



core0
core1
core2
core3
cache
system
c0
c1
c2
c3
c4
t0
t1
t2
t3
t4

Transfer function

LTI (linear, time-invariant systems)

input signal $x(t)$ and output $y(t)$

$$Y(s) = H(s)X(s)$$

or

$$H(s) = \frac{Y(s)}{X(s)}$$

where $H(s)$ is the transfer function of the LTI system

Pole-residue representation

- Pole-zero

$$H(s) = \frac{b_0 + b_1 s + \ldots + b_m s^m}{1 + a_1 s + \ldots + a_n s^n}$$

$$H(s) = K\,\frac{(s - z_1)\cdot(s - z_2)\ldots(s - z_m)}{(s - p_1)\cdot(s - p_2)\ldots(s - p_n)}$$

- Pole-residue

$$H(s) = \sum_{i=1}^{n} \frac{k_i}{s - p_i}$$

Matrix Pencil

- Used for extracting poles and residues.
- Specifically, works for EM transient signal.

$$y_k = \sum_{i=1}^{M} r_i \exp(p_i \Delta t k)$$

- k = 0, 1, …, N-1,
- r_i are the complex residues,
- p_i are the complex poles,
- Δt is the sampling interval.

General Pencil of Function Method

ALGORITHM: GPOF
Input: sampling vectors $\mathbf{y}_i = [y_i, y_{i+1}, \ldots, y_{i+N-L-1}]^T$
Output: poles vector $\mathbf{p}$ and residues vector $\mathbf{r}$

1. Construct matrices Y_1 and Y_2.

   $Y_1 = [\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{L-1}] \quad Y_2 = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_L]$

2. Singular value decomposition (SVD) of Y_1. $Y_1 = UDV^H$

3. Construct matrix Z. $Z = D^{-1}U^H Y_2 V$

4. Eigen-decomposition of Z. $Z_0 = eig(Z)$

   find poles vector: $p_i = \frac{log(z_i)}{\Delta t}$

5. Solve R_1 and R_2 from $Y_1 = Z_1 R Z_2$ and $Y_2 = Z_1 R Z_0 Z_2$.

$$Z_1 = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ z_1 & z_2 & \ldots & z_M \\ \vdots & \vdots & \ldots & \vdots \\ z_1^{N-L-1} & z_2^{N-L-1} & \ldots & z_M^{N-L-1} \end{bmatrix}$$

$$Z_2 = \begin{bmatrix} 1 & z_1 & \ldots & z_1^{L-1} \\ & & \ldots & \\ \vdots & \vdots & \ldots & \vdots \\ 1 & z_M & \ldots & z_M^{L-1} \end{bmatrix}$$

   find residues vector: $\mathbf{r} = \frac{R_1 + R_2}{2}$

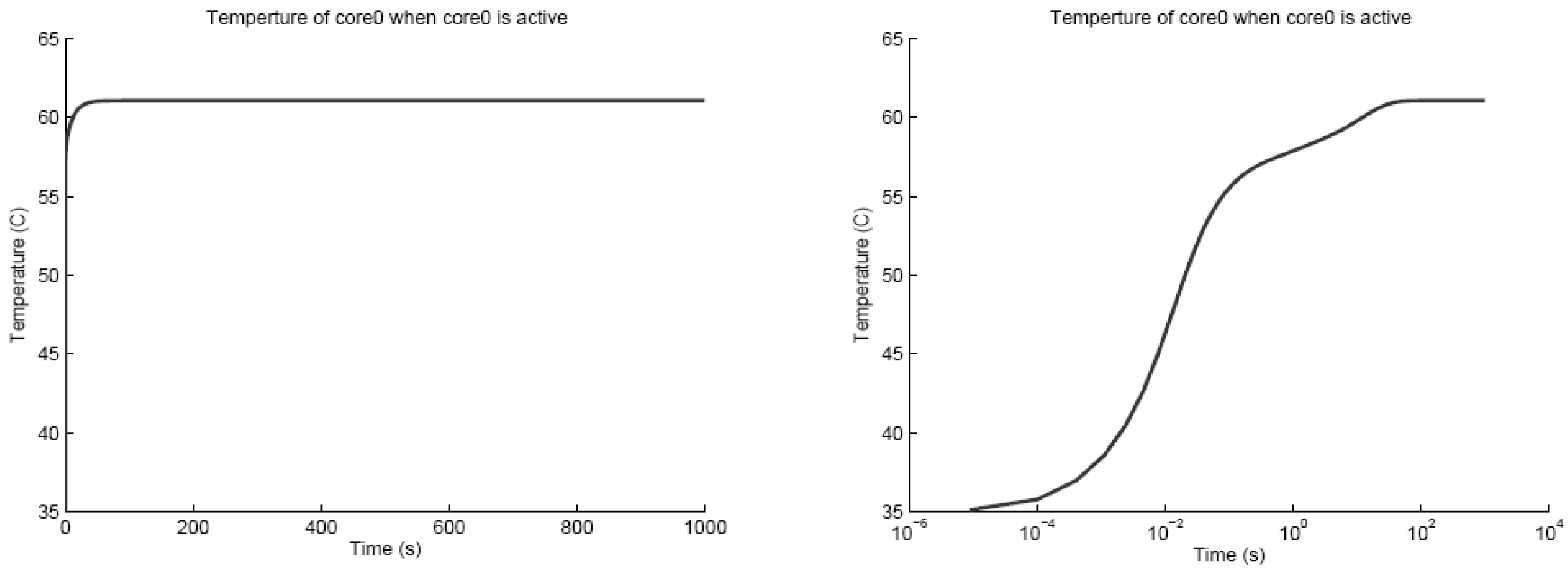
How to choose M and L

- M is model order number.
- L is sampling window size.
- N is the number of total sampled points.
- For GPOF, $M \le L \le N-M$. Allow different window sizes and pole numbers.
- Typically, choosing $L = N/2$ can yield better results.

Sampling issue

- Traditional MP using constant interval time for sampling.
  - Temperature increase dramatically fast in the first few seconds.
- Log-scale sampling is a good way.
- Numerical differentiation for computing impulse response.
  - Need to compute the impulse response instead of step responses, which are given.

Linear vs Log-scale



(a) Linear time scale thermal step response.

(b) Logarithmic time scale thermal step response.

Log-scale sampling

- Temperature increases very fast in a first few seconds.
- Temperature needs a very long time to get steady.
- Offset to make sure it starts at t=0.
- Get the response back

$$\mathbf{y}'(t) = \mathbf{y}(ln(t) - ln(t_0))$$

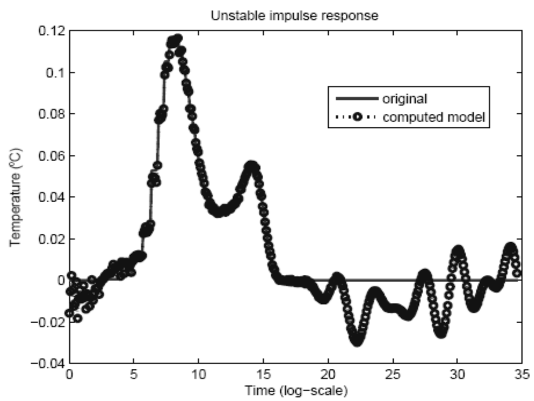
y'(t): response in normal time scale;

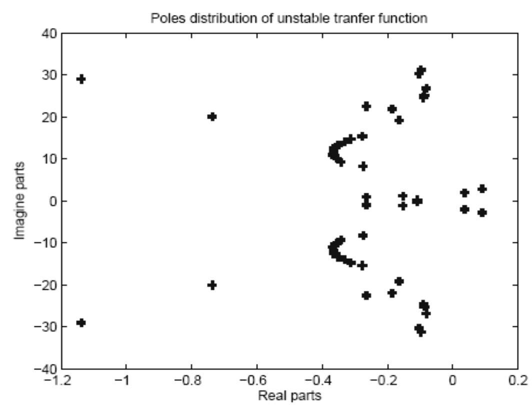
y(t): response in log-scale;

t_0: offset, usually a very small value.

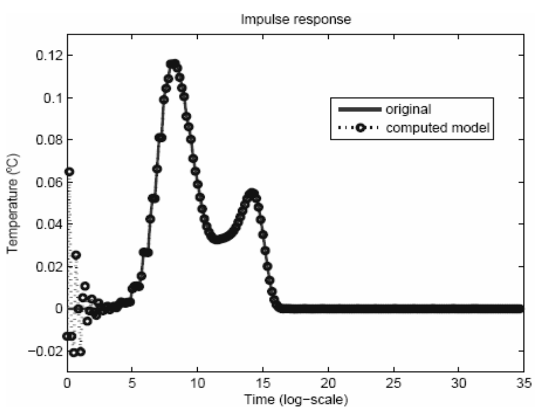
Numerical Differential and Stabilization (1)

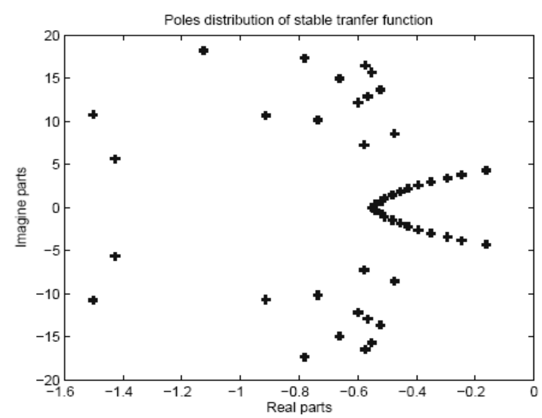
- Stable pole extraction
  - Only negative poles





Impulse responses with some positive poles

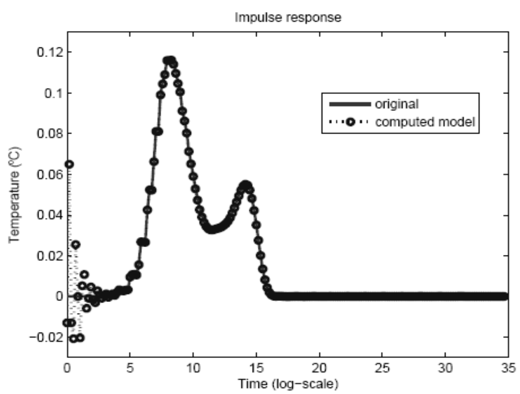


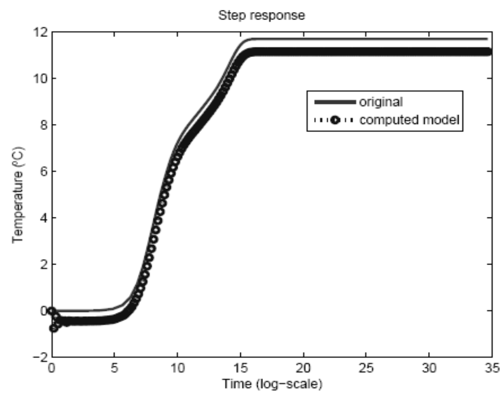


Impulse responses with only negative poles

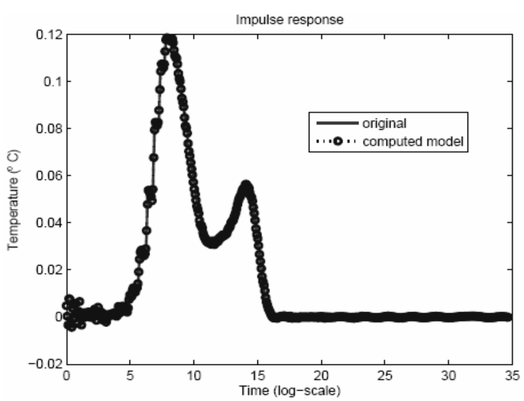
Numerical Differential and Stabilization (2)

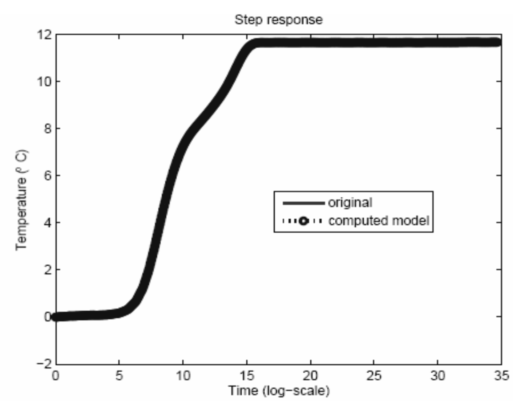
- Stabilizing the starting response
  - Increasing sampling points





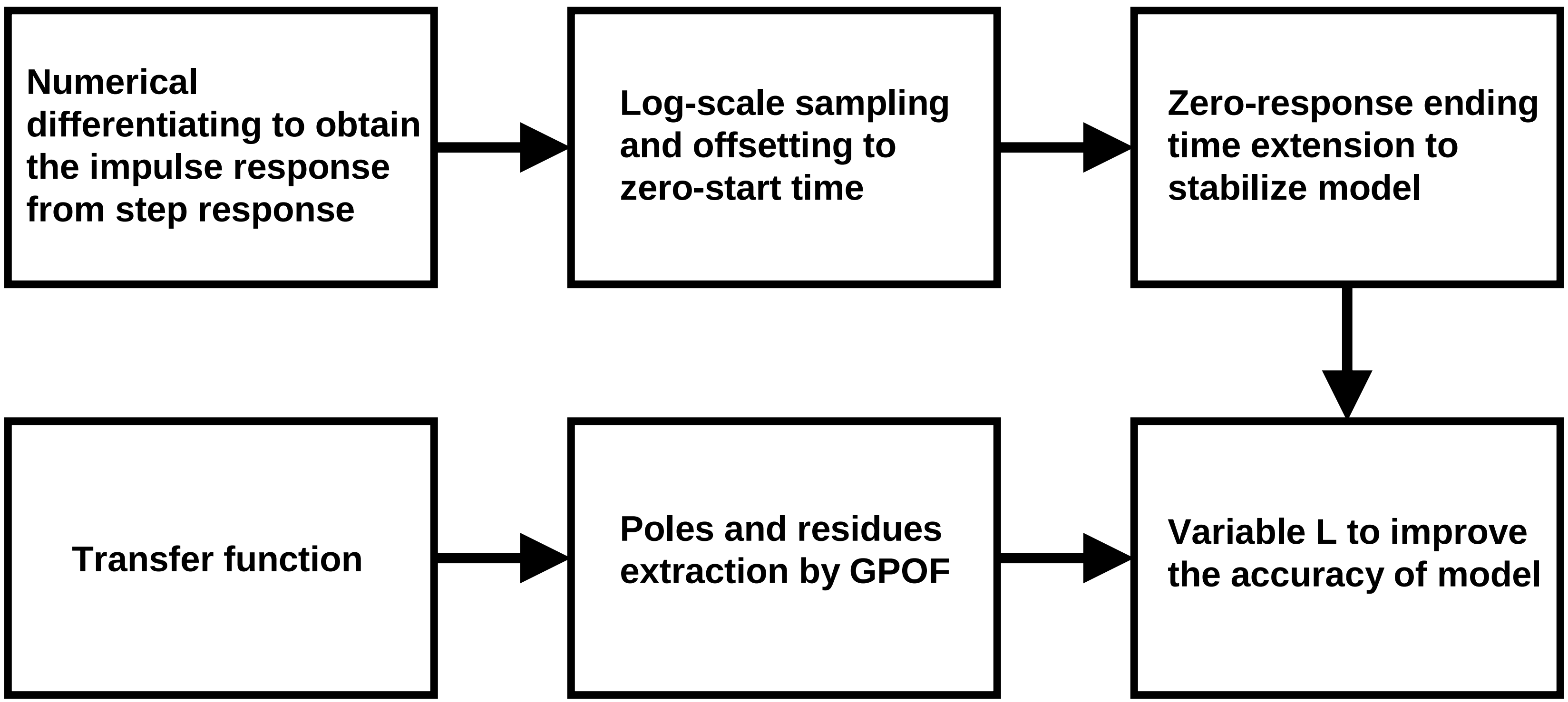
Impulse and step responses for L = 100





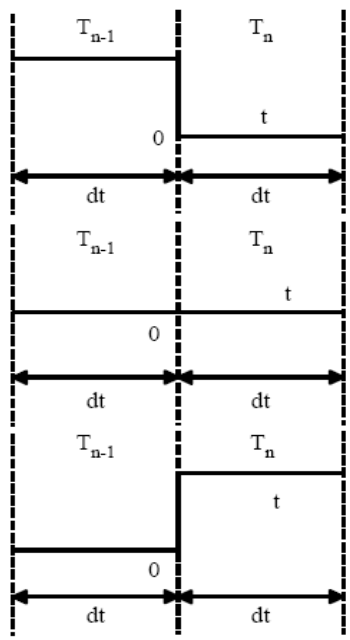
Impulse and step responses for L = 200

Thermal modeling flow



Recursive computation

- Computation complexity is only O(n)
  - n is the number of time segments or the number of power traces



$$y_n(t) = y_{n-1}(t + dt) - y_0(t) \qquad \textbf{a)}$$

$$y_n(t) = y_{n-1}(t + dt) \qquad \textbf{b)}$$

$$y_n(t) = y_{n-1}(t + dt) + y_0(t) \qquad \textbf{c)}$$

Reduction of thermal models

- State-space realization

$$Y(s) = \frac{r}{s-p} + \frac{\overline{r}}{s-\overline{p}} \qquad p = a + bj \qquad r = c + dj$$

$$\mathbf{A}_i = \begin{bmatrix} a & b \\ -b & a \end{bmatrix} \qquad \mathbf{b}_i = \begin{bmatrix} 2 \\ 0 \end{bmatrix} \qquad \mathbf{c}_i^T = [c \quad d]$$

- MOR by PRIMA

$$\mathbf{A}_r = \mathbf{V}^T \mathbf{A} \mathbf{V} \qquad \mathbf{b}_r = \mathbf{V}^T \mathbf{b} \qquad \mathbf{c}_r^T = \mathbf{c}^T \mathbf{V}$$

- Reduced transfer function

$$Y(s) = \sum_{k=1}^{q} \frac{\mu_k \cdot v_k}{s - \lambda_k}$$

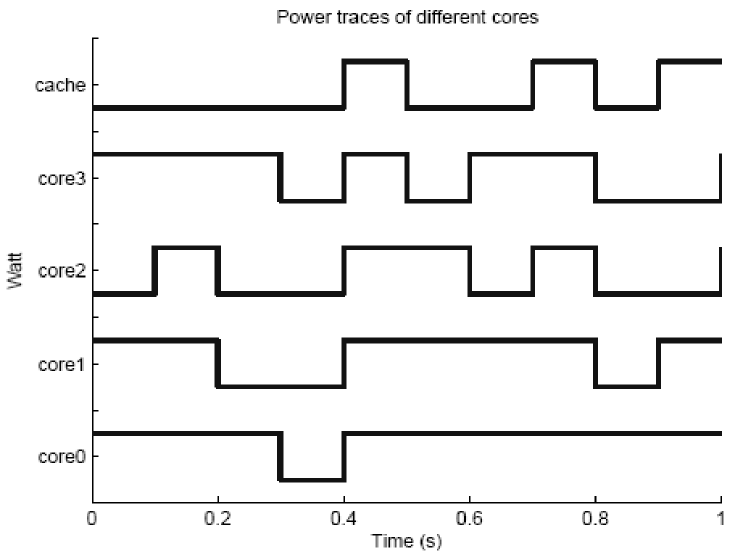
$$\mathbf{A} = \mathbf{P} \Lambda \mathbf{P}^{-1}$$

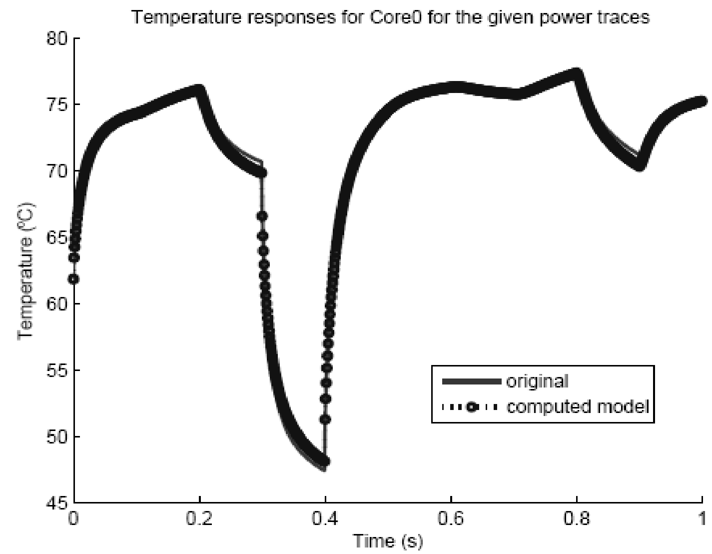
λ_k is the kth diagonal element of Λ, μ_k is the kth element of $c_r^T P$, v_k is the kth element of $P^{-1} b_r$ and q is the reduced order

Training

- Extracting 5 groups of poles and residues using matrix pencil method.
- Obtaining the transfer function of the system.
- Simulating the output of the system (thermal simulation).
- Linear combination
- Benchmark provided by Intel, random power input.

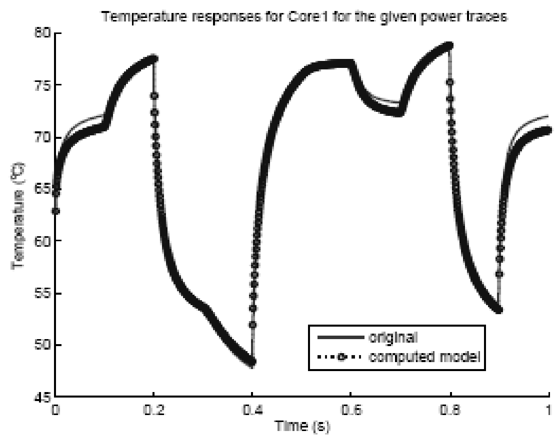
Simulation result (1)

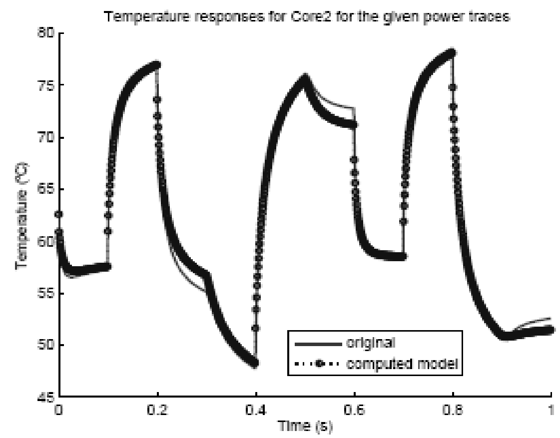




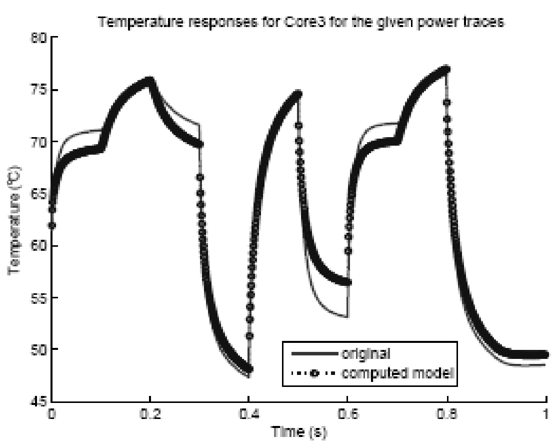
Random power input on all cores and thermal response for Core0.

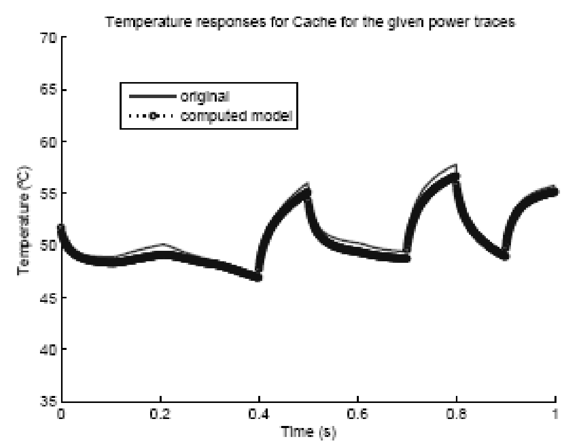
Simulation result (2)





Thermal response of Core1 and Core2





Thermal response of Core3 and Cache

Simulation result (3)

- **Features of the errors between measured and computed temperatures (M = 50)**

	Error (°C)			Error percentage	
	Maximum	Mean	Std. deviation	Maximum	On average
Core0	1.05	0.34	0.23	1.56%	0.50%
Core1	1.67	0.53	0.48	2.44%	0.78%
Core2	1.78	0.61	0.47	2.56%	0.98%
Core3	3.33	1.10	0.82	6.09%	1.80%
Cache	1.05	0.63	0.22	1.84%	1.22%

- **Errors of the maximal and minimum peaks (M = 50)**

	Maximal peak			Minimum peak		
	Measured (°C)	Error (°C)	Percentage	Measured (°C)	Error (°C)	Percentage
Core0	77.27	0.45	0.58%	47.47	0.38	0.79%
Core1	78.86	0.04	0.05%	47.81	0.35	0.73%
Core2	78.55	0.38	0.48%	47.77	0.24	0.51%
Core3	76.48	0.75	0.98%	47.38	0.45	0.95%
Cache	57.80	0.99	1.72%	48.86	0.11	0.23%

Simulation result (4)

- Reduction of thermal models (M = 30)

Errors of the maximal and minimum peaks and means (M = 30)

	Maximal peak		Minimum peak		Mean	
	Error ($^\circ C$)	Percentage	Error ($^\circ C$)	Percentage	Error ($^\circ C$)	Percentage
Core0	0.40	0.52%	0.46	0.96%	0.36	0.48%
Core1	0.12	0.15%	0.49	1.00%	0.47	0.69%
Core2	0.06	0.07%	0.34	0.70%	0.56	0.88%
Core3	0.76	0.98%	0.53	1.11%	1.11	1.66%
Cache	1.01	1.78%	0.01	0.02%	0.03	1.25%

Speedup when M = 30 compared to M = 50

	Run time (s) when $M = 50$	Run time (s) when $M = 30$	Time reduced
Core0	1.31	0.80	38.9%
Core1	1.29	0.78	39.5%
Core2	1.28	0.78	39.1%
Core3	1.28	0.78	39.1%
Cache	1.30	0.79	39.2%

Conclusion

- Efficient on-chip thermal analysis technique is required for on-chip dynamic thermal management study and run-timing DTM.
- Developed a new estimation method to compute real microprocessor Function Units' power.
- Developed behavioral thermal modeling techniques based on matrix pencil.
- Developed thermal reduction techniques.