Dynamic Thermal Management for Multi-core Microprocessors Considering Transient Thermal Effects

Zao Liu, Tailong Xu,
Sheldon X.-D. Tan,
Hai Wang[$]

Department of Electrical Engineering
University of California, Riverside, CA
[$]UESTC, China

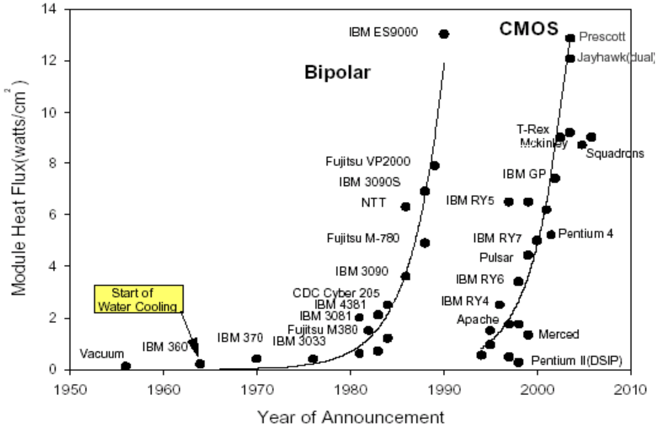
UC RIVERSIDE
UNIVERSITY OF CALIFORNIA

Content

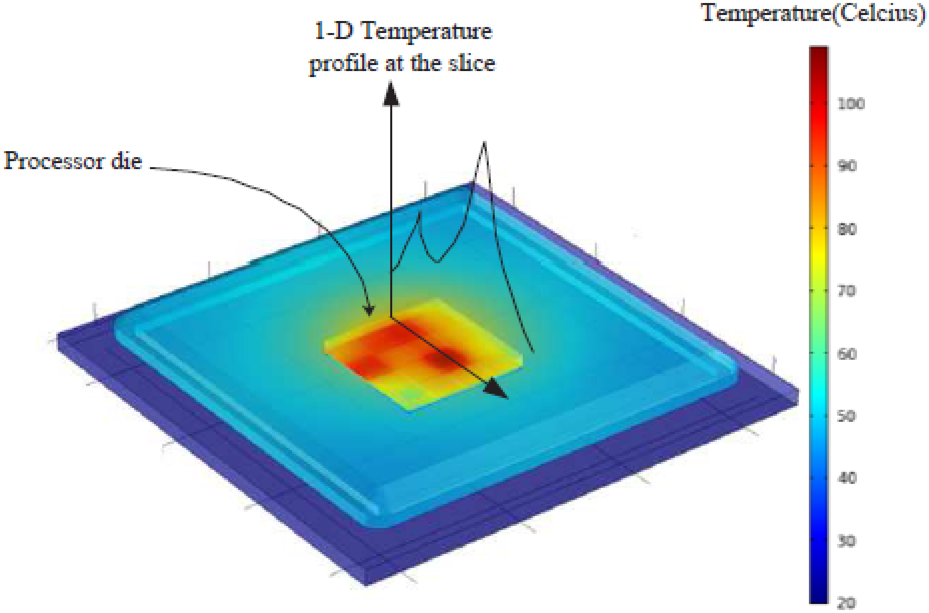
- Introduction to thermal management
  - Motivation
  - Problem description – task scheduling based thermal management
- Proposed method
- Experimental result
- Conclusion

Why dynamic thermal management?

- Thermal management is vital for high performance multi-core processor. [ITRS‘11, Skadron:ISCA’03]
  - Power density keeps increasing, device scaling down, 3-D IC
  - Cooling solution for the worst cases can be expensive
  - High temperature causes reliability issues



Temperature is the limiting factor and first-tiered design constraints



Temperature distribution of multi-core chip

Dynamic thermal management strategies

- Global clock gating
  - Freezing all dynamic operations and turn off clocks
  - Save power but, will hurt performance
- Dynamic voltage and frequency scaling (DVFS) [Herbert: ISLPED’07]
  - Save power also, but hurt performance and difficult for timing (especially for core-based DVFS).
  - But may has difficult for reducing leakage (lower supply voltage leads to large leakage)
- Task migrations[Powell:ACM’03, Ge:DAC’10]
  - Migrate heavy tasks away from the heavily loaded core to avoid elevated temperature
  - Has less impacts on performances. Can reduce the thermal gradients and thus thermal-cycling based reliability issues.

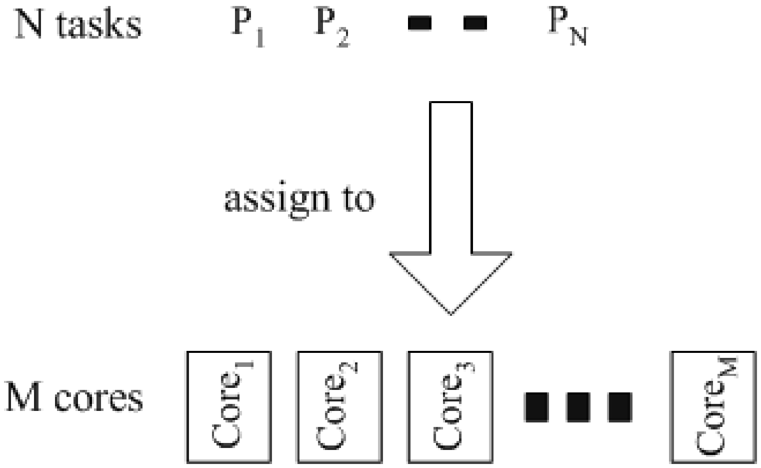
Thermal sensitive long-term reliability issues

- Electro-migration (metal wires)
- Stress migration (metal wires)
- Time dependent dielectric breakdown(devices)
- Temperature cycling (wires and devices)
- Negative biased temperature instability NBTI (PMOS devices)

Many of those failure effects are exponentially depend on the temperature
And are very sensitive to temperature gradients in time and space

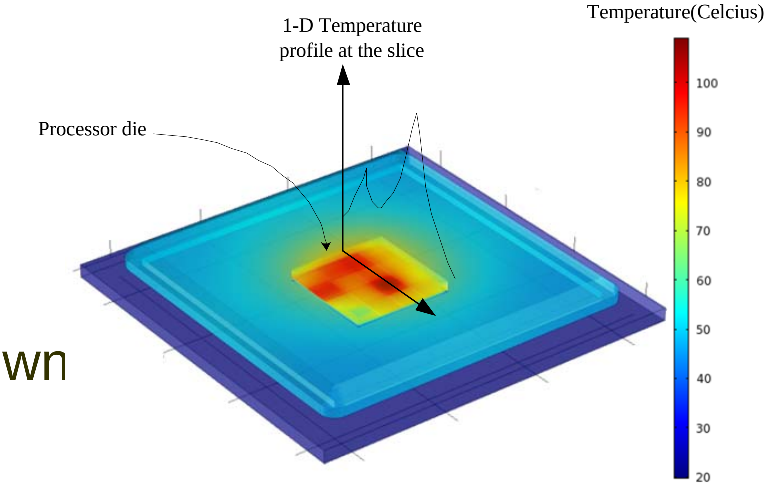
Problem description – task scheduling based thermal management to reduce temperature gradients

- **Motivation:** to avoid excessive on-chip temperature without sacrifice data throughput and reduce transient temperature gradients across a chip
- **Problem formulation:** given N tasks for M processor cores, find a task scheduling method to minimize the temperature variance, and reduce the number of on-chip hot spots.
- **Critical part:** identify a suitable core to take on the heavy load without generating thermal emergency.



Status quo of task migrations for multi-core microprocessors

- Traditional task migrations technique tries to move heaviest task to the core with lowest temperature.
- Based on steady-state temperature (resistance-only thermal circuits)
- It is very expensive to perform full-blown transient thermal analysis during the process.
- Transient and dynamic thermal effects can be very significant for today’s multicore processors and are reliability relevant



Some relevant works in task migration scheme

- Heat-and-run: [Powell:ACM’03, Ge:DAC’10]
  - Method: Migrate the heaviest task to the core with lowest temperature.
  - Problem: Sub-optimized task distribution due to the transient thermal effect because heat capacitance is not considered.
- Other methods:
  - Ad-hoc approach considering neighboring temperature effect. [Liu:DATE’12]

What is new in the proposed thermal-aware task migrations?

- Will consider the thermal dynamic effects
- But still try to avoid the full-blown transient thermal analysis to determine the task migrations to maintain efficiency.
- Propose to apply moment-matching based analysis technique and unique task migration scheme.

Content

- Introduction to thermal management
  - Motivation
  - Problem description – task scheduling based thermal management
- Proposed method
- Experimental result
- Conclusion

Thermal analysis in frequency domain

- A thermal circuit can be descripted by general linear circuit system

$$C\dot{x} + Gx = Bu$$

- Laplace transformation in s domain

$$GX(s) + C(sX(s) - X(0)) = BU(s)$$

  - Moment matching for model order reduction

$$X(0) = 0 \qquad U(s) = 1$$

  - Moment matching with initial state

$$X(0) \neq 0$$

- Taylor expansion at s=0

$$G(T_0 + T_1 s + T_2 s^2 + \ldots) + sC(T_0 + T_1 s + T_2 s^2 + \ldots)$$
$$= sCT(t_0) + B(U_0 + U_1 s + U_2 s^2 + \ldots)$$

Recursively Moment Generation and Pade Approximation

$$\boxed{T_0 = G^{-1}(BU_0 + CT(t_0))}$$

$$T_1 = G^{-1}(BU_1 - CT_0)$$

$$T_2 = G^{-1}(BU_2 - CT_1)$$

$$\ldots$$

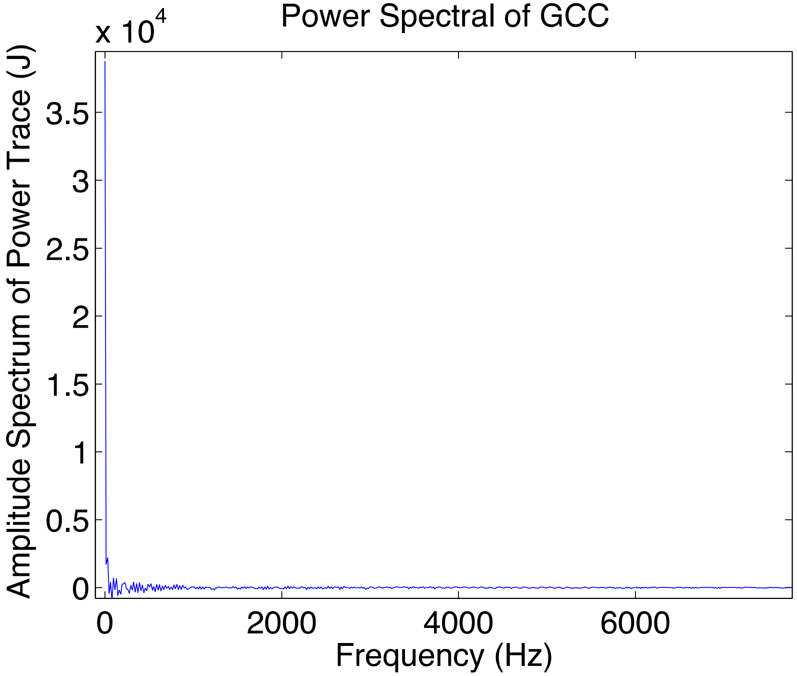
$$T_m = G^{-1}(BU_m - CT_{m-1})$$

$$\Downarrow$$

$$T_0 = G^{-1}BU_0 + G^{-1}CT(t_0)$$

$$= G^{-1}BU_0 + T_{eff}$$

Teff reflects the thermal impacts of previous tasks on current core



0th-moment of dynamic power has most the energy

Physical insight of 0-th moment of T_0

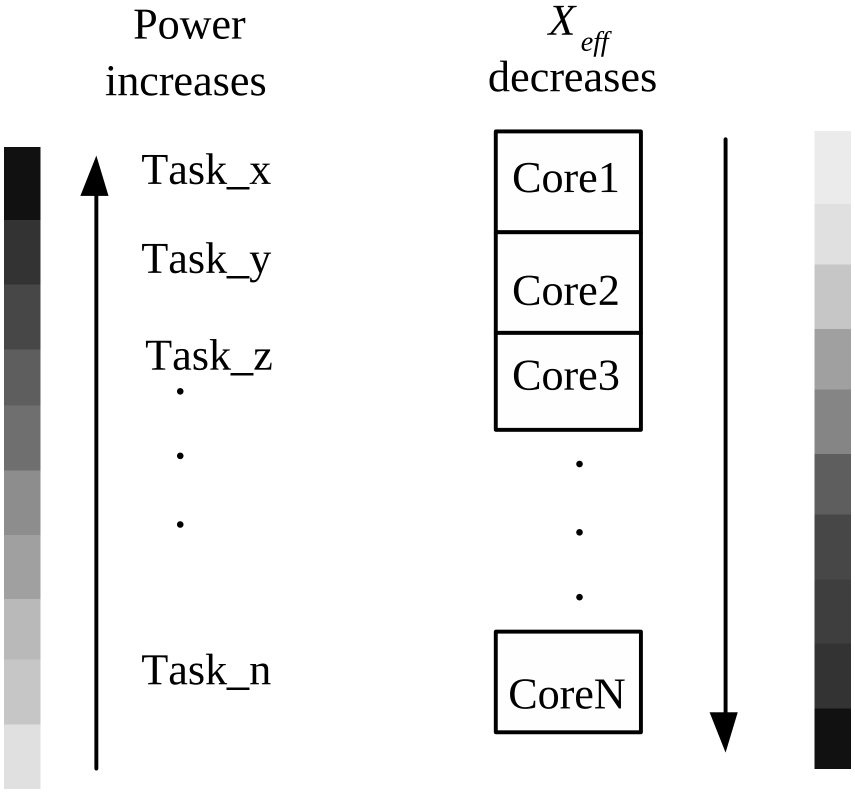
$$T_0 = G^{-1}U_0 + G^{-1}CT(t_0)$$

- $\mathbf{G^{-1}U_0}$: steady state temperature response of current execution cycle.
- $\mathbf{T_{eff}}$ = $\mathbf{G^{-1}CT(t_0)}$: effective initial temperature in frequency domain.
  - *C* - indicate the ability of the system to store energy,
  - *$CT(t_0)$* – energy stored from the previous execution cycle, thus, smaller heat capacitance is favored to reduce temperature since energy stored from the previous cycle is lower.
  - We also need to look at the thermal conducting capability, which is reflected in G^{-1}

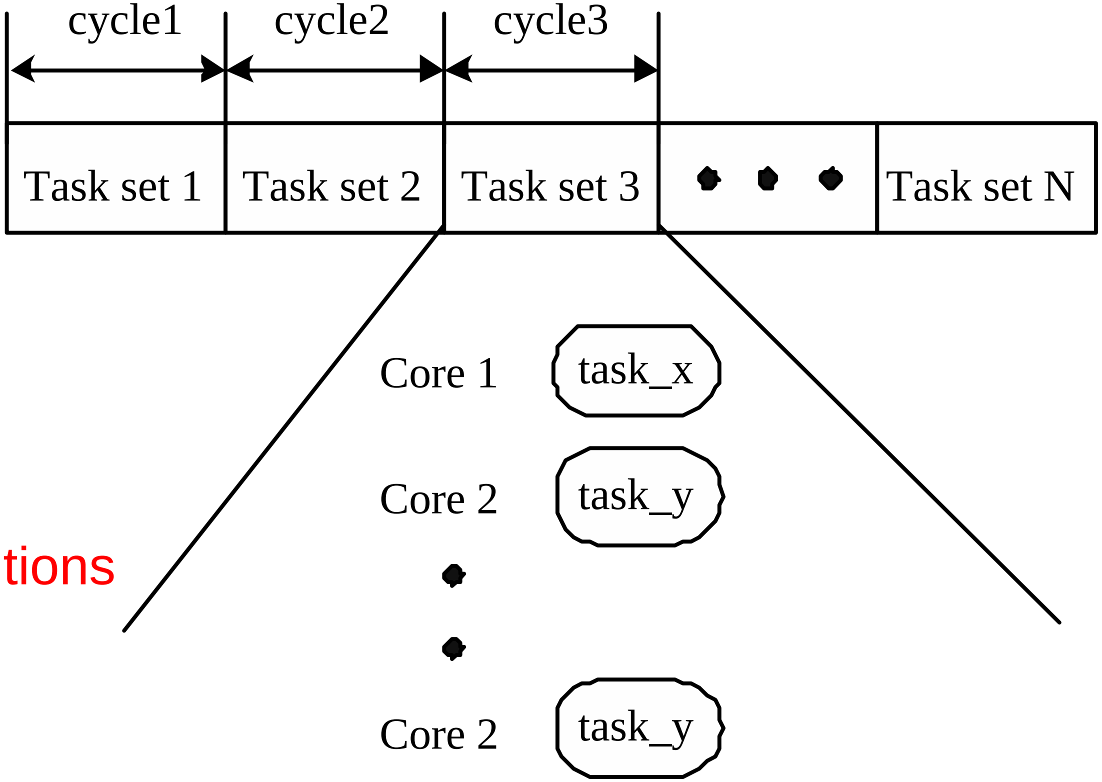
New task migration scheme

Tasks are ranked with power
Cores are ranked with Teff

Heaviest task will be matched
with the lowest Teff



Thermal scheduling framework



Tasks occupies the same time slots.

Task are scheduled Between task executions

Algorithm flow

Algorithm: Thermal Management

- 1. Obtain the power traces for different benchmarks.
- 2. Obtain the frequency domain power spectral for all the power traces.
- 3. If it is the first execution cycle,

  use the known initial temperature.

  Otherwise,

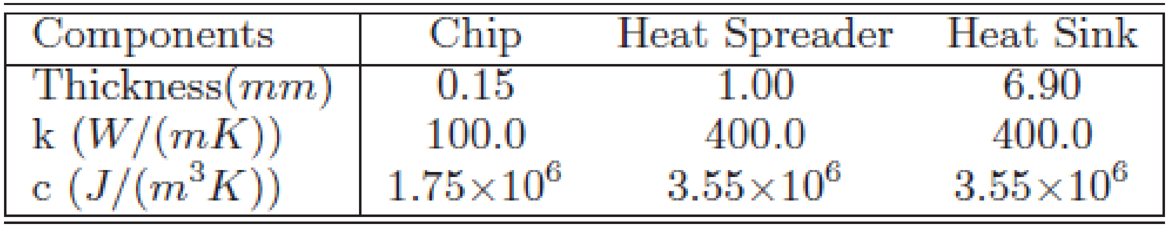
  use the final temperature of the processor at the end of the previous task execution cycle.
- 4. Calculate the frequency domain effective initial temperature T_{eff} or T_0.
- 5. Perform task scheduling where task with largest power is assigned with core with lowest T_{eff} or T_0.

Content

- Introduction to thermal management
- Problem description – task scheduling based thermal management
- Proposed method
- Experimental result
- Conclusion

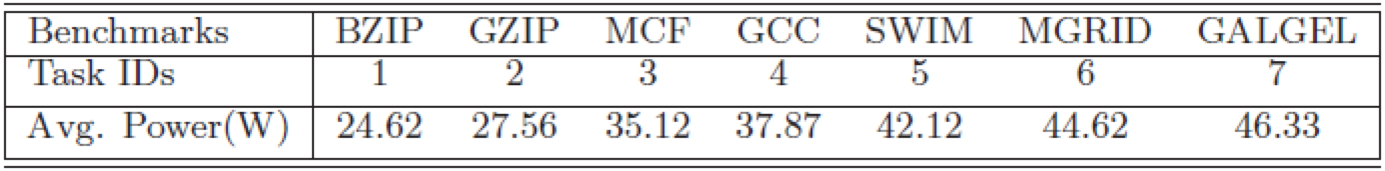
Experiment setup

- Simulation tools:
  - Wattch: Run benchmark to obtain power trace
  - Hotspot: Simulate the temperature response
  - Matlab 7.0: Build task scheduling method
- Platform: 16-core system
- Package structure and thermal properties



Components	Chip	Heat Spreader	Heat Sink
Thickness(mm)	0.15	1.00	6.90
k ($W/(mK)$)	100.0	400.0	400.0
c ($J/(m^3K)$)	1.75×10^6	3.55×10^6	3.55×10^6

- Spec2K Benchmarks (power-ranked) used in simulation



Benchmarks	BZIP	GZIP	MCF	GCC	SWIM	MGRID	GALGEL
Task IDs	1	2	3	4	5	6	7
Avg. Power(W)	24.62	27.56	35.12	37.87	42.12	44.62	46.33

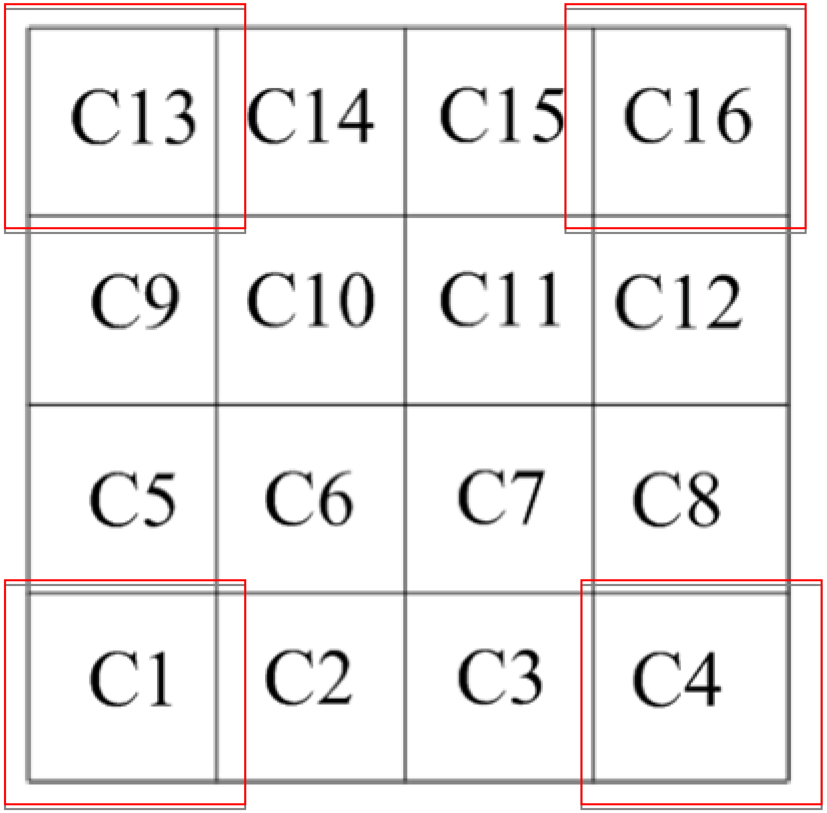
Comparison experiments

- Proposed method:
  - Assign the heaviest task to the core with lowest effective initial temperature $\boldsymbol{T_{eff}}$.
- Comparison 1:
  - **Random scheduling**, that is randomly swap tasks between different cores at the beginning of each execution cycle.
- Comparison 2:
  - **Traditional intuitive approach**, that is always assign the heaviest task to the core with lowest temperature at the beginning of each execution cycle.

Experimental set up

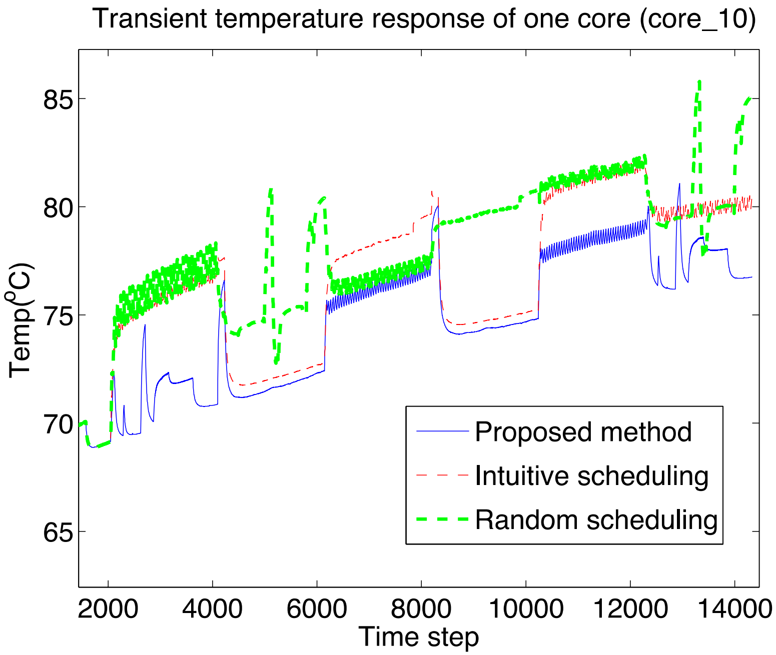
16 core microprocessors
using HotSpot to build
the thermal models.

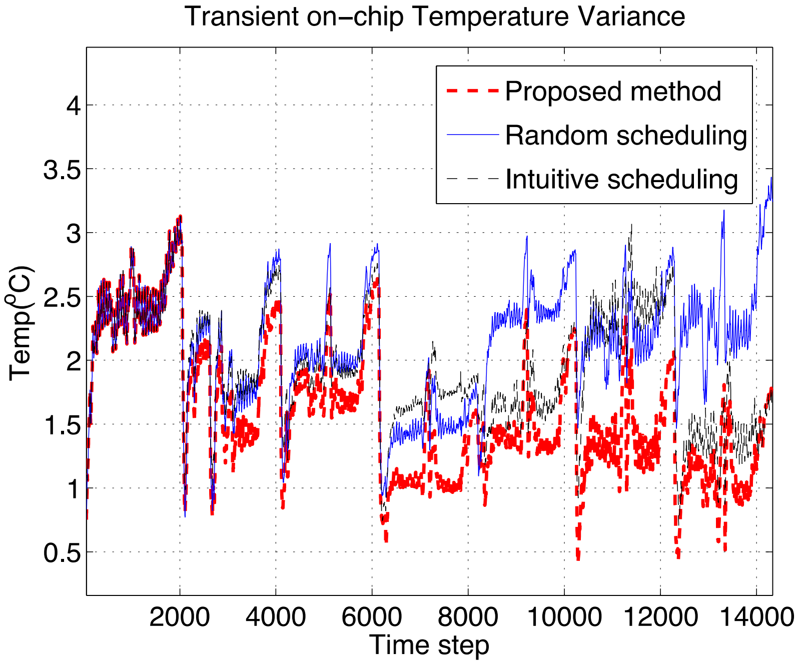
Using Wattch to obtain
The power traces for
SPEC 2000b benchmarks



C13	C14	C15	C16
C9	C10	C11	C12
C5	C6	C7	C8
C1	C2	C3	C4

Comparisons against random and simple lowest temp schemes



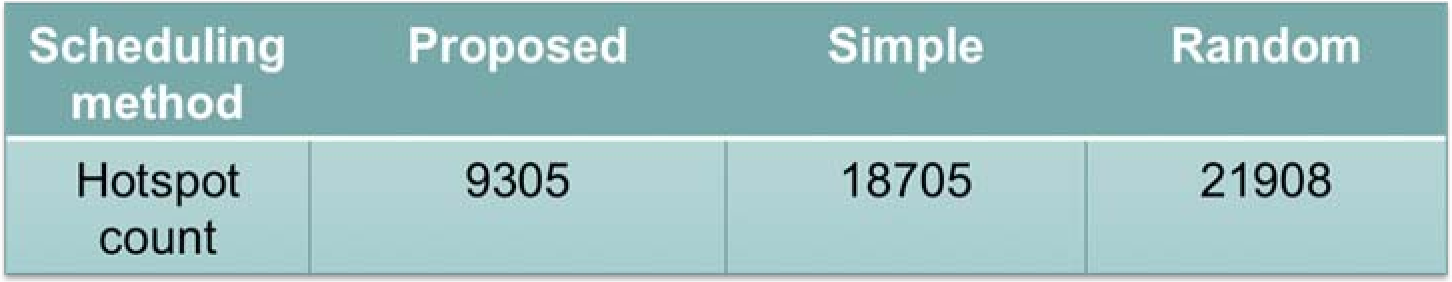


Temperature response comparisons

Temperature response variance comparisons

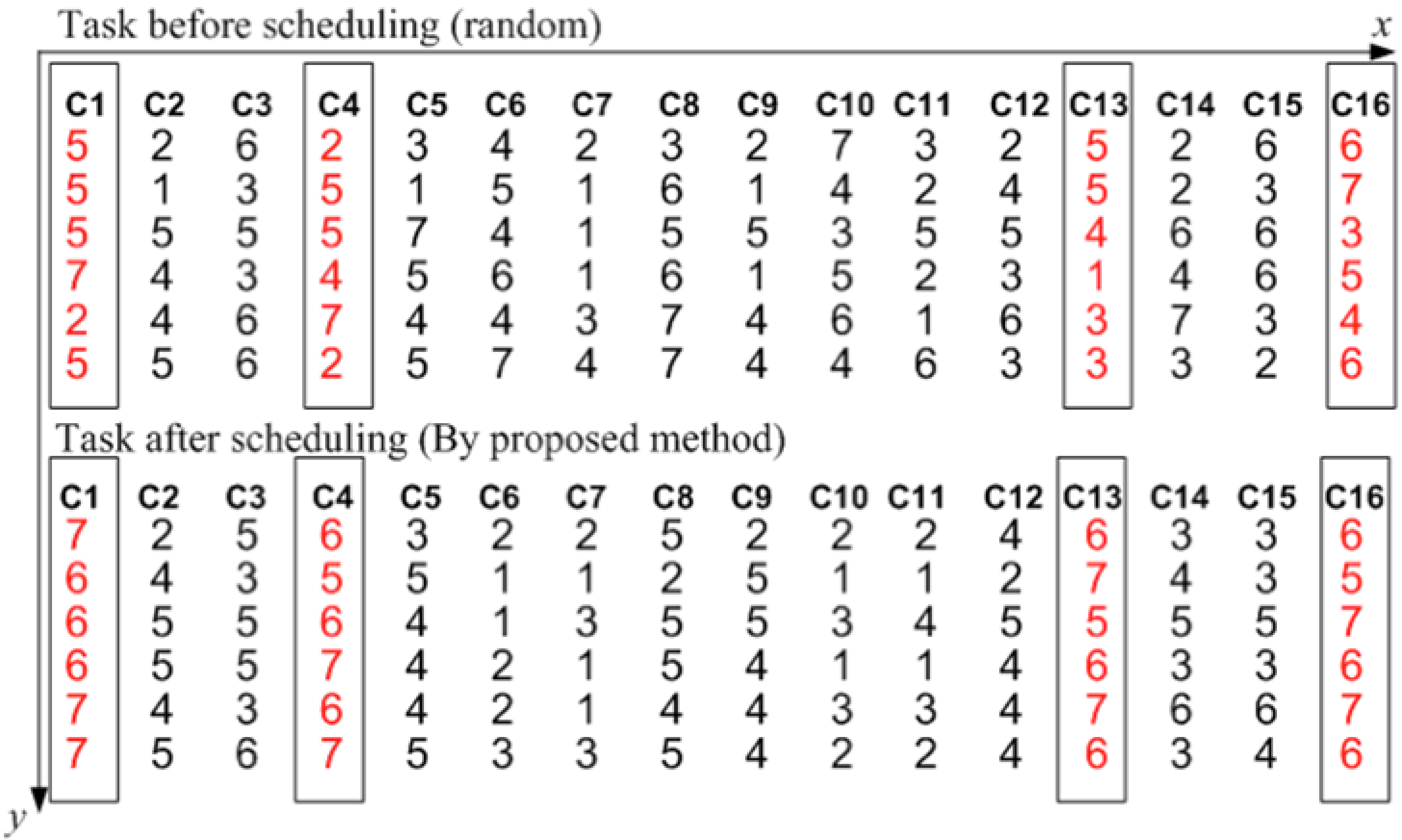
Hot spot comparisons

Statistic of hot spot occurrence during transient simulation



Scheduling method	Proposed	Simple	Random
Hotspot count	9305	18705	21908

Interesting observation



Task before scheduling (random)

C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16
5	2	6	2	3	4	2	3	2	7	3	2	5	2	6	6
5	1	3	5	1	5	1	6	1	4	2	4	5	2	3	7
5	5	5	5	7	4	1	5	5	3	5	5	4	6	6	3
7	4	3	4	5	6	1	6	1	5	2	3	1	4	6	5
2	4	6	7	4	4	3	7	4	6	1	6	3	7	3	4
5	5	6	2	5	7	4	7	4	4	6	3	3	3	2	6

Task after scheduling (By proposed method)

C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16
7	2	5	6	3	2	2	5	2	2	2	4	6	3	3	6
6	4	3	5	5	1	1	2	5	1	1	2	7	4	3	5
6	5	5	6	4	1	3	5	5	3	4	5	5	5	5	7
6	5	5	7	4	2	1	5	4	1	1	4	6	3	3	6
7	4	3	6	4	2	1	4	4	3	3	4	7	6	6	7
7	5	6	7	5	3	3	5	4	2	2	4	6	3	4	6

Task assignments on different cores be- fore (random) and after the proposed scheduling method. Boxed columns are tasks executed on corner cores. The larger task id represents heavier task.

Conclusion

- A new dynamic thermal management method considering transient temperature effect is proposed to reduce on-chip temperature gradient.

- Zero-th order temperature moment is used as thermal predictor.

- The experiment shows that more uniform temperature distribution could be achieved by using the proposed method.